한국지능시스템학회 논문지 2010, Vol. 20, No. 2, pp. 173-180

데이터 익명화 결정 기법

Data Anonymity Decision

정민경* · 홍동권**
Minkyoung Jung and DongKweon Hong

계명대학교 컴퓨터공학

요 약

공개되는 데이터에서 각 개인의 민감한 정보를 보호하기 위한 방법으로 데이터 익명화에 관한 연구가 활발히 이루어지고 있다. 대부분의 연구들은 익명화 요구 사항에 위배되지 않으면서, 효율적인 시간 내 레코드들을 일반화하는 기법을 중심으로 연구를 진행하고 있다. 익명화 작업이 많은 시간이 요구되는 문제임을 고려한다면, 민감한 정보에 대한 프라이버시 침해의 우려가 있는지, 익명화가 요구되는지를 미리 검사하는 것은 개인 정보 보호차원뿐만 아니라 데이터의 활용성 및 시간적 효율성 측면에서도 매우 중요하다. 또한, 그러한 침해의 우려가 있다면 어떤 유형의 공격에 취약한지를 미리 판단함으로써 그에 적절한 익명화 방식을 결정하는 것도 중요하다. 본 논문에서는 민감한 속성에 대한 공격 유형을 크게 2가지로 분류한다. 그리고 데이터가 이들 공격으로부터 안전한가의 여부를 검사할 수 있는 기법을 제시하고, 불안정하다면 어떠한 공격에 취약하고 대략 어떤 방식의 일반화가 요구되는가를 제시한다. 본 연구에서는 익명화되기 전의 테이블뿐만 아니라, 익명화된 테이블, 그리고 익명화가 되었지만 삽입, 삭제로 인해 변경된 테이블도 공격성 검사 대상이 된다. 뿐만 아니라 익명화된 테이블도 민감한 정보를 제대로 보호하고 있는지 혹은 삽입 삭제로 인해 재익명화 작업이 필요한지의 여부도 본 연구의 결과로 결정할 수 있다.

키워드 : 데이터 프라이버시, 민감 정보, 데이터 익명화, 프라이버시 침해

Abstract

The research of the preserving privacy of sensitive information has been popular recently. Many researches about the techniques of generalizing records under k-anonymity rules have been done. Considering that data anonymity requires a lot of time and resources, it would be important to decide whether a table is vulnerable to privacy attacks before being opened in terms of the improvement of data utilization as well as the privacy protection. It is also important to check to which attack the table is vulnerable and which of anonymity methods should be applied in the table. This paper describe two possible privacy attacks based upon related references. Also, we suggest the technique to check whether data table is vulnerable to any attack of them and describe what kind of anonymity methods should be done in the table. The technique we suggest in this paper can also be applied for checking the safety of anonymity tables in which insert or delete operations occurred as well from privacy attacks.

Key Words : data privacy, sensitive information, data anonymity, privacy attacks

1. 서 론

기업, 병원, 다수 공공기관에서 수집된 많은 개개인의 정보들은 외부로 배포되기 전에 개인 신원 정보(예를 들어 주민등록번호 혹은 이름)들은 미리 암호화 되거나 삭제되고 나머지 정보들만이 배포된다. 그러나 이러한 공개된 정보로부터 개개인에 대한 레코드를 식별하거나 아울러 민감한 속성 정보(예를 들어 병명 혹은 학력 사항)까지 추론함으로써 개인 정보의 프라이버시(privacy)를 침해할 수 있다.

관련 연구 논문[1,2,3,5]들을 토대로, 민감한 정보의 프라이버시 침해 방법을 크게 2가지로 정리할 수 있다. 첫 번째로, 배포된 데이터 테이블로부터 특정 속성 값들을 연결하여 개개인에 대한 레코드들을 식별하고 해당 레코드의 민감한 속성 정보를 침해하는 방법이다[2]. 본 논문에서는 이러한 침해 방식을 연결성 공격(Linking Attack)이라 명명한다. 아래의 <표 1>은 연결성 공격을 보여주는 예로, 가상의 환자들의 의료 정보를 나타내며 병명은 민감한 속성에 해당된다.

연결성 공격의 예) 어느 회사의 직원인 홍길동은 병 때문에 병가를 내고 며칠 째 회사에 나오지 않는다. 홍길동의 직장 상사 김민철은 홍길동이 사는 동네(우편번호가 13490), 나이(41세), 성별(남)을 알 것이다. 이러한 정보들을 토대로 김민철은 <표 1>의 R4 레코드가 홍길동의 의료기록이라는 것을 식별할 수 있으며 홍길동이 당뇨병에 걸렸음을 알게 된다.

두 번째로, 익명화된 혹은 되지 않은 테이블에서 특정 속성에 대해 동일한 값을 가지는 레코드들이 민감한 속성에

접수일자 : 2010년 1월 6일
완료일자 : 2010년 4월 1일
** 교신저자
감사의 글 : 본 연구는 2009년 계명대학교 비사연구기금으로 이루어졌음.

대해서도 모두 동일한 값을 가질 때, 민감한 속성값을 추론할 수 있다. 이러한 침해 방식은 동질성 공격(Homogeneity Attack)[1]이라 불리며 아래의 익명화된 가상의 의료 정보 테이블로 설명될 수 있다.

표 1. 환자의 의료 정보
Table 1. Medical information of a patient

	우편 번호	나이	성별	병명
R1	13502	40	남	당뇨
R2	13483	25	남	폐렴
R3	13660	23	여	독감
R4	13490	41	남	당뇨

표 2. 환자의 익명화된 의료 정보
Table 2. Generalized medical information

id	우편 번호	나이	성별	병명
R1	13***	40~45	남	당뇨
R2	13***	40~45	남	당뇨
R3	134**	20	여	독감
R4	134**	20	여	결핵

동질성 공격의 예) 직상 상사인 김민철은 부하직원인 홍길동이 사는 동네(우편번호가 13490), 나이(41세), 성별(남)을 알고 있다. <표 2>를 통하여 R1과 R2 레코들 중 하나가 홍길동의 정보를 기록한 것임을 알 수 있다. 이때 R1과 R2 레코드가 동일한 병명(당뇨)의 민감한 속성정보로 가지고 있기 때문에 김민철은 홍길동이 당뇨에 걸렸다는 것을 알게 된다.

