韓國 컴퓨터情報學會 論文誌
第13卷 第1號, 2008. 1.

2008-13-1-1-8

질의 전처리기를 사용한 스트림 DBMS의 효율적 질의처리

양 영 휴*

An Efficient Query Processing in Stream DBMS using Query Preprocessor

Yang, YoungHyoo *

요 약

유비쿼터스 시대의 텔레매틱스 데이터 관리는 자동차의 위치나 속도, 엔진의 속도, 타이어의 상태, 운전자의 관심사항등의 실시간으로 유입되는 스트림 데이터에 대한 질의를 처리하는데 있다. 본 논문에서는 기존의 스트림 DBMS의 질의처리 연구현황을 비교 분석하고, 스트림 DBMS에서 다뤄야 하는 모든 유형의 질의를 질의 전처리기를 사용하여 하나의 통합된 시스템에서 처리할 수 있는 통합 하이브리드 모델을 제안한다. 최근 각종 장치의 가격은 하락하는 반면, 성능은 수직 상승함에 따라 DB와 큐등을 위한 공간을 추가함으로써 최대의 병렬성을 보장 받을 수 있다. 그 결과 제안된 하이브리드 모델에서는 다양한 유형의 스트림 DBMS 질의들을 단일 시스템 내에서 일괄적이며 효율적으로 처리하여 시스템 성능 향상을 기대 할 수 있다.

Abstract

The telematics data management deals with queries on stream data coming from moving cars. So the stream DBMS should process the large amount of data stream in real-time. In this article, previous research projects are analyzed in the aspects of query processing. And a hybrid model is introduced where query preprocessor is used to process all types of queries in one singe system. Decreasing cost and rapidly increasing performance of devices may guarantee the utmost parallelism of the hybrid system. As a result, various types of stream DBMS queries could be processed in a uniform and efficient way in a single system.

▶ keyword : 텔레매틱스 데이터 관리(Telematics Data Management), 스트림 DBMS(Stream DBMS), 웨어하우징 기법과 분산 기법(Warehousing & distributed approach), 영속질의(Continuous query), 히스토리칼 질의(Historical query),스냅 샷 질의(Snap-shot query)

• 제1저자 : 양영휴
• 접수일 : 2008. 1. 20, 심사일 : 2008. 1. 24 심사완료일 : 2008. 1. 25
* 한양여자대학 인터넷정보과 교수

Ⅰ. 서 론

컴퓨팅 환경의 급격한 변화와 발전은 유비쿼터스 시대를 열었으며, 컴퓨터는 장소와 시간의 제약을 받지 않고 언제 어디서나 눈에 보이지 않는 자원을 사용, 사용자의 존재를 인식하여 스스로 판단하여 정보를 제공하며, 특별히 사용자의 요구가 있지 않더라도 적절한 조치를 취해주는 최적의 환경을 제공하여 주게 되었다. 즉, 유비쿼터스 서비스는 사용자의 필요에 따라 다이내믹하게 맞춤형 서비스를 제공하여 주는 것이다. 이러한 서비스는 사용자의 활성 장치와 상황에 따라 적절하게 맞추어 진다. 이 분야의 서비스로는 환경 모니터링, 스마트 홈, 지능형 빌딩, 실시간 금융서비스, 그리고 텔레매틱스를 들 수 있다. 이 중 텔레매틱스는 텔레커뮤니케이션과 인포매틱스 기술을 통합하여 텔레커뮤니케이션을 이용, 이동 중인 자동차 안의 사용자들에게 정보 서비스를 제공하여 주는 것이다. 서버사이드의 텔레매틱스 구조는 콘텍스트 데이터 관리, 위치, 프로파일, 환경, 온라인 데이터 모니터링과 지식 기반 시스템, 보안과 사생활 보호등을 지원해 주어야 한다. 정보 서비스로는 실시간 교통정보, 인터넷 서비스, M-커머스(Mobile Commerce), VOD, 게임, 뉴스, 날씨와 주식 정보 등이 있다. 콘텍스트 인식 서비스로는 차량 네비게이션, 자동차 보험, LBS(Location-based service,위치 기반 서비스), 차량진단 서비스 등을 들 수 있다. 보안 서비스로는 차량도난 방지, 노약자를 위한 비상 대책 등이 있으며 자원 검색 서비스로는 옐로우페이지 서비스, 식당, 주유소 안내 및 인터넷과 프린터 서비스 등이 있다. 자동차 위치나 속도, 엔진의 온도, 타이어의 상태, 운전자의 관심사항들이 센서에 의해 감지되어 주기적으로 계속 보고 되는데, 이러한 종류의 데이터는 실시간으로 계속 유입되는 특성이 있어 스트림 데이터(stream data)라고 한다. [1,2]

