



南开大学
Nankai University

秘密共享实践实验报告

- 姓名：陈睿颖
- 学号：2013544
- 专业：计算机科学与技术

1. 实验内容

假设有三个同学需要对班里的优秀干部 Alice、Bob、Charles、Douglas 进行投票，最后统计各个班干部获得的票数。这个时候就可以利用 Shamir 秘密共享将各个投票方的投票分享出去并进行隐私求和计算，实现三个人对于他们拥有的数据的平均值的计算。

2. 实验环境

本实验使用的是WSL Ubuntu22.04，IDE为VSCode远程连接WSL。

3. 实验步骤

1. 在桌面新建一个文件夹，命名为 vote。在 vote 下打开终端，执行如下命令，建立文件 `ss_function.py`、`ss_student.py`、`count_student.py` 和 `vote_counter.py`。

ss_function.py 代码如下:

```
1 import random
2
3 #快速幂计算  $a^b \pmod p$ 
4 def quickpower(a,b,p):
5     a=a%p
6     ans=1
7     while b!=0:
8         if b&1:
9             ans=(ans*a)%p
10            b>>=1
11            a=(a*a)%p
12    return ans
13
14 #构建多项式:  $x_0$  为常数项系数,  $T$  为最高次项次数,  $p$  为模数,  $fname$  为多项式名
15 def get_polynomial(x0,T,p,fname):
16     f=[]
17     f.append(x0)
18     for i in range(0,T):
19         f.append(random.randrange(0,p))
20     #输出多项式
21     f_print='f'+fname+'='+str(f[0])
22     for i in range(1,T+1):
23         f_print+=' '+str(f[i])+'x^'+str(i)
24     print(f_print)
25     return f
26
27 #计算多项式值
28 def count_polynomial(f,x,p):
29     ans=f[0]
30     for i in range(1,len(f)):
31         ans=(ans+f[i]*quickpower(x,i,p))%p
32     return ans
33
34 #重构函数  $f$  并返回  $f(0)$ 
35 def restructure_polynomial(x,fx,t,p):
36     ans=0
37     #利用多项式插值法计算出  $x=0$  时多项式的值
38     for i in range(0,t):
39         fx[i]=fx[i]%p
40         fxi=1
41         #在模  $p$  下,  $(a/b) \pmod p = (a*c) \pmod p$ , 其中  $c$  为  $b$  在模  $p$  下的逆元,  $c=b^{(p-2)} \pmod p$ 
42         for j in range(0,t):
43             if j !=i:
44                 fxi=(-1*fxi*x[j]*quickpower(x[i]-x[j],p-2,p))%p
```

```

45         fxi=(fxi*fx[i])%p
46         ans=(ans+fxi)%p
47     return ans
48

```

ss_student.py 代码如下：

```

1  import ss_function as ss_f
2
3  #设置模数 p
4  p = 1000000007
5  print(f'模数 p: {p}')
6
7  #输入参与方 id 以及秘密 s
8  id = int(input("请输入参与方 id:"))
9  s = int(input(f'请输入 student_{id}的投票值 s:'))
10
11 #秘密份额为 (share_x, share_y)
12 shares_x = [1,2,3]
13 shares_y = []
14
15 #计算多项式及秘密份额 (t=2, n=3)
16 print(f'Student_{id}的投票值的多项式及秘密份额: ')
17 f = ss_f.get_polynomial(s, 1, p, str(id))
18 temp = []
19 for j in range(0, 3):
20     temp.append(ss_f.count_polynomial(f, shares_x[j], p))
21     print(f'({shares_x[j]}, {temp[j]})')
22     shares_y.append(temp[j])
23
24 #Student_id 将自己的投票值的秘密份额分享给另外两个学生
25 #将三份秘密份额分别保存到 student_id_1.txt, student_id_2.txt, student_id_3.txt
26 #Student_i 获得 Student_id_i.txt
27 for i in range(1, 4):
28     with open(f'student_{id}_{i}.txt', 'w') as f:
29         f.write(str(shares_y[i-1]))
30

```

其中，代码实现了一个安全多方计算协议中的一部分：每个参与方计算一个多项式并计算其在三个不同点的值，将这三个点的值分别发送给三个不同的参与方，以此保证不同的参与方持有不同的秘密份额。具体地，代码执行的过程如下：

a. 首先通过 `import` 导入了 `ss_function` 模块中的函数。

- b. 代码通过输入 `id` 和 `s` 获取当前参与方的编号和其投票值。
- c. 代码计算了当前参与方的投票值在三个不同点上的值，并将其保存在 `shares_y` 列表中。
- d. 代码通过循环将当前参与方的秘密份额写入到三个不同的文件中。

