交互式发布DP方案评估实验报告

- 姓名：陈睿颖
- 学号：2013544
- 专业：计算机科学与技术

1. 实验内容

参考教材实验5.1，对交互式发布方案进行DP方案设计，指定隐私预算为0.1，支持查询次数为20次，对DP发布后的结果进行评估说明隐私保护的效果

2. 实验目的

根据所给的代码编译出简单的差分隐私拉普拉斯机制噪音产生和加噪程序和直方图加噪发布程序，使用给定的数据集，通过变换输入的隐私预算来观察不同隐私预算下的噪音规模和对数据的影响。

3. 实验步骤

1. 解压实验文件夹：

```
tar -xzvf ./experiment1.tar.gz
```

1. 进入实验文件夹，运行 `make` 命令进行编译：

```
user@user-VirtualBox:~/experiment1/experiment1$ make
gcc -I./include -c csvpackage.c
gcc -I./include -c laplace.c -lm
gcc -I./include -c testraw.c
gcc csvpackage.o laplace.o testraw.o -o testraw  -lm
gcc -I./include -c testhist.c
gcc csvpackage.o laplace.o testhist.o -o testhist -lm
```

2. 运行testraw程序，先输入较大的隐私预算=10，观察生成的噪音和加噪后的数据与原始数据的差别

```
user@user-VirtualBox:~/experiment1/experiment1$ ./testraw
Please input laplace epsilon:10
Under privacy budget 10.000000, sanitized original data with fake animal name an
d laplace noise:
Animals which carrots cost > 55 (original): 90
Added noise:0.686190    Animal 1        1.686190
Added noise:0.524764    Animal 2        88.524764
Added noise:-0.017234   Animal 3        34.982766
Added noise:0.819691    Animal 4        99.819691
Added noise:-0.169384   Animal 5        68.830616
Added noise:-0.024823   Animal 6        13.975177
Added noise:0.005517    Animal 7        77.005517
Added noise:0.976770    Animal 8        53.976770
Added noise:-0.082793   Animal 9        93.917207
Added noise:0.548722    Animal 10       67.548722
Added noise:0.215363    Animal 11       92.215363
Added noise:0.657767    Animal 12       87.657767
Added noise:-0.026346   Animal 13       69.973654
```

3. 指定隐私预算为0.1，查询次数为20次，即每次查询分配的隐私预算为0.005，且每次查询使用的随机数种子均不相同，根据以上对testraw.c作出如下修改：

```c
#include <stdio.h>
#include <string.h>
#include "laplace.h"
#include "csvpackage.h"
#include <time.h>
#include <stdlib.h>

extern int rand();
extern void srand(unsigned);
```

```
10
11  /*
12  函数功能:          对传入的csv文件进行处理，提取其中数据并生成拉普拉斯分布的噪音进行加噪
13  输入参数说明:
14  path                csv文件的存储位置
15  beta                拉普拉斯分布参数
16  seed            长整型指针变量，  *seed 为伪随机数的种子
17  */
18  void csv_analysis(char* path, double beta, long int seed, int num_queries)
19  {
20      FILE *original_file = fopen(path,"r+"); //读取指定路径的数据集
21      struct Animals * original_data = NULL;
22      original_data = csv_parser(original_file);
23      int sum=0,i=0,j;
24      double x = 0;
25          int s=0;
26      for(j=0; j<num_queries; j++) {  // 支持多次查询，循环调用该函数
27          i = 0;  // 每次查询前需要将 i 和 sum 重置为 0
28          sum = 0;
29          while(original_data[i].name)  //循环为原始数据集内各条数据去除标识（动物
    名）、生成拉普拉斯噪音并加噪
30          {
31                          //printf("Added noise:%f\t%s
    %d\t%f\n",x,"Animal",i+1,original_data[i].carrots+x); //此处分别列出了每条具体添加
    的噪音和加噪的结果。当投入较少预算时，可能会出现负数
32              double b=beta/num_queries; //每次分配的预算
33                          x = laplace_data(b,&seed); //产生拉普拉斯随机数
34              if(original_data[i].carrots+x>=55)
35              {
36                  sum++;
37              }
38              i++;
39          }
40          printf("Query %d - Animals which carrots cost > 55 (Under DP): %d\n",
    j+1, sum); //输出加噪后的数据集中，每日食用胡萝卜大于55的动物个数
41              s+=sum;
42                  seed = rand()%10000+10000;//下一次查询前重置seed
43          }
44          double res=s/num_queries;
45          printf("average - Animals which carrots cost > 55 (Under DP): %f\n",
    res);
46  }
47
48  /*
49  参数表:
50  seed            长整型指针变量，  *seed为伪随机数的种子
51  sen                         数据集的敏感度
```

```
x                       用于储存拉普拉斯分布噪音的临时变量
beta                    隐私预算，在输入后根据公式转换为拉普拉斯分布参数
*/
int main()
{
    long int seed;
    int sen = 1;  //对于一个单属性的数据集，其敏感度为1
    double beta;
    srand((unsigned)time( NULL )); //生成基于时间的随机种子（srand方法）
    beta = 0;
    printf("Please input laplace epsilon:");
    scanf("%lf", &beta);
    if(beta<=0 || !beta)//当输入的beta值无效时，默认设定beta值为1
    {
        beta = 1.0;
    }
    printf("Under privacy budget %f, sanitized original data with fake animal name and laplace noise:\n",beta);
    beta = sen / beta; //拉普拉斯机制下，实际公式的算子beta为敏感度/预算
        seed = rand()%10000+10000; //随机种子产生
        csv_analysis("./zoo.csv",beta,seed,20); //先调用原始数据集
        printf("==================Using neighbour dataset==================\n");
        seed = rand()%10000+10000; //随机种子更新
        csv_analysis("./zoo_nb.csv",beta,seed,20); //再调用相邻数据集

        return 0;
}

