데이타 스트림에서의 다중 조인 질의 최적화 방법

(Optimizing Multi-way Join Query Over Data Streams)

박 홍 규 † 이 원 석 ††
(Hong Kyu Park) (Won Suk Lee)

요 약 데이타 스트림이란 실시간에 연속적으로 빠르게 생성되는 데이타 집합을 의미한다. 이러한 데이타 스트림들은 최근 사회가 발달과 더불어 정보 환경도 급속도로 발전함에 따라 센서 데이타, 교통상황 수집 자료, 웹 클릭 모니터링 등과 같은 많은 응용 분야에서 적용되고 있다. 이러한 형태의 데이트 스트림을 처리하기 위해서는 미리 등록된 질의에 대하여 새롭게 들어오는 스트림 데이타의 결과를 계속적으로 생성하게 된다. 이와 같은 이유로 끊임없이 들어오는 스트림 데이타들을 빠르게 처리하는 것이 이 분야에서 주된 이슈가 되었으며, 이를 위한 방법으로 등록된 질의들을 효율적으로 처리하기 위한 질의 최적화 분야에 많은 연구가 있었다. 그러므로 본 논문에서는 기존 연구에서 사용되었던 그리디 방법을 기반으로 비용 모델을 이용하여 최소의 비용을 갖는 질의 계획을 선택하는 확장된 그리디 방법(EGA)을 제시한다. 확장된 그리디 방법은 알고리즘의 정확성이 떨어지는 그리디 알고리즘의 단점을 극복하기 위하여 비용이 가장 작은 연산 하나를 선택하는 대신 비용이 작은 연산들의 집합을 선택한다. 이 연산들의 집합의 크기는 알고리즘의 정확성과 수행 시간에 영향을 끼치며, 두 개의 변수에 의해서 적응적으로 조절될 수 있다. 실험에서는 다양한 스트림 환경에서 대부분 그리디 알고리즘보다 향상된 성능을 보장하고, 두 변수에 의한 알고리즘의 성능 및 수행 시간 차이를 보여줌으로써 본 알고리즘의 효율성을 검증하였다.

키워드 : 데이타 스트림, 질의 최적화, 다중 조인

Abstract A data stream which is a massive unbounded sequence of data elements continuously generated at a rapid rate. Many recent research activities for emerging applications often need to deal with the data stream. Such applications can be web click monitoring, sensor data processing, network traffic analysis, telephone records and multi-media data. For this, data processing over a data stream are not performed on the stored data but performed the newly updated data with pre-registered queries, and then return a result immediately or periodically. Recently, many studies are focused on dealing with a data stream more than a stored dataset. Especially, there are many researches to optimize continuous queries in order to perform them efficiently. This paper proposes a query optimization algorithm to manage continuous query which has multiple join operators(Multi-way join) over data streams. It is called by an Extended Greedy query optimization based on a greedy algorithm. It defines a join cost by a required operation to compute a join and an operation to process a result and then stores all information for computing join cost and join cost in the statistics catalog. To overcome a weak point of greedy algorithm which has poor performance, the algorithm selects the set of operators with a small lay, instead of operator with the smallest cost. The set is influenced the accuracy and execution time of the algorithm and can be controlled adaptively by two user-defined values. Experiment results illustrate the performance of the EGA algorithm in various stream environments.

Key words : Data Stream, Query Optimization, Multi-way join

· 이 논문은 2008년도 정부(교육과학기술부)의 재원으로 한국과학재단의 국가지정연구실사업으로 수행된 연구임(No.R0A-2006-000-10225-0)

† 학생회원 : 연세대학교 컴퓨터과학과
gladiator11@database.yonsei.ac.kr
†† 종신회원 : 연세대학교 컴퓨터과학과 교수
leewo@database.yonsei.ac.kr

논문접수 : 2008년 2월 25일
심사완료 : 2008년 9월 22일

정보과학회논문지: 데이타베이스 제35권 제6호(2008.12)

1. 서 론

최근 산업 발달과 고도 사회로 접어들면서 과학 기술이나 공학 분야 이외에 경제 사회 등의 다양한 분야에서 각종 데이타들을 처리하여 정보를 얻는 것이 중요해지고 있다. 이러한 데이타 처리의 형태가 일정한 저장공간에 데이타를 저장한 후 관리·활용[1,2]하던 예전과는 달리 최근에는 연속적인 특성을 가지는 대용량의 데이타를 관리, 활용하는 곳에 관심이 집중되고 있다. 이러한 종류의 데이타는 USN(Ubiquitous Sensor Network), 주식거래, 금융거래, 전화통화, 교통관리 및 교통상황수집자료, 위성통신 등 여러 응용 분야에서 사용, 적용되고 있으며, 실시간에 연속적으로 발생되어 빠른 속도로 관리 시스템에 입력되어 들어오는 이러한 데이타들을 데이타 스트림(Data Stream)[3-5]라 한다. 기존의 연구에서 오랜 동안 데이타 스트림의 처리 기술을 연구해 STREAM[6-10], Telegraph(Tiny DB)[11,12] 그리고 Aurora[13] 등과 같은 DSMS(Data Stream Management System)이 개발되었다.

DSMS(data stream management system)은 데이타를 모두 저장해놓은 상태에서 한 번 질의를 수행하여 결과를 돌려주는 기존의 DBMS와는 달리 질의가 미리 등록을 되면 새로운 스트림이 입력될 때마다 즉시 혹은 주기적으로 등록된 질의를 수행하는 연속 질의(continuous query) 형태[3]를 띠게 된다. 이와 같이 연속 질의는 한 번 등록되면 반복되어 수행되기 때문에 최적의 실행 계획으로 실행하는 것은 실시간에 스트림 데이타를 처리하는 데 매우 중요한 요소가 된다. 특히 빠른 속도로 입력되는 여러 개의 스트림을 조인하는 다중 조인 질의의 경우, 단순한 질의보다 훨씬 많은 비용이 들 뿐만 아니라 조인의 순서에 따라서 매우 큰 성능 차이가 나기 때문에 최적화의 중요성이 더욱 크다. 일반적으로 다중 조인 질의의 경우 질의 숫자가 증가함에 따라 최적의 조인 조건을 찾기 위한 탐색영역이 기하급수적으로 증가하는 어려움이 있기 때문에 기존의 연구에서는 대부분 현재 들어오고 있는 스트림의 특징을 모니터링하여 저장한 후 그리디 알고리즘을 사용하여 이러한 문제를 해결하고 있다. 비록 그리디 방법은 최적의 방법을 보장하지 못하지만, 다른 최적화 방법에 비해 빠른 시간 안에 질의 계획을 생성할 수 있기 때문에 스트림 환경에서의 질의 최적화 방법에 많이 사용된다[4,7,9,14].

하지만 그리디 알고리즘은 다중 조인 질의를 최적화할 때 치명적인 문제점을 내포하고 있다. 처음 하나의 조인이 선택되고 나면 다음 조인을 선택할 때 많은 조인들이 선택될 수 있는 후보에서 제외되기 때문에 많은 조인 순서들을 고려할 수 없게 된다. 이것은 조인의 특성상 처음에 비용이 가장 작은 조인을 선택하는 것이 항상 최적의 조인 순서를 보장할 수 없을 뿐만 아니라 최적화의 오차가 매우 클 수 있다.

