KIPS Tr. Software and Data Eng.
Vol.4, No.12 pp.543~548 pISSN: 2287-5905

http://dx.doi.org/10.3745/KTSDE.2015.4.12.543

An Efficient Top-k Query Processing Algorithm over Encrypted Outsourced-Data in the Cloud

Jong Wook Kim† · Young-Kyoon Suh††

ABSTRACT

Recently top-k query processing has been extremely important along with the explosion of data produced by a variety of applications. Top-k queries return the best k results ordered by a user-provided monotone scoring function. As cloud computing service has been getting more popular than ever, a hot attention has been paid to cloud-based data outsourcing in which clients' data are stored and managed by the cloud. The cloud-based data outsourcing, though, exposes a critical secuity concern of sensitive data, resulting in the misuse of unauthorized users. Hence it is essential to encrypt sensitive data before outsourcing the data to the cloud. However, there has been little attention to efficient top-k processing on the encrypted cloud data. In this paper we propose a novel top-k processing algorithm that can efficiently process a large amount of encrypted data in the cloud. The main idea of the algorithm is to prune unpromising intermediate results at the early phase without decrypting the encrypted data by leveraging an order-preserving encrypted technique. Experiment results show that the proposed top-k processing algorithm significantly reduces the overhead of client systems from 10X to 10000X.

Keywords : Cloud Computing, Encryption, Top-k Query

아웃소싱 암호화 데이터에 대한 효율적인 Top-k 질의 처리 알고리즘

김 종 욱† · 서 영 균††

요 약

최근 다양한 분야에서 생산되는 데이터의 양이 폭발적으로 증가함에 따라 사용자가 가장 관심 있어 하는 몇 개의 데이터를 검색하는 top-k 질의에 대한 관심이 고조되고 있다. Top-k 질의는 사용자의 점수 함수를 이용하여, 사용자가 원하는 모든 조건을 만족시키는 데이터들 중에서 최상위 (또는 최하위) 점수를 가지는 k개의 데이터를 사용자에게 반환한다. 최근 들어 클라우드 컴퓨팅 서비스의 대중화로 인하여 사용자의 대용량 데이터를 클라우드에 아웃소싱하여 경제적으로 저장 및 관리하는 데이터 아웃소싱이 크게 주목받고 있다. 그러나 데이터 아웃소싱으로 인하여 사용자의 민감한 데이터가 클라우드 서비스 제공자에게 노출될 수 있다는 위험이 존재하며, 이러한 문제를 방지하기 위해서는 사용자의 민감한 데이터를 암호화하여 클라우드에 저장하는 것이 필수적으로 요구된다. 본 논문은 클라우드 컴퓨팅 환경에서 암호화된 데이터에 대한 top-k 질의를 효율적으로 처리하는 알고리즘을 제안한다. 제안되는 알고리즘은 순서보존 암호화 기법을 이용하여, 암호화된 데이터만을 대상으로 top-k 질의 결과에 포함되지 않을 것으로 예상되는 중간 결과들을 클라우드 내에서 미리 제거함으로써 효율적인 top-k 질의 처리가 가능하게 한다. 논문의 실험 결과는 제안된 top-k 질의 처리 알고리즘이 단순 방법과 비교하여 사용자 시스템의 부하를 10배~10000배 줄일 수 있음을 증명한다.

키워드 : 클라우드 컴퓨팅, 암호화, Top-k 질의

1. 서 론

최근 다양한 분야에서 생산되는 데이터의 양이 폭발적으로 증가함에 따라 사용자가 원하는 모든 조건을 만족시키는 데이터들 중에서 가장 관련이 높은 몇 개의 데이터를 검색하는 top-k (조인) 질의에 대한 관심이 고조되고 있다. Top-k 질의에서 사용자의 관심은 점수 함수(score function)로 표현되며, 점수 함수들은 단조함수(monotonic function)라는 특징이 있다. Top-k 질의는 사용자의 점수 함수를 이용하여, 사용자가 원하는 모든 조건을 만족시키는 데이터들 중에서 최상위 (또는 최하위) 점수를 가지는 k개의 데이터를 사용자에게 반환한다[1-3].