텔레매틱스 데이터 관리는 크게 두 가지로 분류된다. 클라이언트 측에서의 데이터 관리와 서버 측에서의 데이터 관리가 그것이다. 클라이언트 측의 데이터 관리는 모바일 디바이스의 DBMS와 임베디드/울트라 타이니/Java 데이터베이스로 구분되는 모바일 DBMS를 포함한다. 서버 측의 데이터 관리로는 이동객체 DBMS와 스트림 DBMS가 있으며, 이동객체 DBMS는 위치기반 서비스를 위한 DBMS이며 이동객체의 위치를 추적하고 예측하는 것이다. 이에 비해 스트림 DBMS는 자동자의 위치나 속도, 엔진의 속도, 타이어의 상태, 운전자의 관심사항등의 실시간으로 계속 유입되는 이러한 stream 데이터에 대한 질의(query)를 처리하는데 그 중점적 역할이 있다.

센서 데이터나 GPS등의 데이터 스트림을 위한 스트림 DBMS의 응용 분야로는 텔레매틱스, 센서 스트림 모니터링과 환경 모니터링에 응용되며, 가장 큰 특징으로는 실시간으로 많은 양의 데이터를 처리하여야 하며 연속 질의를 지원하여야 하므로 기존의 정적인 데이터를 처리하는 DBMS와는 차이가 있다.

Ⅱ. 기존의 스트림 DBMS

텔레매틱스가 구현되는 환경은 기존의 그것과는 차이가 있게 마련이다. 자동차의 위치와 속도, 엔진이나 타이어의 상태, 운전자가 가지는 관심사항들이 센서에 의해 주기적으로 감지되어 보고된다. 이러한 데이터는 실시간이며, 끊임없이 지속적으로 유입되는데 이러한 스트림 데이터는 기존의 정적인 데이터와는 다음의 차이점을 가지고 있다.

가. 데이터가 네트워크를 통해서 실시간으로 지속적으로 들어온다.
나. 데이터베이스 시스템 내에서 이러한 데이터의 도착 순서를 조정할 수 없다.
다. 데이터의 크기가 무제한적이다.
라. 데이터의 형태가 매우 다양하다.

기존의 DBMS는 이미 저장되어 있는 정적인 데이터를 효율적으로 관리하고, 최적화된 상태로 질의를 처리하도록 설계되어 있으므로, 실시간으로 유입되는 대용량의 다양한 형태의 스트림 데이터를 처리하기에는 적합하지 않으므로 새로운 형태의 스트림 DBMS가 요구되어진다.

또한 텔레매틱스 서비스는 기존의 데이터베이스에서는 거의 처리하지 않았던 영속질의 (Continuous Query)를 처리할 수 있어야 한다. 영속 질의란 일정시간 데이터베이스 시스템 내에 존재하면서 이 질의의 조건에 부합되는 새로운 정보가 들어오면 사용자에게 계속 결과를 전달해 주는 질의를 뜻한다.

종전에는 데이터베이스에 미리 저장되어 있는 데이터에 대한 질의들이 대부분이었기 때문에 질의 결과는 항상 즉시 사용자에게 전달되었다. 하지만 텔레매틱스 환경에서는 오랜 시간 데이터베이스에 존재하면서 질의 조건을 검사하는 영속 질의들이 빈번하게 들어온다. 하나의 질의가 장시간 존재하면서 처리되기 때문에 질의처리 시스템은 여러 개의 이러한 영속질의를 동시에 처리 할 수 있어야 한다. 또한 영속질의는 처리 결과가 스트림 데이터가 되는 경우가 종종 있어서 질의 하나가 기존의 질의에 비해 많은 컴퓨팅 자원을 요구하는 경향이 있다. 텔레매틱스 장비나 각종 센서들이 보내는 데이터

들을 처리하는 시스템은 이러한 영속 질의의 특성을 반영하여 설계되어야 한다.

2.1 Cougar Device DBMS

미국 Cornell 대학의 Cougar Device DBMS는 스트림 데이터에 대한 질의를 세 가지 유형으로 정의하고 분류하였다. 히스토리칼(historical) 질의는 이미 데이터베이스에 저장되어 있는 정보에 대한 질의를 의미하며, "지난 2007년의 평균 강우량을 알려 달라" 와 같은 질의가 이에 속한다. 스냅 샷(snap-shot) 질의는 현시점에서 얻을 수 있는 정보에 대한 질의를 뜻하며, "현재 성수대교를 지나는 차량들의 평균속도를 보이시오."와 같은 질의가 스냅 샷 질의이다. 또한 장기 수행(long-running) 질의는 어느 특정 기간 동안에 주기적으로 처리 결과를 알려주어야 하는 질의를 의미하며 "앞으로 10시간 동안 서울지역의 모든 측우기의 강우량을 30분 간격으로 알려 주시오"와 같은 질의가 그것이다.

