지식 기반 추론 엔진을 이용한 자동화된 데이터베이스 튜닝 시스템[1)]

강승석[O 1], 이동주[1], 정옥란[2], 이상구[1]
서울대학교 컴퓨터공학부 지능형 데이터베이스 연구실[1],
University of Illinois at Urbana-Champaign[2]
{pyxis81, therocks, orjeong, sglee}@europa.snu.ac.kr

Automated-Database Tuning System With Knowledge-based Reasoning Engine

Seungseok Kang[O 1], Dongjoo Lee[1], Ok-ran Jeong[2], Sang-goo Lee[1]
IDS Lab., School of Computer Sci. & Eng., Seoul National University[1],
University of Illinois at Urbana-Champaign[2]

1. 요 약

데이터베이스 튜닝은 일반적으로 데이터베이스 어플리케이션을 “좀 더 빠르게” 실행하게 하는 일련의 활동을 뜻한다[1]. 데이터베이스 관리자가 튜닝에 필요한 주먹구구식 룰(Rule of thumb)들을 모두 파악하고 상황에 맞추어 적용하는 것은 비싼 비용과 오랜 시간을 요구한다. 그렇게 때문에 서로 다른 어플리케이션들이 맞물려 있는 복잡한 서비스는 필수적으로 자동화된 데이터베이스 성능 관리와 튜닝을 필요로 한다. 본 논문에서는 이를 해결하기 위하여 지식 도메인(Knowledge Domain)을 기초로 한 자동화된 데이터베이스 튜닝 원칙(Tuning Principle)을 제시하는 시스템을 제안한다. 각각의 데이터베이스 튜닝 이론들은 지식 도메인의 지식으로 활용되며, 성능에 영향을 미치는 요소들을 개체(Object)와 콘셉트(Concept)로 구성하고 추론 시스템을 통해 튜닝 원칙을 추론하여 쉽고 빠르게 현재 상황에 맞는 튜닝 방법론을 적용시킬 수 있다.

자동화된 데이터베이스 튜닝에 대해 여러 분야에 걸쳐 학문적인 연구가 이루어지고 있다. 그 예로써 Microsoft의 AutoAdmin Project[2], Oracle의 SQL 튜닝 아키텍처[3], COLT[4], DBA Companion[5], SQUASH[6] 등을 들 수 있다. 이러한 최적화 기법들을 각각의 기능적인 방법론에 따라 다시 분류하면 크게 Design Tuning, Logical Structure Tuning, Sentence Tuning, SQL Tuning, Server Tuning, System/Network Tuning으로 나누어 볼 수 있다. 이 중 SQL Tuning 등은 수치적으로 결정되어 이미 존재하는 정보를 이용하기 때문에 구조화된 모델로 표현하기 쉽고 사용자의 다양한 요구에 의해 변화하는 조건들을 수용하기 쉽기 때문에 이에 중점을 두고 성능 문제를 해결하는 데 초점을 맞추었다.

데이터베이스 시스템의 일련의 처리 과정에 따라 DBMS를 구성하는 개체들과 속성, 그리고 연관 관계들이 모델링된다. 데이터베이스 시스템은 Application / Query / DBMS Level의 3개 레벨에 따라 구조화되며, 본 논문에서는 개체, 속성, 연관 관계 및 데이터베이스 튜닝에 사용되는 Rule of thumb들을 분석하여 튜닝 원칙을 포함한 지식의 형태로 변환하였다. 튜닝 원칙은 데이터베이스 시스템에서 발생하는 문제를 해결할 수 있게 하는 일종의 황금률로써, 지식 도메인의 바탕이 되는 사실(Fact)과 룰(Rule)로써 표현된다. Fact는 모델링된 시스템을 지식 도메인의 하나의 지식 개체로 표현하는 방식이고, Rule은 Fact에 기반을 두어 튜닝 원칙을 지식의 형태로 표현한 것이다. Rule은 다시 시스템 모델링을 통해 사전에 정의되는 Rule와 튜닝 원칙을 추론하기 위해 사용되는 Rule의 두 가지 타입으로 나뉘며, 대부분의 Rule은 입력되는 값에 따라 다른 솔루션을 취하게 하는 분기의 역할을 수행한다. 사용자는 제한적으

1) 본 연구는 정통부의 대학 IT 연구센터(ITRC)의 지원으로 수행되었습니다.