클라우드 컴퓨팅은 인터넷 기술을 이용하여 클라우드 컴퓨팅 서비스 공급자가 제공하는 IT 자원(예, 소프트웨어, 서

※ 이 논문은 미래창조과학부 및 한국연구재단의 EDISON 사업(NRF-2011-0020576) 지원으로 연구되었음.
† 정 회 원 : 상명대학교 미디어소프트웨어학과 조교수
†† 비 회 원 : 한국과학기술정보연구원 국가슈퍼컴퓨팅연구소 연구원
Manuscript Received : August 26, 2015
First Revision : October 13, 2015
Second Revision : November 23, 2015
Accepted : November 23, 2015
* Corresponding Author : Young-Kyoon Suh(yksuh@kisti.re.kr)

버, 스토리지 등)을 필요한 만큼 빌려 쓰고, 사용한 만큼 비용을 지불하는 서비스 형태를 의미한다. 최근 들어 클라우드 컴퓨팅 서비스의 대중화로 인하여 사용자의 대용량 데이터를 클라우드에 아웃소싱(outsourcing)하여 경제적으로 저장 및 관리하는 데이터 아웃소싱이 크게 주목받고 있다. 데이터 아웃소싱으로 인하여 사용자는 자체적인 IT 인프라 구축 및 운영에 드는 비용 절감의 효과를 얻을 수 있다는 장점이 있다. 그러나 이러한 장점에도 불구하고 사용자의 데이터가 클라우드 스토리지 서버에 저장되기 때문에, 사용자의 민감한 데이터(sensitive data)가 클라우드 서비스 제공자에게 노출될 수 있다는 위험이 항상 존재한다. 그러므로 클라우드 서비스를 이용하여 데이터를 아웃소싱하기 위해서는 사용자의 민감한 데이터를 암호화하여 보호하는 것이 필요하다.

민감한 데이터를 암호화하여 클라우드에 저장하면, 사용자는 외부에 그러한 민감한 데이터를 노출하는 위험을 방지할 수 있다. 그러나 클라우드 환경에서 암호화된 데이터로 인한 성능 저하 문제가 발생할 수 있다. 일반적으로, 데이터에 암호화를 적용하면 기존 인덱스 기법 사용에 제한이 있기 때문에 응답시간 저하가 발생한다. 또한 암호화된 데이터에 대한 직접적인 연산을 수행하는 것이 제한적이므로, 클라우드 서비스만을 이용하여 사용자의 요구사항을 처리할 수 없게 된다. 이 경우 암호화된 데이터로 인하여 클라우드에서 처리 불가한 연산들은 사용자 시스템에서 처리되어야 하는데, 이는 사용자 시스템의 부하를 가중시키는 요인이 될 수 있다. 이에 따라 데이터베이스 커뮤니티를 중심으로 암호화된 데이터에 대하여 사용자 질의를 효율적으로 처리하는 방법에 관한 연구가 활발히 진행되고 있다[4-6]. 그러나 기존 연구에서 지원하는 사용자 질의의 종류는 매우 제한적이며, 특히 암호화된 데이터에 대한 top-k 질의 처리 기능은 아직 지원하지 못하고 있는 실정이다. 이러한 문제를 해결하기 위해, 본 논문은 클라우드 컴퓨팅 환경에서 암호화된 데이터에 대한 top-k 질의를 효율적으로 처리하기 위한 방법을 제안한다.

본 논문은 다음과 같이 구성되어 있다. 2장에서 본 논문에서 다루는 문제를 정의하고, 3장에서 논문에서 제안하는 알고리즘을 설명한다. 4장에서 제안한 알고리즘의 성능 평가를 수행한 후, 5장에서 결론을 맺는다.

2. 문제 정의 및 단순 방법

2.1 문제 정의

Fig. 1은 본 연구에서 가정하는 시스템 구조를 나타낸다.

클라우드 스토리지 서버는 사용자의 모든 요구사항을 정직하게 실행하지만, 아웃소싱된 데이터의 내용에 관하여 궁금해한다고(Honest but curious model) 가정한다. 따라서 사용자의 민감한 데이터를 보호하기 위해, 그 데이터에 대해 암호화를 적용한 후, 클라우드 스토리지 서버에 안전하게 데이터를 저장하는 것이 필수적으로 요구된다. Fig. 1에서 보이는 바와 같이, 사용자들은 스마트폰 혹은 태블릿과 같

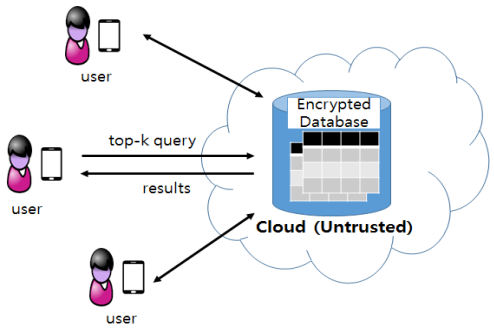


Fig. 1. System Architecture Assumed in This Paper

이 컴퓨팅 자원이 제한적인 클라이언트 기기(사용자 시스템)를 사용하여 top-k 질의와 같은 사용자 요구사항을 클라우드에 요청한다. 클라우드는 사용자 질의를 수행한 후, 그 질의에 대한 결과를 사용자에게 반환한다. 일반적으로 사용자 시스템에서 이용 가능한 컴퓨팅 자원은 매우 제한적이므로, 대용량 데이터에 대한 top-k 질의 처리와 같이 복잡한 연산은 클라우드 환경에서 수행되는 것이 바람직하다.

