International Journal of Database Theory and Application
Vol.8, No.1 (2015), pp.187-196
http://dx.doi.org/10.14257/ijdta.2015.8.1.19

Database Performance Optimization for SQL Server Based on Hierarchical Queuing Network Model

Jingbo Shao[1,2], Xiaoxiao Liu[3],Yingmei Li[1] and Jingyu Liu[1*]

[1]*College of Computer Science and Information Engineering, Harbin Normal University, 150025 Harbin, China*
[2]*Postdoctoral Research Station of Information and Communication Engineering, Harbin Engineering University, 150001 Harbin, China*
[3]*College of Information, Beijing City University, 100083 Beijing, China*
zro_bo@163.com

Abstract

With the ever-increasing complexity and variety of database workload, database application system has been imposed on higher and higher performance requirements. Database system consists of software and hardware. And the factors that affect database performance are uncertain. In order to tackle the issue of database system for SQL server, this paper proposes hierarchical queuing network model for performance prediction, and a model is established for both software resources and hardware resources, the nested resources are linearized by hierarchical calling, thus finding out the main factors for system performance bottleneck, and system performance is adjusted and optimized accordingly. The performance tuning algorithm for SQL server database based on the hierarchical queuing network is presented in detail. And TPC-C benchmark is adopted for simulation. Experimental results show the proposed method achieves 16.8% performance increase on average, and TPS is improved by 40% compared to previous method.

Keywords: *performance optimization; hierarchical queuing network; SQL server database*

1. Introduction

With the rapid development of science and technology, information system has become necessary in people's daily life. The speed of user response, QoS and running performance of database system determine the vitality of information system. With the time going on, performance of SQL server database system degrades, and the response time of it increases. Optimization of database system plays an important role, and runs through the entire life cycle of database applications.

However performance for most database systems is only assessed after the completion of the entire system at early stage. Worse, performance assessment for some database systems is performed after system deployment. Hence system re-design, and re-evaluation are required. And integration of design process brings about two advantages: 1) architecture incorporating hardware, a single entity including network and its supporting database systems forms. 2) An interactive process is formed, which allows the interaction between hardware and database in the early design phase [1]. The earlier optimization work starts, the less costs. Database system performance tuning should be taken into consideration in design stage [2].

* Corresponding author. E-mail: liujingyu926@163.com

ISSN: 2005-4270 IJDTA

Andrew proposed a novel approach to database performance optimization meeting the requirements of query process. Database structure is described in manner of query-driven, so that the database structure is robust and withstands dynamic queries and uncertainties in the changing environment. Quick and timely information retrieval and exchange are feasible. Other than hardware optimization or traditional database tuning technology, but investigation type and attribute for queries are utilized for improvement in efficiency of database query [3].

Sequence match is an operation that searches the sequence with changing pattern and similar sequence to a given query sequence. Kim proposed a method to improve the performance of the entire sequence match significantly in order to overcome performance bottlenecks. First, the pre-experiment is adopted to analyze the CPU processing time and disk access time in index query post processing steps; based on these results the author proves that the post processing steps for sequence is the main bottleneck for pattern match, the post-processing step optimization method is a key issue neglected by previous methods; in order to solve performance bottlenecks, a simple and efficient way to promote efficiency for post-processing step was proposed. Compared to query sequences, rearranging the order of the candidate sequence is performed. Redundant disk access and CPU processing are completely eliminated in the occurrence of the post processing step. The method achieves high efficiency and does not cause any false negative. Extensive and intensive experiments demonstrate the advantages of this method quantitively. Experimental results show that the proposed method performs faster by 43.36-96.75% times than previous method when using datasets for sequence of real-world stock. The method reduces the proportion of post processing step to the whole sequence match by 67-97%. This means that the proposed method successfully overcomes the performance bottlenecks of sequence match. This method is 16.17-32.64% times faster than when dealing with the real world stock sequence, and 8.64-14.29% times faster when treating large-scale integrated sequence [4].

Queuing network model is used to describe the dynamic behaviors of the Oracle database system design [5]. However the traditional queuing network model is only applicable to modeling independent tasks, while hierarchical queuing networks deals with interdependent tasks, and models both hardware and software, finding out performance bottleneck between them in order for implementation of performance adjustment and optimization of SQL server database system.

The authors suggest that database technology should be based on self-tuning loop feedback control system and are bound to build on mathematical models [7].