그러므로 본 논문에서는 이러한 그리디 알고리즘보다 정확한 최적화를 수행함과 동시에 시스템의 로드에 따라서 최적화에 소요되는 부담(overhead)을 적응적으로 조절할 수 있는 확장된 그리디 알고리즘(EGA)을 제안한다. 확장된 그리디 알고리즘은 현재 선택이 가능한 조인들 중 비용이 가장 작은 조인을 하나 선택하는 그리디 알고리즘과는 달리 현재 선택이 가능한 조인 뿐만 아니라 다음에 실행되는 조인의 비용까지 고려하여 비용이 작은 1개 이상의 조인들을 선택함으로써 질의 최적화의 정확성을 향상시킨다. 하지만 이와 같은 방법은 그만큼 많은 조인의 조합을 탐색한다는 의미이기 때문에 무작정 많은 조인을 선택하게 되면 오히려 부담이 너무 커지게 되기 때문에 확장된 그리디 알고리즘은 조인 비용을 기반으로 2개의 사용자 변수를 이용하여 시스템 로드에 합당하게 부담, 즉 선택되는 조인의 갯수를 컨트롤한다. 이 논문의 공헌은 다음과 같이 요약할 수 있다.

- 조인 선택도나 입력 속도가 작은 조인부터 선택하는 기존의 방법과는 달리 조인 비용 모델을 정의하고 그 모델을 기반으로 비용 기반 최적화 방법을 사용함으로써 더 정확한 질의 최적화를 가능하게 하였다.
- 기존에 질의 최적화에 많이 사용되었던 그리디 방법보다 성능이 개선된 알고리즘을 제안하고, 이 알고리즘을 위하여 스트림의 특성들을 새로운 형태로 저장한다.
- 두 개의 사용자 변수를 통하여 제안된 알고리즘의 성능과 최적화 수행 시간을 적응적으로 조절할 수 있다.

본 논문의 구성은 다음과 같다. 먼저 2절에서는 관련 연구에 대해서 살펴보고 3절에서는 본 논문에서 처리하는 질의의 형태와 비용 기반 최적화를 하기 위해 필요한 조인 비용 모델 그리고 통계 카탈로그에 대해서 설명한다. 4절에서는 확장된 그리디 질의 최적화 알고리즘을 제안하고 5장에서는 제안된 알고리즘과 그리디 알고리즘을 여러 가지 스트림 환경을 고려하여 비교 실험한다. 마지막으로 6절에서는 결론 및 앞으로의 연구 방향에 대해 논의한다.

2. 관련 연구

데이타 스트림 환경에서 효율적인 스트림 처리를 위해서 조인의 비용을 예측하기 위한 조인 비용 모델의 연구가 진행되고 있다. [15]에서는 단일 이진 조인에 대해서 스트림 데이타를 처리하는 방식(Hash join, Nested join)에 따라서 각각의 조인 비용 모델을 정의하고 비용

모델과 실제 스트림 처리 실행 시간이 비례함을 실험으로서 증명하였다. [6]에서는 처리 비용과 메모리 사용의 관점에서 효율적인 자원 할당을 예측하기 위해서 비용 모델을 정의하였다.

[17,18]에서는 단일 연속 질의의 조건이 2개 이상의 스트림을 조인하는 질의의 경우 출력 속도를 최대화하기 위한 조인 방법을 제시한다. 대부분의 주어진 메모리 내에서 모든 데이타 스트림들을 저장할 수가 없어 오버플로우가 났을 경우, 부하 분산을 통하여 모든 데이타 스트림에 대한 질의 결과를 주는 것은 아니지만 메모리 내에서 등록된 연속 질의를 처리하거나 질의 계획의 최적화를 다시 함으로써 해결을 하지만, 이 논문에서는 파일을 이용하여 정확한 정보를 도출하며, 질의 계획은 그대로 실행함으로써, 질의 재최적화 및 그에 따른 부담이 없는 장점을 가지고 있다.

[4]는 조인의 중간 결과를 저장하지 않고 스트림에서 데이타가 들어오면 스트림의 입력 속도가 낮은 스트림일수록 중첩 반복의 바깥쪽(outer loop)으로 하는 방식으로 스트림을 처리하는 방법을 제시한다. 이러한 방법은 현재 들어온 스트림은 입력 속도가 1이기 때문에 가장 바깥쪽(outer)에 위치하게 되며, 그 다음부터는 입력 속도 순으로 정렬하여 각각 반복을 통하여 조인을 처리하게 된다. 이 방법의 단점은 입력 속도만으로 조인의 순서를 결정함으로써 조인의 성능에 영향을 미칠 수 있는 조인 선택도에 대한 고려가 하나도 없다는 것이다. 또한 이 논문은 여러 스트림 상황에서의 경험적인 질의 최적화에 대해서도 몇 가지 제안한다.

STREAM[6,7]은 데이타 스트림이 들어오면 주어진 필터(선택 조건)의 순서대로 수행을 하고 결과를 출력하고, 일정 확률로 들어온 데이타를 프로파일링하여 각 필터의 선택도를 각각 계산하여 매트릭스 뷰(Matrix View)에 저장을 한 후, greedy 알고리즘을 이용하여 선택도가 낮은 필터를 먼저 수행하도록 동적으로 필터의 순서를 바꾸는 A-Greedy 알고리즘을 제안한다. 이러한 방법은 시시각각 변하는 스트림 환경에서 각 상황에 따라 최적화된 필터의 순서를 찾아내고 그 순서대로 필터를 처리함으로써 성능의 향상을 기대할 수 있지만, 매트릭스 뷰를 저장, 관리해야 하는 부담도 가지고 있다. 그리고 이 논문에서는 A-greedy 알고리즘을 조인의 순서 결정으로 확장하여 알고리즘을 적용시킨다. 스트림 데이타가 들어왔을 때 각각의 스트림들과 조인 결과가 나오는 지를 확인(drop probing)하는 단계와 조인을 수행하여 결과를 출력하는 단계(output generation), 두 단계에 거쳐서 조인을 수행하게 되며, drop probing 단계에서 조인 결과의 유무를 모니터링하여 이 선택도를 계산하여 A-greedy와 마찬가지로 매트릭스 뷰를 이용하여 저장한 후 가장 작은 선택도를 가진 조인을 먼저 수행하도록 조인 순서를 동적으로 바꿀 수 있도록 한다. 이 논문에서는 각 조인의 중간결과를 저장하지 않는 PPT 조인 방법을 쓰기 때문에 동적으로 조인의 순서를 변동시켜도 부담이 발생하지 않는다.

TelegraphCQ[11]에서는 Eddy[12]을 이용하여 튜플을 라우팅하고 스케줄링하게 되는데, 우선순위가 높은 튜플부터 처리하게 된다. 데이타 스트림의 기아 현상(starvation)을 막기 위해서 최근에 들어온 튜플일수록 더 낮은 우선순위를 가지고 있고, 시간이 지남에 따라서 우선순위가 높아지는 방법을 사용한다.