사용자의 데이터베이스는 n개의 릴레이션($R = \{R_1, R_2, \ldots. R_n\}$)으로 구성되어있고, 각각의 릴레이션 $R_p \in R$는 다음과 같이 나타낼 수 있다고 가정하자.

$$R_p(ID_p, n_{p,1}, n_{p,2}, \cdots, n_{p,l}, s_p)$$

속성 ID_p는 릴레이션 R_p의 기본 키(primary key)에 해당한다. 속성 $n_{p,1}, n_{p,2}, \cdots, n_{p,l}$는 민감하지 않은 데이터(non-sensitive data) 열에 해당하고, 속성 s_p는 민감한 데이터 열에 해당한다. 설명의 편의성을 위하여, 본 논문은 각 릴레이션에 한 개의 민감한 데이터 열이 존재한다고 가정한다. 그럼에도 불구하고 본 논문이 제안한 알고리즘은 다수의 민감한 데이터 열이 존재하는 경우에도 동일하게 적용된다. 사용자의 대용량 데이터는 Fig. 1과 같이 클라우드 서비스를 이용하여 저장 및 관리된다. 이때, 사용자의 민감한 데이터 열에 해당하는 값들이 외부에 노출되는 것을 방지하기 위해, 이 값들은 암호화된 상태로 클라우드 스토리지 서버에 저장된다.

사용자는 사용자 시스템을 이용하여 다음과 같은 top-k (조인) 질의를 요청한다고 가정하자.

```
SELECT select-list
FROM   R1, R2, ···, Rn
WHERE  equi-join-condition(R1, R2, ···, Rn)
ORDER BY f(s1, s2, s3, ···, sn)
STOP AFTER k
```

즉, 사용자는 질의의 조건을 만족하는 데이터들 중에서 최상위(혹은 최하위) 점수를 가지는 k개의 데이터에 관심이 있다. 이때, $f(s_1, s_2, s_3, \cdots, s_n)$는 사용자가 제공한 단조 점수 함수에 해당하며, 점수 함수의 값은 암호화된 상태로 클

라우드에 저장되어있는 민감한 데이터에 의해 결정된다.

2.2 단순 방법

2.1절의 top-k 질의를 처리하기 위해서는 (1) 점수 함수에 의해 정의된 대로 암호화된 데이터에 대하여 연산을 수행한 후, (2) 결과 값들을 비교하여 최상위(혹은 최하위) 값을 가지는 k개의 데이터를 클라우드에서 선택하는 연산이 요구된다. 그러나 최근 암호화 분야의 노력(예, 완전 동형 암호화 기법[7], 준완전 동형 암호화 기법[8], 순서보존 암호화 기법[9, 10])에도 불구하고, 암호화된 데이터에 대하여 top-k 질의를 처리하는 데 요구되는 이러한 연산들을 모두 지원하는 암호화 기법은 아직 개발되지 못한 실정이다. 그러므로 암호화 데이터에 대한 top-k 질의는 다음과 같이 처리될 수 있다.

- 클라우드 서버 : 암호화된 데이터에 대하여 점수 함수 값을 계산하고, 점수 함수 값들을 비교하여 k개의 데이터를 선택하는 것이 불가능하기 때문에, 조인 수행의 모든 결과를 사용자에게 반환한다. 이때, k개의 결과 선택은 사용자 시스템에서 수행되어야 하므로, 점수 함수 값을 계산하기 위해 필요한 모든 값들을 사용자에게 반환하여야 한다. 즉 2.1절의 질의는 다음의 결과를 사용자에게 반환한다.

```
SELECT select-list, s1, s2, s3, …, sn
FROM   R1, R2, …, Rn
WHERE equi-join-condition(R1, R2, …, Rn)
```

  이때, 속성 s_1, s_2, s_3, …, s_n의 값은 암호화된 상태로 사용자에게 반환된다.

