관계형 데이터베이스에서 데이터 그룹화를 이용한 익명화 처리 기법

The De-identification Technique Using Data Grouping in Relational Database

저자
(Authors)

박준범, 진승헌, 최대선
Jun-Bum Park, Seung-Hun Jin, Daeseon Choi

출처
(Source)

정보보호학회논문지 25(3), 2015.6, 493-500(8 pages)
Journal of the Korea Institute of Information Security & Cryptology 25(3), 2015.6, 493-500(8 pages)

발행처
(Publisher)

한국정보보호학회
Korea Institute Of Information Security And Cryptology

URL

http://www.dbpia.co.kr/journal/articleDetail?nodeId=NODE06380845

APA Style

박준범, 진승헌, 최대선 (2015). 관계형 데이터베이스에서 데이터 그룹화를 이용한 익명화 처리 기법. 정보보호학회논문지, 25(3), 493-500

Journal of The Korea Institute of Information Security & Cryptology ISSN 1598-3986(Print)
VOL.25, NO.3, Jun. 2015 ISSN 2288-2715(Online)
http://dx.doi.org/10.13089/JKIISC.2015.25.3.493

관계형 데이터베이스에서 데이터 그룹화를 이용한 익명화 처리 기법

박 준 범,[1†] 진 승 헌,[2] 최 대 선[2‡]
[1]과학기술연합대학원대학교, [2]한국전자통신연구원

The De-identification Technique Using Data Grouping in Relational Database

Jun-Bum Park,[1†] Seung-Hun Jin,[2] Daeseon Choi[2‡]
[1]Korea University of Science and Technology
[2]Electronics and Telecommunications Research Institute

요 약

정부 3.0 공공정보 공유 및 개방, 소셜네트워크서비스의 활성화 그리고 사용자 간의 공유 데이터 증가로 인터넷상에 노출되는 사용자의 개인 정보가 증가하고 있다. 이에 따라 프라이버시를 지키기 위한 익명화 알고리즘이 등장하였으며 관계형 데이터베이스에서의 익명화 알고리즘은 k-익명성(k-anonymity)을 시작으로 ℓ-다양성(ℓ-diversity), t-밀집성(t-closeness)으로 발전하였다. 익명화 알고리즘의 성능 향상 부분은 계속해서 효율적인 방법이 제안되고 있지만, 기업이나 공공기관에서는 알고리즘 성능의 향상보다는 전체적인 익명화 처리 방법이 필요한 실정이다. 본 논문에서는 관계형 데이터베이스에서 데이터의 그룹화를 이용하여 k-익명성, ℓ-다양성, t-밀집성 알고리즘을 처리하는 과정을 구체화하였다.

ABSTRACT

Personal information exposed in the Internet is increasing by the public data opening and sharing, vitalization of SNS(Social Network Service) and growth of information shared between users. Exposed personal information in the Internet can infringe upon targeted users using linkage attack or background attack. To prevent these attack De-identification models were appeared a few years ago. The 'k-anonymity' has been introduced in the first place, and the 'ℓ-diversity' and 't-closeness' have been followed up as solutions, and diverse algorithms have been being suggested for performance improvement nowadays. However, industry or public sectors actually needs a whole solution as a system for the de-identification process rather than performance of the de-identification algorithm. This paper explains a way of de-identification techique for 'k-anonymity', 'ℓ-diversity', and 't-closeness' algorithm using QI(Quasi-Identifier) grouping method in the relational database.

Keywords: K-anonymity, L-diversity, T-closeness, De-identification Algorithms

I. 서 론

정부 3.0 공공정보 공유 및 개방, 소셜네트워크서비스의 활성화 그리고 인터넷 사용자 간의 공유 데이터 증가 등으로 인해 인터넷상에 노출되는 개인 정보가 증가하고 있다. 특히 정부에서는 정부 3.0의 목표로 2017년까지 7.7억 건의 공공정보를 공개할 계획에 있다.[1] 공공정보로 개방된 정보는 인터넷에 공개된 개인 정보와 연결될 경우 사용자 식별이 가능하게 된다. 익명화된 정보와 다른 정보를 결합하여