Eddy에서는 연산자의 처리 순서를 결정하기 위해서 "lottery scheduling"이라는 방법을 사용한다[11]. Eddy에 연결되어 있는 각각의 연산자들은 각각 "ticket account"라는 것을 가지고 있고, 이는 각 연산자의 선택도를 나타내는 지표가 되며, 어떤 연산자에 티켓의 수가 많으면 선택도가 낮음을 의미한다. 튜플이 Eddy에 의해서 수행해야 될 다음 연산자를 선택할 때 연산자가 다음에 수행될 연산자로 뽑힐 확률을 그 각각의 연산자가 가지고 있는 티켓의 수와 비례하도록 하여, 선택도가 낮은 연산자를 우선으로 선택하는 방법이다(이는 lottery 개념을 이용하여, 랜덤하게 선택된다.). 연산자가 lottery에 참여하기 위해서는 여분의 input queue가 있어야 하기 때문에 빠른 연산자가 그 우선권을 갖게 된다. 반면 느린 연산자의 경우 대부분의 시간 동안 queue가 차 있기 때문에 lottery에 참여하는 데 부적합할 것이다. 반면 Eddy에 연산의 결과를 돌려줄 때마다 ticket account가 줄어듦으로써, selectivity가 높은 연산자의 경우, lottery에서 이길 확률이 적게 된다.

Aurora[13]에서는 처음에는 최적화되지 않은 임의의 네트워크를 구성하고 있는 상태이며, 시스템이 실행되면서 box(연산자)가 처리될 때의 평균 비용이나 각 box의 선택율(selectivity) 등의 통계 값을 수집하기 시작하며, 이러한 정보들을 바탕으로 실시간에 최적화를 수행하게 된다. 대량의 질의들에 대해 모든 box들을 병합하여 처리하는 것은 NP-complete 문제가 되어 적합한 해결책이 되지 않는다. 그 대신 최적화기(optimizer)는 전체 네트워크를 여러 개의 서브 네트워크(sub-network)로 나누어 각각 최적화를 수행하며, 하나의 서브 네트워크가 최적화 될 때 다른 서브 네트워크는 정상적으로 질의를 처리하고 있을 수 있다. 최적화 방법은 인접한 box들을 병합함으로써 성능을 향상시킬 수도 있고, 각 box의 튜플 처리 비용을 계산하여 더 적은 비용을 갖는 box를 먼저 수행하는 방법을 사용한다.

NiagaraCQ[14]는 XML데이타에 대한 질의 결과를 처리하고 관심에 대상이 되는 웹 문서의 변화를 감시하

기 위한 시스템으로 연속질의 처리를 위해 개발되었다. 분산된 많은 XML 파일에 대한 대규모의 연속질의 처리를 증가 그룹 처리 방법을 도입하였으며 XML 문서가 변경되었을 경우 변경된 부분만을 고려하는 시스템이며, 이질적인 데이타에 대한 Push과 Poll Model을 모두 적용하여 원천 데이타의 변화를 탐지할 수 있다.

3. 선행 지식

3.1 조인 질의 및 비용 모델

데이타 스트림 처리하는 연산은 여러 가지가 있지만 그 가운데 가장 큰 부하를 가지고 있는 연산은 조인 연산이다. 특히 다중 조인은 연산 자체가 가지는 부하가 클 뿐만 아니라 조인의 순서에 따라서 수행 시간이 많은 차이가 난다. 그러므로 여기에서는 n개의 스트림을 조인하는 다중 조인 질의를 최적화하는 데 그 목적을 두고 있다. 셀렉션 연산이 없다는 제한은 [14]와 같이 PullUp과 PushUp 방법 중 하나를 선택하여 사용될 수도 있고, [11]과 같이 비트맵을 이용하여 조인과 셀렉션을 동시에 처리할 수 있기 때문에 큰 문제가 되지 않으며, 어그리게이션 조건 또한 의미상 셀렉션과 조인 연산을 모두 수행한 후에 처리하는 경우가 많기 때문에 큰 문제가 되지 않는다. 그러므로 본 연구에서 사용하는 질의는 조인 조건만을 가진 질의로 한정 지으며, 다음과 같이 정의될 수 있다.

SELECT *
FROM S_1[W_1 Min], S_2[W_2 Min], S_3[W_3 Min], S_4[W_4 Min] …
WHERE $S_1.attr_1=S_2.$ $attr_1$ and $S_2.$ $Attr_2=S_3.$ $Attr_2$ and $S_3.$ $Attr_3=S_4.Attr_3$ …

비용 기반 질의 최적화를 하기 위해서는 각 조인들의 비용을 정의하는 조인 비용 모델이 필요하다. 여기에서는 Kang[15]이 제안한 조인 비용 모델을 기반으로 한다. 그러나 Kang[15]은 두 개의 스트림을 조인하는 연산에 대해서 비용을 평가하는데, 이는 다중 조인에는 적합하지 않다. 왜냐하면 다중 조인의 경우 조인 결과의 크기가 다음 조인의 입력 속도가 되기 때문에 조인 비용을 측정하는 데 조인 결과의 크기가 중요한 요소가 된다. 그러므로 여기에서 는 Kang의 비용 모델에 조인 결과의 크기를 추가시킨 비용 모델을 적용하였다.

Kang의 조인 비용모델은 2개의 스트림을 조인하는 조인에 대해서 단위 시간당 처리해야 할 일로서 들어온 데이타 스트림을 삽입하는 연산, 상대 스트림을 조사하여 조인 결과를 생성하는 연산, 그리고 윈도우 범위를 벗어난 스트림 데이타를 삭제하는 연산의 합으로 정의하고, 다양한 조인 방식에 대하여 각 비용 모델은 정의하였는데 본 논문에서는 HHJ(2개의 스트림 모두 해시 인덱스를 가진 조인)의 비용 모델을 적용하였다.

하지만 Kang의 연구에 따르면 삽입 연산과 삭제 연산은 실제 조인을 수행하는 비용에 비해 상대적으로 매우 작기 때문에 이 두 연산들을 제외하고 실제 조인을 수행하는 비용만 고려하였으며 이는 다음과 같다.

$$C_i(R,S)=\lambda_R\times\frac{W\times\lambda_S}{|S|}+\lambda_S\times\frac{W+\lambda_R}{|R|} \quad (1)$$

다음 표 1은 비용 모델을 정의하기 위한 용어 정리를 위한 표이다.

표 1 조인 비용 정의를 위한 용어

λ_R	스트림 R의 입력속도
W	윈도우의 크기
|R|	윈도우 R의 해시 버켓의 수
ΔR	스트림 R에서 새로 들어온 데이타
$\sigma_{R,S}$	스트림 R과 S의 조인 선택도
$C_{j(R,S)}$	새로운 스트림이 들어왔을 때 필요한 연산 비용
$C_{o(R,S)}$	조인 결과의 크기
$C_{(R,S)}$	스트림 R과 S의 조인 비용

조인 결과의 크기는 입력 스트림의 입력 속도와 조인 선택도를 이용하여 정의할 수 있으며, 이는 다음과 같다.

$$C_o(R,S)=\lambda_R\lambda_S(2W-1)\sigma_{R,S} \quad (2)$$

$$C(R,S)=a_1C_i(R,S)+a_2C_o(R,S) \quad (3)$$

따라서, 총 조인 비용은 식 (1)와 식 (2)의 합으로 구할 수 있으며 식 (3)과 같다. a_1과 a_2는 각 비용에 가중치를 주기 위한 변수이며, 디폴트 값은 1이다.

