情報保護學會論文誌
第17卷 第3號, 2007. 6

안전한 데이터베이스 환경에서 삭제 시 효과적인 데이터 익명화 유지 기법*

변 창 우,[1†] 김 재 환,[1] 이 향 진,[2] 강 연 정,[2] 박 석[1]
[1]서강대학교 컴퓨터학과, [2]한국정보보호진흥원 암호응용팀

An Effective Anonymization Management under Delete Operation of Secure Database

Changwoo Byun,[1†] Jaewhan Kim,[1] Hyangjin Lee,[2] Yeonjung Kang,[2] Seog Park[1]
[1]Department of Computer Science, Sogang University,
[2]Cryptography & Standardization Team, Korea Information Security

요 약

정보를 배포할 때 개인정보를 보호하기 위해 데이터 소유자는 이름이나 주민등록번호와 같은 명시적인 개인 신원정보를 암호화하거나 삭제한다. 그러나, 배포되는 정보들을 서로 연결함으로써 개인 신원을 확인할 수 있고 결국 개인정보가 노출되게 된다. 배포되는 정보로부터 개인정보를 보호하는 방법에 대한 최근의 연구는 k-anonymity 방법과 ℓ-diversity 방법이다. 그러나, 이들 연구는 데이터의 삽입이나 삭제가 없는 정적인 환경을 가정하고 있다. 따라서, 동적인 데이터베이스 환경에 기존 기법들을 그대로 적용할 경우 갱신된 데이터의 내용이 반영됨으로써 개인정보가 유출되는 취약성이 발견된다. 특히, 삽입 환경에서 발생되지 않는 삭제 환경에서의 고려사항은 k-anonymity와 l-diversity 스킴이 붕괴될 수 있다는 것이다. 본 논문에서는 삭제 환경에서 동적 데이터베이스 환경에서 k-anonymity와 ℓ-diversity를 그대로 따르면서 데이터베이스 익명화를 유지할 수 있는 기법을 제안한다.

ABSTRACT

To protect personal information when releasing data, a general privacy-protecting technique is the removal of all the explicit identifiers, such as names and social security numbers. De-identifying data, however, provides no guarantee of anonymity because released information can be linked to publicly available information to identify them and to infer information that was not intended for release. In recent years, two emerging concepts in personal information protection are k-anonymity and ℓ-diversity, which guarantees privacy against homogeneity and background knowledge attacks. While these solutions are signigicant in static data environment, they are insufficient in dynamic environments because of vulnerability to inference. Specially, the problem appeared in record deletion is to deconstruct the k-anonymity and ℓ-diversity. In this paper, we present an approach to securely anonymizing a continuously changeable dataset in an efficient manner while assuring high data quality.

Keywords : *privacy, publishing data, inference attack, anonymity, diversity, data quality*

접수일: 2006년 12월 20일; 채택일: 2007년 4월 3일
* 본 연구는 "개인정보보호기술 표준화 연구 (2006-P10-47)" 지원에 의해 수행되었습니다.
† 주저자, cwbyun@sogang.ac.kr

Ⅰ. 서 론

많은 기업, 공공기관에서는 수집하는 개인정보(privacy)의 양은 계속적으로 증가하고, 이들 정보는 다양한 목적(예를 들어, 의료 분석, 인구통계 동향 분석, 마케팅 조사 등)으로 사용한다[1,2]. 수집된 개인정보들이 소속된 조직의 통제 속에 있다면 많은 기술들을 통해 보호될 가능성은 높다. 그러나, 개인정보가 배포되었을 때는 더 이상 그것을 수집했던 조직의 통제 하에 있지 않기 때문에 문제가 된다.

수집된 정보가 제 3자에게 배포되는 환경에는 크게 두 가지로 구분할 수 있습니다. 수집 후 장기간 후에 배포를 하는 경우와 단기간 혹은 실시간의 배포의 요구가 있는 환경이다. 전자의 예로 정부기관이나 공공기관에서의 조사(통계조사, 경향 조사)가 될 수 있으며, 이 경우 데이터의 삽입/삭제가 잦은 환경이라도 요구하는 시점에서 원시 데이터를 익명화를 하는 것이 바람직합니다. 후자의 경우는 병원 내의 많은 연구소, 포털사이트를 운영하는 기업 내의 연구소처럼, 소장한 데이터를 기반으로 많은 연구원들의 연구 자료로 실시간 혹은 단기간적으로 활용하는 경우이다. 본 연구는 병원의 DBMS 환경에서 병원 내의 많은 연구소의 연구원들의 자료 요청에 대처하는 상황을 가정한다.

〔표 1〕 배포된 유권자 정보

	이름	나이	성별	ZIP
t1	김상욱	25	M	02144
t2	안은주	29	F	02141
t3	이현주	26	M	02138
t4	윤일국	28	M	02139
t5	김재환	25	M	02140

〔표 2〕 배포된 의료 정보

	나이	성별	ZIP	병명
t1	25	M	02144	감기
t2	25	M	02140	피부염
t3	26	M	02138	감기
t4	28	M	02139	폐렴
t5	29	F	02141	빈혈
t6	35	M	02142	당뇨
t7	38	M	02143	당뇨

다음과 같은 가상의 두 개의 배포된 정보를 가정한다. 〔표 1〕은 모 지역의 배포된 유권자 명부 데이터이고, 〔표 2〕는 같은 지역의 병원에서 배포한 환자에 대한 의료기록이다.

식별자가 제거되어 배포받은 테이블 하나만으로는 개인정보를 식별해 낼 수 없지만 다른 테이블과의 조인을 통해 식별이 가능하다는 문제점이 있다. Sweeny는 이 문제점을 해결하기 위해 k-anonymity를 제안하였다[3,4]. K-anonymity는 데이터 집합에 있는 각 레코드들이 적어도 k-1개의 다른 레코드들과 구분되지 않도록 하여 프라이버시를 보호하는 방법이다. k-anonymity 방법은 구분되지 않는 레코드들의 개인적인 속성값들이 모두 같거나 다양하게 구분되지 않으면 속성 값을 추론할 수 있는 문제가 있어 완벽한 익명화 기법이 아님을 밝힌 Machanavajjhala는 ℓ-diversity를 제안하였다[7]. ℓ-diversity에서는 구분되지 않는 레코드들로 이루어진 각 집합의 민감한 속성들은 적어도 ℓ개의 서로 구분되는 민감한 속성들을 포함 하도록 하는 기법이다.

k-anonymity와 ℓ-diversity는 레코드의 추가나 삭제가 없는 정적인 환경을 가정하기 때문에 레코드가 추가되거나 삭제될 경우 민감한 속성의 값이 유출되는 문제가 발생한다. Byun은 점진적인 레코드 삽입 환경에서 전체 데이터 집합에 대해서 다시 ℓ-diversity 연산을 하지 않고 부분적으로 처리하는 방법과 이때 추론을 할 수 없도록 하는 방법을 제안하여 이 문제를 해결하였다[8].

