# DSMS 에서 환형 큐 버퍼 기반의 해시 색인을 이용한 조인 기법

김상기*, 이연*, 백성하*, 이동욱*, 김경배**, 배해영*
*인하대학교 컴퓨터 정보 공학과
**서원대학교 컴퓨터 교육과
e-mail : {kimsk, leeyeon, shbaek, dwlee}@dblab.inha.ac.kr, gbkim@seowon.ac.kr, hybae@inha.ac.kr

# Join processing using Hash Index based on Ring Queue Buffer in DSMS

Sang-Ki Kim*, Yan Li*, Sung-Ha Baek*, Dong-Wook Lee*, Gyoung-Bae Kim**, Hae-Young Bae*
*Dept. of Computer Science and Information Engineering, Inha University
**Dept. of Computer Education, Seowon University

**요 약**

유비쿼터스 환경에서 센서 데이터는 빠르고 연속적인 데이터 스트림 형태로 전송된다. 이러한 개별적인 데이터 스트림 정보를 연관되어 처리하기 위해 조인 연산이 필요하다. LWO, SWF 와 같은 기존 기법들은 Nested Loop Join 을 이용해 데이터 스트림 환경에서 조인 알고리즘을 제시하였다. 하지만, Nested Loop Join 을 사용하기 때문에 슬라이딩 윈도우의 크기에 따라 처리 속도가 영향을 많이 받고 XJoin 은 디스크 I/O 추가 비용이 필요하다. 제안 기법은 환형 큐 버퍼 기반의 해시 색인을 이용한 조인 기법은 환형 큐를 이용하여 데이터의 삽입, 삭제를 관리하고 해시 색인을 이용해 조인 연산을 처리 하여 제안 기법은 기존의 기법 보다 조인 연산을 보다 효율적으로 처리 할 수 있다.

## 1. 서론

유비쿼터스 환경의 사용자 기반 실시간 서비스를 위해 USN(Ubiquitous Sensor Network), RFID 등 센서 기반 환경에서 발생하는 데이터 스트림 처리에 대한 많은 연구가 진행되었다[1]. 예를 들어, 환경 모니터링, 재난 재해 방지, 주식 정보 등 다양한 정보가 센서를 통해 빠르고 연속적인 데이터 스트림 형태로 전송되기 때문에 기존 DBMS 에서 처리에 한계 있기 때문에 DSMS(Data Stream Management System) 개발이 이루어졌다. 이러한 DSMS 에 “인천과 수원의 최근 5 분 마다 온도와 습도”의 질의가 들어왔을 때 센싱 된 온도의 값과 가스 누출을 종합하기 위해 조인(Join) 연산이 요구된다[2].

현재까지 알려진 DSMS(Data Stream Management System)으로 STREAM, Aurora, Tribeca, TelegraphCQ, NiagaraCQ 등이 있다[3, 4, 5, 6, 7]. 기존의 데이터 스트림 연구에서 질의를 처리하기 위해 일회성 질의가 아니라 질의가 미리 등록되어 입력되는 데이터에 대해 연속적으로 질의를 수행하는 연속질의(Continuous Query) 형태를 갖는다. 이러한 질의 처리를 위해 데이터 범위를 한정 시키는 슬라이딩 윈도우를 사용한다. 슬라이딩 윈도우는 무한한 길이를 갖는 스트림 데이터를 일정시간이나 튜플 개수 단위로 스트림 데이터를 분할하여 질의처리에 이용하는 방식이다. 슬라이딩 윈도우를 이용하여 조인 연산을 처리하는 기법으로 LWO(Largest Window Only), SWF(Smallest Window First)는 조인 관계의 두 스트림 중 하나의 스트림에 신규 데이터가 도착했을 때, 그 데이터는 상대 스트림 윈도우 범위 안의 데이터와 Nested Loop Join 을 이용하여 조인 연산의 결과를 구한다[8]. SHJ 를 확장한 XJoin 은 메모리와 디스크를 사용하여 Join 연산을 한다. 그러나, LWO, SWO 는 Nested Loop Join 을 이용하기 때문에 윈도우 범위 안의 튜플 수에 많은 영향을 받는다. 또한 XJoin 은 디스크를 사용하기 때문에 디스크 I/O 시간이 필요하다[9].

본 논문에서 제안하는 환형 큐 버퍼 기반의 해시 색인 기법은 해시 색인을 이용하여 신규 데이터가 입력되었을 때 조인 연산이 빠르며 윈도우 범위 안에 튜플 수에 영향을 받지 않는 장점이 있다. 또한, 환형 큐 정책을 사용하여 데이터의 삽입 및 삭제에 대한 알고리즘과 슬라이딩 윈도우 관리가 용이하다.