3.2 통계 카탈로그

통계 카탈로그는 현재 정의되어 있는 질의 실행 계획에 따라 조인 연산을 수행하는 과정에서 발생되는 비용들 뿐만 아니라 실시간으로 입력되는 데이타 스트림들에 대한 정보를 이용하여 수행되지 않는 조인 연산에 대해서는 3.2에서 정의된 비용 모델에 따라 비용을 예측하여 저장한다. 즉, 통계 카탈로그는 수행될 수 있는 모든 조인 연산에 대하여 비용을 저장하고 있으며, 과거에 수행된 적이 있는 조인 연산에 대해서는 실제로 수행되었을 때의 비용을 저장하고 있음으로써 통계에 대한 정확성을 높일 수 있다. 비록 스트림 환경에서의 데이타들의 특성(입력 속도, 조인 속성의 도메인 크기, 등)들은 계속해서 바뀌지만, 완전히 랜덤으로 발생되는 것이 아니라 응용 분야의 특성과 각 스트림들의 특성에 따라 특정한 패턴 및 범위를 가질 것이다. 그러므로 위와 같이 과거에 수행된 실제 비용들을 저장함으로써 별도의 런타임-부담 없이 보다 정확한 통계 정보를 가질 수 있다. 각각의 조인 연산의 비용은 현재 측정된 값들로 대체되는 것이 아니라, 과거에 예측된 비용이든 실제 비용

이든, 이 전에 저장되어 있던 비용들과 현재 측정된 비용을 합쳐서 *가우시안 분포*를 띠고 있다고 가정하고 평균과 표준 편차의 형태로 저장된다. 각 조인 연산들의 평균 비용은 현재 시점에서만의 정보 뿐만 아니라 과거의 정보를 모두 종합하여 판단할 수 있게 해 주며, 표준 편차는 그 조인의 비용이 얼마만큼 변하는지에 대한 판단을 할 수 있게 한다. 이와 같은 방법은 본 논문에서 제안되는 비용 기반 질의 최적화 방법에 의미 있게 사용된다.

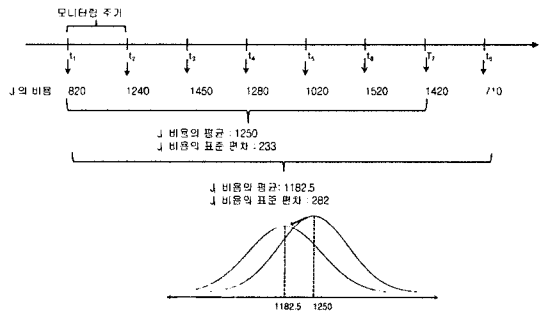

그림 1 통계 카탈로그의 갱신 방법

위의 그림은 통계 카탈로그에 있는 임의의 조인 j_i의 비용이 어떻게 바뀌는지에 대하여 설명한 그림이다. 모니터링의 주기마다 j_i의 비용은 갱신이 되며, 현재 측정된 값과 이전에 저장되어 있던 비용들의 평균과 편차를 계산하여 저장한다. 그러므로 j_i의 평균은 현재까지 모니터링된 모든 비용들의 평균 값이 되며, 그 값들의 변화 정도가 표준 편차로 정의된다. 여기에서 측정이라는 것은 실제 조인이 이루어져서 정확하게 알려진 실제 비용일 수도 있고, 예측된 값일 수도 있다. 예를 들어 A, B, C 세 개의 스트림들을 조인하는 질의가 있고 (A⋈B)⋈C)의 실행 계획을 가지고 있다면, A⋈B 와 (A⋈B)⋈C)의 조인 비용은 실제로 조인을 수행하면서 정확하게 알 수 있을 것이다. 하지만 B⋈C 조인은 실행이 되지 않기 때문에 이 조인의 비용은 예측되어야만 한다. 이런 경우에, 조인 속성의 값들은 균등분포의 형태로 도메인 크기만큼 분포하고 있다고 가정하고, 3.2절에서 사용된 식을 이용하여 그 조인의 비용을 예측할 수 있다.

4. 확장된 그리디 알고리즘(EGA)

확장된 그리디 최적화 방법은 통계 카탈로그(Statistics Catalog)을 기반으로 그리디 방법으로 비용이 가장 작은 조인 연산들을 선택해 나감으로써 실행 계획을 생성하는 방법이다. 스트림 환경에서는 실시간으로 데이타를 처리하여야 하기 때문에 질의 최적화를 통해 빠르게 데이타를 처리할 수 있게 하는 것도 중요하지만, 질의 최적화에 많은 자원을 할당할 수 없다. 그러므로 수행 시간이 빠르지만 비교적 정확한 그리디 방법이 스트림 환경에서 질의 최적화를 하는 데 많이 사용된다. 그러나 다중 조인의 경우 한 번 조인 연산이 선택되면, 그 조인으로 인하여 다른 많은 조인들이 다음 후보에서 제외되기 때문에 알고리즘의 정확성이 많이 저하된다. 예를 들어 6개의 스트림($s_1,\cdots s_6$)을 조인하는 질의가 있을 때 처음에 s_2와 s_3를 조인하는 연산이 선택되어지면 s1과 s2의 조인 연산과 s_3와 s_4의 조인 연산 등이 선택되어질 수 없다. 이와 같이 질의 최적화 과정에서 선택 후보에서 제외되는 조인들의 수가 많기 때문에 그리디 방법은 차선의 질의 계획(sub-optimal query plan)을 생성할 가능성이 매우 크다. 이 장에서 제안하는 확장된 그리디 최적화 방법은 비용이 가장 작은 하나의 후보만을 추적하여 질의 계획을 생성하는 일반적인 그리디 방법이 아니라 여러 개의 후보들을 동시에 추적함으로써 보다 정확한 질의 최적화를 할 수 있는 방법이다. 그러나 동시에 추적되는 후보의 개수가 늘어날수록 최적화를 하는 부담이 커지기 때문에 효율적으로 그 후보를 선택하고 조절하는 방법이 필요하다. 그러므로 확장된 그리디 방법에서는 3.3절에서 설명한 통계 카탈로그와 사용자 정의 변수인 범위 결정 변수를 이용하여 질의 최적화의 조인 선택 후보의 개수를 조절한다. 만약 범위 결정 변수가 0이라면 일반적인 그리디 방법과 동일한 형태로 최적화가 진행되며 범위 결정변수가 0보다 크다면, 그 값에 따라서 추적되는 후보가 결정되며, 이는 최적화 방법의 정확성과 직결되기 때문에 결론적으로는 범위 결정 변수의 값에 따라서 최적화 방법의 정확성을 조절할 수 있다.

4.1 범위 결정 변수를 통한 확장된 그리디 최적화 방법

통계 카탈로그에 저장되는 각 조인에 대한 비용 정보는 질의 최적화를 수행하는 특정 시점의 조인 비용정보를 저장하고 있는 것이 아니라 과거의 임의 시점으로부터 현재까지 모니터링 된 비용 정보들을 종합해 평균과 표준편차의 형태로 저장되어 있기 때문에 단일 값으로 인식하기 힘들다. 그러므로 통계 카탈로그에 저장하고 있는 조인 비용의 평균과 표준편차 그리고 범위 결정 변수(Range Decision Variable) k를 이용하여 각 조인 연산의 비용을 유효 범위로 표시하며, 다음과 같이 정의된다.

