동적 데이터를 위한 프라이버시 보호 기법†

이주창, 김응모
성균관대학교 정보통신공학부
{lordeath, umkim}@ece.skku.ac.kr

Privacy Preserving Data Publication of Dynamic Datasets

Joochang Lee, Ung Mo Kim
School of Information and Communication Engineering
Sungkyunkwan University

요 약

정보기술의 발달로 정보를 수집, 관리, 공유하기가 용이해 짐에 따라 여러 조직이나 기관에서는 개인정보를 수집해 관리하고 있다. 수집한 개인정보를 통계나 연구 등을 목적으로 배포할 때 개인의 프라이버시를 보호하기 위해 *k*-anonymity 와 *l*-diversity 원리가 제안되었고 이를 기반으로 하는 프라이버시 보호 기법들이 제안되었다. 그러나 기존 방법들은 정적인 데이터를 단 한번 배포하는 것을 가정하기 때문에 지속적으로 데이터에 삽입이나 삭제가 발생하는 동적 데이터 환경에 그대로 적용하기 적합하지 않다. 본 논문에서는 동적 데이터 환경에서 *l*-diversity 을 유지하면서 데이터 삽입과 삭제를 효율적으로 처리할 수 있는 기법을 제안한다. 제안 기법은 일반화를 사용하지 않기 때문에 일반화에서 발생하는 정보의 손실이 발생하지 않고 삽입과 삭제의 처리가 간단한 것이 특징이다.

1. 서론

정보기술의 발달로 정보를 수집, 관리, 공유하기가 용이해 짐에 따라 여러 조직에서 개인정보를 수집해 관리하고 있다. 이렇게 수집된 데이터는 연구 등을 목적으로 마이크로데이터(microdata)의 형태로 배포한다. 예를 들어 병원에서 환자의 진료 기록을 연구 목적으로 배포할 수 있다. 마이크로데이터는 미리 집계, 요약되지 않은 테이블 형태의 데이터로 이용자가 직접 분석, 활용할 수 있는 장점이 있다.

마이크로데이터를 배포할 때는 프라이버시를 보호하기 위해 개인을 유일하게 식별할 수 있는 이름이나 주민번호와 같은 속성(attribute)은 제거한다. 그러나 최근 연구에서 이와 같은 방법만으로는 충분히 프라이버시를 보호할 수 없다고 지적되었다[1]. 그 이유는 나이, 성별 등과 같은 속성을 결합해 부식별자(quasi-identifier)로 이용해 데이터에 포함된 개인을 식별하는데 사용할 수 있기 때문이다. 이러한 속성은 데이터의 분석을 위해 필요하기 때문에 삭제하지 않는다.

표 1a 와 같은 병원 진료 기록이 배포되었다고 가정하자. 공격자는 또 다른 곳에서 배포된 표 1b 와 같은 투표자 등록 데이터를 외부 테이블로 가지고 있다. 두 테이블을 이용하면 공격자는 철수의 병명이 간염이라는 것을 알 수 있다. 병원 데이터에서 병명과 같이 노출되지 않도록 보호해야 할 속성을 민감한(sensitive) 속성이라 하고 이러한 공격 형태를 일컬어 링크 공격(link attack)이라 한다.

†본 연구는 정보통신부 및 정보통신연구진흥원의 대학 IT 연구센터 지원사업의 연구결과로 수행되었음 (IITA-2007-C1090-0701-0028)

프라이버시 침해 문제를 해결하기 위해 *k*-anonymity[1]와 *l*-diversity[2] 원리가 제안되었다. *k*-anonymity 는 각 레코드가 최소 *k*-1 개의 다른 레코드와 구분되지 않도록 하여 프라이버시를 보호하는 기법이고 *l*-diversity 는 레코드가 가질 수 있는 민감한 속성의 확률이 최대 1/*l* 이 되도록 제한하는 방법이다. *k*-anonymity 와 *l*-diversity 를 충족시키기 위한 기법으로 부식별자 값을 일반화(generalization) 하거나 은폐(suppression)을 하여 데이터 집합을 같은 부식별자를 가지는 레코드들의 그룹들로 나누는 익명화 방식[1, 6, 7]과 레코드들의 민감한 속성을 교환하거나[8] 또는 일반화를 하지 않으면서 마이크로데이터를 두 개의 테이블로 분리하여[3] 그룹으로 나누는 익명화 방식이 있다. 표 2b 는 표 2a 를 2-anonymity 와 2-diversity 를 충족하도록 일반화 하여 익명화 한 테이블이다. 각 레코드는 최소 다른 1 개의 레코드와 구별되지 않는다. 이렇게 서로 구분되지 않는 레코드들의 집합을 동등 클래스(equivalent class)라 한다.

