Automatic Diagnosis of Performance Problems in Database Management Systems

Darcy G. Benoit
Jodrey School of Computer Science, Acadia University
Wolfville, Nova Scotia, Canada B4P 2R6
darcy.benoit@acadiau.ca

Abstract

Database performance is directly linked to Database Management System (DBMS) resource allocation. Complex relationships between resources make problem diagnosis and performance tuning difficult, time-consuming tasks. Database Administrators (DBAs) currently tune and re-tune the DBMS as the database grows and workloads change. Increased performance and reduced cost of ownership can be achieved by automating the tuning process, starting with resource allocation problem diagnosis. In this paper we overview an automatic diagnosis framework designed to diagnose resource problems. Our diagnosis model and results demonstrate an ability to correctly identify system bottlenecks for a generic On-Line Transaction Processing workload.

1. Introduction

A DBMS is a complex collection of subsystems that each performs a specific task and competes for DBMS resources. Determining which resources need to be adjusted to maximize DBMS performance is a non-trivial task [10]. As databases increase in size and complexity, manually performance tuning becomes impractical [4]. Several calls for the automation of the diagnosis and tuning processes have been made in recent years [2][4][5][7][10]. Automating the diagnosis process is the first step in allowing the DBMS to manage its own resources.

DBMS tuning involves two distinct tasks: diagnosis and resource adjustment. Diagnosis involves determining which resource(s) is responsible for the performance problem. Resource adjustment (or "tuning") involves altering resource allocations to achieve better performance. Automation would allow the DBMS to quickly achieve peak performance at a reduced cost by removing most human interaction.

This paper continues with Section 2 on related research, Section 3 describing diagnosis models, Section 4 presenting test results and Section 5 summarizing our results.

2. Related Work

Related literature has provided information on previous work in the area of automating resource diagnosis. Topics range from automatic memory management in data servers [11], feedback control loops [6], control theory [8], generic agents for performance tuning [3] and goal-oriented resource management [4]. We present a diagnosis system to work within an existing framework described in [1].

3. Diagnosis Framework

We begin our diagnosis system by modeling both the workload and resources. The workload model is used to supply information about the transactions running on the system. The resource model is used to describe the relationships between various DBMS resources. The workload and resource models are used in conjunction with a series of diagnostic rules in order to diagnose the workload. Diagnosis is performed in a feedback loop were the performance of underlying resources are used to gauge the overall performance of the system.

The core of the diagnosis process is the use of the diagnosis model (in tree form) and the resource model. System diagnosis is accomplished by traversing the diagnosis tree. Starting at the root node of the tree, questions are posed about the performance of the DBMS. Depending on the values of particular performance indicators within the DBMS, the tree is traversed until a leaf node is reached. The

Proceedings of the Second International Conference on Autonomic Computing (ICAC'05)
0-7695-2276-9/05 $ 20.00 **IEEE**

leaf node contains a list of one or more resources that should be considered for tuning. We can use the resource model to generate a list of resources that may be affected by adjusting a resource from the current list, allowing us to look ahead at resource adjustments that may need to be made.

4. Testing

To insure that a realistic workload and database were used for our experiments, we used an un-audited TPC-C workload for our test purposes, measuring "tpmC" or "*Transactions Per Minute C*" [9]. Performance information was collected during each workload run.

A naïve tuning strategy was used to determine how diagnosed resources were adjusted. Once a resource was identified by the diagnosis process, we adjusted that resource by a pre-determined amount. The workload was then run again and the new performance was measured.

The workload was run on two difference databases – simulated "small" and "large" databases where the difference was the amount of buffer pool space allocated to each.

4.1. Experiment and Results

For a small database test case, we ran the OLTP workload with a buffer pool size of 100,000 4k pages. Initial performance was 0.0 tpmC (system was deadlocked) and final performance was 6270.76 tpmC. The final performance is compared to the throughput on same system when tuned by a human expert and the throughput when the DB2 Tuning Wizard is used to configure the DBMS. The throughput for the expert configuration is 6355.65 tpmC while the throughput for the wizard configuration is 4966.00 tpmC. The diagnosis system is able to tune the system to 98.67% of the expert throughput and 126.28% of the tuning wizard throughput.

For a large database test case, we simulate database growth by reducing the size of the buffer pool to 50,000 4k pages. We use the same workload and starting resource settings. Initial performance is still 0.0 tpmC (deadlocked system) while the final performance is 4233.65 tpmC. Our diagnosis system is able to tune the system to 98.91% of the expert throughput and 121.95% of the wizard throughput.

In both cases, it was possible for the automatic diagnosis system to correctly identify the resources causing the underlying performance problem and, using a naïve tuning strategy, solve the performance problem. Our system surpassed our goal of matching the DB2 Tuning Wizard performance.

5. Conclusions

This paper demonstrates that a diagnosis framework can be constructed to automatically diagnose a subset of DBMS performance problems. The diagnosis model and the resource model are designed so that they can adapt to a specific DBMS environment.

6. References

[1] Darcy Benoit, Wendy Powley and Patrick Martin. Quartermaster: A Framework for Automatic Tuning of Database Management Systems, School of Computing*, Queen's University, June 1999.*

[2] Phil Bernstein et al*. The Asilomar Report on Database Research,* SIGMOD Record, *Vol. 27, No. 4, pp. 74-80, December 1998.*

[3] J.P. Bigus et al. *AutoTune: A Generic Agent for Automated Performance Tuning,* Practical Application of Intelligent Agents and Multi Agent Technology*, 2000.*

[4] Kurt P. Brown, Manish Mehta, Michael J. Carey and Miron Livny. Towards Automated Performance Tuning For Complex Workloads, Proceedings of the 20th International VLDB Conference*, pp. 72-84, Santiago, Chile, 1994.*

[5] Surajit Chaudhuri. Letter from the Special Issue Editor, Bulletin of the Technical Committee on Data Engineering*, Vol. 22, No. 2, page 2, June 1999.*

[6] Joseph L. Hellerstein. Automated Tuning Systems: Beyond Decision Support. In the Proceedings of the 1997 Computer Measurement Group*, 1997.*

[7] Patrick Martin, Min Zheng, Hoi-Ying Li, Keri Romanufa and Wendy Powley. Dynamic Reconfiguration: Dynamically Tuning Multiple Buffer Pools, Proceedings of the International Conference on Database and Expert System Applications *(DEXA'2000), pp. 92-101, September, 2000.*

[8] Sujay Parekh, Neha Gandhi, Joseph Hellerstein, Dawn Tilbury, T.S. Jayram, and Joe Bigus. Using Control Theory to Achieve Service Level Objectives In Performance Management, in Real-Time Systems, *Vol. 23, No. 1-2, pp. 127-141, 2002.*

[9] TPC-C Specification*, http://www.tpc.org*

[10] Gerhard Weikum, Christof Hasse, Axel Mönkeberg and Peter Zabback. The Comfort Automatic Tuning Project, Information Systems*, Vol. 19, No. 5, pages 381-432, 1994.*

[11] Gerhard Weikum, Arnd Christian König, Achim Kraiss and Markus Sinnwell. Towards Self-Tuning Memory Management for Data Servers, Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*, pp. 3-11, 1999.*