정의 1. 조인 비용의 유효 범위(The effective range of a join cost)

범위 결정 변수 k와 임의의 조인 연산 ji이 통계 카탈로그에 평균($\mu(j_i)$)과 표준편차($\sigma(j_i)$)로 저장되어 있을 때

조인 ji의 유효 범위는 $[\mu(j_i)-k*\sigma(j_i),\ \mu(j_i)+\ k*\sigma(j_i)]$로 나타내어진다. 이 범위의 가장 작은 비용은 $C_{min}(j_i)$, 가장 큰 비용은 $C_{max}(j_i)$로 표시한다.

위와 같이 정의된 각 조인 비용들의 유효 범위는 해당 조인이 수행되었을 때 그 조인의 비용은 이 범위 안에 있다고 가정하는 것이다. 정의 2에 따라서 각 조인 비용의 유효 범위는 조인 비용의 평균을 기준으로 표준 편차의 값이 크거나 범위 결정 변수의 값이 클수록 그 조인은 더 큰 유효 범위를 가질 것이며, 표준 편차의 값이 작거나 범위 결정 변수의 값이 작다면 작은 유효 범위를 가질 것이다. 확장된 그리디 방법은 위와 같이 정의된 각 조인 비용의 유효 범위를 기준으로 비용이 작은 조인 연산을 동시에 추적할 후보군으로 정의한다. 이 때 동시에 추적될 후보들을 ***동시 추적 조인 후보 집합 (Concurrently Traced Join Candidate Set)***이라 하고 동시 추적 조인 후보 집합의 정의는 다음과 같이 정의한다.

정의 2. 동시 추적 조인 후보 집합(Concurrently Traced Join Candidate Set)

선택할 수 있는 모든 조인들의 집합을 J_{all}이라고 하고, 이 조인 후보 집합 중에서 조인 비용의 평균값이 가장 작은 조인을 j_{min}이라고 정의하면, 동시 추적 조인 후보 집합($CTCS$)의 모든 조인들은 다음과 같은 조건을 만족해야만 한다.

$$CTCS = \{j_i \mid j_i \in J_{all},\ \hat{C}_{min}(j_i) < \hat{C}_{max}(j_{min})\}$$

위 정의는 조인의 평균 비용이 가장 작은 조인의 최대값 보다 임의의 조인의 조인 비용의 최소값이 더 작을 경우를 동시 추적 조인 후보 집합으로 본다는 것이다. 즉, 각 조인들의 범위 값들을 그래프로 그렸을 때 조인의 평균 비용이 가장 작은 조인과 겹치는 조인들이 동시 추적 조인 후보 집합의 원소가 된다. 확장된 그리디 방법은 위와 같이 CTCS들을 찾아낸 후 각 원소들에 대해서 동일한 방법으로 그리디 방법을 적용하여 추적함으로써 질의 계획을 완성하며, CTCS를 찾아내는 예제는 그림 2에 잘 나타나 있으며, 전체적인 방법은 그림 3에 자세히 설명되어 있다.

예제 1) 그림 2는 EGA의 CTCS를 구하는 방법과 같은 통계 카탈로그라고 하더라도 범위 결정 변수 k의 값에 따라서 CTCS가 달라짐을 보여준다. 통계 카탈로그에는 앞에서도 설명되었듯이 각 조인 연산자들의 조인 비용이 평균과 표준편차로 계산되어 저장되어 있다. 이 값들과 범위 결정 변수(k)를 이용하여 각 조인 비용들의 유효 범위를 계산한 후에 j_{min}을 선택한다. 위의 그

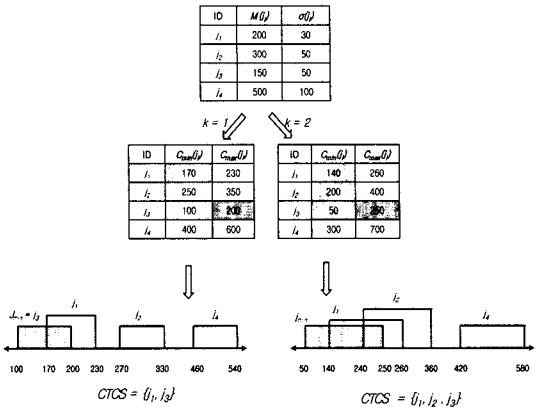

그림 2 동시 추적 조인 후보 집합의 예

림에서 k가 1인 경우, j_1의 유효 범위는 [200-1*30, 200+1*30]이 되며, j_2, j_3, j_4의 유효 범위는 각각 [300-1*50, 300+1*50], [150-1*50, 150+1*50], [500-1*100, 500+1*100]이며, j_{min}은 평균 조인 비용이 가장 작은 조인이므로 J_3이 된다. 정의 2에 따라 CTCS는 $C_{max}(j_{min})$, $C_{max}(j_3)$의 값보다 작은 $C_{min}(j_i)$를 가진 조인이 CTCS가 되기 때문에 j_1과 j_3가 CTCS가 된다. 이와 같은 방법으로 한 질의의 모든 조인 순서가 결정될 때까지 CTCS를 구하게 된다. 만약 k가 2인 경우에도 같은 방법을 적용하여 CTCS를 결정할 수 있는데, 위 그림과 같이 이 경우에는 CTCS가 j_1, j_2, j_3가 된다. 즉, 같은 통계 카탈로그가 있다고 하더라도, 사용자가 정의하는 범위 결정 변수에 따라서 확장된 그리디 최적화 방법의 결과가 달라질 수 있다는 뜻이다. 만약 범위 결정 변수의 값이 0이라면, 모든 조인 비용들의 유효 범위는 각 조인들의 평균 비용으로 대표될 것이며, CTCS의 원소는 항상 jmin, 1개일 것이다. 즉, 이 것은 확장된 그리디 방법이 일반적인 그리디 방법과 동일하게 작동된다는 것을 의미한다. 이와 반대로 범위 결정 변수가 모든 조인이 CTCS에 포함될 만큼 크다면, 확장된 그리디 최적화 방법은 한 질의가 나타낼 수 있는 모든 질의 계획을 추적함으로써 항상 최적의 결과를 생성할 것이다.

우리는 질의가 컴파일될 때(컴파일 시간) 뿐만 아니라 질의가 수행되고 있는 순간(런타임 시간)에도 범위 결정 변수의 값을 조절함으로써 확장된 그리디 최적화 방법의 정확성을 조절할 수 있다. 범위 결정 변수 값에 따라 각 조인 비용들의 유효 범위가 변동하여 동시 추적 조인 후보 집합의 원소를 결정할 때 영향을 미친다. 그러므로 범위 결정 변수 값이 증가할수록 동시 추적 조인 후보 집합에 속하는 조인의 수는 증가한다. 수행 가능한 조인 집합의 크기가 커진다는 것은 그만큼 많은 조인을 추적함으로써 보다 정확하게 최적화를 할 수 있지만, 그

```
Extended Greedy Optimization
    statistics_catalog_file: store the information of all the joins in a query
    range_k: range decision variable
    c_min(j_i): the minimum join cost of j_i
    c_max(j_i): the max join cost of j_i
    ctl_count: the size of statistics_catalog_file
    query_plan_table: the temporary plan table for CTCS
1:  while sum_join_count!=0
2:    for each ctl_count!=0 do
3:      if the selected join satisfies the conditions
4:        c_min(j_i) < c_max(j_i)
5:        add to query_plan_table
6:      end if
7:    end for
8:    sum_join_count--;
9:  end while
10: for size of query_plan_table do
11:   choose the minimum plan cost in the query_plan table
12: end for
```

그림 3 범위 변수를 통한 질의 최적화 알고리즘

만큼 최적화 하는 데 드는 시간이 더 많이 든다는 것을 의미한다.