종래의 익명화 방법들은 정적인(static) 데이터를 배포하는 상황, 즉, 데이터를 배포하는 순간에 모든 데이터가 준비되어 있어 단 한 번 배포하는 상황을 가정하고 있다. 그러나 지속적으로 데이터가 변경되는 환경에서 최근의 정보를 제공하기 위해서는 데이터가 삽입되거나 삭제되는 경우를 고려해야 한다. 이런 환경에서 정적인 데이터를 대상으로 하는 기법들을 그대로 적용할 경우 이전에 배포된 데이터와 새로 배포된 데이터간의 연관 관계로 인해 프라이버시가 침해될 수 있다.

[10]에서는 이러한 문제를 처음 제기하고 데이터를

나이	ZIP	성별	병명
22	11000	M	간염
28	12000	F	감기
33	23000	M	위염
34	25000	F	당뇨
40	29000	F	감기

(a) 병원 진료 데이터

이름	나이	ZIP	성별
상호	39	14000	M
영희	28	12000	F
철수	22	11000	M
길동	41	22000	M
윤정	50	26000	F

(b) 유권자 등록 데이터

표 1. 링크 공격에 취약한 테이블

여러 번 배포함으로써 발생할 수 있는 프라이버시 위협요소를 기술하고 점진적인(incremental) 데이터에 대해 프라이버시를 보호할 수 있는 기법을 제시하였다. 그러나 데이터 삭제를 고려하지 않기 때문에 삽입과 삭제가 모두 발생하는 환경에는 적용할 수 없다. [9]에서는 이러한 점을 지적하고 삭제가 일어나는 경우 삽입되는 레코드를 이용해 다시 일반화를 하거나 모조 레코드(counterfeit)를 삽입해 동적 데이터에 대한 프라이버시 보호 기법을 제시하였다. 그렇지만 일반화에 기반하고 있기 때문에 정보의 손실이 발생하며 프라이버시 침해가 발생했을 때 그 영향이 이전에 배포했던 데이터에까지 전파된다는 단점이 있다.

본 논문에서는 *l*-diversity 원리를 기반으로 동적인 데이터 환경에서 삽입과 삭제를 효율적으로 처리할 수 있는 기법을 제안한다. 제안 기법은 일반화를 사용하지 않기 때문에 일반화에서 발생하는 정보의 손실이 발생하지 않고 삽입과 삭제의 처리가 간단하다는 것이 장점이다.

본 논문의 구성은 다음과 같다. 2 절에서는 동적 데이터에서 발생할 수 있는 프라이버시 침해 상황과 기존의 프라이버시 보호 방법에 기술한다. 3 절에서는 문제를 정의하고 4 절에서 동적 데이터의 프라이버시 보호 기법을 제안한다. 5 절에서 관련 연구 동향을 소개하고 6 절은 결론과 향후 연구과제를 기술한다.

2. 동적 데이터의 프라이버시 보호

본 절에서는 삽입과 삭제가 일어나는 동적인 데이터를 여러 번 배포할 때 발생할 수 있는 문제점에 대해 기술한다.

데이터를 여러 번 배포할 때 기존 정적인 데이터에 기반한 방법을 사용해 익명화 또는 그룹화를 실행해 배포되는 각 테이블이 *k*-anonymity 와 *l*-diversity 을 만족시키도록 하는 것 만으로는 충분하지 않다. 두 개 이상의 테이블로부터 유추할 수 있는 정보로 인해 *k*-anonymity 나 *l*-diversity 가 붕괴되는 경우가 있을 수 있다.

표 2b 를 먼저 배포된 테이블이라 하고 표 3b 를 나중에 배포된 테이블이라 가정하자 (표 2b 에서 삭제되고 표 3b 에서 추가된 레코드를 기울임꼴로 표시하였다). 두 테이블을 따로 봤을 땐 모두 2-anonymous 하고 2-diverse 하지만 공격자는 두 테이블 간의 연관 관계를 이용해 민감한 속성을 유추할 수 있다.