Statistically among methods for database system performance improvement by optimization on the network, hardware and other database parameters, 60% of which comes from optimizing the applied program. Optimization of applied program consists of SQL statement and source code. Due to the need to change the program logic, source code optimization time cost is costly, high risk, and has limited impact on database performance optimization; therefore SQL statements optimization for improvement on database performance systems plays a key role.

In this paper, SQL server database system for distributed systems including software resources and hardware resources is studied, in terms of database query optimization on hardware resources and software resources, a novel SQL server database system performance optimization method is proposed. SQL server database performance model based on hierarchical queuing network is presented. The transactions of database are mapped to hierarchical queuing network. The related algorithm is elaborated on. Analysis of the impact of the proposed method on the performance of SQL server database is described. And simulation is performed on TPC-C benchmark. The extent of database performance optimization is predicted.

2. Modeling for Performance Optimization of SQL Server Database System

2.1. Hierarchical Queuing Network Model

Hierarchical queuing network (HQN) is a hierarchical system which includes a server and service. It represents a system occupied by a number of resources provoking one another. And it is mainly used for performance assessment for system containing software resources and hardware resources. The depth of calling depends on the number of layers for continuous call. Hierarchy is determined mainly by the calling relationships between resources.

Hierarchical queuing network model for SQL server database is shown in Figure 1.

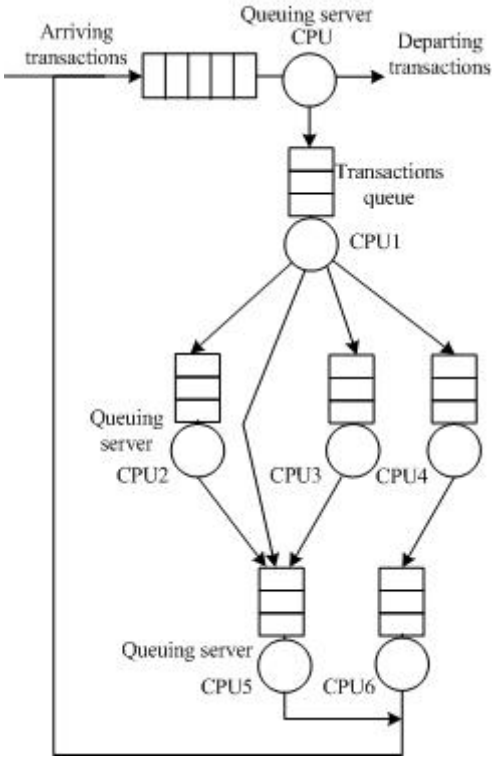

Figure 1. Hierarchical Queuing Network Model for SQL Server Database System

Using queuing network to evaluate database performance, the database designer should use database query optimization techniques to estimate costs of a particular transaction executed on a given database [1].

The SQL server database can be described as a queuing network system. Relations between the server and the transactions can be mapped to a hierarchical queuing network model.

2.2. Working Mechanism for HQN

In this model, the server can be expressed as shared CPU resources; transactions are using these resources. Transactions reach the reception desk (CPU), and CPU serves the transaction immediately when idle, if the reception desk is busy, the transaction needs to wait in queue in front of the reception desk until the previous transaction is served, and leaves the system once service is completed.

A first-in, first-out (FIFO) mechanism is followed. When multiple users use SQL server system simultaneously, each terminal establish a connection to the CPU via network; requests from all the terminals are sent to the controller for processing; and the

final processed information is sent to user terminals. Hierarchical queuing network functions in two ways, *i.e.*, serves as server when required by clients and client when needing service from other servers. All the transactions that arrive at the same disk on the same level queue by means of FIFO, while for those from different disks queue according to the disk they belong to. If different transactions on the same level need CPU for procession, they need to wait in the same way until CPU is idle for service.

3. Algorithm for Performance Optimization of SQL Server Database

Distributed SQL server database system has multi-layered server. Software process running on shared hardware resources may cause contention delays. This is related to grades of thread process, then number of process instance and the allocation of processes to processor. Hierarchical queuing network algorithm mainly deals with distributed SQL server database system performance prediction, based on the results of its predicted results; performance tuning and optimization are implemented.