스트림 환경에서는 스트림 처리기에 들어오는 스트림의 양에 따라서 수시로 시스템의 로드가 변한다. 폭발적으로 스트림 데이타들이 들어와서 실시간으로 스트림을 처리 못하여, 로드 쉐딩을 통해서 결과의 근사치만을 줄 수도 있고, 스트림 데이타가 거의 들어오지 않을 수도 있다. 이렇게 시시각각으로 시스템의 로드(Load)가 변하는 상황에서 이 논문에서 제안하는 확장된 그리디 방법을 적용하고 동적으로 범위 결정 변수의 값을 바꾸는 것은 시스템 로드에 적응적으로 질의 최적화를 수행할 수 있도록 해준다. 만약 처리해야 할 스트림 데이타들이 매우 많아서 질의 최적화에 많은 시간을 할당할 수 없을 경우에는 범위 결정 변수의 값을 작게 함으로써 최적화로 인한 부담을 최소화할 수 있고, 그러지 않은 경우에는 범위 결정 변수의 값을 크게 함으로써 보다 정확하게 질의 최적화를 할 수 있도록 할 수 있다.

4.2 레벨 결정 변수를 통한 그리디 최적화 방법

다중 조인의 경우 조인 연산의 결과가 다른 조인 연산의 입력으로 되기 때문에, 먼저 수행되어진 조인 연산에 따라서 다음 조인이 많은 영향을 받게 된다. 하지만 일반적인 그리디 최적화 방법은 이에 대한 고려가 전혀 없다. 그러므로 본 논문에서는 레벨 결정 변수를 통하여 다음 조인이 될 수 있는 조인들의 비용도 미리 살펴보고 비용이 가장 작은 조인을 결정하게 된다. 이 때 다음 순서가 될 수 있는 조인의 레벨을 결정하는 변수가 바로 레벨 결정 변수(Level Decide Variable)이다. 즉, 레벨 결정 변수는 다음 조인의 비용을 어디까지 고려해야 하는지를 결정하는 변수이며, 그리디 최적화 방법은 레벨 결정 변수로 정의된 곳까지 다음 조인의 비용을 고려해서 하나의 조인을 선택한다. 이 방법 또한 통계 카탈로그를 사용하며, 조인 비용의 평균값을 기준으로 그리디 방법을 적용한다.

만약 레벨 결정 변수를 0으로 지정하면 다음 순서의 조인을 전혀 고려하지 않는다는 뜻이며, $n(n>1)$일 경우에는 n번째 조인까지 고려한다는 의미가 된다. 즉, 레벨 결정 변수가 1이라면, 다음 조인의 비용까지 고려하여 하나의 조인을 선택하는 것이며 다음 조인을 고려한다는 뜻은 모든 조합에 대한 탐색을 하여야 한다는 뜻이다. n개의 스트림을 조인하는 질의의 경우 실행 계획은 (n-1)개의 조인 연산으로 만들어지기 때문에, 레벨 결정 변수는 (n-2) 이상의 값은 가질 수 없다.

4.3 하이브리드 확장된 그리디 최적화 방법

하이브리드 확장된 그리디 최적화 방법은 위에서 제안된 범위 결정 변수와 레벨 제한 변수를 모두 적용시킨 최적화 방법이다. 즉, 레벨 결정 변수로 정의된 조인 수만큼의 조합을 모두 고려하여 각 조인 조합들의 비용을 범위 결정 변수를 이용하여 유효 범위를 계산하고 동시 추적 조인 후보 집합을 결정하게 된다. 그리고 이와 같은 과정을 질의 실행 계획이 완성될 때까지 반복 수행한다.

5. 성능 평가

본 절에서는 스트림 환경에서의 이 논문에서 제안한 예측 질의 최적화 기법의 성능을 평가하고, 범위 결정 변수와 레벨 결정 변수에 따라서 질의 최적화 때문에 발생하는 부담을 조절할 수 있다는 것을 보여준다. 5.1에서는 실험 데이타와 환경에 대해서 설명하고, 5.2에서는 본 논문에서 제안한 알고리즘의 성능을 평가하고, 정의된 변수에 따른 시간의 변화에 대해서도 알아본다.

5.1 실험 데이타 및 실험 환경

본 논문에서 제안한 확장된 그리디 알고리즘을 이용한 질의 최적화 방법은 3절에서 정의된 조인 비용 모델과 유효 범위를 이용하여 각 조인들의 비용의 유효 범위를 결정하고 이 유효 범위를 비교하여 최적의 질의 계획을 찾는 방법이다. 이 것을 위한 가정 중 하나가 모든 조인의 정보(입력 속도, 조인 선택도, 조인 비용)을 알고 있다는 것이다. 그러므로 실험을 위하여 모든 조인 정보를 가지고 있는 통계 카탈로그를 생성하였으며, 이 것은 조인할 스트림의 갯수가 주어지면 실행가능한 모든 조인의 조합을 찾아내고 각 조인들의 정보들을 랜덤하게 생성하는 방법으로 만들었다. 입력 속도와 각 조인의 조인 선택도는 랜덤하게 생성하였고, 각 스트림의 윈도우 크기와 해시의 버켓 크기는 고정하였다. 이와 같이 한 이유는 각 스트림의 윈도우는 질의가 등록될 때 이미 주어지는 것이기 때문에 질의를 컴파일할 때 스트림의 크기를 알 수 있고, 한 번 등록된 질의는 바뀌지 않기 때문이며, 해시의 버켓 크기도 마찬가지로 시스템의 구동 이후에는 바뀌지 않는 가정 하에 모든 실험에서 이 2개의 변수는 고정하였다. 본 논문에서 사용되는 모

표 2 데이타셋과 알고리즘의 명명

IR:i_1~i_2_JS:j_1~j_2	스트림들의 입력 속도의 랜덤 범위는 최소 i_1에서 i_2_의 값 내에서 생성하며, 조인 선택도 또한 최소 j_1에서 j_2의 값 내에서 랜덤하게 생성하는 데이타셋
EGAk_l	범위 제한 변수를 k로 레벨 제한 변수를 l로 정의한 하이브리드 확장된 그리디 알고리즘을 이용한 질의 최적화 방법
Greedy	기존의 일반적인 Greedy 최적화 방법을 이용한 질의 최적화 방법