기존의 시스템에서는 스트림 데이터를 처리하기 위하여 웨어하우징 기법(warehousing approach)을 사용하여 왔다. 모든 센서에서 들어오는 정보를 중앙 집중화된 데이터베이스 시스템이 받아서 저장하고, 거기에서 모든 질의를 처리하는 방법을 사용한 것이다. 이러한 접근법은 히스토리칼 질의를 처리하는 데에는 적합하였지만, 다음과 같은 단점이 있다.

첫째, 웨어하우징 접근법은 측정 장치에 직접 질의를 전달하지 못하는 단점이 있다. 중앙 집중화 된 데이터베이스 시스템은 센서에 보내는 데이터를 수동적으로 받아서 질의를 처리하도록 되어 있다. 따라서 각각의 센서에 대한 제어가 불가능하기 때문에 어느 특정 센서에 대한 특화된 질의를 처리할 수가 없게 된다. 둘째, 웨어하우징 접근법은 센서에서 감지된 정보를 바로 중앙 집중화된 데이터베이스에 전달하므로 많은 자원을 낭비하게 된다. 센서 네트워크내의 모든 데이터가 계속해서 중앙 데이터베이스에 집중된다면, 그 자체로 데이터베이스에게 큰 부하를 주게 될 뿐만 아니라, 불필요한 데이터 전송을 위해 많은 네트워크 자원을 낭비하게 된다.

이러한 웨어하우징 기법의 단점을 보완하기 위한 대안으로 분산 접근법(distributed approach)이 제안 되었다. 여기서는 클라이언트가 보내는 하나의 질의를 여러 개의 하부 질의로 나눈 다음, 이를 센서에 보내서 처리 하는 방법이다. 시간이 지나면서 센서를 제조하는 기술이 향상 되었을 뿐만 아니라, 센서의 가격은 낮아지는 한편 각각의 센서가 가지는 컴퓨팅 능력은 수직 상승하고 있다. 센서가 자체적으로 원 정보(raw data)를 가공해서 데이터베이스에 전달할 수 있게 된 것이다. 질의를 처리하는데 필요한 정보를 센서가 미리 처리해서 데이터베이스에 전달한다면, 원 정보를 데이터베이스에 전달할 필요가 없으므로 데이터 전달에 소모되는 자원을 대폭 줄일 수 있다. 또한, 하부 질의를 센서가 받아서 처리하기 때문에 데이터베이스는 질의를 처리하는데 필요한 컴퓨팅 자원의 소모를 많이 줄일 수 있다. 작은 규모의 모바일 장치들은 각각의 컴퓨팅 파워, 메모리 그리고 통신 기능을 가지고 있으므로 이러한 접근법은 질의처리의 새로운 방법이다. 질의에 따라서 어떤 원격 사이트에 있는 데이터를 추출해야 할지가 결정 될 수 있으며, 또한 질의에 따라서 어떤 장치에서 질의의 일부분이 수행 될지도 결정 된다.

이러한 분산 접근법은 스냅 샷 질의와 장기 수행 질의를 효율적으로 처리 할 수 있다. 또한 분산 접근법과 더불어 현재 SQL에 시간에 대한 항목과 센서에 대한 애트리뷰트를 추가 하면 기존의 데이터베이스 기술들도 스트림 데이터를 효과적으로 처리 하는데 사용할 수 있음을 보였다. [3]

2.2 NiagaraCQ

미국 Wisconsin 대학의 Niagara System은 인터넷 상에 있는 XML 문서들을 XML-QL과 같은 질의어를 통해서 처리할 수 있도록 설계된 분산 데이터베이스 시스템이다. NiagaraCQ는 Niagara System의 하부 시스템으로 많은 영속 질의를 처리 할 수 있도록 하였으며 확장 가능한 시스템이다. Niagara System에서는 영속질의를 시간에 기반 한 영속 질의(Time-based continuous queries)와 변화에 기반 한 영속 질의(Change-based continuous queries)로 구분한다. 시간 기반 영속 질의는 사용자가 명시한 시간 간격마다 처리되며, 정해진 시간 간격마다 질의를 처리해주기만 하면 되므로, 데이터베이스 시스템이 이를 쉽고 효율적으로 처리 할 수 있다. 변화 기반 영속 질의는 조건에 맞는 데이터나 사건이 발생하면 그때 처리 된다. 시스템이 이러한 질의를 처리하기 위해서는 끊임없이 질의조건이 충족되는 지를 모니터링하고 있어야 하므로 시간에 기반 한 영속 질의보다 많은 자원이 소모된다.