접수일(2014년 10월 21일), 수정일(1차: 2015년 2월 23일, 2차: 2015년 4월 8일), 게재확정일(2015년 5월 6일)
† 주저자, usingideal@gmail.com
‡ 교신저자, sunchoi@etri.re.kr(Corresponding author)

개인을 식별하는 것을 재식별이라고 하며 다음과 같은 사례가 있다. Sweeney[2]에서는 메사추세스 주민의 의료기록을 익명화하여 공개하였는데 20달러에 구입한 선거인 명부와 대조하여 주지사의 의료기록을 재식별한 사례를 소개하고 있으며, Barbaro[3]에서는 AOL이 60만 명의 검색어 기록을 공개하였는데 뉴욕타임즈에서 전화번호 목록과 대조하여 재식별한 사례를 소개하고 있다. Narayanan[4]에서는 Netfilx에서 50만 명의 영화평을 공개하고 이를 통해 영화 만족도를 예측하는 대회를 개최하였는데, 익명화하여 제공된 영화평의 신원이 식별된 사례를 제시하였다. 이러한 관계형 데이터베이스 형태로 공개된 정보들을 이용한 재식별을 막기 위해서 대표적으로 k-익명성[5], ℓ-다양성[6], t-밀집성[7] 익명화 모델이 제안되었다.

k-익명성 익명화 모델은 동일한 준식별자(quasi-identifier)로 구성된 튜플들을 k개 이상으로 하여 민감정보(sensitive information)를 특정할 가능성을 낮춘 것이다. 하지만 k-익명성이 적용되어도 민감정보를 특정할 수 있는 경우가 있었는데 그 이유는 k-익명성을 만족하는 그룹의 민감정보 엔트로피가 낮을 경우였다. 예를 들어서 민감정보의 90%가 같을 경우 공격자는 k-익명성이 적용되었더라도 90%의 확률로 민감정보를 특정할 수 있게 된다[6]. 또한, 민감정보 대부분이 비슷한 경우에도 공격자는 민감정보의 평균을 예측할 수 있게 된다. 이런 이유로 k-익명성 모델은 민감정보를 통해 특정인을 식별할 수 있는 문제점이 있었다[7]. k-익명성 모델의 단점을 보완하기위해 ℓ-다양성 모델과 t-밀집성 모델이 제안되었다. ℓ-다양성 모델은 민감정보의 종류가 ℓ개 이상이 되게 하는 것이었다. k-익명성을 만족하는 그룹에서 민감정보의 종류를 다양하게 함으로써 특정인을 식별할 가능성을 낮추었다. t-밀집성은 민감정보의 분포도를 측정하여 특정부분에 정보가 밀집되는 경우를 막는 것이었다. 만약 l-다양성을 만족한 그룹에서 민감정보 대부분이 비슷하다면 특정인의 민감한 정보를 예측할 수 있게 되는데 이를 막기위한 모델이 t-밀집성이다. t-밀집성은 주로 EMD(Earth's Mover Distance)를 이용하여 데이터 분포도를 측정하는데 EMD를 사용하기 위해서는 사전에 민감정보들을 모두 유사도 순으로 정렬해 놓아야 한다. k-익명성, l-다양성, t-밀집성 모델은 모두 관계형 데이터베이스에서 프라이버시를 강화하기 위해 제안된 모델이다. 외국에서는 이러한 익명화