Algorithm for SQL server database performance optimization (HQN-SPO) is as follows:

Input: the number of servers N_p, running time T_p for server P

Output: previous response time r_p for server P, current response time r_c of task, throughput T_p of server P, the allowed response time tolerance ε, the utility rate U_p of server P.

while （$|r_c-r_p|>\varepsilon$）

do {

do while （$|r_c-r_p|>\varepsilon$）

{

for ($L=M$-1,L<=1,L--)

{

Response times are updated from top to bottom level;

}

New performance candidate NC←generated improved performance candidates;

// candidate C, performance requirements R, Property\initial Property//

New iterated candidate C is selected;

}

Return (the optimal performance setting);

}

The performance parameters for SQL server database C, performance requirements R and properties that need optimizing are known. Using hierarchical queuing network model is formulated for performance adjustment and optimization. The algorithm HQN-SPO is presented. When database performance does not meet the requirements, repeated iterations are made until a new database performance meeting the requirements occurs.

4. Experimental Results

Database design contains tables and transactions that can access these tables. The database performance assessment indicators include several types of time. The total waiting time consuming to access the table waiting for other transactions to access the table of the time -consuming, it can be seen as a query; total time spent accessing data for the total time to get the data and complete the data storage operation the total time; return data to the user depending on the time it takes the data between the client and the server and from the speed; process time consuming statements is determined by the processing rate of the client. The total response time of a transaction is equal to the sum of the three [6].

Online transaction processing benchmark TPC-C[8] is adopted for test analysis of performance optimization results for SQL server database. TPC-C can deal with new transactions, transaction ending, and transaction status. For TPC-C, the number of transactions that can be processed per second (TPS) is utilized for system performance measurement after running a certain period of time. The metrics in Benchmark Factory is adopted for autonomous testing of database performance. The hardware configuration is shown as Table 1.

Table 1. Database Configuration for SQL Server System

	Processor	MM	OS
Server	Intel Core MP(2.7G/2M)	4GB	MS Windows Server 2005 SP3
Client	Pentium ®4 (2.26G HZ)	1GB	MS Windows 2005 SP3

The proposed SQL server database performance optimization method based on hierarchical queuing network is implemented on TPC-C. The test is carried out when database scale is 30 and 60. The detailed database configuration is shown in Table 2.

Table 2. Configuration for SQL Server Database

Situation	Database scale	SQL server version	Database size	MM of server
C1	30	7.0	1.87GB	1GB
C2	30	7.0	3.3GB	4GB
C3	60	7.0	1.87GB	1GB
C4	60	7.0	3.3GB	4GB

Table 3 describes the average crucial time and judging time for different transactions. It can be figured out that transaction payment accounts for the most percentage over the whole transaction, while transaction delivery the least. New transactions do not count. The least crucial time is 5s, and 22s for new transactions.

Table 3. TPC-C Benchmark Transaction

Transaction	Least percentage	Average judging time (s)	Least crucial time (s)
New transaction	N/A	15	22
Transaction ending	61	17	5
Transaction status	6	16	7
Transaction delivery	7	9	6

Table 4 shows the initial loaded scale for HQN model, supposing that page of SQL server database is completely loaded with size of 2048 bytes.

Aiming at four different database configurations, the average queuing time is simulated for HQN and the experimental results are shown in Figure 2.

Table 4. Initial Loaded Scale for HQN Model

Name	Base (byte)	Length of row(byte)	Rows /page
Transaction	5000000	701	5
Old transaction	5000000	48	46
New transaction	1250000	6	271

It can be seen from Figure 2, for the same number of clients, the greater SQL server database is, the longer the average queuing time. With the number of clients increase, the average queuing time upgrades, which conforms to the theory of queuing network.

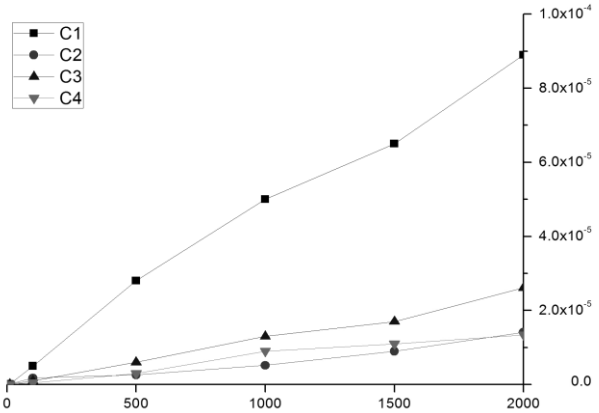

Figure 2. Average Queuing Time for Different Transactions

When the number of clients is over 1000, the increase in the average queuing time degrades, which upgrades when over 1500.

