Journal of Korea Multimedia Society Vol. 18, No. 8, August 2015(pp. 915-924)
http://dx.doi.org/10.9717/kmms.2015.18.8.915

클라우드 환경에서의 암호화 데이터에 대한 효율적인 Top-K 질의 수행 기법

김 종 욱†

Efficient Top-K Queries Computation for Encrypted Data in the Cloud

Jong Wook Kim†

ABSTRACT

With growing popularity of cloud computing services, users can more easily manage massive amount of data by outsourcing them to the cloud, or more efficiently analyse large amount of data by leveraging IT infrastructure provided by the cloud. This, however, brings the security concerns of sensitive data. To provide data security, it is essential to encrypt sensitive data before uploading it to cloud computing services. Although data encryption helps provide data security, it negatively affects the performance of massive data analytics because it forbids the use of index and mathematical operation on encrypted data. Thus, in this paper, we propose a novel algorithm which enables to efficiently process a large amount of encrypted data. In particular, we propose a novel top-k processing algorithm on the massive amount of encrypted data in the cloud computing environments, and verify the performance of the proposed approach with real data experiments.

Key words: Cloud Computing, Encryption, Top-k Query

1. 서 론

최근 국내외에서 클라우드 컴퓨팅은 주목해야 할 차세대 IT 트렌드 중에 하나로 선정되고 있다. 클라우드 컴퓨팅은 개인이나 기업에서 필요로 하는 IT 자원을 별도로 구입하지 않고, 클라우드 컴퓨팅 업체가 제공하는 IT 자원을 필요한 만큼 빌려 쓰는 형태의 서비스를 의미 한다. 개인이나 기업은 클라우드에 대용량의 데이터를 아웃소싱(outsourcing)하여 경제적으로 데이터를 저장 및 관리할 수 있으며, 클라우드에서 제공해주는 IT자원을 이용하여 대용량의 데이터를 효율적으로 분석할 수 있다.

클라우드 서비스를 이용함으로써 사용자들은 자체적인 IT 인프라 구축 및 운영에 드는 비용 절감의 효과를 얻을 수 있다. 그러나 이러한 장점에도 불구하고 사용자의 민감한 데이터(sensitive data)가 클라우드 서비스 제공자(cloud services provider)에게 노출될 위험이 항상 존재한다. 그러므로 사용자의 데이터를 보호하기 위하여, 클라우드 서비스를 이용하기 전에 민감한 데이터에 대하여 암호화를 수행하는 것이 필수적이며, 이에 관한 연구가 활발하게 진행되고 있다[1,2,3].

최근 사회 전반에 걸쳐 빅데이터 활용 붐이 일면서, 빅데이터 안에 숨어 있는 가치를 찾기 위한 빅데이터 분석 기법이 다양한 응용분야에서 연구되고 있다. 특히, 지속적으로 증가하는 빅데이터로부터 사용

※ Corresponding Author : Jong Wook Kim, Address: (110-743) 20-Gil, Hongji-dong, Jongno-gu, Seoul, Korea, TEL : +82-2-781-7590, FAX : +82-2-781-7590, E-mail : jkim@smu.ac.kr
Receipt date : Feb. 26, 2015, Revision date : May 21, 2015
Approval date : June 17, 2015
† Department of Media Software, Sangmyung University
※ This research was supported by a 2015 Research Grant from Sangmyung University.

자의 요구 사항과 연관성이 높은 몇 개의 결과를 검색 하는 top-k 질의는 빅데이터 분석에 많이 활용되고 있는 질의중의 하나이다. Top-k 질의에서 사용자의 요구사항은 점수 함수(score function)로 표현되며, 이러한 점수 함수들은 단조 함수(monotonic function)라는 특징이 있다. Top-k 질의는 사용자의 점수 함수를 이용하여, 최상위 (또는 최하위) 점수를 나타내는 k개의 데이터를 사용자에게 반환 한다. 클라우드 환경에서 top-k 질의를 수행하기 위한 기존 연구가 존재한다[4,5]. 대표적으로 Candan et al.이 제안한 Rankloud는 맵리듀스 기법과 데이터 영역의 유용성을 활용하여, top-k 질의 수행 시 wasted-work을 최소화하기 위한 알고리즘을 제안하였다[4]. 또한, [5]는 맵리듀스를 활용하여 top-k 질의를 효율적으로 수행하기 위한 데이터 분할 방식을 제안하였다. 그러나 기존 방식들은 암호화 기법이 적용되지 않은 평문 데이터(plain data)에 대하여 top-k 질의를 수행하기 때문에, 사용자의 민감한 데이터를 보호할 수 없다는 문제점이 있다. [12], [13]에서는 암호화된 데이터에 대하여 효율적으로 top-k 질의를 수행하기 위한 방법을 제안 하였다. 그러나 [12], [13]에서 제안한 방법들은 순서보존 암호화 기법(order-preserving encryption)을 사용하고 있으며, 이로 인하여 원본 데이터들의 크기순서가 외부에 노출될 위험이 존재한다. 그러므로 본 연구는 클라우드 서비스를 이용하여 암호화된 사용자의 민감한 데이터에 대하여, top-k 질의를 효율적으로 처리하기 위한 방법을 제안한다. 특히, 본 연구에서 제안한 방식은 준동형 암호화 기법을 사용하고 있으며, 이로 인하여 외부에 민감한 데이터에 대한 어떤 정보도 노출되지 않는다는 장점을 가지고 있다.

본 논문은 다음과 같이 구성되어 있다. 2장에서는 논문에서 다루는 문제와 배경 지식에 대하여 설명한다. 3장에서는 논문에서 제시하는 알고리즘을 설명한 후, 4장에서 실험을 통하여 제안한 방식의 성능 평가를 수행한다. 마지막으로, 5장에서는 결론과 향후 연구 과제들을 논하다.