그러나, 삽입 환경과는 달리 삭제 환경의 가장 큰 해결 문제는 ℓ-diversity의 익명화가 무너지는 상황이 발생한다는 것이다. 본 논문은 익명화되어 배포된 테이블에 원시 테이블의 삭제 연산결과를 반영함에 따라 무너진 익명화를 복구하는 방법을 제안한다. 데이터 익명화 기법은 데이터 유형에 종속되어 있지 않지만 논문 전개의 편이를 위해 관계형 데이터베이스로 가정한다.

본 논문의 구성은 다음과 같다. 2장에서는 본 논문과 관련된 k-anonymity 기법과 ℓ-diversity 기법에 대해 간략히 소개한다. 3장에서는 삭제 연산이 k-anonymity와 ℓ-diversity에 미치는 영향과 이로 인하여 발생하는 문제를 기술하고 이를 해결하는 기법을 소개한다. 4장에서 결론을 맺는다.

Ⅱ. 관련연구

2.1 k-anonymity

2.1.1 기본 개념

배포되는 정보들 간의 연결될 수 있는 공통적인 속성을 quasi-identifier라 정의하고 있다[3].

〔**정의 1**〕 **quasi-identifier :** 테이블의 quasi-identifier QT는 배포된 다른 정보와 연결할 수 있는 속성들의 집합이다.

〔표1〕과 〔표2〕에 테이블이 연결될 수 있는 속성 집합은 QT={나이, 성별, ZIP} 이다.

k-anonymity의 목적은 quasi-identifier를 이용하여 인스턴스들 간의 대응관계와 테이블 안의 레코드 사이에 높은 확률을 가지는 관련성을 제거한 테이블을 생성하는 것이다〔3,4〕.

〔**정의 2**〕 **k-anonymity 요구사항** : 테이블 T가 quasi-identifier QT에 의해 k-anonymous 하기 위해서는 T 안의 모든 레코드 r은 QT에 대해 구별이 불가능한 레코드이 적어도 (k-1)개 존재해야 한다.

이와 같은 요구사항에 의해 배포된 정보를 소유한 사람은 특정 개인이 배포된 정보에 포함되어 있다는 사실은 알 수 있지만, 어떤 레코드이 그 사람을 나타내는 지는 1/k 확률보다 큰 확률을 가지고 결정할 수 없게 만든다.

〔표 3〕은 〔표 2〕의 테이블에 대해 QT = {나이, 성별, ZIP}을 가정을 한 2-anonymous 테이블이다.

k가 높을수록 정보 노출 정도는 떨어지지만, quasi-identifier의 정보 정확성이 떨어지는 상관관계가 있기 때문에 적정 수준의 k값으로 quasi-identifier의 값을 일반화(generalization) 할 필요가 있다.

〔표 3〕 2-anonymous table, QT = {나이, 성별, ZIP}

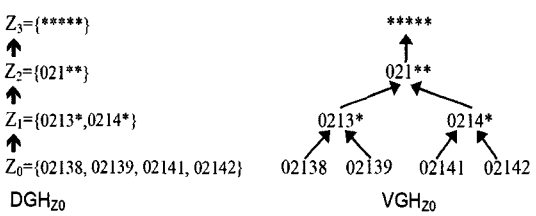
	나이	성별	ZIP	병명
t1	25	M	0214*	감기
t2	25	M	0214*	피부염
t3	〔26-29〕	Person	021**	감기
t4	〔26-29〕	Person	021**	폐렴
t5	〔26-29〕	Person	021**	빈혈
t6	〔35-38〕	M	0214*	당뇨
t7	〔35-38〕	M	0214*	당뇨

2.1.2 k-anonymity와 속성 일반화 계층

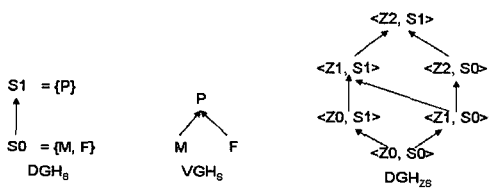
일반화 기법(generalization)은 속성의 값을 보다 일반화된 값으로 대체하는 기법으로 일반화된 도메인 집합으로 가정하고 일반화 과정을 지원하기 위해 데이터베이스 스키마의 도메인 개념을 확장한 기법이다〔5,6〕.

〔그림 1〕의 왼쪽은 ZIP 코드에 대한 도메인 일반화 계층(DGH: Z0, Z1, Z2, Z3) 및 각 도메인에 포함된 값을 표현하고 있고, 오른쪽은 도메인에 포함된 값들을 이용한 값 일반화 계층(VGH)을 나타내고 있다. 이를 이용하여 ZIP 코드 값 {02138, 02139}의 한 단계 높은 일반화 값은 0213*이 되고 이 값은 더 넓은 영역을 표현하는 값이 quasi-identifier를 그룹핑시킬 수 있는 범위가 넓어지고, 이로 인해 민감한 데이터 속성에 대해 다양한 값이 그룹핑되어 그만큼 k-anonymity의 요구사항을 만족시킬 수 있는 가능성이 높아지게 된다.

quasi-identifier를 이루는 속성이 다수인 경우는 각각의 속성에 대한 도메인 일반화 계층과 값 일반화 계층을 혼합한 일반화 계층을 만들어 일반화 과정을 수행한다. 다수의 속성을 혼합한 도메인 일반화 계층을 구성하는 방법은 격자 (lattice) 기법을 이용하는 방법이다. 〔그림 2〕의 왼쪽은 성별 도메인 일반화 계층과 그것의 값 일반화 계층을 보여주고 있고, 가장 오른쪽은 〔그림