데이터 프라이버시를 지키기 위해서 연결성 공격과 동질성 공격을 방지하기 위한 다양한 연구가 활발히 진행되고 있다. (2장 관련연구에서 자세히 소개한다.) 그러나 이러한 연구들은 배포될 정보가 연결성 공격 혹은 동질성 공격을 유발할 것이라는 전제하에서 진행하였다. 따라서 전제가 아닌, 실제로 "배포될 테이블이 민감한 정보를 노출하는가?"를 검사하고, 문제가 있을 경우 어떠한 공격에 취약한가에 대한 결정이 먼저 이루어져야 한다. 본 논문에서는 배포될 테이블이 민감한 정보를 노출하는가 그리고 어떠한 공격에 취약한가를 결정할 수 있는 기법을 제시하며, 본 연구의 필요성은 다음과 같다.

첫째로, 실무에서 많은 데이터 테이블은 프라이버시 침해를 유발하지 않을 수도 있으며 익명화 작업이 불필요할 수 있다. 즉, 배포될 정보들을 모두 익명화 작업이 필요한 대상으로 간주하는 것은 불필요한 작업으로 정보 이용률(utilization) 및 시간적 자원을 소모시킨다.

둘째로, 배포될 테이블이 어떠한 공격을 유발하는가를 미리 판단하는 것은 중요하다. 배포될 테이블이 연결성 공격을 유발한다면, 민감한 속성 정보들을 제외한 특정 속성들만을 기준으로 병합될 최적의 레코드들을 검색하여 일반화한다. 레코드들이 대체로 서로 다른 값을 가진다면 테이블의 모든 레코드들을 대상으로 일반화 작업이 행해져야한다. 하지만 동질성 공격만을 유발하게 된다면, 이는 테이블의 레코드들이 대체로 비슷한 값을 가지며 서로 하나의 그룹을 형성하고 있음을 의미한다. 이 경우, 동질성 공격에 취약한 레코드를 검색하고, 이를 대상으로 자신과 다른 민감한 속성을 가진 그룹으로 병합하는, 즉 전자보다 수월한 작업이 행해진다. 단, 병합으로 인해 기존 그룹의 정보 이용률이 낮아질 수 있다. 이렇듯 공격 유형에 따라 요구되는 일반화 방식이 다르므로 어떠한 공격을 유발하는가를 미리 판단하는 것은 중요하다.

셋째로, 본 연구에서 제시하는 기법은 익명화되기 전의 테이블뿐만 아니라, 익명화된 테이블, 익명화되었지만 삽입, 삭제로 인해 변경된 테이블들에 대해서도 사용될 수 있다. 예를 들어, 익명화된 테이블인 경우, 공격으로부터 민감한 속성을 보호함으로써 올바른 익명화 작업이 행해졌는가를 검사할 수 있다. 또는, 레코드들의 삽입, 삭제가 발생했을 경우, 파손된 블록[7]이 생성되었는지 혹은 식별 가능한 레코드가 생성되었는지의 여부도 판단할 수 있다.

본 논문의 구성은 다음과 같다. 2장에서는 관련 연구를 소개하고, 3장에서는 주어진 데이터 테이블이 어떠한 공격에 취약한가를 검사하는 기법을 소개하고, 그에 따른 익명화 방식에 대해 기술한다. 4장에서는 본 논문에서 제안하는 기법의 시간 복잡도를 기술하고 마지막으로 5장에서는 결론 및 향후 과제를 제시한다.

2. 관련 연구

2.1 데이터 익명화 기법과 관련된 연구

배포되는 정보에서 민감한 개인 정보를 보호하는 방법에 대한 선구자적인 연구는 2002년에 발표된 k-anonymity 논문[2]에서 제안된 데이터 익명화 방법에서 출발한다. 우선 배포될 테이블에서 각 레코드를 유일하게 구별시킴으로써 정보를 노출 시킬 가능성이 있는 몇몇 속성들의 집합을 유사 식별자 또는 준식별자(quasi-identifier)라 한다. 예를 들면, <표 1> 에서 (우편번호, 나이, 성별) 속성은 각 레코드를 구별 지을 수 있으므로 유사 식별자가 되고 익명화가 이루어져야 한다. 익명화의 기본 방식은 유사 식별자 속성의 일부 값을 특수문자로 처리하여 해당 테이블에서 동일한 유사 식별자 속성 값을 가지는 레코드가 적어도 k개가 존재하도록 함으로써 각 레코드들이 유일하게 식별되는 것을 방지한다. 하지만, 이 방식은 연결성 공격은 막을 수 있지만, <표 2>의 R1과 R2 레코드와 같이 하나의 동일 그룹 내에 민감한 속성 정보가 모두 동일하다면 동질성 공격을 막을 수 없다.

이러한 단점을 보완한 여러 가지 연구가 제시되었는데 가장 대표적인 방식이 L-diversity[1] 이다. 이 방식은 동치 클래스 (equivalent class : 익명화된 테이블에서 동일한 유사 식별자 값들을 가진 레코드들의 집합. 예를 들어 <표 2>에서 R1과 R2는 하나의 동치 클래스를 이루며, R3과 R4 역시 하나의 동치 클래스를 구성한다.) 내에 서로 상이한 민감한 속성 값을 가지는 레코드가 L개 이상 존재하도록 함으로써 민감한 속성을 추론할 확률을 1/L로 줄인다. 하지만, 하나의 동치 클래스에서 민감한 속성 값들이 동일한 범주에 속할 경우, 간접적인 정보 획득이 가능하다. 예를 들어, 한 개의 동치 클래스 내에 병명에 대한 속성 값이 "암" 혹은 "에이즈" 인 경우, 상대방의 정확한 병명은 추론할 수 없지만 그 사람이 심각한 질병을 앓고 있음을 알게 된다. 이러한 간접적인 정보 추론을 막기 위한 연구도 최근 진행되었는데 대표적으로 t-closeness 기법과 (p+, α)-sensitive