```

运行结果如下：

```
Please input laplace epsilon:Under privacy budget 0.100000, sanitized original data with fake animal name and laplace noise:
Animals which carrots cost > 55 (original): 90
Query 1 - Animals which carrots cost > 55 (Under DP): 90
Query 2 - Animals which carrots cost > 55 (Under DP): 89
Query 3 - Animals which carrots cost > 55 (Under DP): 90
Query 4 - Animals which carrots cost > 55 (Under DP): 88
Query 5 - Animals which carrots cost > 55 (Under DP): 90
Query 6 - Animals which carrots cost > 55 (Under DP): 91
Query 7 - Animals which carrots cost > 55 (Under DP): 88
Query 8 - Animals which carrots cost > 55 (Under DP): 90
Query 9 - Animals which carrots cost > 55 (Under DP): 88
```

```
Query 10 - Animals which carrots cost > 55 (Under DP): 89
Query 11 - Animals which carrots cost > 55 (Under DP): 89
Query 12 - Animals which carrots cost > 55 (Under DP): 90
Query 13 - Animals which carrots cost > 55 (Under DP): 90
Query 14 - Animals which carrots cost > 55 (Under DP): 91
Query 15 - Animals which carrots cost > 55 (Under DP): 91
Query 16 - Animals which carrots cost > 55 (Under DP): 90
Query 17 - Animals which carrots cost > 55 (Under DP): 89
Query 18 - Animals which carrots cost > 55 (Under DP): 90
Query 19 - Animals which carrots cost > 55 (Under DP): 91
Query 20 - Animals which carrots cost > 55 (Under DP): 89
average - Animals which carrots cost > 55 (Under DP): 89.000000
==================Using neighbour dataset==================
Animals which carrots cost > 55 (original): 89
Query 1 - Animals which carrots cost > 55 (Under DP): 88
Query 2 - Animals which carrots cost > 55 (Under DP): 89
Query 3 - Animals which carrots cost > 55 (Under DP): 89
Query 4 - Animals which carrots cost > 55 (Under DP): 88
Query 5 - Animals which carrots cost > 55 (Under DP): 89
Query 6 - Animals which carrots cost > 55 (Under DP): 88
Query 7 - Animals which carrots cost > 55 (Under DP): 89
Query 8 - Animals which carrots cost > 55 (Under DP): 89
Query 9 - Animals which carrots cost > 55 (Under DP): 85
Query 10 - Animals which carrots cost > 55 (Under DP): 87
Query 11 - Animals which carrots cost > 55 (Under DP): 88
Query 12 - Animals which carrots cost > 55 (Under DP): 88
Query 13 - Animals which carrots cost > 55 (Under DP): 89
Query 14 - Animals which carrots cost > 55 (Under DP): 88
Query 15 - Animals which carrots cost > 55 (Under DP): 90
Query 16 - Animals which carrots cost > 55 (Under DP): 88
Query 17 - Animals which carrots cost > 55 (Under DP): 86
Query 18 - Animals which carrots cost > 55 (Under DP): 88
Query 19 - Animals which carrots cost > 55 (Under DP): 86
Query 20 - Animals which carrots cost > 55 (Under DP): 89
average - Animals which carrots cost > 55 (Under DP): 88.000000