〔그림 1〕 ZIP 코드 도메인과 은폐를 포함한 값 일반화 계층



〔그림 2〕 성별 도메인 및 ZIP 코드 도메인과 혼합된 도메인 일반화 계층

1]의 ZIP 코드와 성별 속성에 대한 격자 구성에 의해 생성된 도메인 일반화 계층을 보여주고 있다. <Z0, S0>에서 <Z0, S1>로 가는 경로는 세 가지가 생기며 각각의 일반화 계층을 기반으로 k-anonymous 테이블을 구성하고 그 중에 가장 일반화가 덜 되면서 k-anonymity 요구사항을 만족하는 테이블을 선택하게 된다.

k-anonymity를 만족하면서 일반화 기법과 혼용하여 사용하는 또 다른 방법은 레코드 은폐 기법이다. 이 기법은 일반화 기법을 이용해도 k-anonymity 요구사항을 만족하지 못하는 레코드들이 생기는 데 그 레코드들에 대해서는 적절하게 은폐하면서 k-anonymity 요구사항을 만족하게 하는 방법이다. 하지만 일반화 기법의 복잡도는 NP-hard이다[7]. 따라서 휴리스틱을 적용한 최적의 기법에 대한 연구가 진행되고 있다[5,6].

2.2 ℓ-diversity

k-anonymity 기법을 이용해 생성된 익명화된 테이블에서 구분되지 않는 레코드들의 민감한 속성의 값들이 모두 같거나 다양하게 구분되지 않으면 속성 값을 추론할 수 있는 문제가 있다.

k-anonymity 공격 예. 이웃인 철수와 영희 사이에서 어느 날 철수가 구급차에 실려 가는 것을 영희가 보았다. 철수가 영희의 이웃이기 때문에 철수의 나이(35세), 성별(남), ZIP (02142)에 대한 정보를 알 것이다. [표 3]의 익명화된 테이블에서 t6과 t7 레코드이 철수의 정보를 기록한 레코드라는 것을 알 수 있다. 이때 해당 레코드의 모든 환자들이 같은 질병(당뇨)을 앓고 있기 때문에 영희는 철수가 당뇨에 걸렸다는 것을 알게 된다.

이런 문제점을 해결하기 위해 방법이 ℓ-diversity 이다[7].

[표 4] 2-diverse table, QT = {나이, 성별, ZIP}

	나이	성별	ZIP	병명
t1	25	M	0214*	감기
t2	25	M	0214*	피부염
t3	[26-29]	M	021**	감기
t4	[26-29]	M	021**	폐렴
t5	[29-38]	Person	0214*	빈혈
t6	[29-38]	Person	0214*	당뇨
t7	[29-38]	Person	0214*	당뇨

[정의 3] ℓ-Diversity 요구사항 : *q*-block[1]에 있는 어떤 민감한 속성 값과 다른 속성 값들이 적어도 ℓ개 이상 있는 경우 이 q*-block은 ℓ-diverse하다. 그리고 테이블에 있는 모든 q*-block이 ℓ-diverse하면 그 테이블도 ℓ-diverse하다.*

결국 ℓ-diversity 요구사항은 $1/\ell$ 보다 작은 확률로 속성 노출의 위험을 갖게 된다.

[표 4]는 [표 3]의 2-anonymous 테이블을 2-diverse 테이블로 표현한 것이다. 영희가 철수의 quasi-identifier의 값(35세, 남, 02142)을 알더라도 2-diverse 테이블을 통해서는 철수가 당뇨에 걸렸다는 결론을 내릴 수 없다.

2.3 삽입 환경에서의 익명화 기법

k-anonymity와 ℓ-diversity는 레코드의 추가나 삭제가 없는 정적 데이터(static data) 환경을 가정하기 때문에 레코드가 추가될 경우 민감한 속성의 값이 유출되는 문제가 발생한다[8]. (25, M, 02145, 폐렴) 레코드가 추가되었을 때, [표 5]는 이를 2-diversity로 표현한 것이다.

ℓ-Diversity 공격 예 1(삽입). 공격자가 25세 남성이 최근 병원에 갔다는 정보를 입수하면, [표 5]에 의해서 병명이 감기, 피부염, 폐렴 중에서 어느 병인지 알 수 없다. 하지만 이전 데이터 집합인 [표 4]를 통하여 첫 번째 그룹에 추가된 레코드의 병명이 폐렴인 것을

[표 5] 새로운 2-diverse table, QT = {나이, 성별, ZIP}

	나이	성별	ZIP	병명
t1	25	M	0214*	감기
t2	25	M	0214*	피부염
t3	25	M	0214*	폐렴
t4	[26-29]	M	021**	감기
t5	[26-29]	M	021**	폐렴
t6	[29-38]	Person	0214*	빈혈
t7	[29-38]	Person	0214*	당뇨
t8	[29-38]	Person	0214*	당뇨

1) 같은 quasi-identifier 값을 갖는 레코드들의 집합. (=equivalence class)

통하여 추론할 수 있다.

Byun은 신규 레코드 삽입의 영향으로 발생하는 추론 공격에 대처하기 위해 다음과 같이 각 연산에서 추론 공격의 여지를 주는 추론 경로를 차단할 필요가 있음을 제안하였다[8]. 제안된 방법은 k-anonymity 기법에서 일반화를 최소화하는 것과 마찬가지로 신규 레코드를 삽입하는 q*-block을 선택하는 데 있어서도 일반화에 의한 정보 손실을 최소화할 수 측정 기준을 [수식 1]을 제안하고 있다.

[수식 1] 정보 손실(Information Loss)

$$IL(q) = |q| \times \sum_{j=1,\ldots,m} \frac{|G_j|}{|D_j|}$$

|q|: q*-block에 있는 레코드의 수,
|Dj|: 속성 aj의 도메인 크기,
|Gj|: 일반화에 참여한 속성 aj의 일반화 정도

정보 손실을 측정하는 방법에 있어서 기존의 방법들은 속성의 일반화에 참여하는 속성값들의 범위의 크기나 그 개수가 기준이 되었지만, Byun[8]의 경우 일반화 계층 구조를 통해 표현했을 때 일반화에 참여하는 단말노드의 수가 많을수록 정보 손실이 많아 데이터의 활용도가 떨어진다고 판단하는 것이다.