k-anonymity 기법을 들 수 있다. 우선 t-closeness[3] 방식은 민감한 속성 값들에 EMD(Earth Mover Algorithmn)[8] 알고리즘을 적용하여 각 동치 클래스내에는 서로 다른 범주에 속하는 민감한 속성들이 존재하도록 레코드들을 병합한다. 예를 들면, "암" 이라는 병명을 가진 레코드는 자신과 비슷한 유사 식별자의 값을 가지면서 "감기"와 같이 가벼운 증상을 가진 레코드들과 병합될 것이다. (p+, α)-sensitive k-anonymity[6] 방식의 경우 테이블에서 민감한 속성 정보들을 민감도의 정도에 따라 그룹화하고 각 그룹에 가중치를 부여한다. 그리고 민감한 정보의 가중치 합계가 α 이상, 그리고 구별되는 민감한 속성값의 개수가 p 이상이 되는 동치 클래스가 생성되도록 레코드들을 병합함으로써 정보의 간접적인 누출을 막는다. 하지만 위의 기법들은 배포될 테이블이 민감한 정보에 대한 프라이버시 침해를 유발할 것이라는 가정 하에 제안되었다.

2.2 데이터 익명화 결정 연구 현황

앞에서 소개한 것과 같이 데이터의 익명화 기법에 관한 여러 가지 방법들이 활발히 진행되고 있는 반면, 데이터 익명화 결정에 관한 연구는 아직 미흡하다. 2007년에 스탠포드 대학에서 배포될 정보의 익명화 적용여부를 결정하는 기법을 Probabilistic Anonymity[4] 라는 논문으로 공개하였다. 하지만 이 기법은 연결성 공격만을 고려하였으며, 배포될 정보가 특정 속성을 기준으로 정렬되어 있다는 가정 하에 진행되었다. 우선, 배포될 데이터 테이블에서 임의의 속성들을 선택하고 관련된 통계정보를 수집한다. 이를 이용하여 해당 테이블에서 선택된 속성들이 각 레코드들을 유일하게 구별 짓는 유사 식별자 역할을 하는가에 대한 확률을 아래의 계산식으로 구한다.

$$\sum_{i=1}^{D} np_i e^{-npi}$$

n은 배포될 테이블의 레코드들의 총 개수이며, D는 테이블에서 선택된 속성에 대하여 유일한 값을 가지는 레코드들의 개수, p_i 는 i^{th} 번째 레코드가 테이블에서 유일하게 식별될 수 있는 확률을 의미한다. 위의 식에서 도출된 결과 값은 선택된 속성들이 익명화가 필요한지의 여부를 결정짓는 역할을 하게 되는데, 이 값이 미리 정한 경계 값보다 클 경우 선택된 속성에 대한 익명화가 필요함을 의미한다. 만약, 값이 작다면 해당 속성들은 유사 식별자가 아닌 것으로 간주하고, 의심되는 다른 속성들을 선택하여 위의 식을 다시 적용한다. 하지만 이 기법은 연결성 공격만을 고려하였기 때문에 동질성 공격에 노출될 수도 있다.

3. Flow Network를 이용한 데이터 익명화 결정 기법

본 장에서는 Flow Network 알고리즘[5]을 이용하여 배포될 테이블이 연결성 공격뿐만 아니라 동질성 공격에도 안전한가를 판단하고 해당 테이블에 대한 익명화 적용 여부를 결정짓는 기법을 제안한다.

이 기법은 아래와 같은 절차로 진행된다.

1) Flow Network 알고리즘을 이용하여 배포될 데이터 테이블에 대한 f를 구한다.

2) 데이터 테이블에서 유사 식별자가 될 수 있는 후보 컬럼을 선택하고 이들에 대해 동일한 값을 가지는 레코드들의 집합, 즉 동치 클래스의 개수[1]를 구한다.

3) f와 테이블의 동치 클래스의 개수를 비교하여 해당 테이블이 연결성 혹은 동질성 공격에 노출되는가를 결정한다.

3.1 관련 데이터 및 관련 용어

본 연구에서 사용될 데이터 테이블은 환자들에 대한 가상의 의료 기록을 포함하고 있는 <표 3>이다.

표 3. 의료 기록 테이블
Table 3. Data table for medical information

ID	우편 번호	나이	성별	보호자와의 관계	...	병명
R1	35070	20	여	모녀	...	Disease1
R2	35112	25	남	모자	...	Disease2
R3	35127	45	남	부부	...	Disease1
R4	35070	20	여	부녀	...	Disease3
R5	35112	25	남	부부	...	Disease1
R6	12405	52	남	부부	...	Disease3
R7	12405	25	여	모녀	...	Disease1
R8	12405	25	여	친척	...	Disease1
R9	12468	60	남	부자	...	Disease4
R10	12473	48	여	친척	...	Disease3
R11	43302	42	여	친척	...	Disease1
R12	43302	42	여	부부	...	Disease2
R13	43521	59	남	친척	...	Disease5
R14	97333	65	남	부녀	...	Disease1
R15	97333	65	남	부부	...	Disease2

<표 3>의 민감한 속성은 병명 컬럼이다. (데이터가 기록된 기관의 성격에 따라 민감한 속성이 결정될 수 있다. 예를 들어, 학생들의 성적에 관한 데이터일 경우 석차 또는 평점이 민감한 속성이 될 것이다.) 반면, 우편 번호, 나이, 성별, 보호자의 관계 컬럼들은 유사 식별자가 될 수 있는 후보 컬럼들이다. 그러나 테이블의 컬럼이 여러 개 혹은 수십 개가 될 수 있으므로 나머지 모든 컬럼들을 후보 컬럼으로 간주하는 것은 옳지 않다. 따라서 본 연구에서는 각 레코드를 구분하는데 큰 영향을 미치지 않는 컬럼들을 유사 식별자의 후보 컬럼에서 제외할 수 있도록 하였다. (관리자 혹은 공급자가 이를 선택할 수 있을 것이며 자세한 내용은 3.3절에서 소개한다.) 본 연구에서는 지금까지 익명화 연구에서 언급되었던 k-anonymity 요구 사항을 토대로 k′-anonymity 요구 사항을 정의한다.