Student1 执行如下命令，输入投票值 0：

```
1 python3 ss_student.py
2 1
3 0
```

Student2 执行如下命令，输入投票值 1。

```
1 python3 ss_student.py
2 2
3 1
```

Student3 执行如下命令，输入投票值 0。

```
1 python3 ss_student.py
2 3
3 0
```

结果如下，在文件夹 `vote` 下会产生 9 个 txt 文件，分别保存三个秘密值的秘密份额

```
资源管理器 ... ss_function.py ss_student.py count_student.py vote_counter.py X
VOTE [WSL: UBUNTU-22.04] vote_counter.py > ...
1 import ss_function as ss f
问题 输出 终端 调试控制台
chenruiying@LAPTOP-LADTBSMC:~/ds/vote$ python3 ss_function.py
1
0
模数 p: 1000000007
请输入参与方 id:1
请输入 student_1 的投票值 s:0
Student_1 的投票值的多项式及秘密份额:
f1=0+181856298x^1
(1,181856298)
(2,363712596)
(3,545568894)
1: 未找到命令
0: 未找到命令
chenruiying@LAPTOP-LADTBSMC:~/ds/vote$ python3 ss_student.py
模数 p: 1000000007
请输入参与方 id:2
请输入 student_2 的投票值 s:1
Student_2 的投票值的多项式及秘密份额:
f2=1+215229505x^1
(1,215229506)
(2,430459011)
(3,645688516)
chenruiying@LAPTOP-LADTBSMC:~/ds/vote$ python3 ss_student.py
模数 p: 1000000007
请输入参与方 id:3
请输入 student_3 的投票值 s:0
Student_3 的投票值的多项式及秘密份额:
f3=0+337588064x^1
(1,337588064)
(2,675176128)
(3,12764185)
chenruiying@LAPTOP-LADTBSMC:~/ds/vote$
```

count_student.py 代码如下:

```
1 p = 1000000007
2 # 输入参与方 id
3 id = int(input("请输入参与方 id:"))
4 # Student_id 读取属于自己的秘密份额
5 data = []
6 for i in range(1, 4):
7     with open(f'student_{i}_{id}.txt', "r") as f: # 打开文本
8         data.append(int(f.read())) # 读取文本
9 # 计算三个秘密份额的和
10 d = 0
11 for i in range(0, 3):
12     d = (d + data[i]) % p
13 # 将求和后的秘密份额保存到文件 d_id.txt 内
14 with open(f'd_{id}.txt', 'w') as f:
15     f.write(str(d))
16
```

在这个方案中，每个参与方输入一个投票值，并将其拆分成多个秘密份额，保存在多个文件中。然后其他参与方可以从这些文件中读取秘密份额，并计算它们的总和，从而得到原始投票值。

具体来说，这段代码实现了以下功能：

- a. 设置模数 p 。
- b. 输入参与方 id 以及投票值的秘密份额。
- c. 计算参与方的多项式及秘密份额，并将秘密份额保存到文件中。
- d. 读取其他参与方的秘密份额文件，计算它们的总和，并将结果保存到一个新的文件中。