- 사용자 시스템 : 사용자 시스템은 클라우드로부터 전송받은 암호화된 값(즉, 속성 s_1, s_2, s_3, …, s_n의 값)을 복호화(decryption)한 후, 점수 함수 값을 계산한다. 최종 단계에서 사용자 시스템은 결과 값들을 비교하여 상위 k개의 데이터를 선택하여 사용자에게 반환한다.

위의 top-k 질의 처리를 위한 단순 방법에서 드러난 가장 큰 문제 중의 하나는, 사용자 시스템에 큰 부하를 줄 수 있다는 점이다. 즉 클라우드 스토리지 서버에 암호화되어 저장된 데이터에 대한 질의 수행에 필요한 많은 연산들이 사용자 시스템에서 수행되게 된다. 일반적으로 사용자 시스템의 컴퓨팅 자원이 제한적이고 계산 능력이 낮다는 것을 고려할 때, 이러한 질의 수행은 매우 비효율적이라고 할 수 있다.

3. 클라우드 환경에서의 암호화된 데이터에 대한 효율적인 TOP-k 질의 수행

암호화된 데이터에 대한 보다 효율적인 top-k 질의 처리 방법은 고성능의 컴퓨팅 자원을 가진 클라우드 서버에서 질의 처리에 필요한 대부분의 작업을 수행하여, 사용자 시스템에서 수행되어야 할 연산을 최소화하는 방식이다. 그러므로 본 절은 사용자 시스템의 부하를 최소화하기 위한 top-k 질의 처리 알고리즘을 제안한다. 특히 본 절에서 제안하는 top-k 질의 처리 알고리즘은 클라우드 서버에서 순서보존 암호화 기법[9, 10]을 이용하여 질의 결과에 포함되지 않을 것으로 예상되는 중간 결과들을 가지치기(pruning)함으로써, 사용자 시스템의 로드를 최소화한다.

3.1 순서보존 암호화 기법을 이용한 가지치기

순서보존 암호화 기법(order-preserving encryption)은 암호화된 데이터들의 크기 순서가 평문 데이터들의 크기 순서를 유지하도록 제안된 알고리즘이다[9, 10]. 즉, 순서보존 암호화 기법은 두 평문 데이터 x, y와 각각의 암호화된 데이터 Enc(x), Enc(y)에 대하여 다음의 조건을 만족한다.

$$x > y \rightarrow Enc(x) > Enc(y)$$

순서보존 암호화 기법은 평문 데이터의 크기 순서가 외부에 노출된다는 단점이 있다. 그러나 순서보존 암호화 기법에 의해 암호화된 데이터들은 암호화된 상태에서 크기 비교가 가능하기 때문에, 데이터베이스 시스템에서 많이 활용되고 있다[4-6].

본 절에서 제안하는 알고리즘을 설명하기 위하여 다음의 사용자 top-k 질의를 가정하자.

```
SELECT    R1.ID1, (0.5*s1 + 0.2*s2)
FROM      R1, R2
WHERE     R1.ID1=R2.ID2
ORDER BY  0.5*s1 + 0.2*s2
STOP AFTER 5
```

위의 top-k 질의의 점수 함수는 가중치가 있는 합계(weighted sum)에 해당하며, 속성 s_1과 s_2의 값은 각각 순서보존 암호화 기법을 사용하여 암호화된 상태로 클라우드 스토리지 서버에 저장되어있다고 가정하자. 논문에서는 설명의 편의성을 위해서 2-way 조인 질의를 사용하지만, 본 절에서 제안한 알고리즘은 일반적인 m-way 조인 질의에 대하여 적용 가능하다. Fig. 2는 위의 top-k 질의의 등가 조인 연산을 수행한 후, 조인 결과를 2차원 상에 나타낸다.

x축과 y축은 각각 속성 s_1과 s_2의 암호화된 값에 해당하며, 순차적으로 정렬되어있다(즉, 원점에서 멀어질수록 높은 값에 해당함). 또한, 검은색 작은 사각형들은 R_1과 R_2의 등가 조인 결과 튜플들에 해당한다. 이때, top-k 질의의 최종 결과는 단조 점수 함수의 특성에 따라 사각형의 네 모서리 중 한곳에 위치하게 되며, 위의 예제에서는 점수 함수가 s_1과 s_2에 대한 단조 증가 함수에 해당하므로 top-k 질의의 결과가 Fig. 2와 같이 오른쪽 상단 모서리에 존재하게 된다. Fig. 2에서는 등가 조인 수행 후, top-k 영역(area)에 이미 k개의 튜플들이 존재한다고 가정한다. 이때, 기준점 (v_1, v_2)를 기준으로 전체 영역을 3개의 다른 영역 (top-k area,