[정의 1]. k′-anonymity 요구사항: *테이블의 각 레코드는 자신과 동일한 유사 식별자 속성 값을 가지면서 민감한 속성에 대해서는 서로 다른 값을 가지는 레코드의 개수가 k′-1개 존재해야 된다. 즉 각 동치 클래스 내에는 동일*

1) 본래 동치 클래스는 익명화된 테이블에서 동일한 유사 식별자 값을 가지는 레코드들의 집합들을 의미한다. 하지만, 본 논문에서는 익명화되기 전 테이블에서도 동일한 의미로 해당 용어를 사용한다.

한 유사 식별자 값을 가지는 레코드가 k′개 있어야 하며, 동시에 다른 민감한 속성 값도 k′개 존재해야 한다.

예를 들어 k′가 3일 경우, (25433, 30, 여, 빈혈)레코드에 대해서 최소한 2개의 (25433, 30, 여, *(빈혈을 제외한 병명)) 레코드가 테이블에 존재해야 된다. 이를 만족하지 않을 경우 해당 테이블은 연결성 공격 혹은 동질성 공격에 노출될 것이며, 익명화 작업이 필요하다.

[정의 2]. s-group(sensitive values′ group): *테이블에서 최소 k′개의 서로 다른 민감한 속성 값들의 집합을 의미한다.*

예를 들어 k′이 3일 경우, 하나의 s-group에는 {Disease1, Disease2, Disease1, Disease3}, 혹은 {Disease1, Disease2, Disease3}등이 올 수 있다. 하지만 {Disease1, Disease2}는 서로 다른 멤버의 개수가 2이므로 s-group이 될 수 없으며, 이 경우 다른 s-group에 병합된다. 테이블에서 생성될 수 있는 s-group들의 멤버들의 총 개수는 테이블의 전체 레코드들의 수와 일치한다.

3.2 *f*의 정의 및 계산

본 연구에서는 배포될 테이블의 s-group들의 최대 개수를 *f*라고 정의하며, 배포될 테이블이 연결성 공격에 노출될 것인지 혹은 동질성 공격에 노출될 것인지를 결정짓는 기준이 된다.

[정의 3]. *f*: *배포될 테이블의 s-group들의 최대 개수를 의미한다.*

예를 들어, k′가 2인 경우 하나의 s-group 에는 최소한 2개의 서로 다른 민감한 속성값들이 존재한다. 즉 테이블내에서 생성될 수 있는 s-group들의 최대 개수를 *f*라고 한다.

Flow Network 알고리즘을 이용하여 *f*을 구할 수 있으며, 우선 중복을 제거한 민감한 속성들에 대한 k′-분할 그래프(k′-sections graph)를 생성한다.

[용어 1]. Si(0<i<=m)(Sensitive Values): *테이블에서 중복이 제거된 민감한 속성값을 나타낸다. m은 테이블에서 중복이 제거된 민감한 속성값의 총 개수를 나타낸다.*

예를 들어 <표 3>에서 m은 5이며 S1 = Disease1, S2=Disease2, S3=Disease3, S4=Disease4. S5=Disease5이다.

[용어 2]. N(Si)]: *테이블내에서 Si의 출현빈도수이다. 예를 들어, <표 3>의 N(S1)은 7이며, N(S2) = 3, N(S3)=3 이다. N(Si)의 모든 총합은 N(S)으로 표현하며 테이블의 전체 레코드들의 수 n과 같다.*

$$n=\overline{N}(s)=\sum_{i=1}^{m}\overline{N}(s_i)$$

<표 3>의 경우 N(S)=7+3+3+1+1=15 이다.

[용어 3]. k′-분할 그래프(k′-sections graph): *다중분할 그래프의 한 유형으로, 최 좌측의 노드가 루트노드가 되며 제일 우측이 단말 노드가 된다. 루트노드로는 소스(Source) 노드가 단말 노드로는 싱크(Sink) 노드가 배치된다. 중간 레벨에는 N(S)/k′ 크기로 나눠진 속성들의 집합이 하나의 레벨을 이루게 되고, 이들 레벨 사이를 연결하는 연결 노드(bridge)가 위치한다.*

k′-분할 그래프에서 소스, 싱크, 연결 노드를 제외한 일반노드들은 (Si, N(Si))의 쌍으로 표현될 수 있다. 소스 노드에 유입되는 흐름의 양은 무한대이며 싱크 노드에서 유출되는 값은 *f*값이 된다. 그리고 일반 노드 i에는 유입되는 간선과 유출되는 간선을 단방향 화살표로 N(Si) 값과 함께 표시한다. 즉 해당 간선을 통해 최대 N(Si) 크기만큼의 흐름(Capacity)를 보낼 수 있다.

지금까지의 내용을 바탕으로 <표 3>의 병명 속성에 대한 k′-분할 그래프를 생성하는 과정은 다음과 같다.

1. 소스 노드와 싱크 노드를 그래프의 처음과 끝에 배치한다.

2. 일반 노드들을 k′개의 그룹으로 나누고, 한 그룹에는 노드들의 N(Si) 총합이 $\lceil \overline{N}(S)/k' \rceil$ 혹은 $\lceil \overline{N}(S)/k' \rceil$ +1 값을 넘지 않도록 배치한다.

k′이 2일 경우, 2개의 그룹이 생성되며 N(S)/k′값은 7.5(15/2)이다. 소수점은 표현될 수 없으므로 그래프의 한 그룹에는 노드들의 N(Si)총합이 7이 되도록, 다른 그룹에는 N(Si) 총합이 8이 되도록 분할한다. 만약 조건에 맞게 분할될 수 없을 경우 하나의 노드를 2개의 노드로 분할하여 각각의 그룹에 배치한다. 그리고 각 그룹 사이에는 연결 노드를 삽입하여 연결한다. (연결 노드를 삽입하는 것은 간선의 개수를 줄임으로써 실행 시간을 단축하기 위함이다.) <표 3>을 예로 들면 아래의 그림과 같다.

