데이터베이스 파라미터 튜닝 기법 비교 분석

권세인 ○ JIN HUIJUN 박상현†
연세대학교 컴퓨터과학과
seinkwon97@yonsei.ac.kr, jinhuijun@yonsei.ac.kr, sanghyun@yonsei.ac.kr

Comparative Analysis of Database Parameter Tuning Techniques

Sein Kwon○ JIN HUIJUN Sanghyun Park†
Dept. of Computer Science, Yonsei University

요 약

대용량의 데이터를 효율적으로 관리하기 위해서 사용하는 데이터베이스의 성능을 향상시킬 수 있도록 데이터베이스 파라미터 튜닝 작업이 필요하다. 대표적인 데이터베이스 파라미터 튜닝 기법은 규칙기반 튜닝과 모델기반 튜닝으로 나눌 수 있다. 규칙기반 튜닝은 제한된 샘플 안에서 튜닝을 진행할 수 있지만 데이터셋에 의존도가 높기 때문에 데이터셋의 퀄리티에 따른 영향을 고려할 필요가 있다. 모델기반 튜닝은 시스템 내부 정보 없이도 튜닝을 진행할 수 있지만 많은 데이터가 필요하여 데이터셋을 구축하는 과정에서 시간이 많이 소요된다. 본 논문에서는 규칙기반 튜닝 기법과 모델기반 튜닝 기법의 대표적인 모델을 소개하고 한계점을 바탕으로 향후 데이터베이스 파라미터 튜닝 연구의 진행방향을 제시하였다.

1. 서 론

스마트 시티화가 진행됨에 따라 현대 사회에서는 많은 데이터를 사용하여 지능형 정보 시스템을 생성 및 저장하고 효과적인 데이터 분석 도구를 사용하여 도시의 인구들을 위한 서비스를 제공한다[1]. 데이터베이스는 이러한 데이터들을 체계적으로 수집할 뿐만 아니라 조회 및 삭제할 수 있는 기능을 가지고 있어 수많은 데이터들을 편리하게 관리할 수 있도록 해준다. 일반적으로 사용되는 데이터베이스 종류에는 데이터를 계층 구조로 관리하는 계층형 데이터베이스[2], 2차원 표 형식으로 데이터를 관리하는 관계형 데이터베이스[3,4,5], 데이터를 키-값 형태로 저장하는 키-값 데이터베이스[6,7] 등이 있다. 위와 같이 다양한 데이터베이스를 사용할 때 대용량 데이터를 효율적으로 처리하기 위해서는 데이터베이스 성능 향상이 추가적으로 요구된다. 데이터베이스의 성능을 향상시키기 위한 방법 중 하나는 데이터베이스 파라미터 튜닝이다. 데이터베이스 파라미터 튜닝이란 사용자가 각 데이터베이스 시스템마다 존재하는 파라미터 설정 값을 조정하여 데이터베이스의 성능 향상을 도출하는 작업이다. 하지만 데이터베이스의 버전이 업데이트 될수록 파라미터가 변동되기 때문에 사용자가 직접 버전 변경에 맞춰 튜닝을 진행하기에는 어려움이 있다.

* 이 논문은 2022년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원(IITP-2017-0-00477, (SW 스타랩) IoT 환경을 위한 고성능 플래시 메모리 스토리지 기반 인메모리 분산 DBMS 연구개발)과 국토교통부의 스마트시티 혁신인재육성사업으로 지원을 받아 수행된 연구임.

† 교신 저자: sanghyun@yonsei.ac.kr

이러한 한계를 해결하기 위해서 자동 데이터베이스 파라미터 튜닝 연구가 많이 진행되어 왔다[8,9,10,11] 본 논문에서는 자동 데이터베이스 파라미터 튜닝에 관한 대표적인 연구에 대해 서술하고자 한다.

2. 데이터베이스 파라미터 튜닝

데이터베이스 관리자가 직접 데이터베이스 파라미터 값들을 선정하기에는 파라미터의 수가 많고 파라미터들 간의 복잡한 관계를 모두 이해하는 것은 어렵다. 또한, 클라우드 데이터베이스와 같이 하드웨어와 소프트웨어의 설정이 자주 변경되는 경우에는 매번 인위적으로 튜닝을 진행하는 것은 효율적이지 않다. 그렇기 때문에 자동으로 최적의 성능을 도출하는 파라미터를 추천해주는 기법이 요구된다. 대표적인 파라미터 튜닝 기법은 데이터베이스 시스템의 내부에 대한 자세한 정보 없이도 튜닝을 진행할 수 있는 규칙기반 튜닝과 훈련된 모델을 통해 파라미터를 튜닝하는 모델기반 튜닝으로 나눌 수 있다.