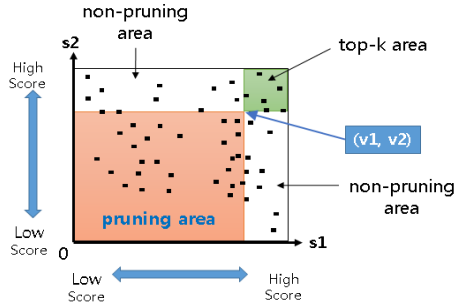


Fig. 2. Pruning Area vs. Non-Pruning Area

non-pruning area, pruning area)으로 분할한다. 이때, top-k 영역과 pruning 영역에 존재하는 모든 튜플들은 다음 조건을 만족한다.

$$\forall\, t_1 \in \text{top-k area},\ \forall\, t_2 \in \text{pruning area},\ score\_t_1 > score\_t_2$$

위의 조건에서 $score_t_1$는 튜플 t_1(즉 Fig. 2의 검은색 작은 사각형들)의 점수 함수 값에 해당하며, 예제 top-k 질의의 경우 다음과 같이 계산 가능하다($score_t_2$도 유사하게 정의된다).

$$score\_t_1 = 0.5*Dec(t_1.s_1) + 0.2*Dec(t_1.s_2)$$

이때, Dec()는 암호화된 값을 복호화하는 함수에 해당한다. 이 경우, top-k 영역에서 이미 k개의 튜플들이 존재하므로, 위의 조건을 이용하여 pruning 영역에 존재하는 튜플들은 복호화 과정 없이 클라우드에서 가지치기가 가능하다.

이와 달리 top-k area와 non-pruning area에 존재하는 튜플들은 다음 조건을 만족한다.

$$\forall\, t_1 \in \text{top-k area},\ \forall\, t_2 \in \text{non-pruning area},\ (score\_t_1 \geq score\_t_2) \mid (score\_t_1 < score\_t_2)$$

Non-pruning 영역에 존재하는 튜플들은 top-k 영역에 존재하는 튜플들보다 높은 점수 함수 값을 가질 수 있으므로, 사용자 시스템에서 복호화 과정을 수행한 후, 실제 값을 비교하는 과정이 필수적으로 요구된다. 즉, non-pruning 영역에 존재하는 튜플들에 대해서는 클라우드에서 암호화된 데이터만을 이용하여 최종 top-k 결과에 포함될지 여부를 판별할 수 없다.

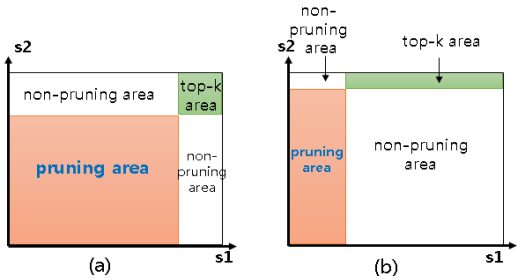


Fig. 3. (a) Optimal vs. (b) Non-Optimal Partitioning

3.2 OPT(Order-Preserving Top-k processing algorithm) 알고리즘

본 절은 암호화된 데이터에 대한 top-k 질의 처리 기법을 제안한다. 먼저, 클라우드에 암호화하여 데이터를 아웃소싱할 때, 각 암호화된 데이터 열에 대하여 새로운 데이터 열을 추가한다. 가령, 2절의 릴레이션 $R_p \in R$의 경우 암호화된 데이터 열 s_p에 대해 데이터 열 o_p를 다음과 같이 추가한다.

$$R_p(ID_p,\ n_{p,1},\ n_{p,2},\ \cdots,\ n_{p,l},\ s_p,\ o_p)$$

추가된 데이터 열 o_p는 순서보존 암호화 기법을 이용하여 암호화되어 저장되어있는 s_p값을 내림차순으로 정렬했을 때, 각 튜플이 몇 번째에 위치했는지를 저장한다. 즉, s_p에 대해 가장 큰 값을 가지는 튜플의 o_p 값은 0이며, 가장 작은 값을 가지는 튜플의 o_p 값은 $(|R_p|-1)$에 해당한다.

Fig. 3과 같이 전체 영역을 3개의 영역(top-k area, non-pruning area, pruning area)으로 분할하는 방법은 다양하게 존재할 수 있다. 이 중 최적의 전략은 pruning 영역을 최대화하여 클라우드에서 가능한 많은 튜플들을 가지치기할 수 있는 데이터 분할 방법이다. 본 논문은 이러한 점을 고려하여 암호화 데이터에 대한 사용자 top-k 질의를 다음과 같이 수행한다.