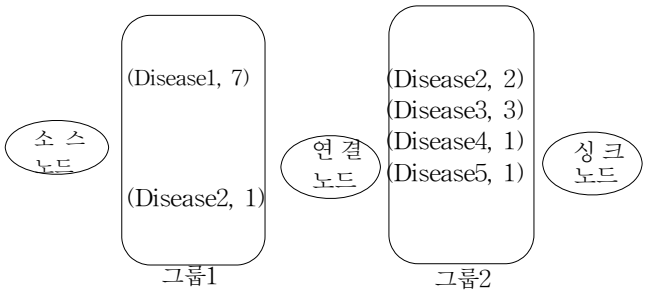

그림 1. k′개의 그룹 생성
Fig. 1. Creation of k′ group

[그림 1]에서 그룹1과 그룹2의 N(Si)총합은 각각 8(=7+1), 7(=2+3+1+1)이다. 그리고 두 그룹 사이에 연결 노드(Bridge)를 삽입하였다. 또한, (Disease2, 3)은 (Disease, 1)과 (Disease, 2)로 분할하여 각 그룹의 배치함으로써 그룹1의 N(Si)총합이 8을 넘지 않도록 하였다. 만약, 노드의 분할을 피하고자 한다면 (Disease1, 7)과 N(Si)값이 1인 다른 노드를 검색하여 함께 배치하여도 무관하다. 혹은 그룹1에 (Disease1, 7) 노드만 배치하고 그룹2에 나머지 노드를 배치함으로써 각 그룹의 N(Si)총합이 7과 8(=3+3+1+1)이 되도록 할 수 있다. 만약 k′이 3일 경우 3개의 그룹을 생성하며, 각 그룹은 N(Si)총합이 5(=15/3)을 넘지 않도록 노드들을 배치한다.

3. 각 그룹에 속해있는 노드들마다 유입되는 간선, 유출되는 간선을 단방향 화살표를 삽입하고 각 간선의 레이블은 자신의 N(Si)값으로 표현한다. [그림 1]은 [그림 2]와 같은 그래프가 될 것이다. 편의상 병명을 D1, D2..등등으로 표현한다.

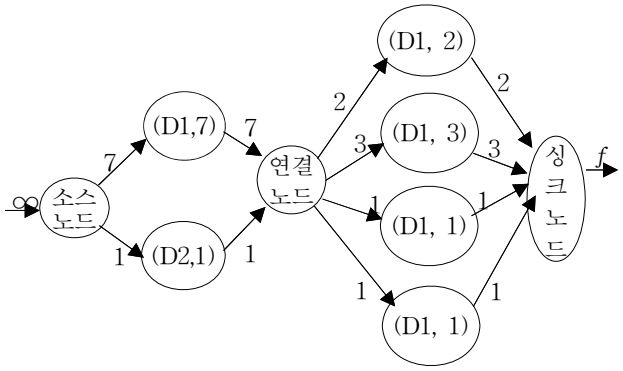

그림 2. k′-분할 그래프
Fig. 2. k′-sections graph

위의 그래프는 k′크기만큼 분할된 최종 그래프이며 최좌측인 소스 노드부터 출발하여 최 우측 싱크 노드로 흐름(flow)을 보내게 된다. 소스 노드로 유입되는 양은 무한대이며 각 노드로 연결된 유입량(capacity) 만큼 흐름을 보내게 된다. 그리고 연결 노드를 거쳐 해당 간선의 유입량만큼 흐름을 계속 보내는데, 자신과 동일한 Si 이름을 가진 노드에는 흐름을 보낼 수 없다. 예를 들어, (D2, 1)노드는 연결 노드를 거쳐 (D2, 2)노드로 흐름을 보낼 수 없다. 이는 동일한 병명을 가진 레코드와 병합될 수 없음을 의미한다. 반면, (D1, 7)노드는 (D2, 1), (D3, 3). D(4, 1), (D5, 1) 노드로 흐름을 보낼 수 있다. 즉 D1이라는 민감한 속성을 가진 레코드는 자신과 다른 병명을 가진 레코드와 병합될 수 있음을 의미한다. 예를 들어 (D1, 7)에서 1만큼의 흐름을 (D4, 1)로 보낸다면, D1병명을 가진 한 개의 레코드는 D4 병명을 가진 레코드 한 개와 병합되어 하나의 s-group을 생성할 수 있음을 뜻 한다. 만약 (D1, 7)에서 3만큼의 흐름을 (D3, 3) 노드로 보낸다면 D1병명을 가진 3개의 레코드들은 D4병명을 가진 3개의 레코드와 각각 일대일로 병합되어 3개의 s-group이 생성됨을 의미한다. 마지막으로 더 이상 흐름을 보낼 간선이 없을 경우 싱크 노드에서 유출되는 흐름의 총합이 해당 그래프의 최대 흐름(maximum flow)이자 본 연구에서 구하고자하는 f값이다. 위의 그래프의 최대 흐름, 즉 <표 3>에서 f는 7이다.

본 장에서 구한 f는 3.3장에서 해당 테이블이 연결성 혹은 동질성 공격에 안전한지 여부를 판단하는 하나의 기준이 된다.