본 논문의 구조는 다음과 같다. 제 2 장에서 DSMS 와 스트림 환경에서 슬라이딩 윈도우 조인에 대해서

본 연구는 건설교통부 첨단도시기술개발사업 - 지능형국토정보기술혁신 사업과제의 연구비지원(07 국토정보 C05)에 의해 수행되었습니다.

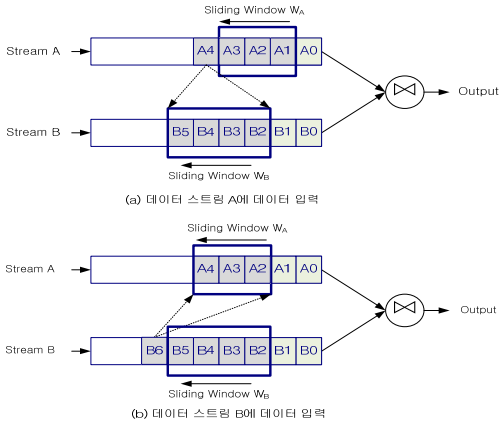
설명한다. 제 3 장에서는 해시 기반 체인 기법을 제안하고 제 4 장에서 성능 평과 결과를 제시하고 마지막으로 제 5 장에서 결론을 맺는다.

## 2. 관련연구

데이터 스트림은 센서로부터 연속적으로 발생하고 빠르게 유입되는 대용량 데이터다. 이러한 데이터는 무한한 길이를 갖기 때문에 일회성 질의가 아닌 연속질의를 사용한다. 연속질의는 질의를 시스템에 등록하여 질의를 처리한다. 또한, 데이터 스트림의 무한한 길이를 갖는 특성 때문에 슬라이딩 윈도우를 사용한다. 이러한 슬라이딩 윈도우를 사용하여 “인천과 수원의 최근 5 분 마다 온도와 습도” 질의를 처리하기 위한 조인 연산이 필요하다.

데이터 스트림의 조인 연산을 수행하기 위해 연속으로 빠르게 들어오는 모든 데이터를 조인을 하는 것은 시스템의 부하가 크다. 따라서, 연관 관계에 있는 다른 스트림을 조인 하기 위해 슬라이딩 윈도우를 사용한다.

슬라이딩 윈도우 조인의 과정은 조사(Probe) 단계, 삽입(Insert) 단계, 삭제(Invalidate) 단계로 이루어진다. 조사 단계는 새로운 데이터가 들어왔을 때 조인 관계에 있는 상대 스트림 윈도우 범위 안의 데이터를 스캔(Scan)하여 조인을 하는 단계이다. 삽입 단계는 새로 들어온 데이터를 윈도우 범위 안에 포함 시키는 단계이고 삭제 단계는 윈도우 범위를 벗어난 데이터를 삭제 하는 단계이다. [8, 10]




[그림 1] 슬라이딩 윈도우 조인

이러한 시간 기반 슬라이딩 윈도우를 이용한 대표적인 연구에 Nested Loop Join 을 이용한 SWO, LWO 는 Nested Loop Join 의 특성상 윈도우 범위 안의 튜플 수에 많은 영향을 받는다[8]. XJoin 은 Symmetric Hash Join 을 확장하여 연속질의를 효율적으로 처리하기 위한 공간조인 기반 연산질의 처리 알고리즘을 제안하고 메모리가 가득 차 더 이상 사용할 수 없을 경우 디스크로 보낸다[9].

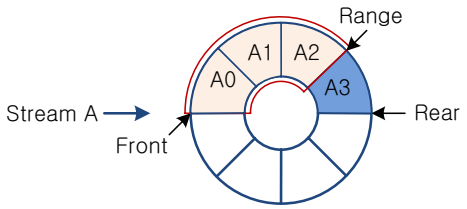
## 3. 환형 큐 버퍼 기반의 해시 색인을 이용한 조인 기법

본 장에서는 튜플 수에 따라 처리시간에 영향을 받는 단점을 보완하기 위해 환형 큐를 이용한 버퍼 관리 정책과 해시 인덱스 구축 기법을 설명한다. 환영 큐 버퍼 관리 정책은 세 개의 포인터를 이용해 신규 데이터의 삽입과 삭제, 슬라이딩 윈도우를 관리하기 쉬운 이점이 있다.

해시 인덱스 구축 기법은 해시 테이블에 튜플의 키 값을 저장하고 같은 키 값을 가진 튜플 들을 체인으로 엮어 조인 연산을 하기 때문에 조사 단계에서 빠른 연산을 할 수 있다.