2.1 규칙기반 튜닝

규칙기반 튜닝이란 정해진 규칙에 따라 데이터베이스 성능을 최적화하는 파라미터를 찾는 기법이다. 규칙기반 튜닝을 사용하는 대표적인 모델은 BestConfig[9]이다. BestConfig에서 모델기반 튜닝이 아닌 규칙기반 튜닝을 사용하는 이유는 총 세가지가 있다. 1) 모델기반 튜닝은 모델의 학습이 불가피하기 때문에 많은 양의 데이터를 요구한다. 2) 사용자에게 모델에 대한 선험적인 지식을 요구하는데 일반 사용자들이 데이터베이스 시스템의 설정과 벤치마킹 사용법, 워크로드에 대한 정보를 알고 있는 것은 어렵다. 3) 모델을 사용할 때 요구되는 하이퍼파라미터를 설정하는 것은 데이터베이스 파라미터 튜닝

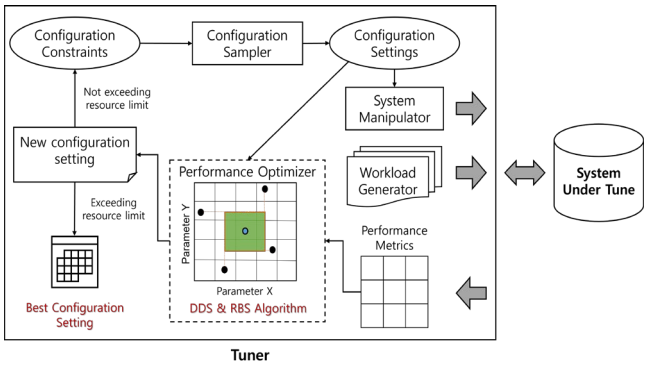

그림 1. BestConfig의 흐름도

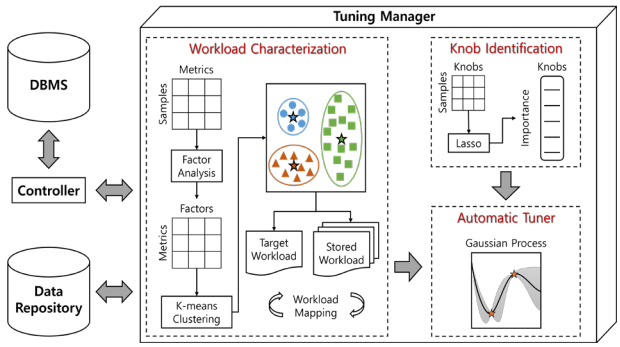

그림 2. OtterTune의 흐름도

을 하는 것만큼 어렵다.

BestConfig 모델의 주 요소는 그림 1과 같이 파라미터 샘플러(Configuration Sampler), 성능 최적화 프로그램(Performance Optimizer), 시스템 조작기(System Manipulator), 워크로드 생성기(Workload Generator)이다. 파라미터 샘플러는 Scalable 샘플링 방법을 구현한다. 성능 최적화 프로그램은 Scalable 최적화 알고리즘을 구현하고 시스템 조작기는 타겟 데이터베이스 시스템의 파라미터 세팅을 업데이트하거나 실행 상태를 모니터링 하는 등의 일을 수행한다. 워크로드 생성기는 타겟 응용 데이터베이스에 알맞은 워크로드를 생성한다. 이 중 시스템 조작기와 워크로드 생성기만이 사용자 데이터베이스 시스템과 연결되어 있기 때문에 다른 데이터베이스 시스템에서 BestConfig를 사용하게 되더라도 시스템 조작기와 워크로드 생성기만 새로운 시스템에 알맞게 수정하면 된다.

파라미터 튜닝의 단계는 파라미터에 대한 제약조건이 주어지게 되면 파라미터 샘플러는 이를 가지고 파라미터 세팅을 생성한다. 생성된 파라미터 세팅은 시스템 조작기에 입력되어 사용자의 데이터베이스 시스템에서 실행되고 성능결과를 성능 매트릭에 저장한다. 이 과정에서의 성능 매트릭과 파라미터 세팅은 성능 최적화 프로그램에 사용되고 주어진 제약조건 안에서 최적의 성능을 도출하는 파라미터 세팅을 찾게 된다.

BestConfig에서는 파라미터 공간을 모두 커버하고 제한된 샘플들을 가지고 성능을 최대화하기 위해 분할 및 발산 샘플링 (Divide&Diverge Sampling, DDS)과 재귀 경계 및 검색(Recursive Bound&Search, RBS) 알고리즘을 제안한다. DDS는 높은 차원에서의 파라미터 공간에서도 넓은 적용범위를 보장하기 위해 n 개의 파라미터가 주어졌을 때 각각의 파라미터를 k 간격으로 나누어 부분 공간을 만드는 샘플링 방법이다. 각 부분 공간에서 랜덤한 점을 찍어 해당하는 부분 공간의 대표값으로 지정한다. RBS는 k 간격으로 나누어진 공간에서의 파라미터 세트에서 가장 좋은 성능을 내는 포인트를 정한 후 그 주변 어떤 범위(bounded space) 안에서 더 좋은 성능을 내는 포인트를 찍는 과정을 반복한다.