이러한 영속 질의들은 많은 양의 정보가 빠르게 변화하는 인터넷상에서 관심 있는 정보의 변화나 사건의 발생을 모니터링 하는데 매우 유용하게 사용될 수 있다. 하지만, 영속 질의들은 오랜 기간 동안에 데이터베이스 시스템에 머물러야 하고, 한명의 사용자가 여러 개의 영속 질의를 하는 경우가 빈번함에 따라, 많은 수의 영속 질의를 한꺼번에 처리 할 수 있도록 시스템을 설계하는 것이 중요하다.

NiagaraCQ에서는 많은 수의 영속 질의들이 그 형태가 유사할 것을 가정하여 이러한 질의들을 그룹화(group

planning)해서 처리 하도록 설계 하였다. 질의의 종류에 상관없이 모두 그룹화가 가능하게 하여 효율적이고 확장성이 있는 데이터베이스 시스템을 구축 할 수 있도록 하였다. 또한, 새로운 영속 질의들은 추가와 삭제가 빈번하게 발생하는데, 이를 점증적인 방식으로 처리하여 전체 시스템 성능을 향상시켰다. 즉, 하나의 새로운 질의가 들어오면 전체 질의들을 다시 그룹화 하는 것이 아니라, 기존의 그룹에 새로운 질의를 추가하는 방식을 취했다. 이런 방식으로 추가를 하면, 매번 새로운 질의가 들어올 때마다 다시 그룹화해야 하는 부담이 없고, 그룹화 하는데 드는 비용이 적다는 장점이 있다. 기존의 그룹과 공통점이 없을 경우에는 새로운 그룹을 만들어서 해당 질의를 추가한다. 그룹화 된 질의는 두 부분으로 나뉘어져 처리 된다. 먼저 서로 공통된 연산을 한꺼번에 처리한 다음 그 결과가 각기 다른 연산을 하는 부분의 입력 값으로 들어가게 된다. 이렇게 하면 공통된 연산을 한꺼번에 처리 할 수 있기 때문에 매우 효율적이 된다.[4]

2.3 Aurora

미국의 Brown 대학교, Brandeis 대학교, 그리고 MIT 대학교에서 스트림 데이터의 처리를 위해 공동으로 수행한 Aurora 프로젝트는 스트림 기반의 애플리케이션들의 요구사항을 효과적으로 충족시키기 위한 기반 인프라를 구축하는 것을 목표로 하였다.

Aurora 프레임워크가 다루는 애플리케이션은 다음의 세 가지가 있다. Real-time monitoring application은 현재 상태를 모니터링한 데이터, 최근의 데이터를 중요시하고 데이터 저장이 중요하지 않거나 필요가 없는 애플리케이션 종류이며, archival application은 시간에 따라 저장되어 있는 대용량의 데이터를 처리하는 애플리케이션을 뜻하며, spanning application은 현재 데이터와 저장되어 있는 대용량의 데이터를 처리하는 애플리케이션을 가리킨다.

이러한 애플리케이션들을 위한 효과적인 데이터 관리와 처리 방법을 제공하기 위하여, 질의의 최적화에서 사용자 인터페이스에 이르는 데이터베이스 설계와 구현의 모든 측면들을 재구성하였다. 현재는 QoS 기반 연산자 스케쥴링, 부하 감쇄 방법, 풀 방식과 푸쉬 방식 데이터 처리를 함께 할 수 있는 복합 데이터 스토리지 최적화 등과 같은 실시간 데이터 처리에 관한 이슈에 초점을 맞추고 있다.

Aurora는 입력되는 데이터 스트림에 대하여 다수의 질의를 동시에 처리하여 애플리케이션으로 질의 처리 결과를 전송해 준다. 질의는 각 애플리케이션에 의해 정의 된다. 그리고 각 애플리케이션은 부분적인 질의 결과 또는 지연되는 질의 결과의 효용성을 설명하는 QoS 명세도 정의한다. Aurora Network에서 데이터 스트림은 데이터 소스에서부터 입력되는 데이터 튜플들의 순서를 말한다. 각 튜플들은 Aurora에 입력되면서 입력시간이 기록된다. 큐에는 연산자 박스에 의해 처리가 될 데이터 튜플들이 들어있게 된다. 처리된 데이터는 다음 연산자 박스의 입력 큐에 쌓인다. 연산자 박스는 입력되는 데이터에 대하여 데이터베이스의 Selection, Join 등과 같은 연산을 수행한다.