Student1 执行如下命令，获得三个投票值的秘密份额相加的结果保存到 d_1.txt：

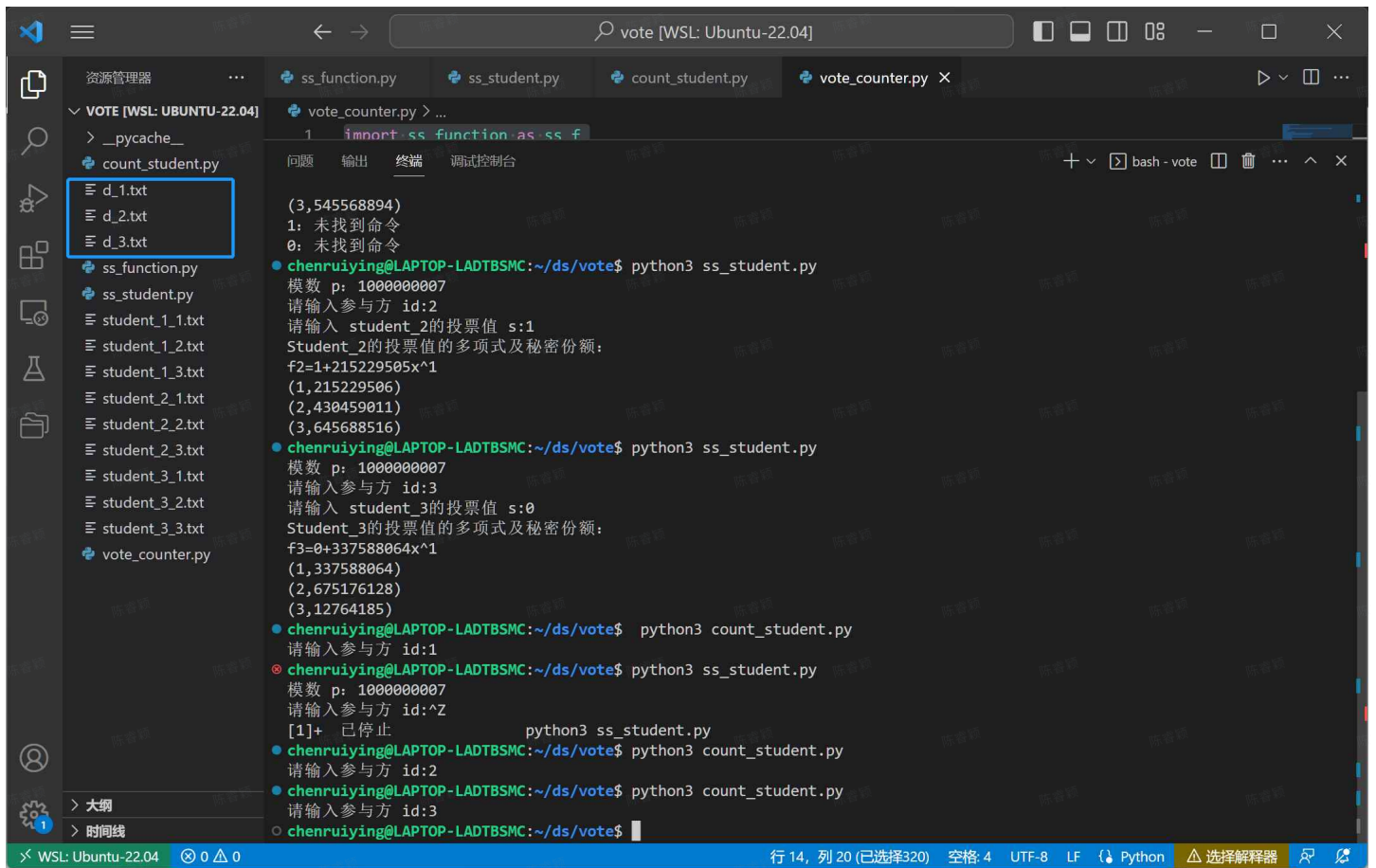
```
1 python3 count_student.py
2 1
```

Student2 执行如下命令，获得三个投票值的秘密份额相加的结果保存到 d_2.txt：

```
1 python3 count_student.py
2 2
```

Student3 执行如下命令，获得三个投票值的秘密份额相加的结果保存到 d_3.txt：

```
1 python3 count_student.py
2 3
```



```
1 import ss_function as ss_f
2 #设置模数 p
3 p=1000000007
4 #随机选取两个参与方, 例如 student2 和 student3, 获得 d2,d3, 从而恢复出 d=a+b+c
5 #读取 d2,d3
6 d_23=[]
7 for i in range(2,4):
8     with open(f'd_{i}.txt', "r") as f: #打开文本
9         d_23.append(int(f.read())) #读取文本
10 #加法重构获得 d
11 d=ss_f.restructure_polynomial([2,3],d_23,2,p)
12 #计算平均得票数
13 d=d/3
14 print(f'得票结果为: {d}')
```

vote_student.py 代码如下:

```
1 import ss_function as ss_f
2 #设置模数 p
3 p=1000000007
4 #随机选取两个参与方, 例如 student2 和 student3, 获得 d2,d3, 从而恢复出 d=a+b+c
5 #读取 d2,d3
6 d_23=[]
7 for i in range(2,4):
8     with open(f'd_{i}.txt', "r") as f: #打开文本
9         d_23.append(int(f.read())) #读取文本
10 #加法重构获得 d
11 d=ss_f.restructure_polynomial([2,3],d_23,2,p)
12 #计算平均得票数
13 d=d/3
14 print(f'得票结果为: {d}')
```

计票员执行如下命令, 得到三位同学投票值的平均值:

```
1 python3 vote_counter.py
```

```
● chenruiying@LAPTOP-LADTBSMC:~/ds/vote$ python3 vote_counter.py  
得票结果为: 0.3333333333333333  
○ chenruiying@LAPTOP-LADTBSMC:~/ds/vote$
```

结果正确!

4. 心得体会

- 掌握了 Shamir's Secret Sharing 算法的原理和实现方式,并学会了如何使用 Python 编程语言实现。
- 学习了快速幂算法、多项式插值法等数学算法。
- 了解了一些常用的数学算法,例如快速幂算法、多项式插值法等。
- 学会了如何读写文本文件,以及如何将数据保存到文本文件中。