2. 문제 정의 및 배경 지식

2.1 문제 정의

본 장에서는 논문에서 다루는 문제를 정의 한다.

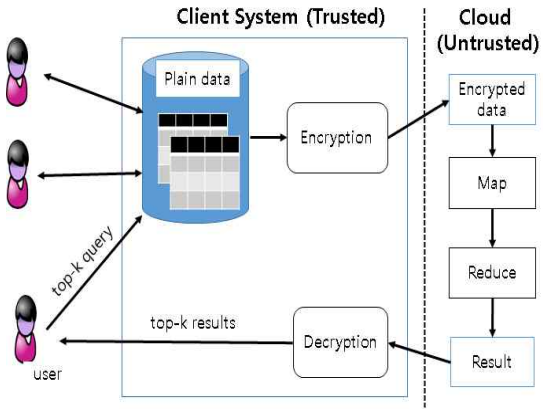

Fig. 1. A system architecture used in this paper.

본 연구에서 가정하는 시스템 구조는 Fig. 1과 같다. 대용량의 데이터들은 신뢰(trusted)할 수 있는 클라이언트 시스템에 저장되어 있다. 클라이언트 시스템은 어플리케이션 제공자가 소유하고 있는 서버에 해당한다. 클라이언트 시스템은 컴퓨팅 자원이 제한적이기 때문에, top-k 질의와 같이 많은 컴퓨팅 자원을 요구하는 사용자 질의의 경우 클라우드 서비스를 이용하여 수행한다고 가정하자. 즉, 클라우드 서비스를 사용하는 주된 목적은 데이터를 클라우드에 아웃소싱하기 보다는, 대용량 데이터를 클라우드 서비스를 이용하여 효율적으로 분석하기 위함이다. 클라이언트 시스템에 저장된 대용량의 데이터들은 평문 형태로 데이터베이스의 릴레이션(relation)에 저장 되어 있다. 이때 $R = \{R_1, R_2, ... R_m\}$를 데이터베이스에 존재하는 모든 릴레이션들의 집합이라고 가정하자. 설명의 편의성을 위해, 각각의 릴레이션 $R_p \in R$는 기본 키(ID_p), 한 개의 민감한 데이터 컬럼(s_p), 민감하지 않은 데이터(non-sensitive data) 컬럼들($n_{p,1}, n_{p,2}, ...,n_{p,l}$)로 구성되어 있다고 가정하자[1]. 즉, 릴레이션 R_p는 다음과 같이 나타낼 수 있다.

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

이때, 사용자는 다음과 같은 top-k 질의를 요청한다고 가정하자.

SELECT select-list
FROM $R_1, R_2, ... R_m$

1) 본 논문에서 제안한 알고리즘은 릴레이션에 여러 개의 민감한 데이터 컬럼들이 존재하는 경우에도 동일하게 적용된다.

WHERE equi-join-expression(ID_1, ID_2, ..., ID_m)
ORDER BY $f(s_1, s_2, s_3, \ldots s_m)$
STOP AFTER k

$f(s_1, s_2, s_3, \ldots s_m)$는 사용자가 정의한 단조 점수 함수(monotonic score function)에 해당하며, 사용자는 최상위 점수를 가지는 k개의 결과에 관심이 있다. 클라우드 서비스를 이용하여 top-k 질의를 수행하기 위하여, 클라이언트 시스템에 저장된 데이터는 사용자 질의와 함께 클라우드로 전송이 된다. 이때, 사용자의 민감한 데이터가 클라우드 서비스 제공자에게 노출되는 것을 방지하기 위하여, 다음과 같이 암호화를 적용한다.

$$R_p(ID_p, s^e_p, n_{p,1}, n_{p,2}, \ldots, n_{p,l})$$

컬럼 s_p에 해당하는 값들은 암호화(s^e_p)를 적용한 후 클라우드에 전송함으로써, 민감한 데이터를 클라우드 서비스 제공자로부터 보호할 수 있다.

클라우드에서는 암호화된 데이터에 대하여 맵리듀스[14]를 이용하여 top-k 질의를 수행한 후, 결과를 다시 클라이언트 시스템에게 반환한다. 클라우드에서 반환된 결과들은 암호화 되어 있기 때문에, 이를 먼저 복호화한 후, 사용자에게 top-k 결과를 반환한다.

2.2 배경 지식: 동형 암호 기법

클라우드 컴퓨팅 환경에서 데이터 암호화 기법을 사용하는 주된 이유는 사용자의 민감한 데이터를 클라우드 서비스 제공자로부터 보호하기 위해서이다. 일반적으로 암호화된 데이터에 대하여 연산(예, 덧셈, 곱셈)을 직접 수행하는 것은 불가능하기 때문에, 먼저 복호화를 수행한 후, 평문 데이터에 대하여 연산을 적용해야 한다. 그러나 복호화 과정을 클라우드에서 수행하면, 민감함 데이터가 클라우드 서비스 제공자에게 노출 될 수 있다는 문제점을 안고 있으므로, 복호화 과정은 반드시 신뢰할 수 있는 클라이언트 시스템에서 진행되어야 한다.

최근 들어, 암호화된 데이터 상태에서 원하는 횟수만큼 연산 수행을 가능케 하는 완전 동형 암호화 기법(fully homomorphic encryption)이 개발되었다[6]. 즉, 덧셈과 곱셈 연산을 보존하는 완전 동형 암호화는 다음과 같은 식으로 표현 가능하다.

$$E(x) + E(y) = E(x+y),\ E(x) \cdot E(y) = E(x \cdot y) \quad (1)$$

완전 동형 암호화 기법은 데이터의 안정성을 보장하면서, 복호화 없이 다양한 연산을 수행 할 수 있다는 장점이 있지만, 실제 응용 프로그램에 적용하기 위해서는 먼저 효율성 문제가 해결되어야 한다.