네트워크 연산자 박스들은 박스 프로세서 (Box Processor)의 연산자 수행 코드에 의해서 실행이 되고 네트워크의 큐들은 스토리지 관리자(Storage Manager)에 의해 관리된다. 라우터(Router)는 스트림 데이터 튜플들을 받아서 타임 스탬프를 붙이고 적합한 큐에 입력될 수 있도록 스토리지 관리자에게 전달한다. 스케쥴러는 카탈로그와 QoS 모니터를 참조하여 어느 한 시점에서 큐의 데이터 중에 얼마만큼을 어떤 박스에서 처리할 지를 결정하여 그 박스를 스케쥴링 해준다. 최종 결과 데이터는 QoS 모니터에도 전달되어 결과가 애플리케이션이 정의한 QoS를 얼마나 충족시키는지를 판단하고 그 정도에 따라 스케쥴링 정책이 변경될 수 있도록 한다. 시스템의 부하가 증가하는 경우에는 부하 감쇄기(Load Shedder)가 부하 감쇄 정책에 따라 부하를 줄이는 작업을 수행한다. [5,6]

2.4 기존 시스템의 비교 및 분석

다음의 표1에서는 이러한 세 가지 system들을 질의의 종류, 접근법, 그리고 특징들로 나누어 비교 분석하였다. 각 시스템에서는 각각의 질의가 가지는 특징에 국한하여 질의를 독립적으로 처리하기 때문에 모든 유형의 질의가 하나의 시스템에서 통합적으로 처리 되는 하이브리드 모델이 요구 되어지며, 이러한 하이브리드 모델에서는 유입되는 질의를 유형별로 그룹화 하기 위하여 질의 전처리기를 사용한다.

표1. Cougar Device DBMS, NiagaraCQ 및 Aurora 시스템의 분석 및 비교.
Table 1. Queries, Approaches and Characteristics of Cougar Device DBMS, NiagaraCQ and Aurora System표 4

	질의의 종류		approach	특징
Cougar Device DBMS	historical		warehousing	① 질의의 작업부하와 장치 액세스의 분리 ② 많은 양의 데이터가 장치로부터 데이터베이스 서버로 이동함에 따른 값진 자원의 낭비
	snap-shot		distributed	① 컴퓨팅 파워 메모리와 통신기능을 모두 가진 소규모 모바일 장치를 사용하여 local processing가능 ② 질의에 의해 remote site의 데이터를 추출함 ③ 질의에 의해 어떤 장치에서 어떤 sub-query가 실행될지 결정됨 ④ 질의 실행 시 장치를 직접 액세스하는 device database system
	long-running			
NiagaraCQ	time-based (continuous query)		distributed	① XML 데이터 세트에 영속질의 수행 ② 변화 기반 질의와 시간기반 질의에 대해 단일화 된 처리 ③ 확장, 축소에 적응된 처리 ④ 영속질의의 점진적인 그룹화 가능
	change-based (continuous query)			
Aurora	continuous & historical query	real-time monitoring	QoS (Query of Service) -driven control	① data-flow system ② 부하관리가 효율적임 ③ 질의 최적화 ④ Stream Storage 관리 ⑤ 실시간 요구와 QoS 지원을 위한 스케줄링
		archival monitoring		
		spanning monitoring		

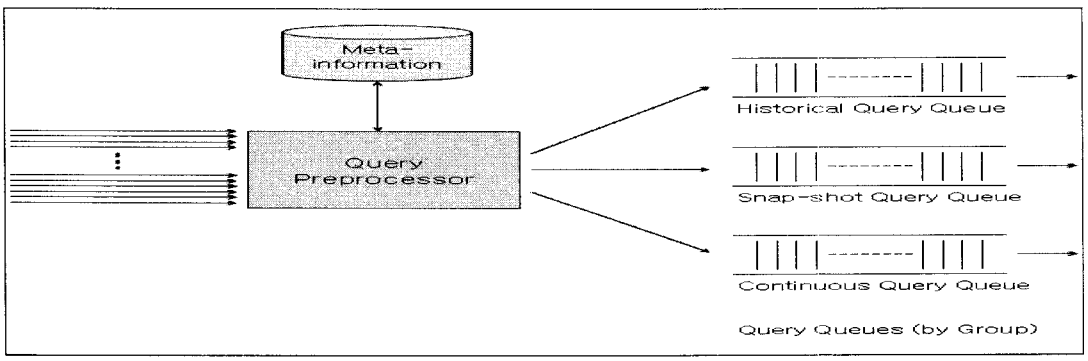

그림1. 하이브리드 모델의 질의 전처리기
Fig. 1. Query Preprocessor of the Hybrid Model

III. Stream DBMS의 효율적 질의 처리를 위한 Hybrid Model

스트림 DBMS에서 처리하게 되는 질의들은 다양하다. 이 다양함은 질의에서 주어지는 시간 조건, 상황 조건 등에서 비롯되는 바, 정적인 데이터에 관한 히스토리칼 질의, 장치 네트워크상의 주어진 시간에 행해지는 질의, 주기적 질의, 혹은 조건이 만족되면 바로 실행되어야 해서 지속적인 모니터링이 필요한 질의 등으로 구분된다.