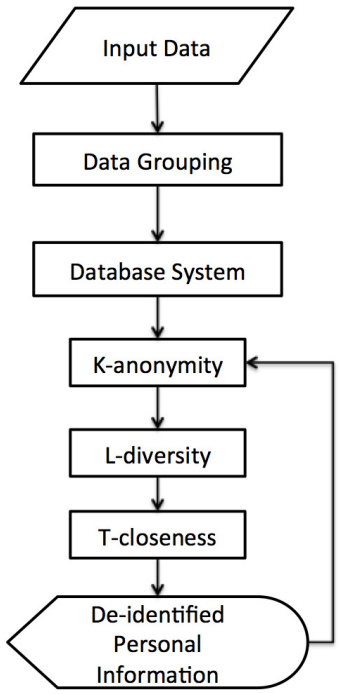

Fig. 1. The de-identification system

모델을 적용시킨 소프트웨어가 [8,9,10] 상업용 혹은 베타 버전으로 배포되어 지고 있다. 하지만 우리나라의 경우 프라이버시의 대한 인식이 최근에 들어서 중요시되고 있기 때문에 아직 배포되어지는 익명화 소프트웨어가 없는 상황이다. 본 논문에서는 익명화 시스템 구현에 도움이 되기위해 전체적인 익명화 처리 방법을 구체적으로 설명하며 데이터 그룹화를 이용해 각 익명성 모델을 만족하는 과정을 보이고 [8]에서 무료로 배포하는 데이터를 이용해 각 모델별로 익명화 처리하여 결과를 분석해보았다. 본 논문의 구성은 다음과 같다.

본 논문은 Ⅱ장에서는 전체적인 익명화 처리 시스템을 다루고 Ⅲ장에서는 각각의 익명화 처리 과정을 설명하며 Ⅵ장에서는 익명화 처리 분석을 그리고 Ⅴ장에서 결론을 맺는다.

Ⅱ. 제안하는 익명화 처리 시스템 구조

Fig.1. 은 본 논문에서 제안하는 데이터베이스를 적용한 익명화 처리 시스템을 나타낸 것이다. 안정적인 시스템 설계를 위해 각 알고리즘 마다 변경된 값들은 데이터베이스에 업데이트되도록 구성하였으며

k-익명성, ℓ-다양성, t-밀집성을 개별적으로 데이터베이스에서 확인할 수 있다. 익명화 처리된 데이터가 익명화 모델에서 설정한 수치를 만족하지 못할 경우 일반화, 마스킹처리, 범위설정 등의 방법[11]을 사용할 수 있는데 본 시스템에서는 마스킹처리 하여 익명화 모델을 만족하도록 하였다.

2.1 익명화 처리 방법

Fig.2.는 익명화 처리 시스템을 통해 마스킹 처리되어지는 과정을 나타낸 것이다. 마스킹된 정도에 따라 각각의 다른 그룹에 속하게 된다. 그룹을 나눈 이유는 k-익명성 알고리즘을 효과적으로 실행시키기 위해서이다. 만약 1개만 마스킹 된 그룹과 2개가 마스킹 된 그룹이 섞여 있다면 k-익명성 알고리즘을 실행할 때 불필요한 연산을 하게 되기 때문이다. 마스킹 처리 과정은 사용자가 설정한 익명화 알고리즘 수치를 만족할 때까지 진행된다. 사용자는 마스킹 되지 않은 그룹을 기본값(default)으로 볼 수 있으며 전체 데이터를 보고 싶은 경우 모든 그룹에 속한 데이터를 가져올 수 있다.