완전 동형 암호화 기법과는 달리 준동형 암호화 기법(partially homomorphic encryption)은 암호화된 데이터 상태에서 제한된 횟수의 특정 연산 수행을 가능하게 하는 방식이다. 준동형 암호화 기법은 빠른 속도의 암호화-복호화로 인하여, 데이테베이스 암호화에 많이 활용되고 있다[7,8]. 본 논문에서는 덧셈 연산을 보존하기 위해 Paillier [9] 암호체계 기반의 준동형 암호화 기법과 곱셈 연산을 보존하기 위해 ElGamal 암호체계 기반 준동형 암호화 기법을 사용한다.

3. 클라우드에서의 TOP-K 질의 처리

3.1장에서는 준동형 암호화 기법을 이용하여 클라우드 환경에서 top-k 질의를 처리하는 단순 방법을 제안한다. 그리고 3.2장에서는 3.1장의 단순 방법의 질의 처리 성능을 개선하기 위한 방법을 제안한다.

3.1 단순 방법: 준동형 암호화 기법 기반 단순 top-k 질의 수행 방법

클라우드에서 준동형 암호화 기법을 이용한 top-k 질의는 Fig. 2와 같이 처리할 수 있다. Fig. 2에서 사용자는 다음과 같은 top-k 질의를 요청했다고 가정하자.

SELECT ID_1, $n_{1,1}$, $n_{2,1}$
FROM R_1, R_2
WHERE $R_1.ID_1 = R_2.ID_2$
ORDER BY $2 \times s_1 + s_2$
STOP AFTER 3

그림에서 사용한 top-k 질의는 설명의 편의성을 위해서 two-way 조인 질의를 사용하였다. 그러나 본 논문의 알고리즘은 일반적인 m-way 조인 질의에 대하여 적용 가능하다. Fig. 2의 top-k 질의의 점수 함수는 가중치가 있는 합계(weighted sum)에 해당하므로, 덧셈 보존 준동형 암호화 기법을 사용하여, 민감한 데이터 컬럼(s_1, s_2)을 각각 암호화 한다(Fig. 2-(a)). 이때, 상수 곱(즉, $2 \times s_1$)은 준동형 암호화 기법을 적용하기 이전에 수행함으로써, 암호화된 데이

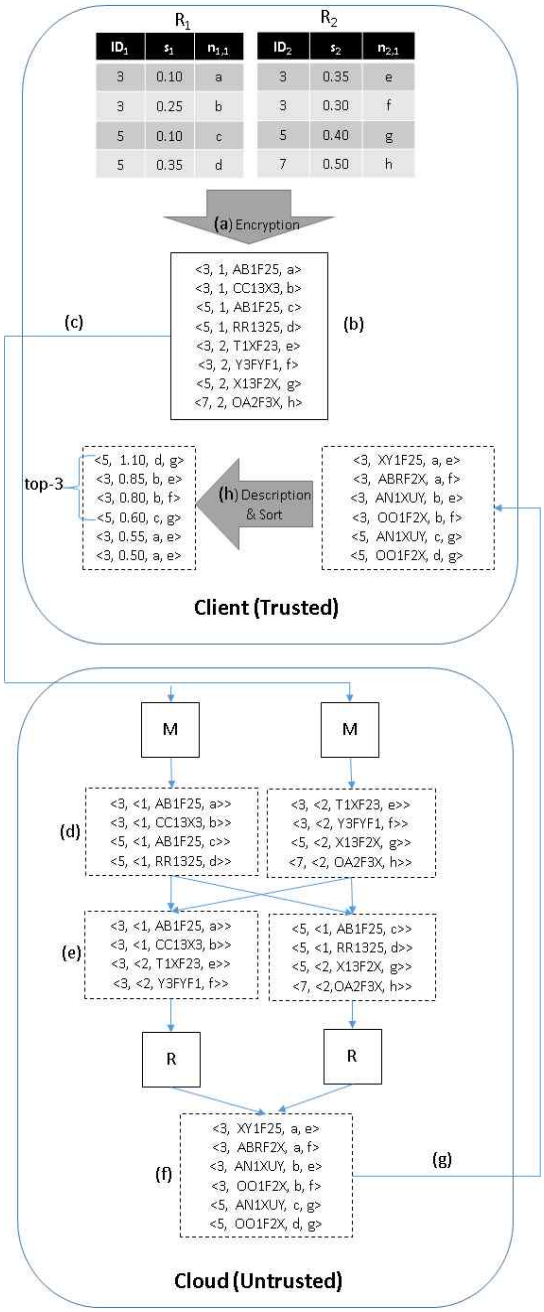

Fig. 2 Naive algorithm for processing top-k queries.

터에는 덧셈 연산만 수행하도록 한다. 만일, top-k 질의의 점수 함수가 곱셈 연산으로만 구성되어 있으면, 곱셈 보존 준동형 암호화 기법을 사용한다. 또한, 3.3장에서는 덧셈과 곱셈 연산을 모두 사용하는 top-k 질의 수행 방법에 관하여 논의한다. 클라이언트 시스템에서 클라우드로 전송할 데이터는 암호화 기법을 수행한 후 다음과 같이 구성된다(Fig. 2-(b)).

〈ID 값, 릴레이션 번호, 암호화된 값 (민감한 데이터), 암호화되지 않은 값 (그 외의 데이터)〉

예를 들면, Fig. 2에서 릴레이션 R_2에 존재하는 튜플 (ID_2, s_2, $n_{2,1}$) = (7, 0.50, h)는 준동형 암호화를 적용한 후, 다음과 같이 변환된다.

〈7, 2, OA2F3X, h〉

이때, OA2F3X는 0.50의 암호화된 값에 해당한다고 가정하자. 클라우드 서비스를 이용하여 top-k 질의를 수행하기 위하여 암호화된 데이터는 클라우드로 전송 된다(Fig. 2-(c)).