기존의 시스템들에서는 시간과 변화에 기반 하여 질의를 다양하게 구분하여 정의하였고, 각 질의의 특성에 따라 각기 다른 접근법을 사용하고 있다. 각 시스템에서 제안된 접근법들은 유형별로 구분된 질의에 대해서는 최적의 처리를 지향하지만, 다양한 유형의 질의를 한 시스템에서 적절하게 처리하기 위해서는 각 접근법의 장점을 통합한 하이브리드 시스템 모델이 제안되어야만 한다. 본 논문에서는 각각의 질의를 전처리기(preprocessor)를 통과시키는 과정에서 질의의 지속시간과 환경변화의 기준에 따라 이들 질의를 클러스터링, 그룹화하며, 각 그룹별로 큐(queue)를 유지하면서 각 큐 별로 기존의 스트림 DBMS 시스템이 사용해온 접근법 중 적절한 접근법을 선택하여 사용하는 하이브리드 시스템 모델을 제안한다.

이 모델에서는 저장된 메타 정보(Meta Information)를 이용하여 질의 전처리기(Query Pre-processor)에서 질의를 유형별로 구별한다. 질의 전처리기를 통과하여 나온 질의들은 히스토리칼 질의, 스냅 샷 질의 그리고 영속 질의 등으로 그룹화 되며, 각 유형의 질의들은 유형별로 유지되고 있는 각각의 큐(queue)에 모여지게 된다.

각각의 큐로 질의들이 유형화되어 들어오면 이때부터는 각 큐별로 유형에 맞는 최적화된 질의 처리가 수행된다.

히스토리칼 질의는 그림 2의 구조에서 보듯이 Cougar Device DBMS처럼 저장된 데이터로부터 질의 결과를 얻기 위해서 query requester를 통해 질의 처리를 요구한다.

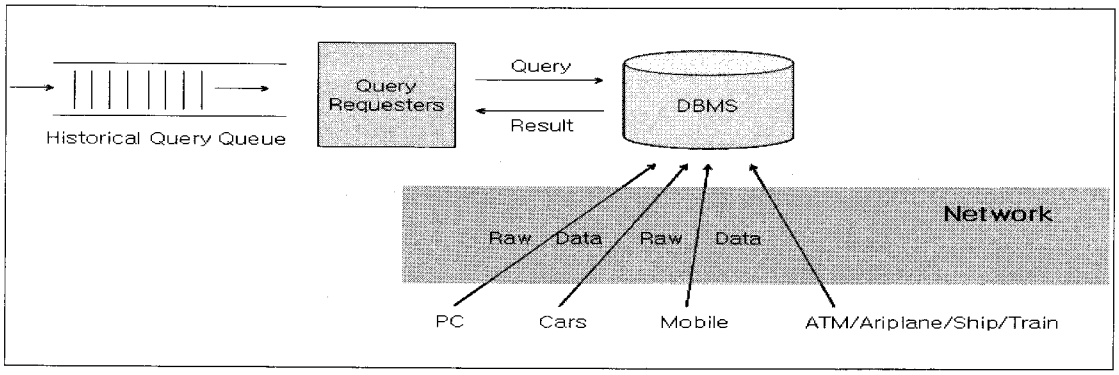

그림2. 히스토리칼 질의 처리 구조
Fig.2. Historical Query Processing Architecture

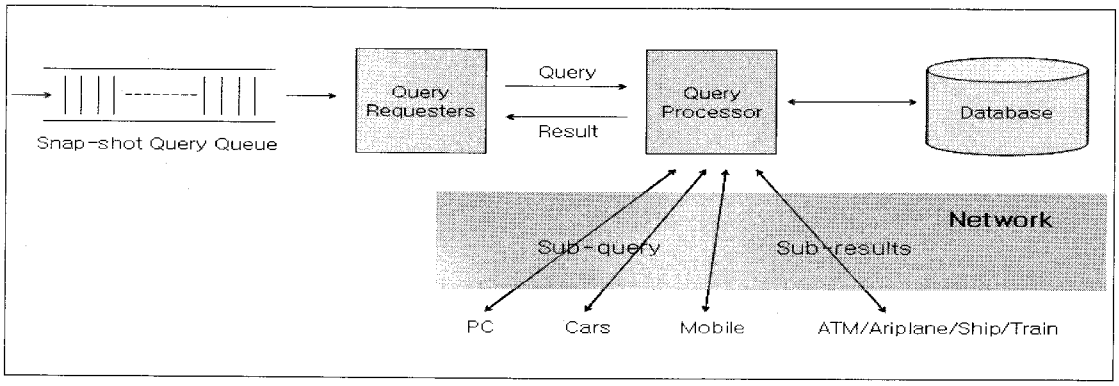

그림3. 스냅 샷 질의 처리 구조
Fig.3. Snap-shot Query Processing Architecture

질의 처리과정에서 DBMS는 네트워크를 통해 PC, 자동차, 모바일, ATM 그리고 비행기, 선박, 기차 등으로부터 원 데이터를 받아들여 사용하며, 질의 처리 결과는 다시 query requester에게 전하여 진다.