2.2 모델기반 튜닝

모델기반 튜닝이란 데이터셋을 수집한 후 기계 학습을 활용하여 최적의 파라미터를 찾는 기법이다. 그 중 대표적인 모델은 OtterTune[10]과 CDBTune[11]이다.

먼저 Ottertune의 구조는 그림 2와 같이 컨트롤러와 튜닝 관리자로 나뉘게 된다. 컨트롤러는 데이터베이스 시스템과 연결되어 있으며 해당 데이터베이스 시스템의 상태에 대한 통계 정보인 내부 매트릭과 데이터베이스 시스템의 성능 정보인 외부 매트릭에 대한 정보를 튜닝 관리자에게 전송한다. 이후 튜닝 관리자는 해당 정보와 이전 튜닝 세션에 대한 정보를 데이터 저장소에 저장한다. 이러한 정보들은 기계학습 모델을 통해 파라미터 값을 추천할 때에 사용된다. Ottertune의 구조는 워크로드 특징 추출(Workload Characterization), 파라미터 식별(Knob Identification), 자동 튜너(Automatic Tuner) 세가지로 구성된다. 먼저 워크로드 특징 추출 단계는 저장소에서 파라미터 파일에 따른 내부 매트릭에 대한 데이터를 가져온다. 내부 매트릭은 데이터베이스 성능과 관련이 없는 매트릭도 포함되기 때문에 요인분석(Factor Analysis)과 K 평균 군집화(K-means Clustering)를 통해 필요하지 않은 매트릭을 제거함으로써, 기계학습 알고리즘의 검색 공간을 감소시켜 전체 튜닝 과정의 속도를 빠르게 한다. 다음으로 파라미터 식별 단계에서 Lasso를 통해 성능에 따른 중요도 순위를 구하여 영향력이 높은 파라미터를 추출한다. 마지막으로 자동 튜너 단계에서 유클리디안 거리(Euclidean Distance)를 통해 목적 워크로드와 데이터 저장소에 저장 되어있는 워크로드의 유사도를 측정하고 유사도가 가장 높은 워크로드와 목적 워크로드를 매핑해준다. 그리고 가우시안 회귀(Gaussian Process Regression)를 통해 목적 워크로드의 성능을 향상시킬 수 있는 최적의 파라미터를 추천한다.

CDBTune도 모델기반 방법론을 사용하지만 파라미터를 튜닝하는 과정에서 OtterTune과는 다르게 시도와 실패(try-and-error)를 통해 학습하는 방식인 강화학습을 적용한다. 또한 CDBTune은 강화 학습을 사용하여 종단 간(end-to-end) 자동 클라우드 데이터베이스(Cloud Database,CDB)를 튜닝해주는 데이터베이스 시스템이다.

CDBTune 모델의 주 구성은 그림 3과 같이 워크로드 생성기(Workload Generator), 매트릭 수집기(Metrics Collector), 추천기(Recommender), 메모리풀(Memory Pool)로 이루어져 있다. 워크로드 생성기는 컨트롤러를 통해 튜닝 요청을 받게 되면 표준 워크로드를 실행하고, 매트릭 수집기는 실행하는 클라우드 데이터베이스의 매트릭 데이터를 수집하고 전처리한다.

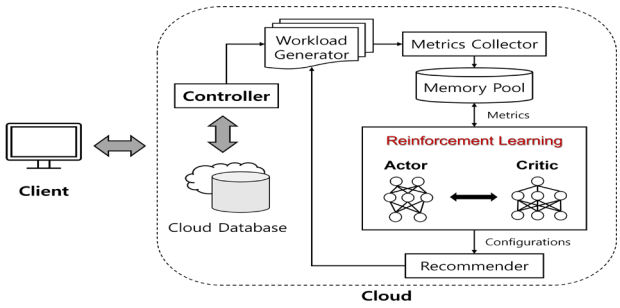

그림 3. CDBTune의 흐름도

처리된 데이터들은 메모리 풀에 저장되어 강화학습 훈련 샘플로 사용된다. 마지막으로 추천기는 강화학습 모델이 파라미터를 추천할 때 클라이언트와 클라우드를 연결해주는 컨트롤러에게 파라미터의 수정을 요청한다. 강화학습 네트워크에서는 특히 DQN(Deep Q Network)과 액터-크리틱이 혼합된 DDPG(Deep Deterministic Policy Gradient) 알고리즘을 사용하여 파라미터 튜닝을 진행한다. DQN이란 어떤 행동에 대한 가치를 리턴하는 함수에서 경험 반복(Experience Replay)과 타깃 네트워크 방법을 추가한 학습 방법이다. 액터-크리틱은 액터에서 정해주는 행동에 대한 가치함수 계산이 크리틱에서 이루어지는 학습 방법이다. 이러한 DDPG 알고리즘은 연속적인 공간에서도 행동가치 값을 얻을 수 있다는 장점을 가지고 있다. 또한 고차원의 상태와 행동, 특히 내부 성능 지표와 파라미터로 정책을 학습할 수 있다.