등가 조인(equi join)을 수행하기 위하여, 클라우드의 맵 함수는 다음과 같은 key-value로 구성된 중간 단계의 결과를 생산한다(Fig. 2-(d)).

〈ID 값, 〈릴레이션 번호, 암호화된 값, 암호화되지 않은 값〉 〉

즉, key는 ID 값에 해당하며, value는 〈릴레이션 번호, 암호화된 값, 암호화되지 않은 값〉으로 구성되어 있다. 맵리듀스에서는 내부적으로 정렬이 수행되어 같은 key 값을 가지는 데이터들은 같은 리듀스의 입력으로 사용된다(Fig. 2-(e)). 리듀스 함수는 등가 조인 및 암호화된 데이터에 대하여 덧셈 연산을 수행한 후 결과 값을 생성한다(Fig. 2-(f)). 이때, 덧셈 보존 준동형 암호화 기법을 사용하기 때문에, 암호화된 데이터에 직접으로 덧셈연산을 수행할 수 있다. 그러나 준동형 암호화 기법은 암호화된 데이터에 대하여 비교 연산을 지원하지 않으므로, 클라우드 내에서 최상위 점수를 가지는 k개의 데이터를 선택할 수 없다. 그러므로 맵리듀스에 의해 생성된 모든 결과들은 클라이언트 시스템에 전송된다(Fig. 2-(g)). 클라이언트 시스템에서는 전송받은 결과 값을 복호화한 후, 점수 함수 값에 의해 정렬한다(Fig. 2-(h)). 마지막으로, 최상위 값을 가지는 k개의 결과물을 사용자에게 반환한다.

Top-k 질의의 경우 사용자는 최상위 점수를 가지는 k개의 데이터에 관심이 있으며, 일반적으로 k는 비교적 작은 값에 해당한다. 그러나 본 장에서 설명한 질의 처리 기법은 이러한 특성을 활용하지 않기 때문에 다음과 같은 문제점이 존재한다.

- 데이터 통신 오버헤드: 릴레이션에 존재하는 모든 튜플들을 클라이언트 시스템에서 클라우드로 전

송하므로 데이터 통신 오버헤드가 발생한다(Fig. 2-(c)). 또한, 이로 인하여 클라우드의 결과 값을 클라이언트로 전송하는 단계에서도 통신 오버헤드가 발생할 수 있다(Fig. 2-(g)).

- 조인 수행 오버헤드: 맵리듀스에 의해 생성된 수많은 결과들 중에서 k개의 데이터만이 사용자에게 반환된다. 그러므로 클라우드에서는 불필요한 조인(즉, 사용자에게 반환되지 않는 데이터를 생산하는 조인)을 수행하게 되며, 이는 IT 자원을 낭비하는 결과를 초래한다.(Fig. 2-(d), (e), (f)). 대부분의 클라우드 서비스는 IT 자원을 사용하여 사용량에 따라 지불(Pay-As-You-Go)하는 방식을 택하고 있으므로 이는 사용자의 비용을 증가시키는 부정적 영향을 초래한다.

3.2 조인 선택도를 이용한 top-k 질의 성능 개선 방법

본 장에서는 3.1장의 알고리즘을 확장하여 클라우드 서비스를 이용하여 민감한 데이터에 대하여 top-k 질의를 효율적으로 처리하기 위한 방법을 제안한다. 먼저, 3.1장에서 사용한 top-k 질의를 예로 들어보자. Fig. 3에서 x축은 s_1, y축은 s_2를 나타낸다. 그림에서 작은 사각형들은 R_1과 R_2의 등가 조인 결과들에 해당한다. 이때, top-k 질의의 최종 결과는 점수 함수가 단조함수에 해당하므로 사각형의 네 모서리 중 한곳에 위치하게 되며, 3.1장의 top-k 질의의 경우 오른쪽 상단 모서리에 top-k 질의의 결과들이 위치하게 된다. 그러므로 s_1, s_2의 점수 분포도 및 점수 함수의 특성을 이용하여, top-k 질의의 결과들이 위치할게 될 영역(Fig. 3의 점선 사각형에 해당함)을 추정할 있으며, 이 영역 안에 존재하는 데이터들만을 사용하여 top-k 질의를 수행할 수 있다. 이 경우, 전체 데이터 영역 중 아주 작은 영역만을 사용하기 때문에, 앞 장에서 언급한 오버헤드 문제를 크게 해소할 수 있다. 본 논문에서 제안하는 알고리즘은 (a) 조인 선택도 계산, (b) 후보 영역 선택, (c) top-k 질의 처리로 구성된다.

3.2.1 조인 선택도(Join Selectivity) 계산

본 연구에서는 카운팅 블룸 필터(counting bloom filter)을 이용하여 특정 데이터 영역에 대한 조인 선택도를 계산한 후, 이를 top-k 질의 처리에 활용한다.

릴레이션 $R_p(ID_p, s_p, n_{p,1}, n_{p,2}, ...,n_{p,l})$의 컬럼 s_p가 가질 수 있는 값의 전체 영역을 v개의 균등한 배타적 영역으로 나눈 후, 각각의 영역에 대하여 하여 카운팅 블룸 필터(f^p)를 생성한다. 이때, 컬럼 ID_p가 가질 수 있는 고유한 값(distinct value)의 개수를 w라 가정하면, 카운팅 블룸 필터의 길이를 w로 설정함으로써, 조인키 값들과 블룸필터의 비트들 사이에 일대일 매핑이 형성 되도록 한다. 이때, h(•)를 조인키 값들과 블룸필터의 비트들 사이에 일대일 매핑함수(one-to-one mapping function)라 가정하자. 예를 들면, Fig. 2에서, 컬럼 s_1과 s_2의 최소값과 최대값을 각각 0과 1이라고 가정하자. 또한, s_1과 s_2를 5개의 균등한 배타적 영역으로 분할한다고 가정하자. 이때, 릴레이션 R_1의 2번째 영역 $0.2<s_1\leq0.4$에는 두 개의 튜플(즉, 2번째, 4번째)이 존재하며, 이 영역의 카운팅 블룸 필터($f^1_{0.2<s1\leq0.4}$)는 매핑 함수(h)를 사용하여 Fig.