스냅 샷 질의의 처리는 그림3의 구조에서 보듯이 이 질의들이 query requester를 통해 질의 처리기에 전달되는데, 처리기는 저장 데이터베이스와 데이터를 주고 받음과 동시에, 네트워크를 통하여 독립적 처리 장치를 가진 PC, 자동차, 모바일, ATM, 비행기, 선박 그리고 기차 등에 각각의 sub-query를 실행하도록 전달하고, 장치들로부터 수행 후의 sub-result들을 전달 받는다. 네트워크를 통해 연결된 장치들은 각자 독립적인 처리능력을 가지며 향상된 성능을 가진 처리장치들이므로 하나의 질의를 여러 sub-query들로 나누어 병렬적으로 수행함으로써 전체적인 처리효율이 높아지게 되는 것이다.

앞의 그림1에서 질의 전처리기를 통과하여 영속질의로 분류되어 continuous query queue로 들어오게 되는 질의는 스트림 DBMS에서 가장 많이 처리하게 되는 유형의 질의이다. 오랜 시간 시스템에 머물면서 time-interval을 두고 주기적으로 처리되거나, 일정시간 동안의 통계를 필요로 하게 되는 질의도 있고, 항시 변화를 모니터링하고 있다가 실행 조건이 충족될 때마다 실행되어야 하는 영속 질의도 있다. 이러한 질의들은 시스템에 오랜 시간 머물게 될 뿐만 아니라 사용자 한 명이 여러 영속 질의의 처리를 요구하는 경우도 있으므로 실시간적으로 병행 처리 되어야 한다.

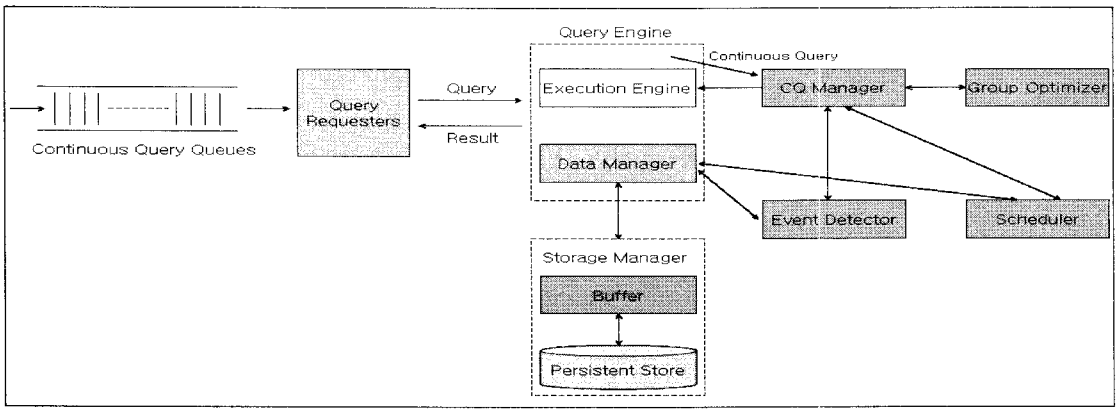

그림4. 영속 질의 처리 구조
Fig.4. Continuous Query Processing Architecture

그림4에서는 이러한 복합적인 요소를 가진 질의를 효율적으로 처리하기 위해서 Execution Engine과 Data Manager를 함께 모아 Query Engine을 구성 하며 이 안에서 질의처리 요구를 받아들이고 처리 결과를 돌려준다.

Execution Engine은 CQ(Continuous Query) Manager로부터 현재 수행되어야 할 질의들을 입력받아 Data Manager로부터 전달받은 데이터와 함께 처리하게 된다. CQ Manager는 장시간 시스템에 머물고 있는 질의들 중, Group Optimizer, Scheduler 그리고 Event Detector와의 상호작용을 통해 실행되어야 할 질의들을 결정해서 Query Engine으로 보내게 된다. Data Manager는 질의 수행에 필요한 데이터를 Storage Manager로부터 전달받고 필요에 따라 처리 결과 데이터를 데이터베이스에 저장하기로 한다. Data Manager 역시 Event Detector와 Scheduler와의 정보교환이 요구된다.

이상으로, 기존의 스트림 DBMS에서 와는 달리 모든 유형의 질의를 하나의 통합된 시스템에서 처리할 수 있는, 제안된 하이브리드 모델의 구조를 살펴보았다.