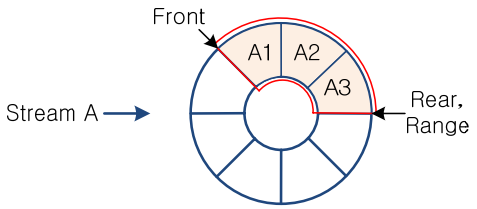
### 3.1 환형 큐를 이용한 버퍼 관리 정책

데이터 스트림 입력을 받는 버퍼는 환형 큐로 구현되어 관리 된다. 버퍼의 삽입 정책을 일반적인 환형 큐 알고리즘과 같다. [그림 2]와 같이 새로운 데이터 A3 가 버퍼에 삽입 될 경우 Rear 가 삽입 된 데이터의 끝으로 이동해 다음에 삽입될 데이터를 기다린다.




[그림 2] 환영 큐를 이용한 버퍼 삽입 정책

다음으로 버퍼의 삭제 정책은 [그림 3]과 같이 Range 가 Sliding Size 만큼 이동하게 되고 Front 는 Range 와 Range Size 를 유지하기 위해 이동하면서 Range Size 범위에 포함되지 않는 데이터를 삭제한다.

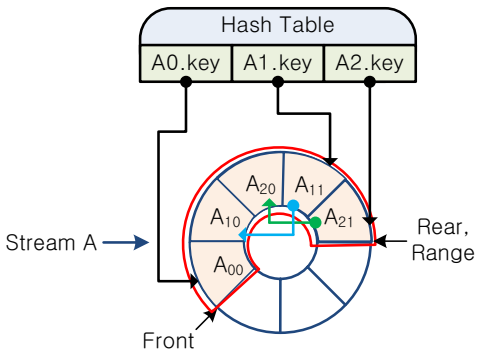



[그림 3] 환영 큐를 이용한 버퍼 삭제 정책

### 3.2 해시 색인 구축 기법

해시 색인을 구축을 위해 키 값을 갖는 해시 테이블을 갖고 있다. 스트림 A 에 데이터 $A_{00}$, $A_{10}$, $A_{20}$, $A_{11}$, $A_{21}$ 가 차례대로 삽입된다고 가정하자. 먼저 $A_{00}$ 가 스트림 A 에 삽입될 때 $A_{00}$.key 값을 해시 함수로 보낸다. 해시 함수는 $A_{00}$.key 계산한 해시 값과 해시 테이블을 비교하여 $A_{00}$.key 값이 없으면 해시 테이블에 $A_{00}$.key 값을 저장하고 $A_{00}$ 는 버퍼에 삽입된다. 해시

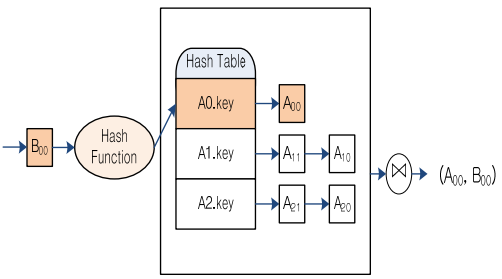
테이블에 $A_{00}$ 의 키 값을 갖고 있는 버켓은 버퍼에 삽입된 튜플 $A_{00}$ 의 위치를 가리킨다. 다음으로 $A_{10}$, $A_{20}$ 가 삽입된다면 역시 $A_{00}$ 와 마찬가지의 과정을 거치게 된다. $A_{10}$, $A_{20}$ 의 삽입이 끝나고 $A_{11}$ 이 스트림 A 에 삽입된다면 위와 마찬가지로 해시 함수를 통해 해시 값을 계산하고 키 값이 있다면 현재 해시 테이블 키 값과 연결된 튜플을 새로 들어온 튜플과 체인을 구성하고 해시 테이블의 키 값은 새로 들어온 데이터의 위치를 갖는다. 이와 마찬가지로 $A_{21}$ 가 들어왔을 경우 위와 같은 과정을 거쳐 해시 인덱스를 구축한다. 아래 [그림 4]는 스트림 A 에 $A_{00}$, $A_{10}$, $A_{20}$, $A_{11}$, $A_{21}$ 순서로 전송 되어 인덱스가 구축되어 처리 된 그림이다.




[그림 4] 환영 큐 버퍼 정책 기반 해시 색인 구축

### 3.3 조인 처리 과정

조인 관계에 있는 스트림을 각각 A, B 라 가정한다. 스트림 A 는 [그림 4]와 같이 구성되어 있고 스트림 B 에 새로운 데이터가 전송되어 삽입된다. 이 때의 신규 데이터를 $B_{00}$ 라고 가정한다. $B_{00}$ 와 스트림 A 의 슬라이딩 윈도우 범위 안에 있는 데이터와 조인 연산을 하기 위해 $B_{00}$.key 값을 해시 함수에 넣어 해시 값을 얻는다. $B_{00}$.key 의 해시 값을 이용해 스트림 A 의 해시 테이블에 접근하고 키 값이 존재하면 해시 테이블이 가리키는 포인터를 이용해 스트림 A 의 튜플에 접근하고 튜플과 체인으로 연결된 모든 튜플들과 조인 연산을 하여 $(A_{00}, B_{00})$ 결과를 나타낸다.