이름	나이	ZIP	병명
철수	22	11000	간염
영호	28	12000	감기
민재	37	17000	폐렴
영희	30	21000	위궤양
지훈	33	23000	위염
수진	34	25000	당뇨
동원	39	26000	빈혈
은정	40	29000	골절
재영	46	31000	간염
유진	50	34000	폐암

(a) Microdata T(1)

GID	나이	ZIP	병명
1	[22,28]	[11k,12k]	간염
1	[22,28]	[11k,12k]	감기
2	[30,37]	[17k,21k]	폐렴
2	[30,37]	[17k,21k]	위궤양
3	[33,34]	[23k,25k]	위염
3	[33,34]	[23k,25k]	당뇨
4	[39,40]	[26k,29k]	빈혈
4	[39,40]	[26k,29k]	골절
5	[46,50]	[31k,34k]	간염
5	[46,50]	[31k,34k]	폐암

(b) Generalization T'(1)

표 2. 첫 번째 배포된 마이크로데이터

이름	나이	ZIP	병명
철수	22	11000	간염
영희	30	21000	위궤양
지훈	33	23000	위염
미연	29	22000	장염
동원	39	26000	빈혈
은정	40	29000	골절
재영	46	31000	간염
민석	43	47000	폐암
현석	51	38000	디스크
정민	59	44000	당뇨

(a) Microdata T(2)

GID	나이	ZIP	병명
1	[22,30]	[11k,21k]	간염
1	[22,30]	[11k,21k]	위궤양
2	[29,33]	[22k,23k]	위염
2	[29,33]	[22k,23k]	장염
3	[39,40]	[26k,29k]	빈혈
3	[39,40]	[26k,29k]	골절
4	[43,46]	[31k,47k]	간염
4	[43,46]	[31k,47k]	폐암
5	[51,59]	[38k,44k]	디스크
5	[51,29]	[38k,44k]	당뇨

(b) Generalization T'(2)

표 3. 두 번째 배포된 마이크로데이터

예제 1 공격자가 *철수*가 두 테이블에 모두 포함되어 있다는 사실을 알고 있다면 다음과 같이 유추 할 수 있다. 표 2b 로부터 *철수*의 병명이 간염 또는 감기라는 사실을 알 수 있다. 표 3b 로부터 *철수*의 병명이 간염 또는 폐렴 이라는 사실을 알 수 있다. 따라서 두 사실을 결합해 *철수*의 병명이 간염이라고 결정한다. 표 3b 를 다른 방법으로 일반화 하더라도 *철수*의 병명이 간염이라는 사실은 항상 노출된다. 표 2b 에서 *철수*와 가질 수 있는 병명은 간염과 감기 두 가지 경우였는데 데이터에 변경이 일어나 표 3a 에는 감기를 병명으로 가지는 레코드가 없다. [9]에서는 이러한 현상을 치명적 결여(critical absence)라 칭하였다. 치명적 결여는 레코드가 삭제되는 경우에 발생한다는 점에 주목할 필요가 있다. 삽입만 발생하는 경우는 이러한 현상이 일어나지 않는다. □

2.1 m-invariance 원리

본 절에서는 레코드 삽입/삭제가 일어나는 데이터에서 프라이버시 보호 문제를 논한 m-invariance 기법 [9]을 소개한다.

예제 1 에서 *재영*의 레코드를 고려하자. 재영이 속한 동등 클래스에서 *유진*의 레코드가 삭제되고 *민석*의 레코드가 추가 되었지만 *재영*이 가질 수 있는 병명은 간염 또는 폐암으로 유지되었다. 즉, 동등 클래스 내에서 민감한 속성의 값들이 변하지 않으면 해당 클래스에 속한 레코드들의 민감한 속성이 노출되지 않는다. 이를 m-invariance 원리라 한다. m-invariant 일반화 기법은 삭제되는 레코드와 삽입되는 레코드의