- 클라우드 서버 : 클라우드 서버는 알고리즘 1과 같이 수행한다. 먼저, 1번에서처럼 사용자 질의를 변환하여 데이터베이스 관리 시스템 (DBMS: Database Management Systems)에 결과를 요청한다. 이때 변환된 질의의 결과 값은 $sum(o_1, o_2, \cdots, o_n)$에 의해 정렬된다. 2번과 3번에서 함수 getK-thTuple()을 이용하여 $sum(o_1, o_2, \ldots, o_n)$ 값에 의해 정렬된 결과 튜플 리스트(tuple_list)로부터 k-번째 튜플을 추출한 후, 전체 영역을 분할하는 기준점을 구한다. 가령, 3.1절의 예제 질의의 경우 k 값은 5에 해당하므로 함수 getK-thTuple()를 이용하여 tuple_list로부터 5번째 튜플을 추출한 후, Fig. 4에서와 같이 그 값을 이용하여 영역을 분할하는 기준점 (3,3)을 구할 수 있다. 4번에서 9번까지는 tuple_list에 존재하는 튜플들을 순차적으로 검사하여, pruning 영역에 존재하지 않는 튜플들을 Set_cand에 추가한다. 마지막으로 10번에서 Set_cand에 존재하는 튜플들을 사용자 시스템에 전송한다.
- 사용자 시스템 : 사용자 시스템에서는 클라우드로부터 전송받은 암호화된 값을 복호화한 후, 점수 함수 값을 계산한다. 마지막 단계에서는 결과 값들을 비교하여 상위 k개의 데이터를 선택한다.

본 절에서 제안하는 top-k 질의 수행 기법은 2.2절의 단순 방법과 달리 클라우드 서버에서 중간 결과들을 가지치기함으로써, 계산 능력이 낮은 사용자 시스템의 부하를 줄일 수 있고, 클라우드와 사용자 시스템 간에 전송되는 데이터 양을 최소화할 수 있다는 장점이 있다.

Input: top-k query
SELECT select-list
FROM R_1, R_2, ... R_n
WHERE equi-join-condition(R_1, R_2, ... R_n)
ORDER BY f(s_1, s_2, s_3, ... s_n)
STOP AFTER k

1: tuple_list ←
SELECT select-list,
s_1, s_2,..., s_n,
sum(o_1, o_2... o_n) AS sum
FROM R_1, R_2, ... R_n
WHERE equi-join-condition(R_1, R_2, ... R_n)
ORDER BY sum(o_1, o_2,...,o_n)
2: t ← getK-thTuple(tuple_list, k)
3: (v_1, v_2,.....,v_n) ← (t.sum, t.sum,....., t.sum)
4: Set_cand ← Φ
5: **for** each t of tuple_list
6: **if** (t.o_1<v_1) **AND** (t.o_2<v_2) **AND** ... **AND** (t.o_n<v_n)
7: // do nothing
8: **else**
9: add t to Set_cand
10: **send** Set_cand **to** Client

Algorithm 1. Pseudo-Code for Computing the Point to be Used for Selecting the Pruning Area

4. 실 험

본 절은 이전 절에서 제안한 OPT 알고리즘의 성능 평가를 수행한다. 이를 위해, TPC-H 벤치마크[11]의 데이터베이스 스키마를 사용한다. TPC-H 벤치마크 데이터베이스 스키마로부터 3개의 릴레이션(LINEITEM, PARTSUPP, ORDERS)을 선택하였으며, 각각의 릴레이션에는 6,000,000, 800,000, 1,500,000개의 튜플들이 존재한다. 본 논문은 암호화된 데이터에 대하여 top-k 질의를 수행하는 데 있어서, 제한된 컴퓨팅 자원을 가진 사용자 시스템에서 수행되어야 할 연산(즉, 암호화된 데이터를 복호화하여 점수 함수값을 계산한 후, 결과 값들을 비교하여 상위 k개의 데이터를 선택하는 작업)을 최소화하는 데 초점을 두고 있다. 그러므로 실험에서는 사용자 시스템에서 처리해야 할 튜플들(즉, 클라우드에서 사용자 시스템으로 전송하는 튜플들)의 수를 측정하였다. 실험에서는 2.2절의 단순 방법(Naive)과 3절에서 제안한 OPT 알고리즘 간의 성능을 비교하였다.