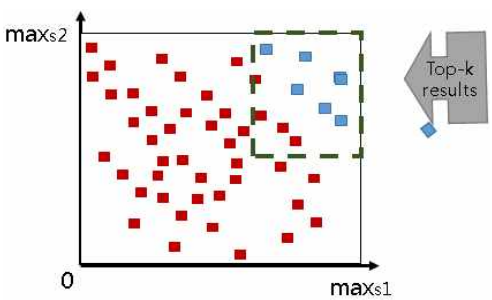

Fig. 3 Total Data Region vs. the results of top-k query.

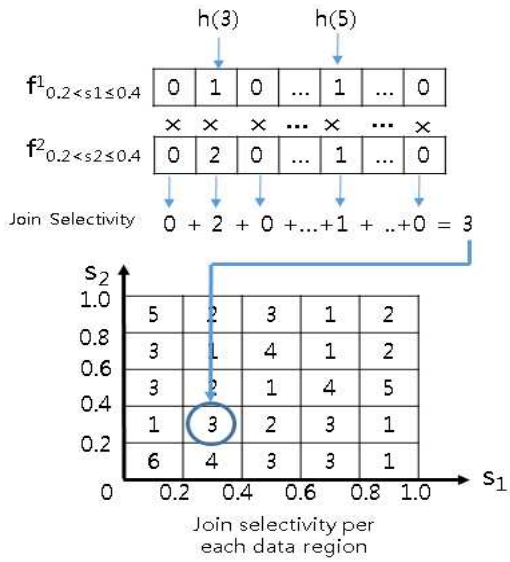

Fig. 4. Join selectivity computation.

4과 같이 생성할 수 있다. 유사하게 릴레이션 R_2의 두 번째 영역 $0.2<s_2\leq 0.4$에 대한 카운팅 블룸 필터를 생성할 수 있다. 이때, 교차 영역(Fig. 4의 원에 해당하는 영역)의 조인 선택도는 다음과 같이 비트 곱의 합으로 구할 수 있다.

$$\sum_{i=1}^{w}(f^1_{0.2<s1\leq 0.4}[i]\times f^2_{0.2<s2\leq 0.4}[i]). \quad (2)$$

이때, $f^1_{0.2<s1\leq 0.4}[i]$는 $f^1_{0.2<s1\leq 0.4}$의 i-번째 비트의 값에 해당한다($f^2_{0.2<s2\leq 0.4}[i]$도 유사하게 정의된다).

일반적인 2-way 조인의 조인 선택도는 다음과 같이 계산할 수 있다. 릴레이션 R_1의 a-번째 영역에 해당하는 카운팅 블룸 필터를 f^1_a, 릴레이션 R_2의 b-번째 영역에 해당하는 카운팅 블룸 필터를 f^2_b라 가정하자. 이때, R_1의 a번째 영역과 R_2의 b번째 영역이 교차하는 영역의 조인 선택도는 다음과 같이 계산 가능하다[2].

$$JS_{a,b}=\sum_{i=1}^{w}(f^1_a[i]\times f^2_b[i]). \quad (3)$$

Pseudo-code for selecting the candidate regions
Input: Data regions T_1, T_2,.. T_q,
the corresponding join selectivity, JS_{T1}, JS_{T2}, ..., JS_{Tq}, and
k (top-k)
Output: A set of candidate data regions, SetCAND

```
1: SetCAND ← Φ
2: minscore ← 0
3: count ← 0
4: List_T_Max ← SortByMaxScore(T_1, T_2,.. T_q)
5: for each T_cur of List_T_Max
6:     count ← count + JS_Tcur
7:     add T_cur to SetCAND
8:     if (count ≥ k)
9:         minscore ← possible minimum score of
           T_cur
10:    break
11: for each T_cur of (T_1, T_2,.. T_q)
12:    curcore ← possible maximum score of T_cur
13:    if (T_cur ∉ SetCAND and curscore ≥ minscore)
14:        add T_cur to SetCAND
15: return SetCAND
```

Algorithm 1. Pseudo-code for the candidate region selection.

2) M-way 조인인 경우 유사하게 확장된다.

Fig. 5에서와 같이 교차 영역별 조인 선택도는 클라우드 서비스의 맵리듀스를 이용하여 효율적으로 계산할 수 있다. 카운팅 블룸 필터는 데이터 크기를 줄이기 위하여 run-length codes를 이용하여 압축된 상태로 클라이언트 시스템에 저장되어 있다. 클라우드에서는 카운팅 블룸 필터를 이용하여 데이터 영역별 조인 선택도를 계산한 후(Fig. 5-(b)), 결과를 클라이언트 시스템에 반환 한다(Fig. 5-(c)). 조인 선택도 계산 단계에서는 단지 카운팅 블룸 필터만이 클라이언트 시스템에서 클라우드로 전송되므로, 이 단계에서는 사용자의 민감한 데이터가 클라우드 서비스 제공자에게 노출될 위험은 존재하지 않는다.