Using the proposed HQN model for SQL server database performance optimization, the related results are shown in figure 3.

In Figure 3, y-axis represents the time of database running (month) and x-axis the total amount of transactions processed per second by database TPS (million), TPS1 and TPS2 represent TPS for SQL server database before and after using HQN.

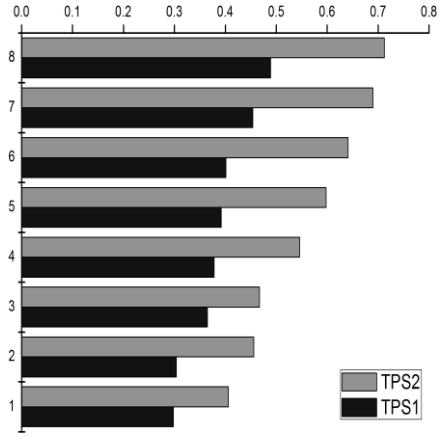

Figure 3. Performance Comparison for SQL server2000, 4GB

Query time and response time for SQL server database system are the main indicators that measure performance of the database. Simulation is performed when scale of user workload is 1-20; experimental results are shown in Figure 4.

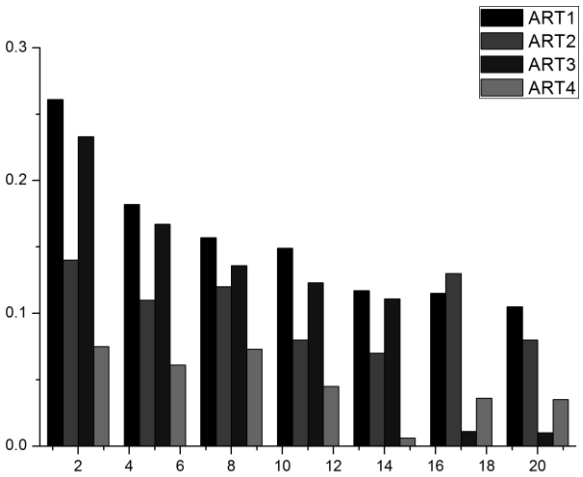

Figure 4. The Average Response Time for Database Under Four Conditions

It can be figured out that the ART decreases with the size of database increases.

In Figure 4, y-axis shows the ART for SQL server database (ms), defined as the average interval from the launch of transaction and the ending of it, x-axis shows the number of user workload. Data from each clique is the average response time under four different database configurations.

The number of bytes fetched from one SQL statement is shown in Figure 5. It can be easily inferred that with the scale of user workload and database increases, Byte1-Byte2 upgrades.

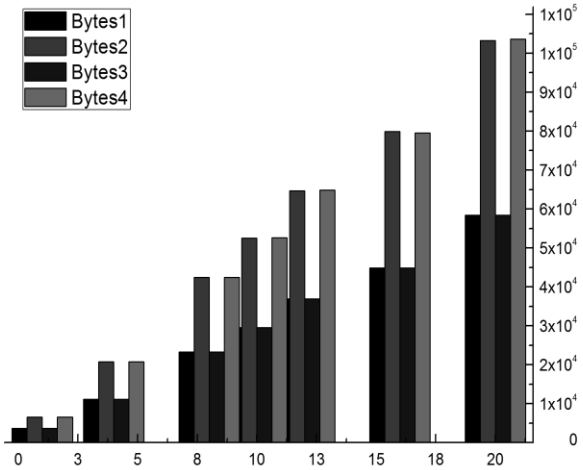

Figure 5. Database Bytes Comparison Under Four Different Conditions

It can also be figured out from Figure 3-6 that, TPS, or ART for database under four different configurations have subtle difference. Moreover with the number of bytes increase, ART decrease instead.

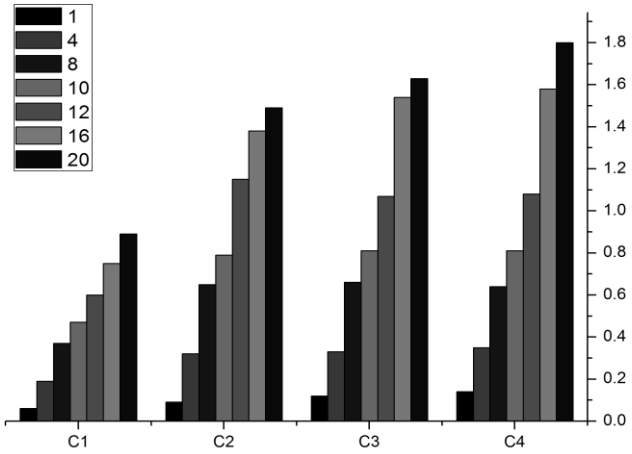

Figure 6a. TPS for Database using Method in [1]