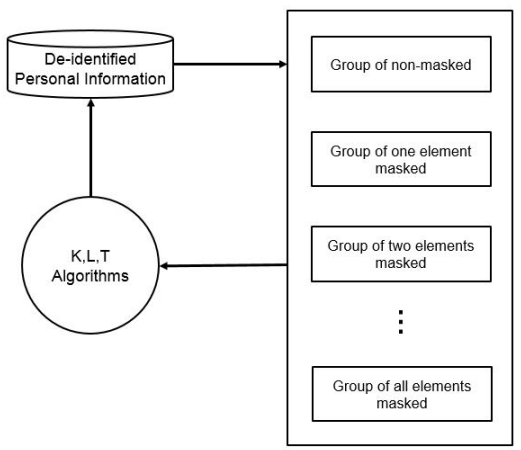

Fig. 2. The process of data masking

III. 데이터 그룹화 익명화 기법

Ⅲ장에서는 익명화 처리 과정에서는 전체적인 익명화 시스템 구성과 k-익명성, ℓ-다양성, t-밀집성의 구체적인 알고리즘 처리 과정을 다룬다. 익명화 처리 시스템 구성을 위해 몽고(mongo)[12] 데이터베이스를 활용하였으며 각 알고리즘 처리 단계에서 데이터베이스와 상호작용하여 그룹화된 값이 익명화 처리 과정을 만족하는지에 대한 유무가 데이터베이스에 업데이트되도록 하였다. k-익명성, ℓ-다양성, t-밀집성의 알고리즘은 파이썬(python) 2.7 버전으로 구현하였으며 데이터 그룹화를 위해 파이썬에서 제공하는 라이브러리인 딕셔너리(dictionary)[13]를 사용하였다. 3.1장에서는 준식별자를 이용한 데이터 그룹화와 k-익명성 알고리즘 구현을 다루며 3.2장에서는 ℓ-다양성과 t-밀집성 알고리즘 구현을 다룬다. 3.3장에서는 익명화 과정을 통해 추출된 데이터의 마스킹 과정에 대해 다룬다.

3.1 그룹화 기법

관계형 데이터베이스에서의 익명화 과정은 대용량의 데이터를 처리해야 할 경우가 많으므로 각 익명화 알고리즘이 효과적으로 진행되도록 해야한다. 본 시스템에서는 준식별자를 이용해 그룹의 키를 생성하여 데이터를 분류하도록 하였으며 알고리즘은 다음과 같다.

data grouping using quasi-identifier

```
1 : Input : PI(Personal Information)
2 : Output : Grouped_PI
3 : grouped_PI = dict()
4 : for identifier, quasi in PI.items():
5 :   key = quasi[0]+quasi[1]+quasi[2]
+quasi[3]
6 :   if key in grouping_PI:
7 :     grouping_PI[key].append(identifier)
8 :   else:
9 :     grouping_PI[key] = []
10:     grouping_PI[key].append(identifier)
11: return grouping_PI
```

대용량의 데이터를 익명화 처리하는 데에는 많은 시간이 소요된다. 그래서 k-익명성, ℓ-다양성, t-밀집성 알고리즘을 적용하기 전에 다음과 같이 그룹화를 해주어야 한다. 1~2번째 줄은 들어오는 데이터와 결과물로 출력되는 데이터를 나타낸다. 3번째 줄의 grouped_PI는 그룹화된 PI(Personal Information)을 나타낸다. 4~5번째 줄은 준식별자를 이용해 그룹의 키를 생성하는 부분이다. 준식별자들을 스트링으로 이어서 그룹의 키를 생성한다. 6~10번째 부분은 생성한 키를 이용해 데이터를 분류

하는 부분이다. 준식별자를 이용해 생성한 키가 그룹에 존재한다면 해당 그룹에 식별자(Identifier)를 추가해 줌으로써 데이터를 그룹화 처리 한다.

3.2 그룹화가 적용된 익명화 처리 기법

데이터 그룹화를 통해 데이터들이 그룹화 되어져 있으므로 k-익명성 알고리즘은 그룹별로 포함된 데이터를 측정함으로써 처리할 수 있다.

basic k-anonymity algorithm

```
1 : Input : grouped_PI, limited_k
2 : Output : k_data
3 : k_data = dict()
4 : for key, identifiers in grouped_PI.items():
5 :   k_anonymity = len(identifiers)
6 :   if k_anonymity >= limited_k :
7 :     k_data[k]=identifiers
8 : return k_data
```

k-익명성 알고리즘은 그룹화된 데이터 배열의 길이를 측정함으로써 구할 수 있다. 4~5번째 줄은 k-익명성을 측정하는 부분이다. 딕셔너리로 그룹화된 데이터에서 식별자 값만을 이용하여 k-익명성을 측정한다. 6~7번째 줄은 측정한 길이가 일정수준을 만족한다면 k_data라는 k-익명성을 만족하는 그룹에 저장하는 부분이다. 이 부분을 계산하면 k-익명성을 만족하는 데이터를 %(퍼센트)로 표현할 수 있다. k-익명성 처리가 그룹 단위로 구현된 상황에서 ℓ-다양성과 t-밀집성 알고리즘은 각각의 개념을 충족하는 배열을 생성함으로써 구현할 수 있다. ℓ-다양성 알고리즘의 슈도코드는 아래와 같다.

