In-memory DB queue를 이용한 IoT 빅데이터 처리 SW

강정훈, 채철승, 김형구, 민수영, 이명수, 박부식, 이상엽
전자부품연구원
e-mail:budge@keti.re.kr, ccs7006@keti.re.kr, goo4168@keti.re.kr,
minsy@keti.re.kr, leems@keti.re.kr, pusik@keti.re.kr, syublee@keti.re.kr

IoT Big Data Processing SW Using In-Memory DB Queue

JeongHoon Kang*, ChulseoungChae*, HyeongGoo Kim*
*Korean Electronics Technology Institute

요 약

본 논문은 손실이 큰 IoT 빅데이터 처리 고속화 SW의 Queue를 In-memory DataBase로 이용하여 전처리 프레임워크 기술에 대하여 제안하였다.

1. 서론

커넥티드 자동차 데이터는 자동차 진단 기술인 OBD(On-Board Diagnostics) IoT를 이용하여 데이터를 수집한다. 이 표준은 자동차에 부착된 센서에서 전자제어 장치로 달된 자동차 주요 계통 정보나 고장 등의 정보를 직렬(Serial) 통신기능을 이용하여 자동차의 콘솔이나 외부 장치에서 볼 수 있도록 한 기능이다[1]. 본 논문에서 사용한 데이터는 OBD로부터 수집된 데이터이다. OBD장치는 자동차의 OBD단자의 전원으로 동작하기 때문에 자동차의 시동이 켜져 있을 때에만 데이터가 수집된다. 또한 OBD장치는 센서에서 측정한 데이터를 수집하기 때문에 센서 혹은 장비의 오류로 인해 데이터가 손실되는 경우가 발생한다[2][3]. 따라서 하루 기준으로 일부 구간에서 데이터가 손실되는 경우가 발생한다. 데이터 처리 방식은 원본 데이터를 읽어 전처리하고 데이터베이스에 저장하는 과정을 멀티프로세싱을 이용하여 병렬로 처리하는 형식이었다. 이는 수집된 데이터가 매초마다 거의 완벽하게 있는 데이터들에 대해서는 각 프로세스마다 비슷한 양의 데이터가 처리되기 때문에 다른 처리 방식과 비슷한 처리속도이지만 현재 자동차 데이터 수집기능으로 수집된 손실이 있는 데이터에 대해서는 오히려 처리시간이 더 오래 걸릴 수 있다. 기존 알고리즘은 날짜별로 DB에서 데이터를 쿼리하여 각 프로세서에 전달하여 전처리하는 형식이다. 하지만 손실이 있는 데이터에 대해서는 전처리 알고리즘에서 동작중인 프로세스가 있는 반면에 데이터가 적은 부분의 프로세스는 작업이 먼저 끝나 동작을 하지 않는 현상이 생긴다는 문제가 있다. 이를 해결하기 위해서는 원본 데이터 쿼리, 전처리, 데이터베이스에 저장, 이러한 세 가지 과정을 분할하고 병렬처리 과정에서 각 프로세스가 작업이 끝나면 다른 새로운 작업을 즉시 실행하도록 구성 하였다.

본 논문에서는 DB에서 데이터를 쿼리하여 전처리하여 DB(데이터베이스)에 저장하는 작업을 매번 새롭게 정의하지 않고 단순히 전처리 작업 및 간단한 드라이버 함수만 작성하면 되는 프레임워크의 큐 부분을 In-memory Database로 사용하여 구현 하였다.

2. 디자인 패턴 및 프레임워크 구성 내용

필요 작업 내용 탐색-데이터 전처리-데이터 저장/전송 작업은 모두 하나의 파이프라인 상에서 동작해야 한다. 이와 같은 작업 특성을 고려하여 생산자-소비자 디자인 패턴을 적용하였으며 각각의 작업은 다음과 같은 과정을 거쳐 수행된다.

데이터 전처리 작업이 진행되기 위해서는 생산자 프로세스에게 처리해야 할 작업 내용을 전달해야 한다. 따라서, 작업 목록을 읽어서 생산자(데이터 전처리 작업 수행 프로세스)에게 '처리해야 할 작업 내용'을 전달한다. 마지막으로, 생산자는 '데이터 전처리'를 수행하고 소비자는 전처리된 데이터를 'DB로 전송'하는 작업을 수행한다.