[표 5]의 (t1, t2, t3)에 의해 이루어진 q*-block에 대한 정보 손실은 다음과 같다.

$$IL(t1, t2, t3) = 3 \times (1 + 1 + \frac{|2|}{|10|}) = \frac{33}{5}$$

|q|: 3
$D_{나이}$: 1 (원시 데이터를 그대로 사용), $D_{성별}$: 1,
D_{zip}: 10 (02140에서 02149)
$G_{나이}$: 1 (일반화 없음), $G_{성별}$: 1,
G_{zip}: 2(한단계 일반화)

새로 추가되는 레코드들의 집합 R = {R1, R2, ..., Rn}이라 하자. 임의의 q*-block Qi에 Ri가 추가된 결과를 Qi'라 한다면 Ri를 추가함으로써 나타나는 정보 손실은 IL(Qi') - IL(Qi)이다. 이 개념을 바탕으로 정보 손실을 최소화하면서 신규 레코드를 갱신하는 방법을 다음과 같이 기술하고 있다.

add 연산 : R에 있는 레코드들로 이루어진 집합 그 자체가 ℓ-diverse q*-block을 구성하고 기존에 존재하는 q*-block과 중복되지 않는다면 새로운 q*-block으로 추가한다.

insert 연산 : 새로운 q*-block에 포함되지 못한 레코드들은 기존에 존재하는 q*-block에 삽입되어야 한다. 새로 삽입되는 레코드 Ri는 정보 손실을 최소화하기 위해 IL(Ej ∪ {Ri}) - IL(Ej) 값이 최소가 되는 q*-block Ej에 삽입한다.

split 연산 : add나 insert가 완료된 후 구분되는 민감한 속성값이 2ℓ보다 큰 경우 정확성을 향상시키기 위해 두 개의 q*-block으로 분할한다.

Ⅲ. 연구동기

삭제 환경에서도 삽입 환경에서처럼 [표 6]에 있는 삭제 전과 후에 배포된 테이블을 공격자가 가지고 있고, 최근에 60대 여성인 누군가가 퇴원을 했다는 정보를 가지고 있으면 그 여성은 기관지염이라는 것을 추론할 수 있다. 때문에 각 q*-block마다 waiting-list를 두어 waiting-list가 ℓ-diverse한 경우에 레코드 삭제를 반영한 익명화 처리 방법을 적용해야 한다.

삭제 환경의 경우 추가되는 문제점은 레코드 삭제의 결과가 익명화된 테이블에 반영됨에 따라 익명화 요구사항 자체를 무너뜨린다는 것이다. 병원 환자들의 질병 정보를 4-anonymous 2-diversity 테이블인 [표 6]-(a)에서 기관지염을 앓고 있는 환자의 레코드를 삭제하면 [표 6]-(b)와 같이 된다. 이 경우 나이 [61-70]인 여성으로 익명화된 부분의 레코드 수는 3개가 되고 서로 구분되는 질병이 감기 하나뿐이므로 4-anonymous 2-diversity를 위배하게 된다. 즉, [표 6]의 테이블에 속성 질병이 기관지염인 레코드의 삭제를 반영함에 따라 두 번째 q*-block은 레코드의 수가 3개가 되어 4-anonymous하지 않게 되고, 3개의 레코드의 민감한 속성값의 종류가 한 가지이기 때문에 2-diverse하지 않게 된다.

이처럼 삭제 연산을 반영함에 따라 익명화 정책에 위반되는 q*-block을 파손블록(broken q*-block)이라 명명하고 다음과 같이 정의 한다.

[정의 4] 파손블록(Broken q*-block) : 임의의 q*-block이 삭제 연산이 반영됨에 따라 k-anonymous하지 않게 되거나 ℓ-diverse하지 않게 될 때 이러한 q*-block을 파손블록(Broken q*-block)이라고 한다.

〔표 6〕 레코드 삭제에 의한 익명성 파괴

나이	성별	질병
〔21-60〕	M	폐렴
〔21-60〕	M	소화불량
〔21-60〕	M	소화불량
〔21-60〕	M	폐렴
〔61-70〕	F	감기
〔61-70〕	F	감기
〔61-70〕	F	감기
〔61-70〕	F	기관지염
〔71-90〕	F	당뇨
〔71-90〕	F	요도결석
〔71-90〕	F	위염
〔71-90〕	F	감기

(a) 삭제 전

나이	성별	질병
〔21-60〕	M	폐렴
〔21-60〕	M	소화불량
〔21-60〕	M	소화불량
〔21-60〕	M	폐렴
〔61-70〕	F	감기
〔61-70〕	F	감기
〔61-70〕	F	감기
〔71-90〕	F	당뇨
〔71-90〕	F	요도결석
〔71-90〕	F	위염
〔71-90〕	F	감기

(b) 삭제 후

Ⅳ. 삭제 환경에서의 익명화 기법

이처럼 삭제 연산의 결과를 배포하는 테이블에 그대로 반영할 경우 파손블록이 발생하는 문제가 있기 때문에 삭제 연산 후 파손 블록을 처리할 필요가 있다. 가장 단순한 방법으로 전체 테이블의 익명화를 다시 수행하는 방법이 있지만 삭제연산을 반영할 때 마다 익명화 연산을 수행하는 것은 계산량에 있어서의 낭비가 심해진다. 본 장에서는 파손된 q*-block을 처리하기 위해 원 시테이블 전체를 연산하지 않고 파손된 q*-block과 정상적인 q*-block을 합병하는 부분적인 연산으로 파손된 q*-block을 처리하는 방법을 제안한다.

4.1 합병을 통한 파손블록 제거

파손블록을 다른 정상적인 q*-block과 합병하여 파손블록을 처리하기 위해서는 합병으로 만들어진 새로운 레코드의 집합이 k-anonymous ℓ-diverse해야 한다. 다음의 정리1과 그 증명을 보면 이것이 가능하다는 것을 알 수 있다.

정리 1: 임의의 q*-block이 k-anonymous ℓ-diverse 요구사항을 만족한다면 파손된 q*-block을 합병하더라도 항상 k-anonymous ℓ-diverse하다.