basic ℓ-diversity algorithm

```
1 : Input : k_data, limited_l
2 : Ouput : l_data
3 : l_data = dict()
4 : for key, identifiers in k_data.items():
5 :   l_list = []
6 :   for idenfier in identifiers:
7 :     user_info = data[identifier]
8 :     user_si = user_info[4]
9 :     if user_si in l_list :
10:       pass
11:     else :
12:       l_list.append(user_si)
13:   if len(l_list) > limited_l:
14:     l_data[key] = identifiers
15: return l_data
```

ℓ-다양성 알고리즘은 k-익명성 알고리즘을 통해 생성된 k_data 데이터를 입력값으로 받는다. 3번째 줄은 ℓ-다양성을 만족하는 값을 저장하는 l_data 변수를 선언하는 부분이다. 6~8번째 줄은 k-익명성을 만족하는 데이터의 식별자값을 가지고 해당 식별자값의 민감정보를 가져오는 부분이다. 이 부분에서 성능을 향상하려면 해쉬테이블을 만들어 민감정보를 더 빠르게 가져올 수 있다. 9~12번째 줄은 단순히 l_list 배열에 들어갈 민감정보의 중복성을 제거해주는 부분이다. 이 과정을 통해 생성된 배열 l_list의 길이는 l-diversity의 l값이 된다. 13~14번째 줄은 설정한 l-diversity의 l을 만족할 경우 l_data에 저장하는 부분이다. 15번째 줄은 최종적으로 ℓ-다양성을 만족하는 데이터를 반환하는 것을 나타낸다.

basic t-closenesss algorithms

```
1 : Input : l_data, limited_t
2 : Output : t_data
3 : t_data = dict()
4 : for key, identifiers in l_data.items():
5 :   t_length = len(identifiers)
6 :   t_list = []
7 :   for identifier in identifiers:
8 :     user_info = data[identifier]
9 :     user_si = user_info[4]
10:     t_list.append(user_si)
11:     t_list.sort()
12:     t_closeness = EMD(t_list)
13:     if t_closeness < limited_t:
14:       t_data[key] = identifiers
15: return t_data
```

t-밀집성의 기본 개념은 데이터의 분포도를 산정하는 것이기 때문에 ℓ-다양성 알고리즘과 배열을 생성하는 단계까지는 동일하지만 ℓ-다양성과는 달리 민감정보가 원본 그대로 배열에 저장되어야 한다. 10~12번째 줄은 배열에 저장된 데이터가 내림차순으로 정렬된 후에 데이터의 분포도를 측정하는 부분이다. 데이터의 분포도 측정은 EMD(Earth Mover's

Distance)[14]연산을 하여 t값을 산출하며 산출된 t값이 일정 수치를 넘는다면 해당 블록을 일반화 처리하여 익명화 수치를 충족하도록 한다.

basic earth mover's distance algorithms

```
1 : Input : t_list
2 : Output : EMD(Earth Mover's Distance)
3 : total_range=[]
4 : for n in range(100) :
total_range.append(n)
5 : total_length = len(total_info)
6 : static_part = total_length / len(t_list)
7 : extra_part = (float(static_part) %
float(len(t_list)))
8 : extra_part = extra_part.split('.')[0]
9 : balance_value = len(t_list) - (extra_part)
10: active_loop = true
11: count_t=0, hap=0
12: for a in t_list :
13:   count_t += 1
14:   for i in range(static_part) :
15:     gap = int(a)-(total_info[count_m])
16:     if gap < 0:
17:       gap *= (-1)
18:       hap += gap
19:   if count_t>=balance_value and
active_loop==true:
20:     active_loop = false
21:     static_part += 1
22: hap = float(hap) / float(total_length)
23: return hap
```