필요 작업 내용 탐색-데이터 전처리-데이터 저장/전송 작업은 모두 서로 다른 프로세스에서 실행된다. 각각의 작업은 이전 작업 결과에 의존하는 만큼, 기존의 시스템에서는 이전 작업의 결과를 전달하기 위한 방안으로 '메시지 큐'를 사용하였다. In-memory DB를 이용한 큐 사용과 관련한 성능 및 구조상의 이점과 관련해서는 '4. In-memory DB를 이용한 큐'에 기술하였다.

3. 시스템 구성 및 핵심 기능

본 논문의 시스템에서 가장 핵심적인 부분은 '데이터 전처리', '데이터 저장/전송' 부분이다. 이 부분은 Workers 클래스에 구현되어 있다. Workers 클래스는 내부적으로 N:M 생산자-소비자 프로세스를 생성 및 실행하는 역할을 한다. 이때 생산자 프로세스 내부적으로 특정 전처리 작업을 수행한다. 해당 전처리 작업을 사용자가 '전처리 함수명'을 전달함으로써 지정할 수 있으며, Preprocessing이라는 별도의 모듈에 정의되어 있다.

다음으로, '필요 작업 내용 탐색' 작업을 수행하는 유틸리티 기능은 별도의 모듈(Utils)로 구현하였다. Utils 모듈에는 사용자 목적에 맞는 데이터 준비 작업을 수행할 수 있는 여러 함수가 정의되어 있다. 그중 하나로 'Database에서 데이터를 쿼리하는 기능'이 있다. 본 논문에서 제시하는 시스템에서 활용한 데이터는 시계열 Database인 openTSDB에 저장되어 있기에 '필요 데이터 준비'에 해당하는 기능이다.

마지막으로, 위에서 언급한 Workers, Utils 모듈을 사용하여 하나의 query_driver를 작성함으로써 하나의 빅데이터 처리 시스템이 구현되었다고 볼 수 있다. 이때 사용자는 단순히 자신의 목적에 맞게 전처리 함수를 지정/정의(Preprocessing 모듈)함으로써 손쉽게 빅데이터 처리 시스템을 실행할 수 있다는 장점이 있다.

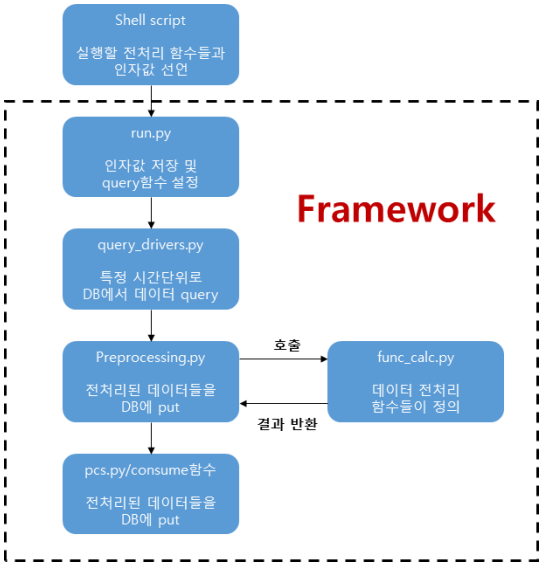

그림 1 전처리 Framework 모식도

4. In-memory DB를 이용한 Queue

기존의 SW 시스템에서는 DB에서 데이터를 쿼리요청하여 리턴되는 결과와 전처리 후 결과 데이터를 정의된 프로세서들에 전달하기 전 중간파이프인 Message Queue를이용하여 구성하였다. 개선된 시스템에서는 Message Queue대신 In-memory DB인 REDIS DB를 사용하였다.

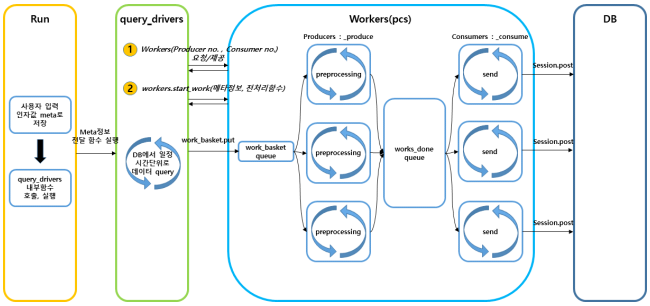

그림 2 기존 Message Queue를 이용한 시스템구성

레디스(Redis)는 Remote Dictionary Server의 약자로서, "키-값" 구조의 비정형 데이터를 저장하고 관리하기 위한 오픈 소스 기반의 비관계형 데이터베이스 관리 시스템(DBMS)이다. Memory 위에서 동작하는 Key/value 저장소(Store)인 레디스는 NoSQL DBMS로 분류되며 동시에 Memcached와 같은 인메모리(In-memory) 솔루션으로 분리된다. 성능은 Memcached의 장점을 기반으로 만들어졌기 때문에 Memcached보다 우수 하고, 다양한 데이터 구조체를 지원하기 때문에 Message Queue, Shared memory, Remote Dictionary 용도로 사용될 수도 있다.