증명: 임의의 q*-block, Q의 레코드들의 수를 $|r|$, 서로 구분되는 민감한 속성들의 수를 $|s|$라 하자. Q는 k-anonymous ℓ-diverse하기 때문에 $k \le |r|$, $\ell \le |s|$, 그리고 $\ell \le k$를 만족한다. 파손된 q*-block의 레코드 n개를 합병한 q*-block, Q'의 레코드 수를 $|r'|$, 서로 구분되는 민감한 속성들의 수를 $|s'|$라 하자. $|r'|=|r|+n$이고 $n>0$, $k \le |r|$이기 때문에 $k \le |r'|$를 만족한다. 추가된 n개의 레코드의 민감한 속성값들이 Q에 있는 민감한 속성값에 포함된다면 $|s'|=|s|$가 되고, 하나도 중복되지 않으면 $|s'|=|s|+n$이 된다. 결국 $|s| \le |s'| \le |s|+n$을 만족한다. $k \le |r|$고 $|r| \le |r'|$이기 때문에 $k \le |r'|$이다. $\ell \le |s|$고 $|s| \le |s'|$이기 때문에 $\ell \le |s'|$이다. 따라서 A'은k-anonymous ℓ-diverse하고 위의 정리1은 참이다.

이처럼 파손 블록을 정상 q*-block과 합병하여 익명성을 유지할 수 있음을 알 수 있다. 하지만 여기서 고려해야 할 새로운 문제는 파손블록과 합병을 할 q*-block을 선정하는데 있어서 적용해야 할 기준을 정하는 것이다. 다음 절에서 이 부분에 대해 자세히 다루고자 한다.

4.2 합병상대 선정기준

본 논문은 합병 상대를 선택하는 과정을 다음의 세 가지 측정 과정을 통해 합병 상대를 선택한다. 〔그림 3〕은 합병 상대 선택을 설명하기 위한 4개의 q*-block을

보여주고 있으며, [그림 4]는 quasi-identifier인 zip-code와 성별의 속성 일반화 계층 메타데이타를 보여주고 있다. q*-block q_1의 네 번째 레코드를 삭제하는 상황이다.

q_1

#	ZIP	성별	병명
1	130XX	P	감기
2	130XX	P	감기
3	130XX	P	감기
4	130XX	P	기관지염

q_2

#	ZIP	성별	병명
5	131XX	M	바이러스
6	131XX	M	바이러스
7	131XX	M	기관지염
8	131XX	M	심장질환

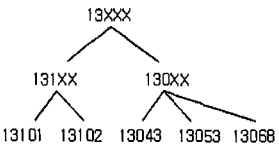
q_3

#	ZIP	성별	병명
9	13101	F	당뇨
10	13101	F	당뇨
11	13101	F	기관지염
12	13101	F	기관지염

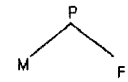
q_4

#	ZIP	성별	병명
13	13043	P	심장질환
14	13043	P	암
15	13043	P	암
16	13043	P	암

[그림 3] q*-block 집합 예제



(a) ZipCode의 일반화 계층



(b) 성별의 일반화 계층

[그림 4] 속성 일반화 계층 메타데이타

4.2.1 합병으로 인한 추론경로와 그 제거방법

기관지염의 환자가 퇴원을 함에 따라 그 레코드를 삭제를 하게 될 경우 해당 q*-block은 파손블록이 되어 다른 적절한 q*-block과 합병을 해야 한다.

예 1: 합병으로 인한 추론

우선 [그림 3]의 경우를 생각해 보자. 공격자가 13012의 주소에 살고 있는 James가 병원에서 오랫동안 치료를 받아왔다가 퇴원했다는 사실을 알고 있다고 가정하자. q_1은 파손블록이 되어 q*-block q_4와 합병하여 [표 7]과 같은 q*-block merge_q1_q4를 생성하여 배포하였다고 하자.

공격자가 (q_1 ∪ q_4) - merger_q1_q4에 의해 {130XX, P, 기관지염}을 얻게 되어 James가 기관지염을 앓고 있었다는 사실을 추론하게 된다.

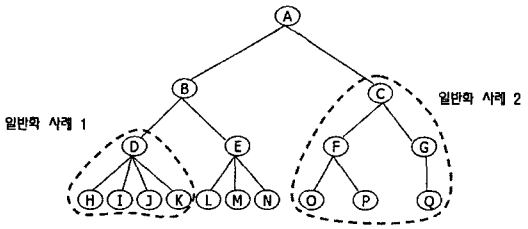
이처럼 합병에 의한 추론 경로를 제거하기 위해서는 합병 상대를 선정하는 시점에 삭제된 레코드들의 민감한 속성값과 같은 민감한 속성값을 갖고 있는 레코드가 있는 q*-block을 합병 상대로 선정해야 한다.

4.2.2 Level

속성 일반화는 속성 일반화 계층에 기반하여 이루어진다. [그림 5]에 두 가지 속성 일반화 경우가 있다. 경우 1은 일반화에 참여한 속성들의 수는 4개 이지만 한 레벨 일반화 되었다. 경우 2는 속성 일반화에 참여한 속성들의 수는 3개 이지만 두 레벨 일반화 되었다. 기존의 일반화에 의한 정보손실을 측정하는 방법은 일반화에

[표 7] q1_q4 합병 결과

ZIP	성별	병명
130XX	P	심장질환
130XX	P	감기
130XX	P	감기
130XX	P	감기
130XX	P	암
130XX	P	암
130XX	P	암



〔그림 5〕 속성 일반화의 두 가지 사례

참여하는 속성들의 수를 기준으로 하였다. 하지만 익명화된 테이블의 활용도, 즉 익명화된 테이블에 할 수 있는 질의의 구체성은 일반화의 정도 즉, 속성값이 속성일반화 계층상의 어느 정도의 레벨까지 일반화 되었는지에 따라 결정된다. 속성값이 일반화 계층상에서 단말노드에 가까울수록(level이 높을수록) 일반화에 의한 정보손실이 적어 더욱 구체적이고 다양한 질의를 할 수 있다. 따라서, 합병에 의해 일반화된 속성값이 속성 일반화 계층에서 레벨이 높은 경우를 합병 상대로 선정해 테이블의 활용도를 높일 수 있도록 해야 한다.

합병 상대를 찾는 두 번째 측정 처리는 합병을 위한 속성의 최소공통조상 노드의 Level이다.