EMD 알고리즘은 데이터의 분산정도를 구하는 것으로 숫자로 표시되는 속성 값들이 열거된 도메인을 $\{v_1, v_2, \ldots, v_m\}$이라고 하고 v_i는 i번째로 가장 작은 값이라 할 때 다음 식으로 표현된다.[14]

$$\text{ordered\_dist}(v_i, v_j) = \frac{|i-1|}{m-1} \quad (1)$$

위 슈도코드는 수식(1)을 표현한 것이다. 슈도코드를 살펴보면 먼저 데이터를 분산 정도를 측정하기 위해 데이터의 전체적인 범위를 설정해주어야 한다. 3~5번째 줄은 전체적인 범위를 설정해주는 부분으로 설정된 배열에 정수값이 순서대로 삽입되도록 한다. 6~8번째 줄은 EMD를 계산하기 위해 나눗셈 연산을 하게 되는데 이때 나누어 떨어지지 않을 경우 여분으로 연산해주어야 할 때를 계산하는 부분이다. 12번째 줄은 민감정보들을 하나씩 가져와 각각의 데이터의 분산도를 측정하는 부분이며 14~18번째 줄은 데이터간의 거리를 측정하는 부분이다. 19~21번째 줄은 6~8번째 줄에서 계산한 여분의 연산을 하는 부분으로 나누어 떨어지지 않는 수에 대해서는 배열의 마지막 부분에서 처리 한다.

IV. 익명화 처리 분석

4.1 실험 데이터

실험을 위해 991,463개의 데이터를 사용하였으며 사용된 데이터는 무료로 배포되는 CAT(Cornell Anonymization Toolkit)의 샘플 데이터이다[8]. 데이터의 구성은 개인번호, 나이, 성별, 인종, 교육, 소득으로 되어있다. 개인번호는 사용자를 식별할 수 있는 식별자(identifier)를 의미하고 나이, 성별, 인종, 교육은 각각을 조합하여 사용자를 식별할 수 있는 준식별자(quasi-identifier)를 의미하며 소득은 사용자의 민감정보(sensitive information)를 의미한다. Table 1.은 CAT에서 제공하는 샘플데이터의 속성 중 교육에 관한 최종학력을 18가지로 분류한 것으로 스트링(string)값들을 정수형(integer)으로 매핑시킨 것을 나타낸 것이다. 매핑테이블을 만드는 것은 Table 1.의 No.04번과 같이 특정 범위를 지정하여 분류할 수 있다는 장점이 있지만, 각각의 속성에 대해 매핑테이블을 만드는 데에 시간이 소요되는 점과 정수형을 보고 매핑테이블을 연결해 정보를 얻어야 하는 단점이 있다.

4.2 k-익명성, ℓ-다양성, t-밀집성 분석

Fig.3, Fig.4, Fig.5는 k-익명성, ℓ-다양성, t-밀집성 알고리즘을 적용한 결과를 나타낸 것이다. 익명화 알고리즘을 만족하는 데이터가 충분히 많았기 때문에 일반화 처리는 하지 않고 각각의 익명화 알고리즘을 만족하는 데이터를 측정하였다.

일반적인 데스크탑 2 GHz Intel Core i7에서 알고리즘 성능 테스트를 실행한 결과 991,463개의 데이터를 k-익명성, ℓ-다양성, t-밀집성 처리하는데 데이터가 로드되는 시간을 제외하고 약 2.5~3초 정도의 시간이 소요되었다. 전체적인 익명화 처리 과정에서