```

可以看到，原始数据集20次查询的平均值为89，与原来的90不同，数据可用性降低，但相邻数据集的查询结果也不相同，能够更好地抵御差分攻击的影响。

4. 另一种差分隐私发布方法的演示，即差分隐私的直方图发布。在该发布方式下，加噪的对象不再是数据本身，而是对数据进行分桶统计后的计数值进行加噪，修改后的代码支持隐私预算为0.1，查询次数为20次：

```c
#include <stdio.h>
#include <string.h>
#include "laplace.h"
#include "csvpackage.h"
#include <time.h>

extern int rand();
extern void srand(unsigned);
/*
函数功能:          对传入的csv文件进行处理，提取其中数据并生成拉普拉斯分布的噪音进行加噪
输入参数说明:
path                csv文件的存储位置
beta                拉普拉斯分布参数
seed            长整型指针变量，  *seed 为伪随机数的种子
*/
void csv_analysis(char* path, double beta, long int seed,  int num_queries)
{
        FILE *original_file = fopen(path,"r+"); //读取指定路径的数据集
        struct Histobuckets * original_data = NULL;
        original_data = hb_csv_parser(original_file);
        int sum=0,i=0,j;
        double x = 0;
        double s[5]={0};
        for(j=0; j<num_queries; j++){

printf("====================Query%d====================\n",j+1);
                i=0;
                sum=0;
                while(original_data[i].bucket)  //循环为原始数据集内各桶数据生成拉普拉斯噪音并加噪
                {
            double b=beta/num_queries;
                        x = laplace_data(b,&seed); //产生拉普拉斯随机数
                        printf("Added noise:%f\t%s\t%f\n",x,original_data[i].bucket,original_data[i].count+x); //此处分别列出了每条具体添加的噪音和加噪的结果。当投入较少预算时，可能会出现负数

                        s[i]+=original_data[i].count+x;
                        i++;
                }
        seed = rand()%10000+10000;
        }
        printf("\n");
        printf("-----------------------------------------------\n\n");
        for ( i = 0; i < 5; i++)
        {
                /* code */
```

```
44
45
46          printf("[Average]
    \t%s\t%f\n",original_data[i].bucket,s[i]/num_queries); //此处分别列出了每条具体添加
    的噪音和加噪的结果。当投入较少预算时，可能会出现负数
47          }
48  }
49
50  /*
51  参数表:
52  seed              长整型指针变量，  *seed为伪随机数的种子
53  sen                           数据集的敏感度
54  x                           用于储存拉普拉斯分布噪音的临时变量
55  beta                    隐私预算，在输入后根据公式转换为拉普拉斯分布参数
56  */
57  int main()
58  {
59          long int seed;
60          int sen = 1;  //对于一个单属性的数据集，其敏感度为1
61          double x, beta;
62          srand((unsigned)time( NULL )); //生成基于时间的随机种子（srand方法）
63          beta = 0;
64          printf("Please input laplace epsilon:");
65          scanf("%lf", &beta);
66          if(beta<=0 || !beta) //当输入的beta值无效时，默认设定beta值为1
67          {
68                  beta = 1.0;
69          }
70          printf("Under privacy budget %f, sanitized original bucket with
    laplace noise:\n",beta);
71          beta = sen / beta; //拉普拉斯机制下，实际公式的算子beta为敏感度/预算
72          seed = rand()%10000+10000; //随机种子产生
73          csv_analysis("./medicaldata.csv",beta,seed,20); //先调用原始数据集
74      printf("==================Using neighbour dataset==================\n");
75      seed = rand()%10000+10000; //随机种子更新
76      csv_analysis("./md_nb.csv",beta,seed,20); //再调用相邻数据集
77          return 0;
78  }
```