이름	GID	나이	ZIP	병명
철수	1	[22,30]	[11k,21k]	간염
c1	1	[22,30]	[11k,21k]	감기
c2	2	[30,37]	[17k,21k]	폐렴
영희	2	[30,37]	[17k,21k]	위궤양
지훈	3	[33,59]	[23k,44k]	위염
정민	3	[33,59]	[23k,44k]	당뇨
동원	4	[39,40]	[26k,29k]	빈혈
은정	4	[39,40]	[26k,29k]	골절
재영	5	[43,46]	[31k,47k]	간염
민석	5	[43,46]	[31k,47k]	폐암
미연	6	[29,51]	[22k,38k]	장염
현석	6	[29,51]	[22k,38k]	디스크

(a) 모조레코드를 이용한 일반화

GID	Count
1	1
2	1

(b) 모조레코드 통계

표 4. m-invariant 일반화 기법

민감한 속성 값이 같으면 삭제 되는 레코드를 삽입된 레코드로 대체하여 다시 일반화 하고 치명적 결여가 발생했을 때는 모조 레코드를 삽입한다.

표 4a 는 m-invariant 일반화 기법으로 표 3a 를 일반화한 테이블이다. 표 4b 는 동등 클래스별 모조레코드 개수를 저장하고 있는 보조테이블로 표 4a 와 함께 배포된다.

m-invariant 일반화 기법의 단점은 다음과 같다. 첫째, 삭제되는 레코드를 삽입되는 레코드를 이용해 다시 일반화하는 과정에서 발생하는 정보의 손실이 커질 수 있다. 표 2b 에서 *지훈*의 레코드에서 나이 속성은 [33,34] 구간으로 일반화 되었지만 표 4a 에서는 [33,59] 구간으로 일반화 되었다. 이와 같은 일반화에 의한 정보 손실은 데이터의 유용성(utility)을 감소시킨다. 둘째, 삭제되는 레코드와 삽입되는 레코드의 민감한 속성의 값이 다르면 모조레코드를 삽입해야 한다는 것이다. 모조레코드 통계를 담고 있는 보조테이블을 같이 배포하더라도 결국 모조레코드를 삽입한다는 것은 달리 말하면 삭제를 하지 않는 것과 동일하다. 최악의 경우 삽입되는 모든 레코드의 민감한 속성 값이 삭제되는 레코드와 전혀 다를 경우 모조 레코드 수는 삭제되는 레코드 수와 같다. 셋째, 배포된 특정 마이크로데이터에 프라이버시 침해가 발생하면 그 효과가 이전/이후에 배포한 마이크로데이터로 전파된다는 것이다. 표 2b 와 표 4a 가 차례로 배포되었고 표 4a 에서 *철수*의 병명이 간염이라는 사실이 유출되었다고 가정하자. 공격자는 이전에 배포된 표 2b 에서 *영호*의 병명이 감기라는 사실을 유추할 수 있다.

3. 문제 정의

배포할 마이크로데이터 테이블을 T 라 하자. T 의 컬럼들은 레코드를 유일하게 식별하는 ID 와 n 개의 부식별자 $QI_1,...,QI_n$, 그리고 민감한 속성 S 로 구성된다. 예제 1 의 경우 환자의 병명이 S 가 된다. 본 논문에서는 보호해야 할 한 개의 민감한 속성 S 가 있고 S 는 이산적인 값을 가지는 범주형(categorical) 데이터로 가정한다. 부식별자 $QI_1,...,QI_n$ 는 범주형이거나 숫자형(numerical) 데이터가 될 수 있다.

시간이 지남에 따라 T 에 삭제와 삽입 연산이 발생하고 데이터 소유자는 특정 시간에 프라이버시 침해 없이 익명화 된 T 를 배포하고자 한다. i 번째 배포할 시점의 T 의 스냅샷을 $T(i)$ 라 하자.

정의 1 (부식별자) 부식별자는 테이블 T 에서 민감하지 않은 속성 $QI_1,...,QI_n$ 들의 집합으로 외부테이블과 연결되어 테이블에 포함된 개인의 민감한 속성 S 를 식별하기 위해 사용된다. 부식별자를 일반화해 익명화된 테이블에서 같은 부식별자 값을 가지는 레코드들의 집합을 **동등 클래스**라 한다. 전체 동등 클래스의 개수를 m, j 번째 동등 클래스를 EC_j 라 하면 $\bigcup_{j=1}^{m} EC_j = T$ 이고 어떤 $1 \le j < k \le m$ 에 대해 $EC_j \cap EC_k = \emptyset$ 이다.