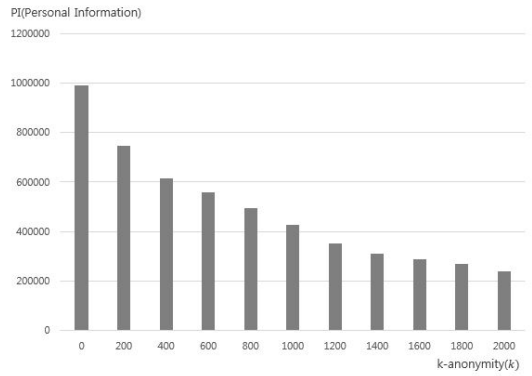

Fig. 3. Graph of k-anonymity

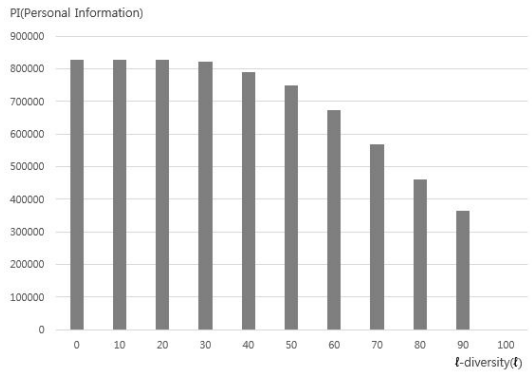

Fig. 4. Graph of ℓ-diversity

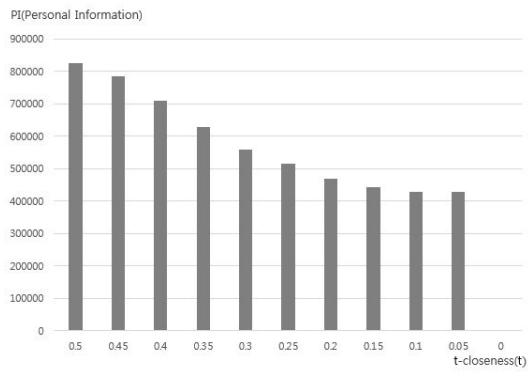

Fig. 5. Graph of t-closeness

약간의 시간상 오차가 발생하는 것을 알 수 있었는데 이유는 k-익명성의 k값에 따라서 처리되는 연산의 양이 달라지기 때문이었다. 예를 들어서 k=200인 경우 k-익명성을 만족하는 데이터가 744,611개, k=2,000인 경우 k-익명성을 만족하는 데이터가

Table 1. Attribute of education

No	Education
00	Not applicable
01	No school completed
02	Nursery school
03	Kindergarten
04	1st-4th grade
05	5th-8th grade
06	9th grade
07	10 th grade
08	11 th grade
09	12 grade. no diploma
10	High school graduate, or GED
11	Some college, no degree
12	Associate degree, occupational program
13	Associate degree, academic program
14	Bachelor's degree
15	Master's degree
16	Professional degree
17	Doctorate degree

237,734개로 측정되었는데 k=200일 경우에는 k=2,000일 경우의 1/3에 해당하는 ℓ-다양성, t-밀집성 연산을 수행하게 하게 되기 때문에 시간이 단축되었다. t-밀집성 실행 시 평균적으로 약 1.8초가 걸리는 것을 볼 수 있는데 그 이유는 t-다양성 알고리즘은 ℓ-다양성 알고리즘과는 다르게 k-익명성으로 그룹화된 데이터가 모두 정렬되어 데이터의 분산 정도를 연산하는 EMD 알고리즘을 연산해야 하기 때문이었다. 본 논문의 실험에서는 대표적인 익명화 처리 알고리즘의 슈도코드를 구현하였으며 이에 따른 익명화 처리 분석을 하였다. 실험에서는 익명화 알고리즘을 충족하지 않는 블록에 대해서 일반화 처리를 하지 않고 데이터의 위험도만을 측정하였는데 그 이유는 전체적인 일반화 처리보다 익명화 알고리즘 구현에 중점을 두고 실험을 진행하였기 때문이었다.

V. 결 론

본 논문에서는 슈도코드를 이용해 관계형 데이터베이스에서 데이터 그룹화를 이용한 익명화 처리 기법을 구체화하였다. 관계형 데이터베이스에서 익명화 처리해야 할 데이터가 증가하면 전체 데이터를 검색

하지 않고 [15]와같이 밀접한 데이터를 분석하여 k-익명성을 적용하거나 [16]과 같이 탐욕 알고리즘을 적용하여 효과적으로 k-익명성을 적용해야 한다. 본 논문에서는 익명화 알고리즘의 기초가 되는 데이터 그룹화 과정을 슈도코드를 이용해 구현하였으며 그룹화를 이용한 방법은 매핑테이블을 생성하지 않고 데이터 그대로 알고리즘에 적용할 수 있기 때문에 알고리즘의 확장성을 높일 수 있다. 본 연구를 이어 전체적인 데이터 익명화를 위해 일반화 처리 부분을 구체화할 것이며 [8]에서 제공하는 데이터 이외에 소셜 네트워크서비스, 공공정보 등에서의 익명화 알고리즘에 대해서도 연구할 예정이다.