〔정의 5〕 Level: 속성값의 계층에서 노드 A와 노드 B의 최소공통조상(least common ancestor; LCA) 노드를 C라 하자. 속성값의 계층에서 C의 레벨을 lev(A, B)라 한다.

$$Lev(b1, b2) = \sum_{j=1,\ldots,n} lev(A_j^{b1}, A_j^{b2})$$

b1과 b2: q*-block 또는 파손블록
Ajb1 와 Ajb2: b1과 b2의 j번째 속성

대부분의 경우 quasi-identifier인 속성은 복수이다. 따라서, 합병에 참여하는 두 block에 있는 각 속성의 Level 값의 합을 두 block의 Level로 계산한다.

4.2.1절에 의해 q*-block q_4는 합병 상대에서 제거되었기에 q_2와 q_3을 고려하게 된다. zipcode 속성에서 q_1과 q_2의 최소공통조상 노드는 13XXX 즉 level=1이다. 또한. 성별 속성에서 최소공통조상 노드는 P 즉, level=1이다. 따라서, Lev(q_1, q_2)=2이다. 같은 방법으로 zipcode 속성에서 q_1과 q_3의 최소공통조상 노드는 13XXX 즉, level=1이고 성별 속성에서의 최소공통조상 노드 역시 P이므로 level=1이다. 따라서, Level(q_1,q_3)=2이다.

4.2.3 Kinship

이전 절에서 속성 일반화 계층의 레벨을 기준으로 한 정보손실을 측정하는 기준을 제안하였다. 하지만 Level만으로는 정보손실을 정확히 반영하지 못하는 문제점이 있다. 4.2.2절의 예제 결과처럼 q_2와 q_3중 어느 q*-block과 합병을 하든지 간에 속성 일반화의 결과는 같다. 그러나 q_2와 합병한 경우는 zipcode와 성별 속성 모두 한 단계씩 일반화가 된 반면, q_3과 합병한 경우는 성별속성은 한 단계 일반화가 되었지만, zipcode 속성 측면에서 q_3은 두 단계 일반화가 되었다. 두 경우 모두 lev의 값이 같기 때문에 합병된 테이블의 정보손실은 같지만 합병을 함에 따라 합병 이전에 비하여 전체 테이블의 정보손실 증가량은 다르다. 즉, 계층구조상에서 인접해 있는 노드와 일반화 하는 것이 일반화에 의한 정보손실이 가장 적은 경우이다.

합병 상대를 찾는 세 번째 측정 처리는 속성 일반화 계층에서의 가까운 정도를 측정하는 Kinship이다.

〔정의 6〕 Kinship: 속성값의 계층에서 노드 A와 노드 B의 최소공통조상(least common ancestor; LCA) 노드를 C라 하자. A와 C의 레벨 차이와 B와 C의 레벨 차이의 합을 A와 B사이의 kinship 이라 하고 kin(A,B)이라 정의한다. 또한 합병이 되는 q*-block(또는 파손블록) q1과 q2에 대해 이 두 집합 사이의 kinship은 q*-block(또는 파손블록)의 속성 각각의 kinship값의 합이라고 정의한다.

$$Kinship(b1, b2) = \sum_{j=1,\ldots,n} kin(A_j^{b1}, A_j^{b2})$$

b1과 b2: q*-block 또는 파손블록
Ajb1 와 Ajb2: b1과 b2의 j번째 속성

언급한 예를 다시 살펴보면, q_1과 q_2를 합병하는 경우 kin(q_1, q_2)=3인 반면, kin(q_1, q_3)=4이다. 따라서, 최종적인 합병 상대는 q_2가 된다. 〔표 8〕과 같은 합병에 의한 q*-block merge_q1_q4이 생성되어 배포된다.

4.2.1절의 Level 측정을 먼저 하는 이유는 파손 블록과 임의의 합병 상대인 q*-block의 합병된 후의 quasi-identifier 속성의 최소 일반화 레벨을 선택하는 것이다. 그 후, 합병 상대 후보군에서 기존의 일반화 레벨에서 선정된 합병 후의 일반화 레벨까지의 최소 레벨 단위를 측정하는 Kinship 측정을 통해 quasi-identifier의

〔표 8〕 q1과 q2의 합병 결과

ZIP	성별	병명
13XXX	P	바이러스
13XXX	P	감기
13XXX	P	감기
13XXX	P	감기
13XXX	P	바이러스
13XXX	P	기관지염
13XXX	P	심장질환

정보 손실을 최소화하는 합병 상대를 선택하게 된다.

4.3 합병 이후의 블록 분리를 위한 쪼갬연산(split)

삭제 환경에서 파손블록이 생기는 경우의 합병 과정을 살펴보았다. 계속적인 파손블록의 발생으로 인해 계속적인 합병 과정을 수행하면, 익명화된 테이블에 있는 모든 블록이 합병이 될 수 있다. 또한, 합병의 결과는 quasi-identifier의 일반화를 상당히 높게 하기 때문에 정보손실도 따라서 증가한다. 따라서, 합병 후 합병된 블록이 분할(split)이 가능한 조건을 만족하면 쪼갬 연산(split)을 통해 분할을 할 필요가 있다. 합병된 후 q*-block구분되는 민감한 속성 값이 2ℓ보다 크면 데이터의 정확성을 향상시키기 위해 두 개의 q*-block으로 분할한다. Byun이 제한한 쪼갬연산의 제약사항을 이용한다.

q*-block Qi이 Qj1과 Qj2 로 나뉘는 경우, ℓ-diversity 요구사항을 만족하지 않는다면 추론에 의한 공격에 취약해진다. 따라서 쪼갬연산으로 Qi를 나누기 전에 ℓ-diversity 요구사항을 만족하는 경우에만 연산을 수행한다.

4.4 알고리즘

〔그림 6〕은 삭제 환경에서 데이터베이스 익명화 알고리즘이다. 알고리즘은 크게 두 부분으로 나뉠 수 있다.

첫 번째 단계는 익명화된 테이블을 각각의 q*-block을 추출해 내는 것이다(line 13-22).