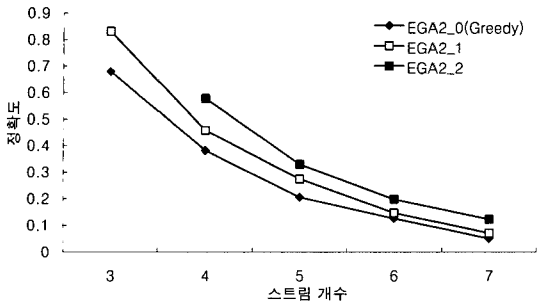

그림 4 스트림의 개수에 따른 성능 비교

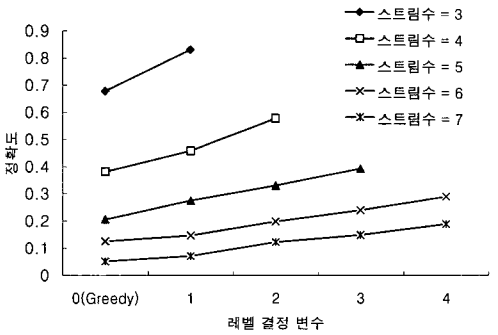

그림 5 레벨 결정 변수에 따른 성능 비교

든 해시 버켓의 크기는 20으로 윈도우의 크기는 10으로 고정하였다. 각 기본 스트림의 입력 속도와 조인 선택도는 랜덤 생성되는 수들의 범위를 정하고, 그 범위 내에서 랜덤하게 수를 생성하였다. 기본 스트림간의 조인일 경우에는 랜덤하게 생성된 각각의 입력 속도와 조인선택도, 그리고 고정되어 있는 해시 버켓의 크기와 윈도우 크기로 조인 비용을 계산하였고[15], 한 조인의 결과가 또 다른 조인의 입력 스트림으로 사용되는 경우에는 선행되는 조인의 출력 속도를 계산하여 다음 조인의 입력 속도로 활용하였고, 구하는 식은 다음과 같다.

$$\lambda_{AB} = \frac{(T_A + T_B)\lambda_A \times \lambda_B \times join\_selectivity(A,B)}{T_{AB}} \quad (4)$$

그림 3는 실험에서 사용되는 실험 데이타의 특성과 각 알고리즘의 이름을 정리한 것이다.

그림 4에서는 조인해야 될 스트림의 갯수가 늘어남에 따른 알고리즘의 정확성을 측정하였다. 여기에서 알고리즘의 정확성은 각 알고리즘으로 선택된 질의 계획과 모든 가능한 질의 계획 중 가장 작은 비용을 갖는 질의 계획이 같을 확률로 정의하였다. 이 실험의 데이타셋은 IR:1~100_JS:0~0.001이었으며, 각각 10000번 실행하여 그 값들을 구하였다.

그림 4에서 보는 바와 같이 조인 스트림의 갯수가 많아질수록 두 알고리즘 다 정확성은 떨어지지만 확장된 그리디 알고리즘을 이용한 질의 최적화 방법이 그리디 알고리즘보다 항상 좋은 성능을 보였다.

그림 5은 레벨 제한 변수의 값을 고정시키고 범위 제한 변수의 값이 성능에 어떤 영향을 끼치는 지에 대한 실험 결과이다. 데이타셋은 위 실험과 마찬가지로 IR:1~100_JS:0~0.001 이다.

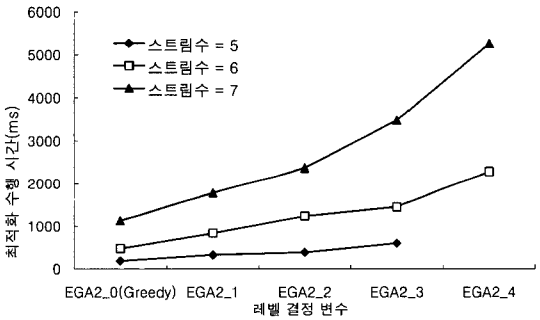

그림 6 레벨 결정 변수에 따른 수행 시간 비교

그림 5에서와 같이 레벨 제한 변수의 값이 커질수록 동시 추적 조인 후보 집합들이 많아지게 되고, 이렇게 더 많아진 조인들에 대해서 각각 다음 조인에 대한 비용을 고려하기 때문에, 즉 질의 최적화를 수행할 때 좀 더 많은 조인 순서에 대해서 검사를 하기 때문에 확장된 그리디 알고리즘의 성능은 향상된다. 하지만 레벨 제한 변수의 값이 "스트림의 갯수 - 2"보다 커지는 것은 의미가 없음을 알 수 있다.

그림 6은 레벨 결정 변수에 따른 수행 시간의 변화를 나타낸다. 레벨 결정 변수의 값이 커질수록 비용을 비교해야 하는 조인의 집합들이 많아지기 때문에 더 많은 질의 최적화 수행 시간이 걸리게 된다.

그림 7(a)와 7(b)에서는 범위 제한 변수의 값에 따라 확장된 그리디 알고리즘이 어떤 성능을 나타내는지에 대한 실험이다.

그림 7의 데이타셋은 모두 IR:1~100_JS:0~0.001이며, 그림 7(a)는 레벨 제한 변수를 2로 하였을 때의 결

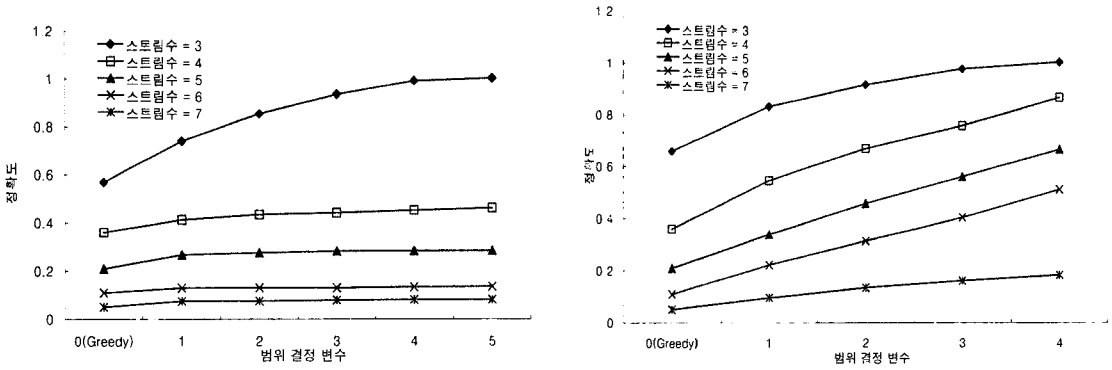

그림 7 범위 결정 변수에 따른 확장된 그리디 알고리즘의 성능 비교

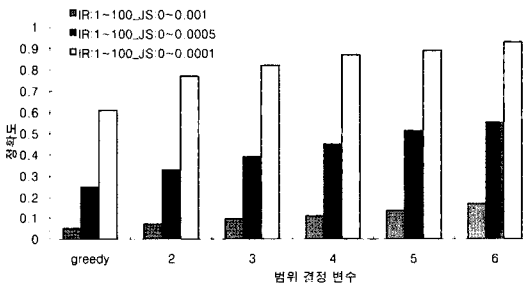

그림 8 조인 선택도의 랜덤 범위에 따른 성능 비교

과이며, 그림 7(b)는 레벨 제한 변수를 4로 하였을 때의 실험 결과이다. 이 두 그래프를 보면 레벨 제한 변수가 작을 경우에는 스트림의 개수가 작을 때에는 범위 제한 변수가 알고리즘의 성능에 어느 정도 영향을 미치나, 스트림의 갯수가 늘어날수록 알고리즘의 성능 향상에 큰 도움이 되지 못함을 알 수 있다. 하지만 레벨 제한 변수의 크기를 크게 하였을 때에는 스트림의 갯수가 많아질수록 알고리즘의 성능에 미치는 영향은 작아지지만, 레벨 제한 변수의 크기가 작을 때보다 더 많은 영향을 끼침을 알 수 있다.