References

[1] The News Wire. (2013, Sep). Announced a performance of public information open and sharing. Available : http://www.newswire. co.kr/newsRead.php?no=714253

[2] L. Sweeney, "Weaving technology and policy together to maintain confidentiality," Proc. of the J Law Med Ethics 1997

[3] M. Barbaro, T. Zeller, "A Face Is Exposed for AOL Searcher No. 4417749," Journal of New York Times, 2006

[4] A. Narayanan, V. Shimatikov, "Robust de-anonymization of large datasets," Proc. of the 29th IEEE Symposium on Security and Privacy, pp. 111-125, 2008

[5] L. Sweeney, "k-anonymity: A model for protecting privacy," Int. J. Uncertain Fuzz, 10(5):555-570, July 2002

[6] A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkita-subramaniam, "l-diversity: Privacy beyond k-anonymity," In Proc. 22nd Intel. Conf. Data Engg.(ICDE), pp. 24, March 2006

[7] N. Li, T. Li, and S. Venkatasubramanian, "t-Closeness: Privacy Beyond k-Anonymity and l-Diversity," Proc. IEEE Int. Conf. Data Eng(ICDE), pp. 106-115, 2007

[8] CAT(Cornell Anonymization Toolkit), url : http://anony-toolkit.sourceforge.net/

[9] ARX Data Anonymization Tool, url : http://arx.deidentifier.org/

[10] MAT(Metadata Anonymisation Toolkit) url : https://mat.boum.org/

[11] Sharma. V, "Methodsfor Privacy Protection Using K-Anonymity," Optimization, Reliabilty, and Information Technology (ICROIT), pp. 149-152, Feb. 2014

[12] Mongo Database, url :http://www.mongodb.org/

[13] Python Dictionary tutorial, url : http://www.tutorialspoint.com/python/python_dictionary.html

[14] Yun Kyung. Shin, "A Study on Several Measures of K-anonymity," kookmin university, 2009

[15] Hua Zhu, Xiaojun Ye, "Achieving k-Anonymity Via a Density-Based Clustering Method," Advances in Data and Web Management Lecture Notes in Computer Science, Volume 4505, pp 745-752, June 2007

[16] Ji-Won Byun, Ashish Kamra, Elisa Bertino, Ninghui Li, "Efficient k-Anonymization Using Clustering Techniques," Advances in Databases: Concepts, Sysrems and Applications Lecture Notes in Computer Science, Volume 4443, pp. 188-200, Nov. 2007

〈저 자 소 개〉

박 준 범 (Jun-bum Park) 정회원
2013년 2월: 한서대학교 항공전자공학과, 컴퓨터공학과 졸업
2013년 8월~현재: 과학기술연합대학원대학교 정보보호공학과 석사과정
〈관심분야〉 빅데이터 프라이버시, 개인정보 익명화/재식별, 정보보호

진 승 헌 (Seung-Hun Jin) 종신회원
1993년: 숭실대학교 전자계산학과 졸업
1995년: 숭실대학교 전자계산학과 석사
2004년: 충남대학교 컴퓨터과학과 박사
1994년~1996년: 대우통신, 1996년~1999년 : 삼성전자
1999년~현재: 한국전자통신연구원 사이버보안기반연구부장/책임연구원
〈관심분야〉 컴퓨터/네트워크 보안, 정보보호 (PKI, ID관리, 개인정보보호, 모바일 지불결제 보안)

최 대 선 (Deaseon Choi) 종신회원
1995년: 동국대학교 컴퓨터공학과 졸업
1997년: 포항공과대학교 컴퓨터공학과 석사
2009년: 한국과학기술원 전산학과 박사
1997년~1999년: 현대정보기술
1999년~현재: 한국전자통신연구원 인증기술연구실장/책임연구원
〈관심분야〉 인증, 개인정보보호, 빅데이터 분석