3.3 데이터 익명화의 결정

민감한 속성을 제외한 컬럼들을 일반화 되어야 할 유사 식별자의 후보 컬럼으로 간주 할 수 있다. 하지만 본 연구에서는 의미상으로 볼 때 각 레코드를 구별하는데 큰 영향을 미치지 않는 컬럼들을 일반화 후보 컬럼에서 제외한다. (실제로 이 부분은 관리자 혹은 책임자가 정할 수 있다.) 예를 들어, <표 3> 경우 관리자는 보호자와의 관계 컬럼을 일반화 후보 컬럼에서 제외할 수 있다. 어느 20대 직장 부하가 50대 여성과 병원에 간 것을 확인하였다면, 그 여성이 직장부하의 보호자인지 혹은 어떤 다른 관계인지 확신할 수 없다. 또한, 직장 부하가 처음으로 병원에 갔을 당시 보호자를 누구로 기록했는지 확신할 수 없다. 따라서 보호자 관계 컬럼이 공개되더라도 각 레코드를 식별하는데 큰 영향을 미치지 못하므로 일반화 후보 컬럼에서 제외한다. <표 3>에서 일반화 후보 컬럼이 되는 것은 우편 번호, 성별, 나이이다. 우편 번호, 성별, 나이와 같은 정보는 육안으로 또는 사소한 친분으로도 쉽게 노출되는 성격을 지니므로 반드시 익명화되어야 할 것이다.

3.3.1 연결성 공격의 검사

주어진 테이블이 연결성 공격에 민감한지를 판단하기 위해서, 우선 테이블내에 생성되는 동치 클래스의 개수를 파악한다. 모든 레코드들이 최소 k′개씩 동일한 유사 식별자 값을 가진다면 최소한 연결성 공격에는 노출되지 않을 것이다. 하지만 테이블에서 유일한 값을 가지는 레코드들이 있거나 k′개 미만의 레코드들이 1개의 동치 클래스를 이룰 경우, k′-anonymity 요구사항에 위배되며 각 레코드들이 유일하게 식별될 수 있다. 본 연구에서는 동치 클래스의 개수와 f를 비교하여 연결성 공격을 검사할 수 있다. f는 n개의 레코드를 가진 테이블에서 최소 k′개의 서로 다른 민감한 속성값을 가지도록 레코드들을 병합했을 때 생성될 수 있는 그룹의 최대 개수이다. k′가 클수록 f값은 작아지며 k′가 작을수록 f의 값은 커지는 반비례관계이며 아래와 같이 표현할 수 있다.

$$f \propto \frac{1}{k'}$$

마찬가지로, 동일 테이블에서 생성되는 동치 클래스의 개수를 e로 두고 각 클래스를 구성하는 레코드들의 최소 개수를 p로 둔다면 아래와 같은 관계를 도출할 수 있다.

$$e \propto \frac{1}{p}$$

만약 e가 f보다 크다면 k′보다 적은 개수의 레코드들이 하나의 동치 클래스를 구성함으로써 상대적으로 많은 수의 그룹이 생성되었음을 알 수 있다. 즉 동치 클래스를 구성하는 레코드들의 최소 개수, p가 k′ 보다 작음을 의미한다. 따라서 이 경우, 테이블내에는 k′-1개의 레코드들로 구성된 동치 클래스가 존재하며 만약 k′가 2라면 레코드 한 개가 하나의 그룹을 이루고 있으므로, k′-anonymity의 요구사항을 위배함과 동시에 연결성 공격에 노출된다. 즉 일반화가 필요하다.

[정리 1]. *연결성 공격의 검사: 테이블의 동치 클래스(일반화 되었을 경우 equivalent class의 개수)의 개수가 f보다 크다면, 해당 테이블은 k′-anonymity 요구사항을 위배함과 동시에 연결성 공격에 노출된다.*

우선 테이블의 동치 클래스의 개수는 아래의 SQL문이 리턴하는 행의 개수로 파악할 수 있으며, "Probability Anonymity" 논문[4]에서도 특정 속성값들의 통계정보를 구하는데 사용되었다.

```
Select col1, col2, col3...
from Table
Group by col1, col2, col3...;
```

예를 들어, <표 3> 경우 아래의 쿼리문으로 구할 수 있으며 결과행의 개수는 10이다.

```
Select 우편 번호, 성별, 나이
from Table 3
```

Group by 우편 번호, 성별, 나이;

결과행의 개수 10은 앞 장에서 구한 f값(=7)보다 작다. 이는 <표 3>에서 1개 혹은 k′개미만의 레코드들로 구성된 동치 클래스가 존재한다는 것을 의미하므로 연결성 공격에 취약하며 익명화가 필요하다. 단, 연결성 공격 검사를 통과하였다 하더라도 예외의 경우 연결성 공격에 노출 될 수 있다. 예를 들어 <표 5>와 같이 n-1개의 레코드들과 1개의 레코드가 각각의 동치 클래스를 생성할 경우, 연결성 공격에 노출됨에도 불구하고 해당 검사를 통과할 수 있다. 또한 연결성 공격 검사는 동질성 공격에 대한 취약 여부는 나타내지 않으므로 익명화시 다른 병명을 가진 레코드를 병합하여 동질성 공격도 미리 방지하는 것이 바람직하다. 우선, 개별적으로 존재하는 레코드들이 많을 경우(앞에서 제시한 쿼리문의 결과행들이 많을 경우), 일반화할 레코드가 많음을 의미하므로 테이블의 전체 레코드들을 검사할 필요가 있을 것이다. 우선 테이블의 각 레코드를 기준으로 최대한 비슷한 유사 식별자 값을 가지면서 서로 다른 민감한 속성값을 가지는 레코드들을 찾아 병합하는 작업을 모든 레코드들에 적용함으로써, 개별적으로 존재하는 레코드를 제거하는 방식의 일반화가 필요하다.

3.3.2. 동질성 공격의 검사

동질성 공격의 검사는 테이블의 동질성 공격에 노출되는지의 여부뿐만 아니라 연결성 공격에서 인식할 수 없었던 예외의 경우를 찾아내는데 사용되는 기법이다. 한 개의 동치 클래스내의 레코드들의 민감한 속성값이 서로 같거나, 대다수의 레코드들이 서로 동일한 값을 가지는데 비해 이들과 동떨어진 값을 가지는 단일 레코드가 존재하는 경우(예를 들어 <표 5>), 본 검사를 통해 민감한 속성값을 노출하게 되는지 여부를 결정할 수 있다.