로 자동 생성된 Fact와 Rule을 통해 튜닝 원칙을 추론하여 데이터베이스 시스템에 적용할 수 있으며, 요구나 필요에 따라 GUI를 통해 상황에 맞는 Fact와 Rule을 수동으로 추가할 수도 있다.

지식 도메인에서 튜닝 원칙을 추론하기 위해 JAVA 기반의 추론 엔진인 JESS가 사용된다. JESS는 스크립트 언어를 사용하는 전문가 시스템[7]으로, 선언적 룰(Declarative Rule)을 이용하여 지식을 표현하고 추론을 수행하는 추론 엔진의 한 종류이다. JESS의 지식 표현 방식은 튜닝 원칙을 쉽게 표현하고 수용할 수 있는 구조를 가지고 있으며, 작은 크기와 빠른 추론 성능을 가지기 때문에 실시간으로 처리되는 어플리케이션 튜닝에 적합하다. 지식 기반 모듈의 가장 큰 역할은 주어진 데이터베이스 시스템의 모델을 통하여 필요한 새로운 지식을 생성하고 저장하는 것이다. 이를 위하여 Fact와 Rule은 지식 표현의 기본 단위인 트리플(Triple)의 형태로 표현된다, 트리플은 Subject, Property, Object의 3가지 요소로 구성되며, 대부분의 Fact와 Rule들은 트리플의 기본 형태 또는 트리플의 조합으로 이루어진 Condition과 Action의 두 부분의 결합으로 구성된다. 이와 같이 데이터베이스 시스템 모델의 개체들과 속성, 그리고 연관 관계들을 표현함으로써 지식들이 추론 엔진의 Fact와 Rule로 기능할 수 있다.

본 시스템에서는 이를 구현 및 실험하기 위하여 웹 기반 서버-클라이언트 시스템을 가정하였다. 서버는 Process Controller, Parser, Rule Database, JESS Reasoning Engine으로 구성되어 있으며, 클라이언트는 Rule Manager Interface와 Result Viewer로 구성되어 있다. 실험을 통해 얻어지는 튜닝 원칙 적용 전후의 실행 시간 측정 등 데이터베이스 시스템 성능 척도를 비교함으로써 시스템의 효용을 판단하였으며, 실험 결과 적용 전에 비하여 튜닝 원칙을 적용한 경우 최대 1초 미만의 전처리에 따른 부하 시간 추가와 최소 약 1.5배에서 최대 약 3배까지의 처리 시간 개선을 확인하였다.

본 논문에서 제안하는 시스템은 튜닝 원칙을 자동으로 생성하고 지식 형태로 변형시킴으로써 새로운 튜닝 원칙을 파생하여 제공하고, 성능에 영향을 미치는 요소와 함께 직접 Fact과 Rule을 추가함으로써 커스터마이징된 튜닝을 수행할 수 있게 하는 장점을 가진다. 추후 쿼리 자체의 튜닝 및 인덱스 최적화 등의 프로세스 자동화와 Rule을 효율적으로 정의하고 추가하는 방법, 그리고 시스템 모델링을 효과적으로 구성하는 방법에 대한 연구를 통해 본 연구를 더욱 개선시킬 수 있을 것이다.

2. 참고 문헌

[1] Dennis Shasha, Philippe Bonnet, <Database Tuning - Principles, Experiments and Troubleshooting Techniques>, pp.19, 2003

[2] “AutoAdmin:Self-Tuning and Self-Administering Databases", http://research.microsoft.com/dmx/autoadmin/

[3] Benoit Dageville, Mohamed Zait, “SQL Memory Management in Oracle 9i", 28th International Conference on Very Large Data Bases(VLDB), pp.962-973, 2002

[4] Karl Schnaitter et al., “COLT:Continuous On-Line Database Tuning",ACM SIGMOD/PODS 2006 Conference, 2006

[5] Stephane Lopes et al., "DBA Companion: A Tool for Logical Database Tuning", 20th International Conference on Data Engineering(ICDE), 2004

[6] A.M. Boehm et al., "A Tool for Analyzing and Tuning Relational Database Applications: SQL Query Analyzer and Schema EnHancer (SQUASH)", Workshop Grundlagen von Datenbanken 2006, pp.45-49, 2006.

[7] "Jess, the Rule Engine for the Java Platform", http://herzberg.ca.sandia.gov/jess/