3. 논의

3.1 한계점

규칙기반 튜닝은 데이터에 의존적이기 때문에 낮은 데이터 퀄리티와 적은 양의 데이터에 대해서는 튜닝 성능이 불안정할 수 있다. 규칙기반 튜닝의 대표적인 모델인 BestConfig는 검색 과정을 매번 재시작해야하기 때문에 이전에 튜닝했던 부분들은 사용할 수 없다는 단점이 있다. 또한 튜닝의 성능이 초기 데이터셋의 퀄리티에 의존적이기 때문에 데이터셋의 디자인과 생성이 중요하게 여겨진다.

모델기반 튜닝은 모델들이 하이퍼파라미터의 영향을 많이 받기 때문에 해당 하이퍼파라미터를 찾는 것 또한 하나의 어려운 과제가 될 수 있다. 모델기반 튜닝을 사용하는 모델 중 하나인 OtterTune은 고차원의 연속적인 공간에서는 사용하기 어려울 수 있으며 파이프라인 학습 모델을 사용하기 때문에 이전 단계에서 최적의 결과값이 다음 단계에서도 최적의 결과값일 것이라고 보장할 수 없으며 모델의 여러 단계가 서로 연동되지 않을 수 있다는 한계점이 존재한다[11]. 또 다른 모델인 CDBTune에서는 튜닝을 진행하는데 있어 파라미터에 대한 정보만 사용하였기 때문에 워크로드의 변화에 따라서 최적의 파라미터를 다시 찾아야하는 한계점이 있다.

3.2 범용성 튜닝 알고리즘의 부재

데이터베이스를 튜닝함에 있어서 모든 저자들은 익숙한 특정 한 개 또는 몇 개의 데이터베이스에 대해서만 연구와 실험을 진행하였기 때문에 이외의 데이터베이스에서 해당 모델과 연구를 적용하기 위해서는 데이터생성과 워크로드 디자인, 파라미터, 벤치 마킹 툴 등을 모두 새롭게 생성하거나 학습해야 할 필요가 있다. 그러므로 특정한 데이터베이스가 아닌 임의의 데이터베이스에서도 적용이 가능한 범용성 튜닝 시스템에 대한 연구가 필요하다.

4. 결론

본 논문에서는 데이터베이스의 성능 향상을 도출해 낼 수 있는 튜닝 기법에 대한 기술 동향에 대해 살펴보았다. 본 논문에서는 데이터베이스의 버전이 계속해서 업데이트됨에 따른 자동 데이터베이스 파라미터 튜닝 기법들을 소개하고 한계점을 서술하였다. 또한 범용성 튜닝 알고리즘의 부재에 대한 연구 필요성을 제시하였다. 향후 연구는 앞서 서술한 한계점을 보완할 수 있는 모델기반의 파라미터 튜닝 기법을 연구하고자 하며 특히 범용성을 고려한 튜닝 알고리즘에 대해 연구를 진행할 계획이다.

참고문헌

[1] Mouchili, Mama Nsangou, Shadi Aljawarneh, and Wette Tchouati. "Smart city data analysis." Proceedings of the First International Conference on Data Science, E-learning and Information Systems. 2018.
[2] MongoDB. https://www.mongodb.com/kr
[3] MySQL. https://www.mysql.com/
[4] PostgreSQL. https://www.postgresql.org/
[5] Oracle. https://www.oracle.com/index.html
[6] RocksDB. http://rocksdb.org/
[7] Redis. https://redis.io/
[8] Li, Guoliang, et al. "Qtune: A query-aware database tuning system with deep reinforcement learning." Proceedings of the VLDB Endowment 12.12, 2118-2130 (2019)
[9] Zhu, Yuqing, et al. "Bestconfig: tapping the performance potential of systems via automatic configuration tuning." Proceedings of the 2017 Symposium on Cloud Computing. 2017.
[10] Van Aken, Dana, et al. "Automatic database management system tuning through large-scale machine learning." Proceedings of the 2017 ACM international conference on management of data. 2017.
[11] Zhang, Ji, et al. "An end-to-end automatic cloud database tuning system using deep reinforcement learning." Proceedings of the 2019 International Conference on Management of Data. 2019.