우선 <표 4>는 대체로 비슷한 유사 식별자 값을 가지며 연결성 공격을 통과할 것이다. 하지만 (97333, 32, 남) 값을 가지는 R3, R4, R5의 병명 속성값은 Disease3으로 모두 동일하므로 동질성 공격에 노출된다.

표 4. 동질성 공격에 노출되는 테이블 I
Table 4. Vulnerable table I exposed to homogeneity attack

ID	우편번호	나이	성별	병명
R1	25433	20	여	Disease1
R2	25433	20	여	Disease1
R3	97333	32	남	Disease3
R4	97333	32	남	Disease3
R5	93733	32	남	Disease3

<표 5>의 f값은 2이며 동치 클래스의 개수도 2이므로 연결성 공격을 통과한다. 하지만 R1에서 R4의 레코드들은 서로 동일하여 하나의 그룹을 생성하는데 비해 R5 레코드는 이들과 동떨어진 값을 가지며 자신이 1개의 그룹을 이루고 있다. 즉 유일하게 식별가능하며 k′-anonymity 요구 사항에 위배된다.

표 5. 동질성 공격에 노출되는 테이블 II
Table 5. Vulnerable table II exposed to homogeneity attack

ID	우편번호	나이	성별	병명
R1	25433	20~	여	Disease1
R2	25433	20	여	Disease1
R3	25433	20	여	Disease3
R4	25433	20	여	Disease2
R5	93733	26	여	Disease4

이러한 경우들을 찾기 위해서는, 본 논문에서 고안한 아래의 SQL문을 실행하여 1개의 민감한 속성값을 가지는 동치 클래스가 존재하는지 확인한다. 쿼리문이 리턴 하는 결과값이 있을 경우 해당 테이블은 동치 클래스내에 1개의 민감한 속성값을 가지거나 혹은 식별가능한 레코드가 존재하여 동질성 공격뿐만 아니라 연결성 공격에도 노출될 수 있음을 의미한다.

```
Select col1, col2, col3..
from Table
Group by col1, col2, col3..
Having count(distinct sensitive_column) = 1;
```

<표 5>를 예로 들면 아래의 SQL문으로 실행될 수 있다.

```
Select 우편 번호, 나이, 성별
from Table
Group by 우편 번호, 나이, 성별
Having count(distinct 병명) = 1;
```

동질성 공격 검사를 통과하지 않을 경우, 해당 테이블은 익명화될 필요가 있다. 우선 위의 쿼리문에서 리턴 하는 레코드를 검사하고, 자신과 최대한 비슷한 유사 식별자의 값을 가지면서 다른 병명으로 구성된 동치 클래스로 병합하는 방식으로 익명화 할 수 있다. 단, 병합으로 인해 그룹 내 레코드들의 수가 필요이상으로 증가하면, 과도한 일반화가 발생하고 정보의 손실을 초래할 수 있다. 이 경우, 해당 동치 클래스가 k′-anonymity 요구사항에 위배되지 않도록 분할하여 여러 개의 새로운 동치 클래스들을 생성할 수 있다.

4. 데이터 익명화 결정 기법의 시간 복잡도

본 연구에서 제안하는 기법의 시간 복잡도는 Flow Network 알고리즘을 이용하여 f값을 구하는데 걸리는 시간과 3.3 절에서 제시한 쿼리문들의 실행시간의 합으로 나타낼 수 있다. 앞에서 설명한 쿼리문들은 어떠한 속성들을 기준으로 레코드들을 그룹화하는 것으로, 일반적으로 테이터들을 정렬한 뒤 그룹화 하게 된다. 하지만, 자세한 과정은 테이블 내부의 통계 정보 및 인덱스들을 이용하는 등 데이터베이스 시스템의 자체 메커니즘에 따라 다를 수 있으므로 객관적인 시간을 측정하는 것은 부적합하다. 따라서 본 연구에서는 f값을 구하는데 걸리는 시간만을 고려한다.

Flow Network 알고리즘을 이용하여 최대 유입량을 구하는 데 걸리는 시간은 Residual Graph에서 Argument Path를 찾는 방법에 따라 2부분으로 나뉜다. 첫 번째로 Ford-Fulkerson algorithm 방식이다. 이는 Residual Graph에서 깊이우선탐색으로 운행하여 Argument Path를 찾는

기법으로 시간 복잡도는 O(E max |f|)이다. 여기서 E는 간선의 개수이며 f는 최대 유입량을 의미한다. 두 번째로 Edmonds-Karp algorithm방식이다. 이 방식은 너비우선탐색으로 Residual Graph를 운행하면서 Argument Path를 찾으며 시간 복잡도는 $O(VE^2)$이다. V는 정점의 개수를 E는 간선의 개수를 의미한다.[5]

본 연구에서는 너비우선탐색을 적용할 것이므로 시간 복잡도는 $O(VE^2)$이 된다. 우선 k′-분할 그래프의 정점의 최대 개수는 아래와 같다.

$$V = n+2k'$$

k′-분할 그래프에서 정점의 개수는 소스, 싱크, 연결 노드, 그리고 일반 노드의 개수의 총합이다. 우선 소스 노드와 싱크 노드는 모두 2개이다. 일반 노드는 중복이 제거된 민감한 속성값으로 표현되므로 중복이 제거된 속성값들의 개수를 의미하며 n으로 표현한다. 그리고 연결 노드는 k′개의 그룹을 연결하는 역할을 하므로 k′-1개가 된다. 예를 들어 k′이 3일 경우 일반 노드들은 3개의 그룹으로 나눠지므로 이들을 연결하는데 2개의 연결 노드가 추가될 것이다. 만약, k′개의 그룹으로 나뉠 때 2개로 분할되는 노드들이 존재할 수 있다. 3.2장의 <표 3> 경우 (Disease2, 3)은 2개의 노드로 분할되었다. 즉 k′개의 그룹이 생성될 때, 마지막 그룹을 제외한 k′-1개의 그룹에는 1개씩의 분할된 노드들이 존재할 수 있다. 이들 모두의 개수를 더하면 2+n + 2(k′-1) = n+2k′이 된다.

k′-분할 그래프의 간선의 최대 개수는 다음과 같다.

$$E = 2n$$

각 일반 노드마다 유입되는 간선, 유출되는 간선 2개씩 존재하므로 간선의 총 개수는 2n이다. 소스 노드로 무한대의 흐름을 보내는 간선은 Argument Path를 탐색하는데 사용되지 않으므로 제외한다.