외부층에서 레디스의 데이터 모델은 키를 값에 맵핑하는 자료구조의 디렉토리이다. 레디스 그리고 기타 구조화된 스토리지 시스템 사이의 주요 차이점 중 하나는 레디스는 스트링뿐만 아니라 자료구조의 추상적 자료형(Abstract Data Type)도 지원을 한다는 점이다[4].

이로인해 메모리를 활용하면서 영속적인 데이터 보존이 가능해졌다. 또한 value 값으로 문자열, 리스트, Set, Sorted set, Hash 등 여러 데이터 형식을 지원하고 리스트형 데이터 입력과 삭제가 MySQL에 비해서 10배정도 처리 속도가 빠르고, 리스트, 배열과 같은 데이터를 처리하는데 유용하기 때문에 다양한 방식으로 데이터를 활용이 가능해졌다. 그리고 여러 프로세스에서 동시에 같은 key에 대한 갱신을 요청할 경우, Atomic 처리로 데이터 부정합 방지 Atomic처리 함수를 제공하기 때문에 멀티프로세싱 사용에 호환이 가능하다.

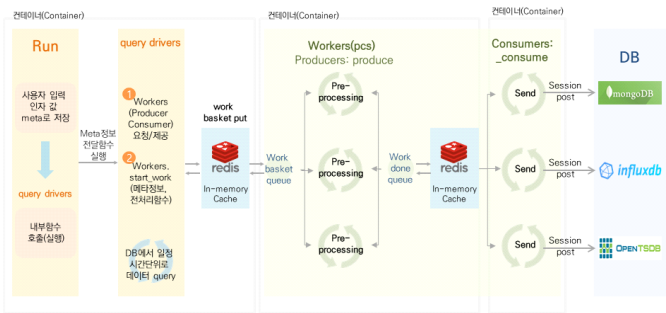

그림 3 redis DB queue를 이용한 개선된 시스템

5. 결론

본 논문에서는 기존 전처리 기술인 손실이 있는 차량 빅데이터의 처리 고속화 SW의 Queue를 In-memory DataBase를 이용한 기술에 대하여 제안하였다. 이로인해 명령어로 명시적으로 삭제, expires를 설정하지 않으면 데이터가 삭제되지 않음으로 메모리를 활용하면서 영구적인 데이터 보존이 가능해졌다. 또한 데이터의 value 값으로 문자열, 리스트등 여러 데이터 형식을 지원하고 리스트형 데이터 입력과 삭제가 MySQL에 비해서 10배정도 빠름으로인해 리스트, 배열과 같은 데이터를 처리하는데 유용하기 때문에 다양한 방식으로 데이터를 활용이 가능해졌다.

본 논문의 시스템은 전처리 부분과 데이터베이스에 저장하는 부분을 분할하고 각 작업과 함수를 모듈화하는 시스템으로 이루어졌다. 이로 인해 손실이 있는 데이터에 대하여 좀 더 빠른 속도로 처리가 가능하다. 또한 멀티프로세싱 과정에서 각 프로세스가 작업이 종료되면 정지상태가 되지 않고 바로 다음 작업을 실행할 수 있도록 하여 시간적인 낭비가 발생하지 않도록 하였다.

제안된 시스템의 Queue를 In-memory DataBase를 이용한 기술로 대용량 빅데이터 처리 SW에 대한 적용이 용이해질것으로 예상된다.

ACKNOWLEDGMENT

본 논문은 산업통상자원부 및 한국산업기술평가관리원 산업핵심기술개발사업의 일환으로 수행하였음. (20004537, 제조 빅데이터 공동활용을 제공하는 클라우드 플랫폼 기술 개발 및 실증)

참고문헌

[1] https://ko.wikipedia.org/wiki/온보드진단기

[2] 서원태, and 현병언. "IoT 기반의 택배안전 서비스에 관한 연구." 한국통신학회 학술대회논문집 (2017): 433-434.

[3] Chen, Shi-Huang, Jeng-Shyang Pan, and Kaixuan Lu. "Driving behavior analysis based on vehicle OBD information and adaboost algorithms." Proceedings of the International MultiConference of Engineers and Computer Scientists. Vol. 1. 2015.

[4] https://ko.wikipedia.org/wiki/redis