3.2.2 후보 영역(Candidate Region) 선택

앞 장에서 계산한 교차 영역별 조인 선택도를 이용하여, 본 장에서는 top-k 결과가 존재할 것으로 예상되는 후보 영역을 알고리즘 1을 이용하여 구한다(Fig. 5의 (d)에 해당함). 알고리즘 1의 입력은 교차 영역(T_r)과 그에 해당하는 조인 선택도(JS_{Tr}), 사용자 지정 k 값으로 구성되고, 출력은 후보 교차 영역들의 집합(SetCand)에 해당한다. 1번에서 후보 교차 영역의 집합은 공집합으로 초기화 된다. 4번과 5번에서 사용자의 top-k 질의 점수 함수에 대하여 각각의 교차 영역이 가질 수 있는 최대값을 구한 후, 이 값에 의해 내림차순으로 정렬된 리스트($List\_T_{Max}$)를 구한다. 5번부터 10번에서는 리스트 $List_T_{Max}$를 k개의 튜플이 발견될 때까지 순차적으로 방문하면서, 이미 방문한 교차 영역은 SetCAND에 추가 한다. 또한, 9번에서 minscore는 마지막에 방문한 교차 영역이 가질 수 있는 최소값으로 설정한다. 5번부터 10번에 의해 SetCAND에 존재하는 교차 영역에서 이미 k개의 튜플이 발견되었기 때문에, 이러한 k개의 튜플들보다 큰 점수를 가질 수 있는 교차 영역들만을 SetCAND에 추가 하면 된다. 이미 발견된 k개의 튜플이 가질 수 있는 최소값은 minscore에 해당한다. 그러므로 SetCAND에 존재하지 않는 특정 교차 영역이 가질 수 있는 최대값이 minsocre부터 크면, 이 교차 영역에서 존재하는 튜플은 이미 발견한 k개의 튜플보다 더 큰 점수를 가질 수 있다. 이러한 특성을 이용하여 11번부터 14번에서는 이미 발견한 k개의 튜플보다 더 큰 점수를 가질 수 있는 튜플이 존재하는 교차 영역을 SetCAND에 추가 한다. 마지막으로,

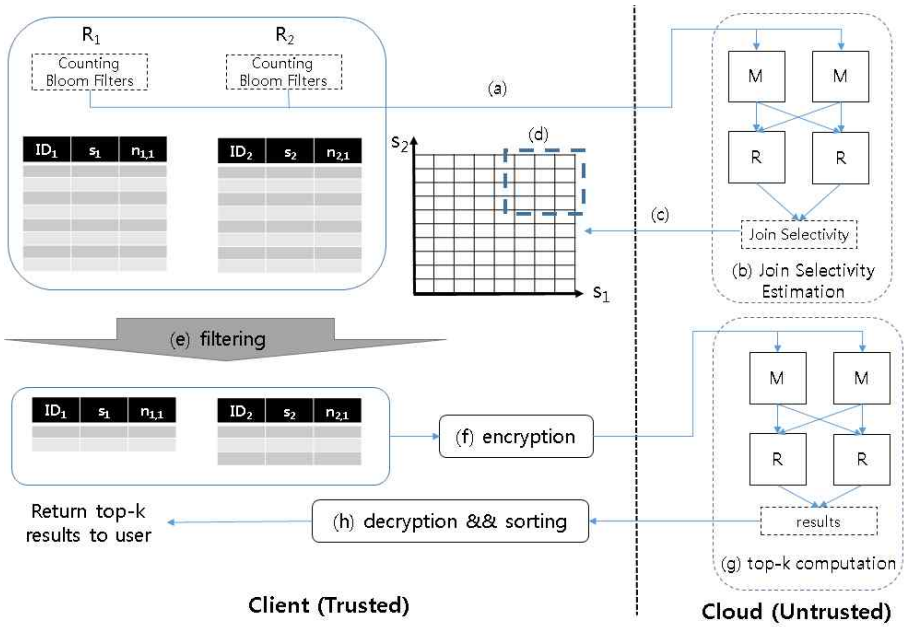

Fig. 5 Proposed algorithm for processing top-k queries.

15번에서 알고리즘 1은 SetCAND를 반환한다.

3.2.3 후보 영역 기반 top-k 질의 처리

마지막 단계는 앞 장에서 선택한 후보 영역을 이용하여 클라우드에서 top-k 질의를 처리하는 것이다. 단순 방법과는 달리 후보 영역에 존재하는 튜플들을 SQL의 SELECT문을 이용하여 추출한다 (Fig. 5-(e)). 추출된 튜플들을 이용한 top-k 질의 처리는 3장의 단순방법과 유사하다. 먼저, 추출된 튜플들의 민감한 데이터에 대하여 암호화를 수행한 후, 암호화된 데이터를 클라우드에 전송한다(Fig. 5-(f)). 클라우드에서는 맵리듀스를 이용하여 등가 조인과 암호화된 데이터에 대하여 연산을 수행(Fig. 5-(g))한 후 결과를 클라이언트 시스템에 반환한다. 클라이언트 시스템에서는 전송된 결과를 복호화한 후, 최상위 k개의 결과를 사용자에게 반환한다(Fig. 5-(h)). 본 논문에서 제안하는 방식(Fig. 5)은 단순 방식(Fig. 2)에 비해, 후보 영역만을 이용하여 top-k 질의를 수행하므로 데이터 통신 오버헤드 및 조인 수행 오버헤드를 크게 줄일 수 있다.

본 논문에서 제안한 top-k 질의 수행 방법은 준동형 암호화 기법을 사용하므로, 순서 보존 암호화 기법을 사용하는 기존 알고리즘과는 달리 원본 데이터의 어떤 정보도 클라우드 서비스 제공자에게 노출되지 않는다 [7].

3.2.4 서로 다른 연산으로 구성된 점수 함수

앞 장에서 다룬 top-k 질의의 점수 함수는 클라우드에서 한 번의 연산(즉, 곱셈 혹은 덧셈)을 수행함으로써 계산 가능한 경우에 해당한다. 이로 인해 민감한 데이터에 대하여 한 번의 준동형 암호화(즉, 덧셈 보존 혹은 곱셈 보존 암호화) 기법을 적용하여 top-k 질의를 수행하는 것이 가능하였다. 그러나 top-k 질의 점수 함수가 서로 다른 연산으로 구성되어 있는 경우 민감한 데이터에 대하여 여러 번의 암호화와 복호화 과정이 필수적으로 요구된다. 다음과 같은 사용자 질의를 예로 들어보자.