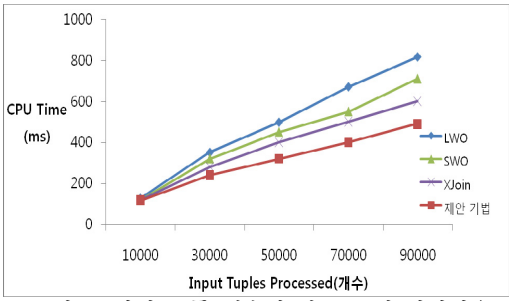



[그림 5] 환영 큐 버퍼 정책 기반 해시 색인 구축 기법을 이용한 조인 처리 과정

## 4. 성능평가

성능평가는 기존의 데이터 스트림 조인 기법인 SWO, LWO 와 제안기법의 처리 시간을 비교하였다. 모든 실험은 4GB 메모리의 인텔 펜티엄 4 2.6GHz 프로세서와 윈도우 XP 프로페셔널 운영체제에서 수행되었으며 알고리즘 구현은 Microsoft Visual Studio 2005 에서 C 언어로 구현하였다.

실험 방법은 시간 기반 윈도우를 사용하였고 윈도우 크기 3000ms, 슬라이딩 크기 1000ms 해시 테이블 크기는 100 개 이다. 삽입 튜플의 개수를 10000 개에서 2000 개 씩 증가하면서 평가하였다.




[그림 6] 삽입 튜플 개수에 따른 조인 처리비용

[그림 6]와 같이 환영 큐 버퍼 기반의 해시 색인을 사용할 경우 신규 데이터가 상대 스트림과 빠른 조인을 위해 슬라이딩 윈도우 범위 안의 데이터를 해시 인덱스에 구축된 키 값으로 같은 데이터가 체인으로 연결되어 있기 때문에 제안기법이 상대적으로 빠른 조인 처리 성능을 보인다. 하지만, LWO, SWO 는 Nested Loop Join 을 기반으로 처리하기 때문에 튜플 수가 증가할수록 처리시간이 증가 하였으며, XJoin 은 Symmetric Hash Join 을 사용하여 LWO 나 SWO 보다는 작은 처리 시간을 보였다.

## 5. 결론 및 향후 연구

본 논문은 데이터 스트림에서 환경 조인 연산을 처리하기 위해 환형 큐 버퍼 기반의 해시 색인을 구축하였다. DSMS 환경에서 조인 연산을 위해 제안기법을 사용할 경우 빠른 조인 처리가 가능하다.

향후 연구로는 해시 색인 구성 시 체인을 구성하는 비용이 크기 때문에 이를 줄이기 위한 연구가 필요하다.

## 참고문헌

[1] B. Babcock, S. Babu, M. Datar, R. Motwani, and J. Widom, "Models and Issues in Data Stream Systems," In Proc. of the ACM Symposium on Principles of Database Systems, 2002, pp. 1-16

[2] J. Kang, J. F. Naughton, and S.D. Viglas, "Evaluating window joins over unbounded streams," In ICDE, Feb 2003.

[3] A. Arasu, B. Babcock, S. Babu, M. Datar, K. Ito, I. Nishizawa, J. Rosen stein, and J. Widom, "STREAM: The Stanford Stream Data Manager," IEEE Data Engineering Bulletin. Vol.26 No.1, 2003, pp. 19-26

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

[5] M. Sullivan, and A. Heybey, "Tribeca: A System for Managing Large Databases of Network Traffic," In Proc. of USENIX Annual Technical, 1998

[6] S. Chandrasekaran, O. Cooper, A. Deshpande, M. J. Franklin, J. M. Hellerstein, W. Hong, S. krishnamurthy, S. R. Madder, V. Raman, F. Reiss, and M. A. Shah, "TelegraphCQ: Continuous Dataflow Procesing for an Uncertain World," Proceedings of the CIDR conference, 2003.

[7] Chen J. et al., "NiagaraCQ: A Scalable Continuous Query System for Internet Databases," In Proc. Int'l Conf. on Management of Data, ACM SIGMOD, Dallas, Texas, pp. 379-390, June 2000.

[8] M. A. Hammad, M. J. Franklin, W. G. Aref, and A. K. Elmagarmid, "Scheduling for shared window joins over data streams,", In VLDB, pages 297-308, Sep 2003.

[9] T. Urhan, and M. Franklin, "XJoin: A reactively scheduled pipelined join operator," IEEE Data Engineering Bulletin, 2000

[10] Song Wang, Samrat Geanguly, "State-Slice: New Paradigm of Multi-query Optimization of Window-based Stream Queries", In VLDB, Sep 2006