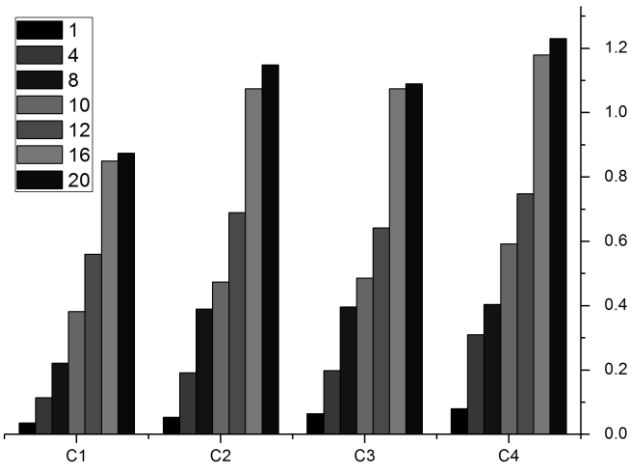

Figure 6b. TPS for Database using Proposed Method

It can be seen From Figure 3 that after using the proposed method HQN, performance for SQL server database is improved by 16.8% and TPS by 40%.

5. Conclusion

A novel database performance optimization model for SQL server is proposed and established. The hierarchical queuing network model is described, formulated and the related algorithm is outlined. The final analysis of SQL server database performance is made. Experimental results on the TPC-C benchmark using HQN indicate that SQL server database performance optimization achieves 40% more performance improvement than the classical algorithm on average.

Acknowledgements

The author, Jingbo Shao would like to thank her university for the support and assistance to the research work. This research work is supported by Heilongjiang provincial Postdoctoral fund under grant No. 326310147, Scientific Research Fund of Heilongjiang provincial Education Department under grant No.12541237 and Advanced Research Project for Science and Technology Development of Harbin Normal University under grant No.901-220601094.This research work is also supported by Innovation Team for Internet of Things on education, Provincial Key Laboratory and engineering research

and development center of college of Computer Science and Information Engineering, Harbin Normal University.

References

[1] R. Osman, I. Awan and M. E. Que, "PED:-Revisiting queuing networks for the performance evaluation of database designs", Simulation Modeling Practice and Theory, vol. 19, no. 1, **(2011)**.
[2] S. Balsamo, A. Di Marco, P. Inverardi and M. Simeoni, "Model-based performance prediction in software development: a survey", IEEE Transactions on Software Engineering, vol. 30, no. 5, **(2004)**.
[3] A. N. K. Chen, "Robust optimization for performance tuning of modern database systems", European Journal of Operational Research, vol. 171, **(2006)**.
[4] S.-W. Kim and B.-S. Jeong, "Performance bottleneck of subsequence matching in time-series databases: Observation, solution, and performance evaluation", Information Sciences, vol. 177, **(2007)**.
[5] R. Osman, I. U. Awan and M. E. Woodward, "Queuing networks for the performance evaluation of database designs", 24th UK Performance Engineering Workshop, Dept of Computing, Imperial College London, London, United Kingdom, **(2008)** July 3-4.
[6] B. Dageville, D. Das and K. Dias, "Automatic SQL tuning in Oracle 10g", Proceedings of the 30th International Conference on Very Large Databases, Toronto, Canada, **(2004)** August 31-September 3.
[7] G. Weikum, A. Moenkeberg, C. Hasse and P. Zabback, "Self-tuning database technology and information services: from wishful thinking to viable engineering", Proceedings of the 28th International Conference on Very Large Databases, Morgan Kaufmann, Hong Kong, China, **(2002)** August 20-23.
[8] G. Powell, "Oracle performance tuning for 10gR2", (Second Edition), Digital Press, **(2006)**.

Author

Jingbo Shao, received the M. E. degree and Ph.D. degrees from Harbin Engineering University, Harbin, China, in 2007, and 2008, respectively. She is currently an associate Professor with the Department computer science and information engineering, Harbin Normal University, Harbin, China. Her current research interests include computer aided design, VLSI design automation.