정의 2 (*l*-diversity 원리) 어떤 동등 클래스 EC_j 내에서 가장 빈번히 발생하는 민감한 속성의 개수와 클래스에 속하는 전체 레코드 개수의 비율이 $1/l$ 보다 작으면 EC_j 는 l-diversity 를 만족한다. 모든 $1 \le j \le m$ 에 대해 EC_j 가 l-diversity 를 만족하면 테이블 T 는 l-diversity 를 만족한다.

정의 3 (프라이버시 침해) 익명화 된 테이블의 전체 집합을 $U = \bigcup_{i=1}^{n} T'(i)$ 라 하자. 공격자가 $T'(1),...T'(n)$ 을 이용해 특정 개인의 레코드 $t \in U$ 의 민감한 속성 S 의 정확한 값을 유추할 수 있다면 프라이버시가 침해되었다고 정의한다.

정의 4 (프라이버시를 보호하는 데이터 재배포 문제) 공격자가 이전에 배포된 $n-1$ 개의 T 를 익명화한 테이블 $T'(1),...T'(n-1)$ 를 가지고 있다고 가정하자. 데이터 재배포 문제의 첫 번째 요구사항은 $T'(1),...T'(n)$ 로부터 프라이버시 침해가 발생하지 않도록 방지하면서 $T(n)$ 을 $T'(n)$ 으로 익명화하는 것이다. 두 번째는 익명화 과정에서 손실되는 정보의 손실을 최소화하는 것이다.

4. 제안기법

본 절에서는 동적인 데이터 환경에서 일반화를 하지 않고 l-diversity 을 유지하면서 삽입과 삭제를 처리할 수 있는 기법을 제안한다.

4.1 데이터 표현

k-anonymity 나 l-diversity 에 기반한 종래의 익명화 기법은 레코드를 여러 개의 그룹으로 나누는 접근 방법을 취한다. 이러한 레코드 그룹화 기법들의 단점은 같은 그룹에 속한 레코드들의 익명성이 서로 다른 레코드에 의존한다는 것이다. 어떤 그룹 내에서 레코드의 삽입이나 삭제가 발생하면 같은 그룹에 속한 다른 레코드들에게 영향을 미친다. 따라서 제안 기법에서는 레코드들을 그룹화 하는 대신 각 레코드가 가질 수 있는 민감한 속성 값들을 따로 저장한다.

이름	나이	ZIP	Row ID
철수	22	11000	1
영호	28	12000	2
민재	37	17000	3
영희	30	21000	4
지훈	33	23000	5
수진	34	25000	6
동원	39	26000	7
은정	40	29000	8
재영	46	31000	9
유진	50	34000	10

(a) 부식별자 테이블

Row ID	병명	확률
1	간염	0.5
1	감기	0.5
2	간염	0.5
2	감기	0.5
3	폐렴	0.5
3	위궤양	0.5
4	폐렴	0.5
4	위궤양	0.5
…	…	…
10	간염	0.5
10	폐암	0.5

(b) 민감한 속성 테이블

표 5. 제안 기법으로 변형된 테이블

표 5 에서 제안 기법을 사용해 표 2a 의 마이크로데이터를 변형한 모습을 보이고 있다. 기존 익명화 기법을 사용해 그룹화된 결과를 이용해 원본 테이블을 부식별자 테이블과 민감한 속성 테이블로 분리하고 각 레코드 마다 가질 수 있는 민감한 속성, 즉, 병명을 확률과 함께 따로 저장한다. 두 테이블에는 Row ID 컬럼이 추가되어 조인 연산에 사용된다. 제안 기법에서는 부식별자 테이블의 속성들을 일반화 하지 않는데 일반화를 사용하지 않더라도 공격자가 알아내고자 하는 개인이 배포된 데이터에 포함되어 있다는 사실을 알고 있으면 민감한 속성을 알아낼 수 있는 확률은 기존 일반화 기법과 차이가 없다.