다음의 표2에서는 기존의 스트림 DBMS 시스템들과 제안된 하이브리드 모델을 비교분석 하였다. 표2에서와 같이 기존 시스템에 비해서 제안된 하이브리드 모델은 모든 유형의 질의를 하나의 시스템 안에서 처리할 수 있으며, 질의 큐와 데이터베이스의 공간을 추가함으로써 질의처리의 병렬성이 높아짐을 볼 수 있다. 최근 각종 장치의 가격은 하락하고, 성능은 수직상승함에 따라 저장장치의 추가는 용이하며 이로써 얻어지는 하이브리드 모델에 성능평가는 급상승하게 됨을 알 수 있다.

표2. 하이브리드 모델과 기존 스트림 DBMS의 비교
Table2. Hybrid model vs. other 3 stream DBMSs

	처리 가능한 질의 유형	질의 큐의 수	최대병렬도수	DB수
Hybrid Model	스트림DB의 모든 질의 유형 · historical query · snap-shot query · 모든 유형의 continuous query	4	3.5	2 (1 meta-DB 1 stream DB)
Cougar Device	historical, snap-shot, long-running	사용안함	1	1
NiagaraCQ	continuous query	사용안함	1	1
Aurora	continuous & historical query	4	1.5	1

5. 결론 및 향후 과제

지금까지 이 논문에서는 처리 결과를 얻기 위해 스트림 DBMS에 들어오는 다양한 유형의 질의들을 의무적으로 질의 전처리기를 통과하도록 한 다음 질의 전처리기 내에서 유형별로 질의들을 그룹화한 후 유형에 맞는 적절한 접근법으로 처리하도록 하는 통합적 모델을 제안하였다. 질의 전처리기에서 메타 정보를 이용하여 분류되어 나오는 히스토리칼 질의, 스냅 샷 질의 및 영속 질의는 모두 현실적으로 스트림 DBMS에서 처리해야 하는 유형의 질의 형태이다. 질의 전처리기를 시스템 내에 추가함으로써, 단일 시스템 내에서 모든 유형의 질의 처리가 일괄적이며 동시에 이루어지므로 스트림 DBMS에서의 질의처리 성능향상에 큰 효과를 볼 것이다.

향후 연구되어야 할 사항들은 첫째, 통합 모델에서 제안된 질의 전처리기가 메타 데이터베이스를 사용하는 구체적인 알고리즘이 명시되어야 할 것이며 둘째, 각각 3개의 큐와 연결된 서브시스템들의 병렬성에 관한 분석 연구가 이루어져야 할 것이다.

참고문헌

[1] 텔레매틱스 개론, 송 준화 외 공저. 홍릉과학출판사.

[2] 김창환, "텔레매틱스 기술 동향"

[3] Yong Yao, J. E. Gehrke. The Cougar Approach to Network Query Processing in Sensor Networks. Sigmod Record, Volume 31, Number 3 September 2002.

[4] J. Chen, D. DeWitt, F. Tian, and Y. Wang. NiagaraCQ: A scalable continuous query system for internet databases. In Proceedings. of the ACM SIGMOD Conference. on Management of Data, 2000.

[5] D. Abadi, D. Carney, U. Cetintemel, M. Cherniack, C. Convey, S. Lee, M. Stonebraker, N. Tatbul, S. Zdonik. Aurora: A New Model and Architecture for Data Stream Management. In VLDB Journal, August 2003.

[6] www.sc.brown.edu/research/aurora/

[7] D. Carney, U. Cetintemel, M. Cherniack, C. Convey, S. Lee, G. Seidman, M. Stonebraker, N. Tatbul, and S. Zdonik. Monitoring Streams: A New Class of Data Management Applications. In proceedings of the 28th International Conference on Very Large Data Bases (VLDB'02) Hong Kong, China, August 2002.

[8] B. Bobcook, S. Babu, M. Datar, R. Motwani, and J. Widom. Models and Issues in Data Stream Systems. In Proceedings of. ACM SIGMOD/SIGS CT Conference. on Principles. of Database System, pp.1-16, Madison, Wisconsin, USA, June 2002.

[9] Samuel R. Madden and Michael J. Franklin. Flooding the Stream: An Architecture for Queries over Streaming Sensor Data ICDE Conference, February, 2002, San Jose.

[10] www.db.stanford.edu/stream

저 자 소 개

양 영 휴

1981년 2월: 이화여자대학교 영어영문학과 문학사

1981년9월 ~ 1984년8월 : 미국 Pittsburgh 대학교 전자계산과

1986년 9월: 미국 Northeastern대학교, School of Computer Science, M.S.

1992년 2월:서강대학교 컴퓨터공학과 박사과정수료

1993년 3월 ~ 현재: 한양여자대학 인터넷정보과 교수

관심분야: 데이터베이스, AI, Ubiquitous Computing.