두 번째 단계는 합병 상대를 탐색하는 과정이다. 우선 레코드가 삭제된 블록이 k-anonymity와 ℓ-diversity를 만족하지 않는 파손블록인지 여부를 판단하여야 한다. 파손블록이 아닌 경우에는 익명성에 문제가 발생하지 않기 때문에 알고리즘 24줄에서처럼 익명화된 테이블에서 삭제 연산을 반영하고 연산을 종료한다. 파손블록일 경우에는 합병에 의한 추론공격이 가능한 경우와 가능하지 않는 경우로 나누어 처리한다. 파손블록이 k-anonymity만 위반할 경우에는 합병에 의한 추론공격이 불가능하기 때문에 Level과 Kinship을 통해 합병 상대를 바로 선정할 수 있다(line 27-30). K-anonymity와 ℓ-diversity의 요구사항을 위반할 경우 삭제된 레코드의 민감한 속성값과 같은 속성값을 지닌 레코드를 포함하는 q*-block과 합병하지 않을 경우 추론공격이 가능하기 때문에 제거한다. 그런 후 나머지 q*-block들을 이용하여 Level을 계산하여 가장 큰 Level 값을 나타낸 q*-block들을 CandidatePartner로 정한다. 그 다음 Kinship을 계산하여 가장 작은 Kinship 값을 갖는 q*-block들을 MergePartner로 정한다. 파손블록과 합병 상대를 합병한 결과와 나머지 q*-block을 통해 익명화된 테이블을 재생성하면 삭제연산 처리는 종료된다.

위의 예제의 경우 최종적으로 MergePartner에 남은 q*-block은 하나이다. 하지만 quasi-identifier의 수가 많아지고 데이터의 양이 커지면 Level과 Kinship을 통해서 최적의 합병 상대를 찾아도 복수개의 결과가 나올 수 있다. 이 경우에는 배포된 테이블을 사용하는 환경에 따라 합병 상대를 선정하여야 한다. 위의 예제의 경우 익명화된 테이블을 통해 얻고자 수행하고자 하는 질의가 zipcode를 이용한 질의를 많은 환경일 경우 복수개의 합병 상대 중 zipcode가 적게 익명화된 상대를 합병 상대로 선정하는 것이 테이블의 활용도 측면에서 유리할 것이다. 따라서, 관리자는 각 quasi-identifier에 우선순위를 정하고 합병 상대가 복수일 경우 우선순위에 따라 익명화가 적게 진행된 테이블을 최종 합병 상대로 선정해야 한다. 이 내용은 본 논문의 연구 범위를 벗어나기 때문에 추후 연구로 고려하고 있다.

V. 결 론

현실세계에서 다루는 각종 데이터들은 끊임없이 삽입과 삭제가 반복되고 있다. 하지만 지금까지 연구된 데이터 익명화 기법들은 데이터 집합의 갱신이 없는 정적인 환경을 가정했지만, 빈번한 레코드 삽입과 삭제에 의한 동적인 환경에서의 데이터 익명화 기법에 대한 연구는 미비하다.

```
1  Input:  Dimmension Table                                  //속성 일반화 계층정보
2          AT                                                //익명화된 테이블
3          Did                                               //삭제된 레코드의 레코드 ID
4  OutPut: 파손블록이 처리된 Anonymized Table
5          Qid                                               // q*-block의 block ID number
6          BQ                                                // 파손블록의 block ID number
7          CandidatePartner                                  // 후보 합병 상대의 block ID
8          MergePartner                                      // 합병 상대의 block ID
9          numQB                                             // q*-block의 수
10         maxLevel=0
11         minKinship=10000
12         DelSensAttr                                       // 삭제된 레코드이 민감한 속성값
13 // 단계1: Anonymized Table에서 q*-block 추출
14         Table AttSet := {SELECT DISTINCT (quasi-identifier) FROM AT}
15         numQB = {AttSet의 레코드의 수}
16         for i:=1 to numQB do
17                 Table q_i := {SELECT * FROM AttSet(i)}
18                 if (q_i에 레코드 ID가 Did인 레코드가 존재)
19                         BQ := i
20                         DELETE FROM q_i WHERE ID = Did
21                 end if
22         end for
23 // 단계 2: 합병 상대 탐색
24         if ( q_BQ is k-anonymous & l-diverse) then return AT
25         else
26             if(q_BQ is l-diverse and NOT k-anonymous) then
27                 // 추론경로 제거 불필요
28                 Level(q_BQ, q_i)가 최대인 i의 list를 CandidatePartner에 삽입
29                  CandidatePartenr에 있는 i들 중 Kinship(q_BQ,q_i)가 최소인 i를 MergePartner에 삽입
30             else if (q_BQ is not l-diverse) then
31                 // 추론경로 제거 필요
32                DelSensAtttr가 있는 레코드가 존재하는 q_i 중 Level(q_BQ, q_i)가 최대인 i의list를
33                CandidatePartner에 삽입
                  CandidatePartenr에 있는 i들 중 Kinship(q_BQ,q_i)가 최소인 i를 MergePartner에 삽입
34             end if
35         end if
36 // Merge
37         INSERT INTO q_MergePartner SELECT * FROM q_BQ
38         Anonymize q_MergePartner
39
40 // re-Generate AnonymizedTable
41         DELETE FROM AT
42         for i:=1 to numQB except i=BQ do
43                 AT :=  SELECT * FROM q_i
44                 DROP q_i
45         end for
```

〔그림 6〕 삭제 환경에서의 익명화 알고리즘

데이터 익명화 기법(예, k-anonymity, ℓ-diversity)은 레코드의 추론을 허용하지만, 그 추론의 확률을 어느 제한점(1/k, 1/ℓ)까지 줄여보고자 하는 방법이다. 이와 같은 방법을 삽입 환경이나 삭제 환경으로의 확장을 하는 것이 Byun[8]과 본 논문이다. 본 논문에서 합병(merge)를 사용하는 목적은 삭제되는 레코드에 의해 익명화를 하는데 사용했던 그 제한점 (k, ℓ)이 무너지는 것을 막아 추론 확률을 1/k 혹은 1/ℓ 만큼 유지하는 것이다. 추가로 쪼갬(split)을 사용하는 목적은 삭제에 의해 레코드의 수가 줄어들었을 때 익명화의 기준으로 잡았던 추론 확률의 허용 기준1/k 혹은 1/ℓ 만큼은 유지하면서 합병했을 때 k 혹은 ℓ보다 과도한 익명화(예,

2k 혹은 2ℓ)가 나왔을 때 데이터의 정확성(data quality)를 유지하는 것이다.