4.2 삽입/삭제 처리

4.1 절에서 기술한 기법을 이용해 데이터를 저장하면 간단한 방법을 사용해 레코드 삽입과 삭제를 처리할 수 있다. 레코드 삭제는 삭제할 레코드에 해당하는 행들을 두 테이블에서 단순히 삭제하면 된다. 그룹화를 하지 않기 때문에 특정레코드가 삭제되더라도 다른 레코드들이 가질 수 있는 민감한 속성의 확률은 변화가 없다. 레코드 삽입은 삽입할 레코드들을 기존 익명화 기법을 이용해 익명화 한 후 그 결과를 4.1 절의 방법대로 변형시키고 이를 추가한다. 이 때 삽입할 레코드들은 적격(eligibility) 조건[2]을 만족해야 한다. 즉, 삽입할 레코드들의 전체 건수가 n 개 일 때 가장 자주 발생하는 민감한 속성 값을 가진 레코드의 수가 n/l 보다 크지 않아야만 l-diverse 한 그룹화가 가능하다.

5. 관련 연구

k-anonymity 원리와 이를 기반으로 일반화와 은폐를 사용해 데이터 배포시 프라이버시를 보호하는 기법은 Samarati 와 Sweeney 에 의해 처음 소개되었다[1, 6]. 이후 k-anonymity 요구사항을 만족시키기 위한 여러 알고리즘들이 제안되었다[3, 5, 7, 8].

k-anonymity 모델은 익명성을 제공하기는 하지만 민감한 속성의 값의 분포에 관한 요구사항은 정의하지 않기 때문에 완벽한 익명화 기법은 아니다. Machanavajjhala el al. 은 이러한 점을 지적하고 동등클래스 내에서 민감한 속성들을 가질 수 있는 확률을 최대 $1/l$ 로 제한하는 l-diversity 모델을 제안하였다[2]. 그 외에도 동등클래스 내의 민감한 속성의 확률 분포를 전체 데이터의 확률 분포와 근접하도록 하는 t-closeness[4], 수치(numeric) 값을 가지는 민감한 속성에 대해 동등클래스 내 민감한 속성 값의 구간을 고려하는 (k,e)-anonymity[8] 기법들이 데이터 익명화 요구사항으로 제안되었다.

기존 익명화 기법들은 데이터를 단 한번 배포하는 상황에는 적합하지만 삽입과 삭제가 발생하는 동적 데이터에는 그대로 적용할 수 없다. Byun 은 이러한 문제를 제기하고 점진적인 데이터에 대해 익명성을 유지할 수 있는 기법을 제안하였지만 레코드 삭제를 고려하지 않고 있다[10]. Xiao 는 삽입과 삭제를 모두 처리할 수 있는 m-invariant 일반화 기법을 제안하였다[9].

6. 결론

본 논문에서는 삽입과 삭제가 발생하는 동적인 데이터 환경에서 마이크로데이터를 여러 번 배포할 때 간단하게 l-diversity 요구사항을 충족시킬 수 있는 기법을 제안하였다. 제안 기법은 일반화를 사용하지 않기 때문에 일반화에서 발생하는 정보의 손실이 발생하지 않고 삽입과 삭제의 처리가 간단하다는 것이 장점이다.

향후 연구 과제로는 제안된 데이터 표현 방법을 사용하는 새로운 익명화 알고리즘과 수치 데이터 형식의 민감한 속성에 적용할 수 있는 익명화 기법이 있다.

참고문헌

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

[2] A. Machanavajjhala, J. Gehrke, and D. Kifer. l-Diversity: Privacy beyond k-anonymity. In ICDE, 2006.

[3] X. Xiao and Y. Tao. Anatomy: Simple and effective privacy preservation. In VLDB Conference, 2006.

[4] N. Li, T. Li, and S. Venkatasubramanian. t-closeness: Privacy beyond k-anonymity and l-diversity. In ICDE, 2007.

[5] D. Kifer and J. Gehrke. Injecting utility into anonymized datasets. In SIGMOD 2006.

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

[7] K. LeFevre, D. J. Dewitt, and R. Ramakrishnan. Incognito: Effective full-domain k-anonymity. In SIGMOD, 2005.

[8] Q. Zhang, N. Koudas, D. Srivastava, and T. Yu. Aggregate query answering on anonymized tables. In ICDE, 2007.

[9] X. Xiao and Y. Tao. m-Invariance: Towards privacy preserving re-publication of dynamic datasets. In SIGMOD, 2007.

[10] J.-W. Byun, Y. Sohn, E.Bertino, and N. Li. Secure anonymization for incremental datasets. In SDM, pages 48-63, 2006.