그림 8은 조인 선택도의 랜덤 범위가 알고리즘의 성능에 어떤 영향을 미치는지에 대한 실험이다.

위 실험에서 보면 조인 선택도의 랜덤 범위가 낮은 값들의 범위를 가질수록 알고리즘의 성능이 좋아지는 것을 알 수 있다. 이러한 이유는 조인 선택도의 값이 작아지면 조인을 수행할수록 조인의 중간 결과들의 크기가 작아지게 되고 중간 결과들의 크기가 작아진다는 것은 레벨의 낮은 조인의 비용이 총 질의 계획의 비용에 더 많은 영향을 끼치는 것을 의미하기 때문이다.

6. 결 론

매우 다양하고 광대한 변동 요인이 있는 스트림 환경 하에서의 다수의 스트림을 조인하는 조건을 갖는 연속 질의를 처리하는 것은 다른 연산에 비해 많은 비용이 필요하며, 어떤 질의 계획에 따라 처리하느냐에 따라 처리 시간이 많이 차이가 나기 때문에 질의 최적화가 매우 중요하다. 기존에 존재하던 DBMS와는 달리 데이타 스트림 환경에서는 실시간으로 들어오는 스트림 데이타들에 대해서 계속적으로 최적화된 질의 계획이 달라질 수 있으므로, 지속적인 질의 최적화가 필요하게 된다. 본 논문에서는 이러한 조인 연속 질의의 최적화를 하기 위한 방법으로 확장된 그리디 알고리즘을 이용한 질의 최적화 방법을 제시한다. 확장된 그리디 최적화 알고리즘은 축적된 스트림 정보들을 이용하여 조인 조건을 갖는 단일 연속 질의를 최적화 하는 알고리즘이다. 이 알고리즘은 범위 결정 변수와 레벨 결정 변수라는 2개의 변수에 따라서 수행하는 방법이 달라지게 되고, 이것은 질의 최적화의 정확도와 수행 시간에 영향을 끼친다. 2개의 변수 값이 커질수록 최적화의 성능은 좋아지는 반면에 최적화를 하는 데 걸리는 시간은 더 오래 걸리고 반대로 변수값들이 작아질수록 최적화의 성능은 떨어지는 반면, 최적화하는 데 걸리는 시간은 적게 걸린다. 본 논문에서 제안하는 알고리즘은 동적으로 최적화를 수행할 때 생기는 부담을 조절할 수 있으며, 이는 계속적으로 스트림의 특성이 변화하고 이를 예측할 수 없는 스트림 환경에 시스템 로드에 따라서 적응적으로 질의 최적화할 수 있다.

참 고 문 헌

[1] A. Krishnamurthy, H. boral, and C. Zaniolo. Optimization of nonrecursive queries. In proc. Of the 1986 Intl. Conf. In Very Large Data Bases, pages 128-137, Aug. 1986.

[2] Terry, D. et al., "Continuous Queries over Append-Only Databases," In Proc. Int'l Conf. on Manage-

ment of Data, ACM SIGMOD, San Diego, California, pp. 321-330, June 1992.
[3] B. Babcock, S. Babu, M. Datar, R. Motwani, and J. Widom. Models and Issues in Data Stream Systems Invited paper In Proc. of PODS 2002, June 2002.
[4] Lukasz Golab M. Tamer Ozsu. "Processing Sliding Window Multi-Joins in Continuous Queries over Data Streams," Proceedings of the 29th VLDB Conference, Berlin, Germany, 2003.
[5] Motwani, R. et al., "Query Processing, Approximation, and Resource Management in a Data Stream Management System," In Proc. the First Biennial Conf. on Innovative Data Systems Research, Asiloma, California, pp. 245-256, Jan. 2003.
[6] The STREAM groups STREAM: The Stanford Stream Data Manager (short overview paper) IEEE Data Engineering Bulletin, March 2003.
[7] S. Babu, R. Motwani, K. Munagala, I. Nishizawa, J. Widom, Adaptive ordering of pipelined stream filters, In Proceedings of the 2004 ACM SIGMOD International Conference on Management of Data. pp. 407-418 2004.
[8] Carney, D., Cetintemel, U., Rasin, A., Zdonik, S., Cherniack, M., Stonebraker, M. "Operator Scheduling in a Data Stream Manager," Proceedings of the 29th VLDB Conference, Berlin, Germany, 2003.
[9] U. Srivastava, K. Munagala and J. Widom. "Operator Placement for In-Network Query Processing," In Proc. of PODS 2005, June 2005.
[10] Shivnath Babu, Kamesh Munagala, Jennifer Widom, and Rajeev Motwani. Adaptive Caching for Continuous Queries In Proc. Int. Conf. on Data Engineering (ICDE), 2005.
[11] S. Madden, M. Shah, J. Hellerstein, and V. Raman. Continuously adaptive continuous queries over streams. In Proc. Of the 2002 ACM SIGMOD Intl. Conf. on Management of Data, pages 49-60, June 2002.
[12] R. avnur and J.M. Hellerstein. Eddies: Continuous Adaptive Query Processing. In ACM SIGMOD, Dallas, TX, May 2000.
[13] Abadi, D. J. Carney,D., Centintemel, U., Cherniack, M., Convey, C., Lee, S., Stonebraker, M., Tatbul, N., Zdonik, S. Aurora: A New Model and Architecture for Data Stream Management. VLDB Journal, 2003.
[14] J. Chen, D. DeWitt, F. Tian, and Y. Wang. NiagaraCQ: A scalable continuous query system for internet databases. In ACM SIGMOD, 2000.
[15] J. Kang, J. F. Naughton, and S. Viglas. Evaluating window joins over unbounded streams. In Proc. of the 2003 Intl. Conf. on Data Engineering, Mar. 2003.
[16] Michael Cammert, J¨urgen Kr¨amer, Bernhard Seeger, Sonny Vaupel A Cost model for adaptive Resource Management in data stream systems IEEE Transactions on Knowledge and Data Engineering Volume 20, Issue 2 February 2008.
[17] Stratis Viglas, Jeffrey F. Naughton, and Josef Burger. Maximizing the output rate of multi-way join queries over streaming information sources. Proceedings of the 29th VLDB Conference, Berlin, Germany, 2003.
[18] S. Viglas and J. F. Naughton. Rate-based Optimization for Streaming Information Sources, In Proc. ACM SIGMOD Int. Conf. on Management of data, 2002, pp. 37-48.

박 홍 규
2004년 2월 연세대학교 컴퓨터과학과 학사. 2007년 2월 연세대학교 컴퓨터과학과 석사. 1007년 3월~연세대학교 컴퓨터과학과 박사 과정. 관심분야는 Data Stream, Query Processing

이 원 석
1985년 7월 Boston University 컴퓨터과학 공학학사. 1987년 7월 Purdue University 컴퓨터과학 공학석사. 1990년 7월 Purdue University 컴퓨터과학 공학박사. 2004년 3월~현재 연세대학교 컴퓨터과학과 정교수. 관심분야는 Database, Data Mining