SELECT $ID_1, n_{1,1}, n_{2,1}, n_{3,1}$
FROM R_1, R_2, R_3
WHERE $R_1.ID_1 = R_2.ID_2$ AND $R_2.ID_3 = R_3.ID_3$
ORDER BY $2 \times s_1 + s_2 \times s_3$
STOP AFTER 3

위의 top-k 질의의 점수 함수는 덧셈과 곱셈연산을 모두 요구하므로, 두 번의 준동형 암호화 기법을 적용하여 다음과 같이 수행 가능하다.

- 후보 영역 선택: 3.2.1장과 3.2.2장에서 제안한 방

법을 사용하여 후보 영역을 선택한다. Top-k질의가 3개 릴레이션의 등가 조인을 요구하므로, 후보 영역은 육면체 모서리 중 한곳에 해당한다.

- 곱셈 보전 준동형 암호화 적용: 릴레이션 R_2와 R_3의 후보 영역에 속하는 튜플들에 대하여, 곱셈 보존 준동형 암호화를 수행한 후, 3.2.3장에서 설명한 방법을 이용하여 R_2와 R_3의 등가 조인과 ($s_2 \times s_3$) 연산을 수행한다. 등가 조인과 연산 수행 결과를 클라이언트 시스템에 전송한 후, 암호화된 데이터를 복호화 한다. 이때, ($s_2 \times s_3$)의 결과를 $result_{23}$이라 가정하자.
- 덧셈 보전 준동형 암호화 적용: 릴레이션 R_1의 후보 영역에 속하는 튜플들과 전 단계에서 결과물에 덧셈 보전 준동형 암호화를 적용한 후, 3.2.3장에서 설명한 방법을 이용하여 R_1과 R_2의 등가 조인과 (s_1 + $result_{23}$) 연산을 수행한다.

Top-k 질의 점수 함수가 서로 다른 연산 수행을 요구하는 경우 위의 예에서 볼 수 있듯이, 여러 번의 암호화와 복호화 과정을 거쳐야 한다. 또한, 이로 인하여 클라이언트 시스템과 클라우드 사이의 데이터 전송도 여러 번 요구 되는 것을 알 수 있다. 그러나 본 논문에서 제안한 방법은 후보 영역만을 이용하여 top-k 질의를 수행하므로, 여러 번의 데이터 암호화와 복호화 및 데이터 전송으로 인해 발행하는 오버헤드를 최소화할 수 있다.

4. 실 험

이 절에서는 제안하는 알고리즘의 성능을 실험적으로 평가한다. 실험에서는 TPC-H 벤치마크[10]의 데이터베이스 스키마를 사용하였으며, 특히 다음과 같은 형태의 top-k 질의를 사용하였다.

```
SELECT   P.SUPPKEY, AVERAGE
         (P.SUPPLYCOST,
         L.EXTENDEDPRICE)
FROM     PARTSUPP as P, LINEITEM as L
WHERE    P.SUPPKEY = L.SUPPKEY
ORDER BY  AVERAGE (P.SUPPLYCOST,
                   L.EXTENDEDPRICE)
STOP AFTER  k
```

이때, 컬럼 SUPPLYCOST와 EXTENDEDPRICE는 사용자의 민감한 데이터에 해당한다고 가정하였다.본 논문에서 제안한 방법의 효율성을 실험하기 위해 Amazon EC2 [11]를 사용하였으며, 실험에서는 6개의 노드로 구성된 하둡 클러스터를 사용하였다.

4.1 실험 결과

Fig. 6는 릴레이션 LINEITEM의 튜플 수를 1M에서 5M으로 변화 시켰을 때 실행 결과를 비교한 결과이다. 실험에서 릴레이션 PARTSUPP의 튜플 수는 0.8M으로 고정되었으며, top-k의 k값은 10으로 설정하였다. 그림에서 보이는 바와 같이, LINEITEM의 튜플 수가 증가 할수록, 3.1장의 단순방법(Naive)에 의한 top-k 질의 수행 시간은 점차 증가하는 것을 알 수 있다. 이와 달리, 3.2장에서 제안한 top-k 질의 수행 기법(Proposed)은 튜플 수의 증가에 크게 영향을 받지 않음을 Fig. 6를 통해 알 수 있다. 이러한 결과가 나타나는 주된 이유는 본 논문에서 제안한 top-k 질의 수행 방법은 전체 데이터 영역 중 일부 영역만을 이용하여 질의를 수행하므로, 단순방법에 비해 릴레이션 튜플 수의 변화에 적게 영향을 받기 때문이다.

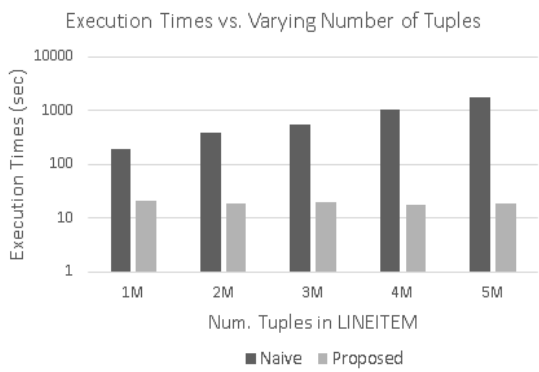

Fig. 6 Performance comparison on varying the number of tuples.