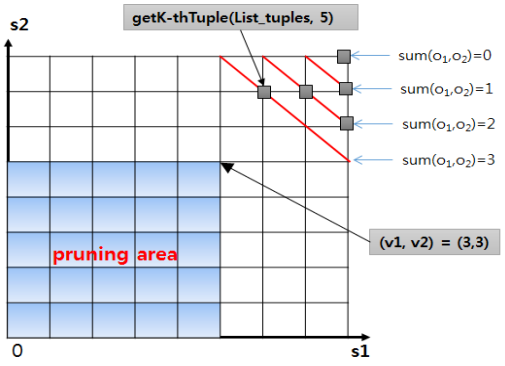
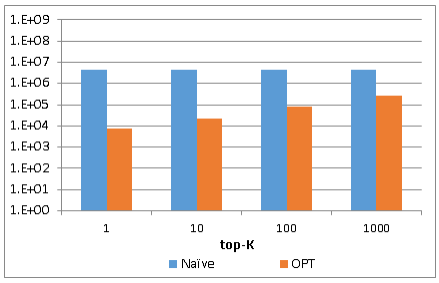


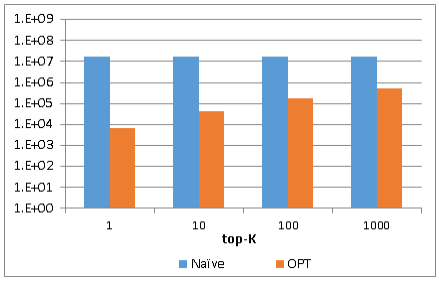
Fig. 4. Computing the Pruning Area for the Example Top-k Query in Section 3.1

Fig. 5은 top-k 질의의 k값을 1에서 1,000으로 변화 시켰을 때 실행 결과를 비교한 결과이다. 실험에서 사용한 질의는 두 릴레이션(LINEITEM, PARTSUPP) 간의 조인으로 구성되어있으며, 조인 선택도(join selectivity, $(R_1 \bowtie R_2)/(R_1 \times R_2)$)가 각각 (a) 9.4×10^{-7}, (b) 3.7×10^{-6}, (c) 7.5×10^{-5}이 되도록 등가 조인 조건을 선택하였다. Fig. 5에서 보는 바와 같이 k값이 감소할수록 사용자 시스템에서 처리해야 할 튜플들의 수가 크게 감소함을 알 수 있다. 이러한 결과가 나타나는 이유는 k값이 감소할수록, pruning 영역이 증가하여 가지치기 가능한 튜플들의 수가 증가하기 때문이다. 특히, k값이 10이하인 경우 본 논문에서 제안한 top-k 질의 처리 방법에 의해, 사용자 시스템에서 처리해야 할 튜플들의 수를 10^{-2}~10^{-4}로 줄일 수 있음을 알 수 있다. 일반적으로 대부분의 응용 프로그램에서 사용하는 k 값은 비교적 작은 값에 해당하므로, 본 논문에서 제안한 top-k 질의 수행 방법을 이용함으로써, 사용자 시스템의 부하를 크게 줄일 수 있다. 반면에 단순 처리 방법은 클라우드에서 중간 결과 튜플들을 가지치기 할 수 없으므로, k 값에 영향을 받지 않는다.

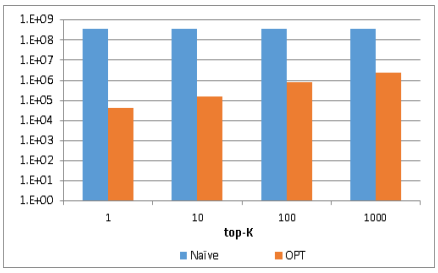
Fig. 6은 세 릴레이션(LINEITEM, PARTSUPP, ORDERS) 간의 조인을 요구하는 top-k 질의에 대하여 k 값을 1에서 1,000으로 변화시켰을 때 실행 결과를 비교한다. 그림에서 알 수 있듯이 단순 방법과 비교하여 본 논문에서 제안한 OPT알고리즘은 클라이언트 시스템에서 처리해야 할 튜플들의 수를 10^{-1}~10^{-2}로 줄일 수 있음을 알 수 있다. 특히, k값이 10 이하인 경우 본 논문에서 제안한 top-k 질의 처리 방법에 의해, 사용자 시스템에서 처리해야 할 튜플들의 수를 10^{-2}로 크게 줄일 수 있음을 알 수 있다.