본 논문에서는 레코드 삭제에 의해 발생되는 익명화 기준에 위반되는 데이터 집합에 대한 재익명성을 처리하는 방법으로 익명화에 의한 정보손실을 최소화 할 수 있는 방안으로 속성 일반화 계층의 레벨과 근접도를 기반으로 하는 Level과 Kinship 측정 기준을 제안하였다.

삽입 환경과 삭제 환경 독립적으로 일어나기보다는 일반적인 데이터베이스 운영 환경에서는 빈번한 레코드 삽입/삭제가 이루어진다. 추후에는 통합적인 처리 기법에 대한 연구가 필요하다. 또한, 기존의 익명화 기법은 하나의 레코드가 한 사람의 정보와 관련된 환경을 가정으로 하고 있다. 그러나 배포 환경, 혹은 배포되는 데이터의 특성 (예를 들어, 시공간 데이터베이스)에 따라 여러 레코드가 한 사람에 대한 데이터 정보인 환경에서의 익명화 기법도 필요하다. 또한, 모바일 환경 혹은 유비쿼터스 컴퓨팅 환경에서 개인의 위치 정보를 보호하는 기법으로의 활용과 통신 프로토콜 상에서 개인정보를 보호하는 기법으로의 활용을 고려할 수 있다[9,10].

참고문헌

[1] P. Samarati, and L. Sweeney, "Generalizing data to provide anonymity when disclosing information(Abstract)", In Proc. of the 17th ACM-SIGMOD-SIGACT-SIGART Symposium on the Principles of Database Systems(PODS'01), pp. 188, Seattle, WA, USA, 2001.

[2] P. Samarati, "Protecting respondents' identities in microdata release", IEEE Transactions on Knowledge and Data Engineering, 13(6), pp. 1010-1027, 2001.

[3] L. Sweeney. "k-anonymity: A model for protecting privacy", International Journal on Uncertainty, Fuzziness and Knowledge-based Systems, 10(5), pp. 557-570. 2002.

[4] L. Sweeney, "Achieving k-anonymity privacy protection using generalization and suppression", International Journal on Uncertainty, Fuzziness and Knowledge-based Systems, 10 (5), pp. 571-588. 2002.

[5] G. Aggarwal, T. Feder, K. Kenthapadi, R. Motwani,R. Panigrahy, D. Thomas, and A. Zhu. "Anonymizing tables", In Proc. of the 10th Int'l Conference on Database Theory, LNCS 3363, pp. 246-258, 2004.

[6] R. J. Bayardo, and R. Agrawal, "Data privacy through optimal k-anonymization", In Proc. of the 21st International Conference on Data Engineering(ICDE'2005), pp. 217-228, Tokyo, Japan, 2005.

[7] A. Machanavajjhala, J. Gehrke, and D. Kifer, "ℓ-diversity: Privacy beyond k-anonymity", In Proc. of the International Conference on Data Engineering(ICDE'06), pp. 24-35, Atlanta, GA, USA, 2006.

[8] J. W. Byun, Y. Sohn, E. Bertino, and N. Li, "Secure Anonymization for Incremental Datasets", 3rd VLDB Workshop, Secure Data Management 2006, pp. 48-63, Seoul, Korea, 2006.

[9] B. Gedik, and L. Liu, "A customizable k-anonymity model for protecting location privacy", In Proc. of the 25th International Conference on Distributed Computing Systems, Columbus, Ohio, USA, 2005.

[10] G. Yao, and D. Feng, "A new k-anonymous message transmission protocol", In Proc. of the 5th International Workshop on Information Security Application(WISA'04), pp. 388-399, Jeju Island, Korea, 2004.

<著者紹介>

변 창 우 (Chang-Woo Byun)
1999 서강대학교 컴퓨터학과(학사)
2001 서강대학교 컴퓨터학과(석사)
2001 ~ 현재 서강대학교 컴퓨터학과 박사과정.
<관심분야> Transaction Management for Dynamic Database, Role-based Access Control, XML Access Control, Privacy
E-mail : cwbyun@sogang.ac.kr

김 재 환 (Jaewhan Kim)
2005 서강대학교 컴퓨터학과(학사)
2007 서강대학교 컴퓨터학과(석사)
2007 ~ 현재 동양시스템즈 기술연구소 TA팀
<관심분야> Role-based Access Control, Privacy
E-mail : baum1982@nate.com

이 향 진 (Hyangjin Lee)
2000년 성균관대학교 전기전자컴퓨터 공학부(학사)
2002년 성균관대학교 전기전자컴퓨터 공학부(석사)
2002년 ~ 현재 한국정보보호진흥원 연구원
<관심분야> 개인정보보호기술, 데이터베이스 보안
Email : jiinii@kisa.or.kr

강 연 정 (Yeonjung Kang)
2003 한양대학교 전자전기공학부(학사)
2006 한양대학교 수학과(석사)
2006 ~ 현재 한국정보보호진흥원 암호응용팀
<관심분야> 개인정보보호기술, 데이터베이스 보안
E-mail : ,yjkang@kisa.or.kr

박 석 (Seog Park) 종신회원
1978년 2월 : 서울대학교 계산통계학(이학사)
1980년 2월 : 한국과학기술원 전산학(공학석사)
1983년 8월 : 한국과학기술원 전산학(공학박사)
1983년 9월~현재 : 서강대학교 컴퓨터학과 정교수
1997년 2월~현재 : 한국정보보호학회 이사
1998년 9월~현재 데이터베이스 연구회 운영자문위원
2006년 : 국세청 정보화 자문위원
2005년 : 한국정보과학회 부회장
2004년 1월~2005년 12월 : 한국정보과학회지 편집위원장
2002년 ~ 2004년 : University of Virginia 방문교수
2006년 : VLDB Panel Co-chair
2006년 : KCC 2006 한국컴퓨터종합학술대회 프로그램위원장
1999년 ~ 현재 : DASFAA Steering Committee
2004년 : DASFAA 2004 Organization Chair
<관심분야> Database Security, Transaction Management, Data Management in Sensor Network, XML, Streaming Data Process, Ubiquitous Computing, Role-Based Access Control, Privacy
E-mail : spark@sogang.ac.kr