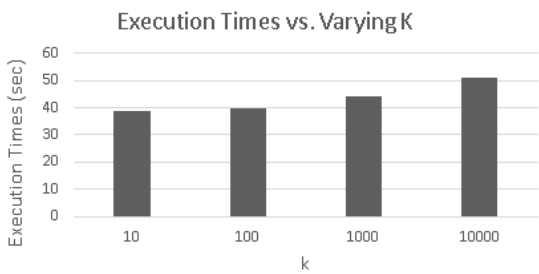

Fig. 7 Execution times on varying k.

Table 1. The amount of data transferred from client systems to cloud (MB)

Num. of Tuples in LINEITEM	1M	2M	3M	4M	5M
Naive	555.3	863.9	1172.4	1480.9	1789.5
Proposed	8.0	8.8	9.5	10.2	11.0

Fig. 7은 top-k 질의의 k 값을 10에서 10,000으로 변화 시켰을 때 3.2장에서 제안한 알고리즘의 실행 결과를 비교한 결과이다. 실험에서 릴레이션 PARTSUPP의 튜플 수와 릴레이션 LINEITEM의 튜플 수는 각각 0.8M과 5M으로 고정되다. 그림에서 보는 바와 같이 k값이 증가할수록 top-k 질의의 수행 시간이 약간 증가함을 알 수 있다. 이러한 결과가 나타나는 주된 이유는 k값이 증가할수록, top-k 질의 수행에 필요한 후보 역영의 크기가 증가하며, 이로 인하여 전체 수행 시간이 증가하기 때문이다.

마지막으로 Table 1은 클라이언트 시스템에서 클라우드로 전송되는 데이터 크기를 비교한 결과이다. 실험에서 릴레이션 PARTSUPP의 튜플 수는 0.8M으로 고정되었으며, 릴레이션 LINEITEM의 튜플 수를 1M에서 5M으로 변화 시켰다. 또한, k값은 10으로 설정하였다. 표에서 보이는 바와 같이 LINEITEM의 튜플 수가 증가 할수록, 3.1장의 단순방법(Naive)에서의 데이터 전송 크기는 크게 증가함을 알 수 있다. 그러나 3.2장에서 제안한 방법(Proposed)은 후보 영역을 이용하여 top-k 질의를 수행함으로써, 클라이언트 시스템에서 클라우드로 전송되는 데이터 크기를 최소화하고 있음을 알 수 있다.

5. 결 론

본 논문에서는 클라우드 환경에서 암호화된 대용량 데이터에 대하여 top-k 질의를 효율적으로 수행할 수 있는 방법을 제안하고, 이를 실험적으로 평가하였다. 본 논문의 실험 결과는 제안한 알고리즘이 단순 방법에 비하여 top-k 질의 처리에 있어서 매우 우수하다는 것을 보인다. 본 연구 결과는 민감한 데이터를 다루는 응용 프로그램에서 활용될 수 있을 것으로 예상된다.

REFERENCE

[1] C. Orencik and E. Savas, “Efficient and Secure Ranked Multi-keyword Search on Encrypted Cloud Data,” *Proceedings of the 2012 Joint EDBT/ICDT Workshops*, pp. 186-195, 2012.

[2] Z. Xu, W. Kang, R. Li, K. Yow, and C.Z. Xu, “Efficient Multi-Keyword Ranked Query on Encrypted Data in the Cloud,” *Proceeding of IEEE 18th International Conference on Parallel and Distributed Systems*, pp. 244-251, 2012.

[3] C. Wang, N. Cao, J. Li, K. Ren, and W. Lou, “Secure Ranked Keyword Search over Encrypted Cloud Data,” *Proceedings of IEEE 30th International Conference on Distributed Computing Systems*, pp. 253-262, 2012.

[4] K.S. Candan, J.W. Kim, P. Nagarkar, M. Nagendra, and Y. Renwei, "RanKloud: Scalable Multimedia Data Processing in Server Clusters," *IEEE MultiMedia*, Vol. 18, No. 1, pp. 64-77, 2010.

[5] C. Doulkeridis and K. Norvag, "On Saying "Enough Already!" in MapReduce," *Proceeding of the 1st International Workshop on Cloud Intelligence*, 2012.

[6] C. Gentry, "Fully homomorphic encryption using ideal lattices," *Proceedings of the Forty-first Annual ACM Symposium on Theory of Computing*, pp. 169-178, 2009.

[7] S. Tu, M.F. Kaashoek, S. Madden, and N. Zeldovich, "Processing Analytical Queries over Encrypted data," *Proceedings of the Very Large Data Bases Endowment*, pp. 289-300, 2013.

[8] T. Ge and S. Zdonik, "Answering Aggregation

Queries in a Secure System Model," *Proceedings of the International Conference on Very Large Data Bases*, pp. 519-530, 2007.

[9] 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.

[10] TPC-H, http://www.tpc.org/tpch, 2015. 9. 1

[11] Amazon EC2, http://aws.amazon.com/ec2, 2015. 9. 1

[12] M. Jang, A. Cho, and J. Chang, "Cache View Based Top-k Query Processing for Encrypted Data Analysis," *Lecture Notes in Electrical Engineering*, Vol. 330, pp. 725-730, 2015.

[13] X Liao and J. Li, "Privacy-preserving and Secure Top-k Query in Two-tier Wireless Sensor Network," *Proceedings of the IEEE Global Communications Conference*, pp. 335-341, 2012.

[14] J. W. Kim "Data Partitioning on MapReduce by Leveraging Data Utility," Journal of Korea Multimedia Society Vol. 16, No. 5, pp. 657-666, 2013.

김 종 욱

1994년 3월~2000년 8월 고려 대학교 전산과학과 학사
2000년 9월~2002년 8월 한국과학기술원 전산학과 석사
2004년 1월~2009년 12월 Arizona State University Computer Science 박사
2009년 10월~2010년 8월 Technicolor Member Research Staff
2010년 9월~2013년 8월 Teradata, Software Engineer
2013년 9월~현재 상명대학교 미디어소프트웨어학과 조교수