(a) Join Selectivity = 9.4×10^{-7}



(b) Join Selectivity = 3.7×10^{-6}



(c) Join Selectivity = 7.5×10^{-5}

Fig. 5. The Number of Tuples to be Processed by the Client System (2-way join)

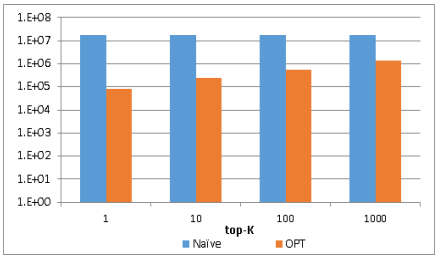


Fig. 6. The Number of Tuples to be Processed by the Client System (3-way join)

5. 결 론

본 논문은 클라우드에 아웃소싱된 암호화된 데이터에 대한 효율적인 top-k 질의 처리 기법을 제안한다. 또한, 본 논문은 제안된 기법에 대해 TPC-H 벤치마크 데이터를 이용한 성능 평가를 수행하였다. 본 논문이 제안하는 top-k 질의 수행 기법은 순서보존 암호화 기법을 이용하여 top-k 질의 결과에 포함되지 않을 것으로 예상되는 중간 결과들을 클라우드에서 가지치기함으로써, 사용자 시스템의 부하를 최소화할 수 있다. 본 논문의 실험 결과는 제안된 top-k 질의 처리 알고리즘이 단순 방법과 비교하여 사용자 시스템의 부하를 크게 줄일 수 있음을 증명한다.

References

[1] R. Fagin, "Combining Fuzzy Information from Multiple Systems," *Proceedings of the 15th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pp.216-226, 1996.

[2] R. Fagin, A. Lotem, and M. Naor, "Optimal Aggregation Algorithms for Middleware," *Proceedings of the 21th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, pp.102-113, 2001.

[3] J. W. Kim and K. S. Candna, "Skip-and-prune: Cosine-based Top-k Query Processing for Efficient Context-sensitive Document Retrieval," *Proceedings of the 2009 ACM SIGMOD International Conference on Management of Data*, pp.115-126, 2009.

[4] S. Tu, M. F. Kaashoek, S. Madden, and N. Zeldovich, "Answering Aggregation Queries in a Secure System Model," *Proceedings of the 33th International Conference on Very Large Data Bases*, pp.519-530, 2007.

[5] C. Doulkeridis and K. Norvag, "Processing Analytical Queries over Encrypted Data in MapReduce," *Proceedings of the VLDB Endowment*, pp.289-300, 2013.

[6] W. K. Wongm, B. Kao, D. W. L. Cheung, R. Li, and S. M. Yiu, "Secure Query Processing with Data Interoperability in a Cloud Database Environment," *Proceedings of the 40th ACM SIGMOD International Conference on Management of Data*, pp.1395-1406, 2014.

[7] C. Gentry, "Fully Homomorphic Encryption Using Ideal Lattices," *Proceedings of the 41th Annual ACM Symposium on Theory of Computing*, pp.169-178, 2009.

[8] P. Paillier, "Public-key Cryptosystems Based on Composite Degree Residuosity Classes," *Proceedings of the 17th International Conference on Theory and Application of Cryptographic Techniques*, pp.223-238, 1999.

[9] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu, "Order Preserving Encryption for Numeric Data," *Proceedings of the 30th ACM SIGMOD International Conference on Management of Data*, pp.563-574, 2004.

[10] A. Boldyreva, N. Chenette, Y. Lee, and A. O'Neill, "Order-preserving Symmetric Encryption," *Proceedings of the 28th EUROCRYPT*, pp.224-241, 2009.

[11] TPC-H [Internet], http://www.tpc.org/tpch.

김 종 욱

e-mail : jkim@smu.ac.kr
2000년 고려대학교 전산과학과(학사)
2002년 한국과학기술원 전산학과(석사)
2009년 Arizona State University Computer Science(박사)
2010년~2013년 Teradata, Software Engineer
2013년~현 재 상명대학교 미디어소프트웨어학과 조교수
관심분야 : Database Systems, Data Mining

서 영 균

e-mail : yksuh@kisti.re.kr
2003년 경북대학교 컴퓨터과학과(학사)
2005년 한국과학기술원 전산학과(석사)
2015년 Dept. of Computer Science, University of Arizona(박사)
2005년~현 재 한국과학기술정보연구원 국가슈퍼컴퓨팅연구소 연구원
2008년~2015년 Research Associate, University of Arizona
관심분야 : Database Design, Database Systems, Database Ergalics