마지막으로, 총 시간 복잡도는 아래와 같다.

$$VE2 = (n+2k')((2n)2) = 4n3 + 8n2k' = O(n3)$$

따라서 Flow Network 알고리즘을 이용하여 f값을 구하는데 걸리는 시간 복잡도는 $O(n^3)$ 이다.

5. 결론 및 향후 과제

어떠한 정보가 외부로 공개되기 전에 민감한 정보에 대한 프라이버시 침해 우려가 있는지를 미리 검사하는 것은 개인 정보 보호차원에서 아주 중요한 일이다. 또한, 그러한 침해 우려가 있다면 어떤 공격에 취약한지를 미리 판단하는 것도 중요하다. 이는 레코드에 대한 익명화 기법을 좌우하므로 미리 파악될 필요가 있다. 최근, 익명화 알고리즘 자체에 대한 연구가 많은 논문을 통해 제안되었지만, “과연 배포될 테이블이 공격에 취약한가? 그러하다면 어떠한 공격에 노출되는가?”를 미리 결정할 수 있는 데이터 익명화 결정 기법에 대한 연구는 활발히 이루어지지 않았다.

본 논문에서는 배포될 테이블에 대해 발생될 수 있는 공격 유형을 최근의 연구 논문들을 토대로 크게 2가지로 정리하고 기존의 k-anonymity 기법을 토대로 k′-anonymity 요구사항을 정의하였다. 그리고 배포될 테이블이 k′-anonymity 요구사항에 위배될 경우, 어떠한 공격에 취약한가를 미리 파악할 수 있는 기법을 제시하고 각 공격에 대해서는 어떠한 방식의 익명화가 이루어져야하는지 제시하였다. 연결성 공격에 취약하다면, 배포될 테이블에는 유일하게 식별 가능한 레코드가 존재한다는 것을 의미하므로 유사 식별자 값들을 익명화하여 동일한 값을 가지는 레코드가 동치 클래스에 최소한 k′개 존재하도록 한다. 만약, 연결성 공격 검사는 통과하되 동질성 공격 검사를 통과 못 할 경우, 동일한 한 개의 민감한 속성값을 가지는 동치 클래스가 있음을 의미하므로 본 연구에서 제시한 쿼리문이 리턴 하는 결과 행을 검색하고 이와 다른 민감한 속성값을 가지는 다른 동치 클래스의 레코드들과 병합 및 일반화한다.

본 연구에는 배포될 테이블뿐만 아니라 이미 익명화가 이루어진 테이블에 대해서도 이와 동일한 방법으로 공격성 검사를 적용할 수 있다. 익명화된 테이블이 민감한 정보를 제대로 보호하고 있는지의 여부를 확인해야한다면, 본 논문에서 제시하는 공격성 검사 기법을 적용하면 될 것이다.

본 논문에서 정리한 공격 외에도 앞으로 새로운 유형의 공격이 발생할 수 있을 것이다. 데이터 테이블에서 일어날 수 있는 새로운 형태의 공격을 발견하고, 이를 미리 파악할 수 있는 그리고 그에 적절한 익명화 기법에 대한 연구가 활발히 이루어져야 할 것이다.

참 고 문 헌

[1] A.Machanavajjhala, J.Gehrke, D.kifer. l-Diversity: "Privacy Beyond k-anonymity," *In proceedings of the International Conference on Data Engineering*, pp. 24. 2006

[2] L.Sweeney. k-anonymity: “A model for preserving privacy. International Journal on Uncertainty”, *Fuzziness and Knowledge-based Systems*, vol. 10, no.5, pp.557-570, 2002.

[3] N.Li, T.Li, “S.Venkatasubramanian. t-Closeness: Privacy Beyond k-anonymity and l-diversity,” *In proceedings of IEEE 23rd Conference on Data Engineering*, ICDE 2007

[4] S.Lodha, D.Thomas. “Probabilistic Anonymity,” *PinKDD Workshop with KDD* 2007

[5] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest *Introduction to Algorithms,* pp. 579~631, 1990

[6] X.Sun, H.wang, J.Li, T.M.Truta, P.Li. (p+, α)-sensitive k-anonymity: “A new enhanced privacy protection model,” *IEEE* 2008

[7] 변창우, 김재환, 이향진, 강연정, 박석, “안전한 데이터베이스 환경에서 삭제 시 효과적인 데이터 익명화 유지 기법,” *정보보호학회 논문지,* 2007

[8] "Earth movers distance" 2009년 10월 검색 *http://en.wikipedia.org/wiki/Earth_mover's_distance*

저 자 소 개

정민경 (Minkyoung Jung)
2003년 : 계명대 회계학과 졸업
2006년 : 계명대 컴퓨터공학과 석사졸업
2006년 ~ 현재 : 동대학원 박사과정

관심분야 : 스트림데이터 처리, 데이터베이스 보호 기술
Phone : 011-515-5639
E-mail : skallet@kmu.ac.kr

홍동권 (Dong-Kweon Hong)
1985년 : 경북대학교 전자과 공학사.
1992년 : U. of Florida 컴퓨터공학 석사
1995년 : U. of Florida 컴퓨터공학 박사.
1997년 ~ 현재 : 계명대컴퓨터공학과 교수

관심분야 : XML, 데이터베이스, 질의 최적화
Phone : 053-580-5281
Fax : 053-580-5165
E-mail : dkhong@kmu.ac.kr