运行结果如下：

原始数据集：

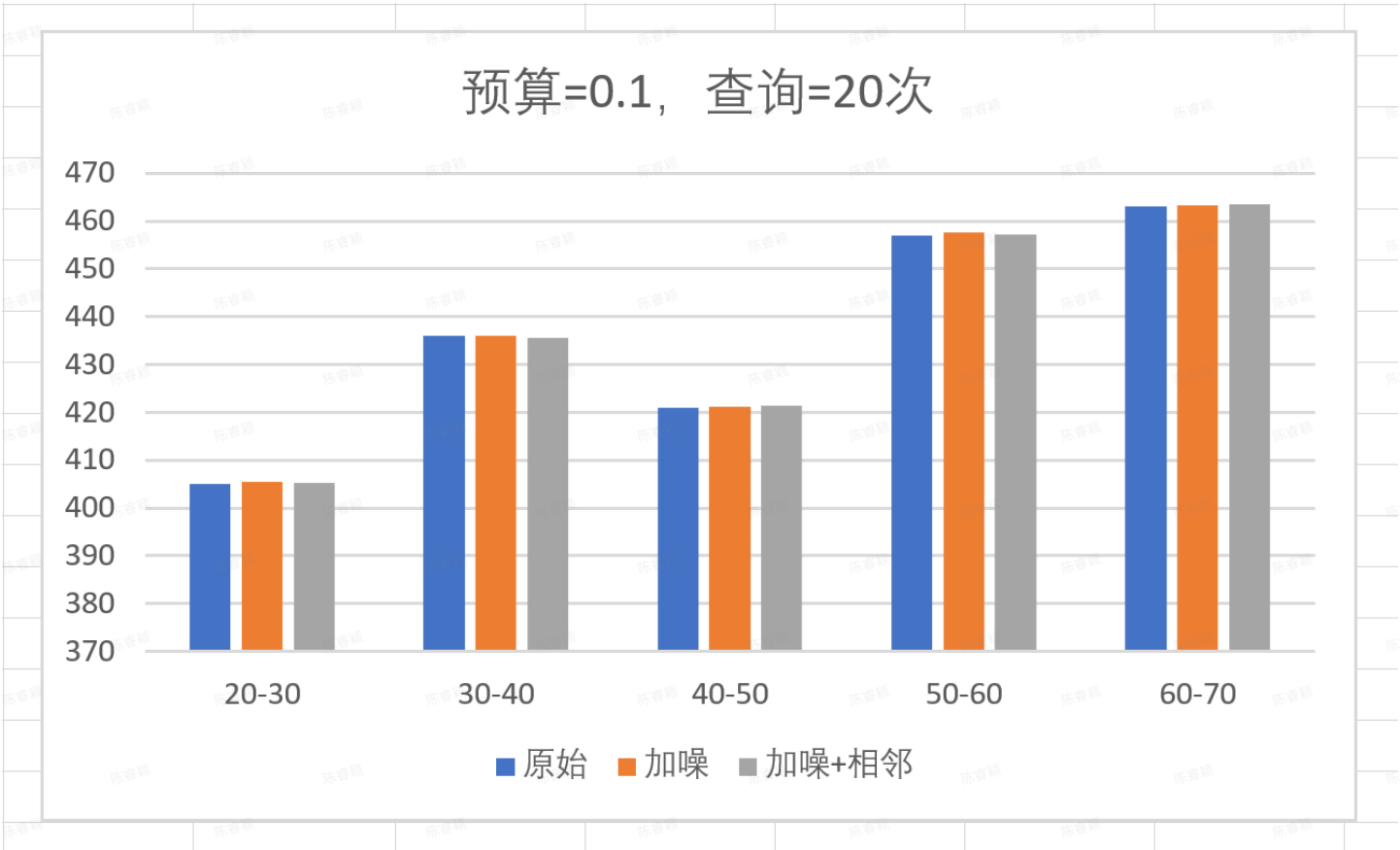
```
[Average] Added noise:0.128739  20-30   405.128739
[Average] Added noise:0.080561  30-40   436.080561
[Average] Added noise:0.540015  40-50   421.540015
[Average] Added noise:0.265186  50-60   457.265186
[Average] Added noise:0.280577  60-70   463.280577
```

相邻数据集：

```
[Average] Added noise:0.473206  20-30   405.473206
[Average] Added noise:0.653519  30-40   435.653519
[Average] Added noise:0.148636  40-50   421.148636
[Average] Added noise:0.475320  50-60   457.475320
[Average] Added noise:0.336312  60-70   463.336312
```

可以看到，由于噪音规模的提高，在相邻数据集的变化影响下，查询结果不减反增。即，虽然数据可用性变差，但能保护实际数据的变化不被攻击者获取，可抵御差分攻击。

绘制直方图如下：



4. 心得体会

- 学习了根据所给的代码编译出简单的差分隐私拉普拉斯机制噪音产生和加噪程序和直方图加噪发布程序
- 学习了在原有代码基础上进行修改，使得其支持多次查询
- 学习使用给定的数据集，通过变换输入的隐私预算来观察不同隐私预算下的噪音规模和对数据的影响