**第9周上机实验报告**

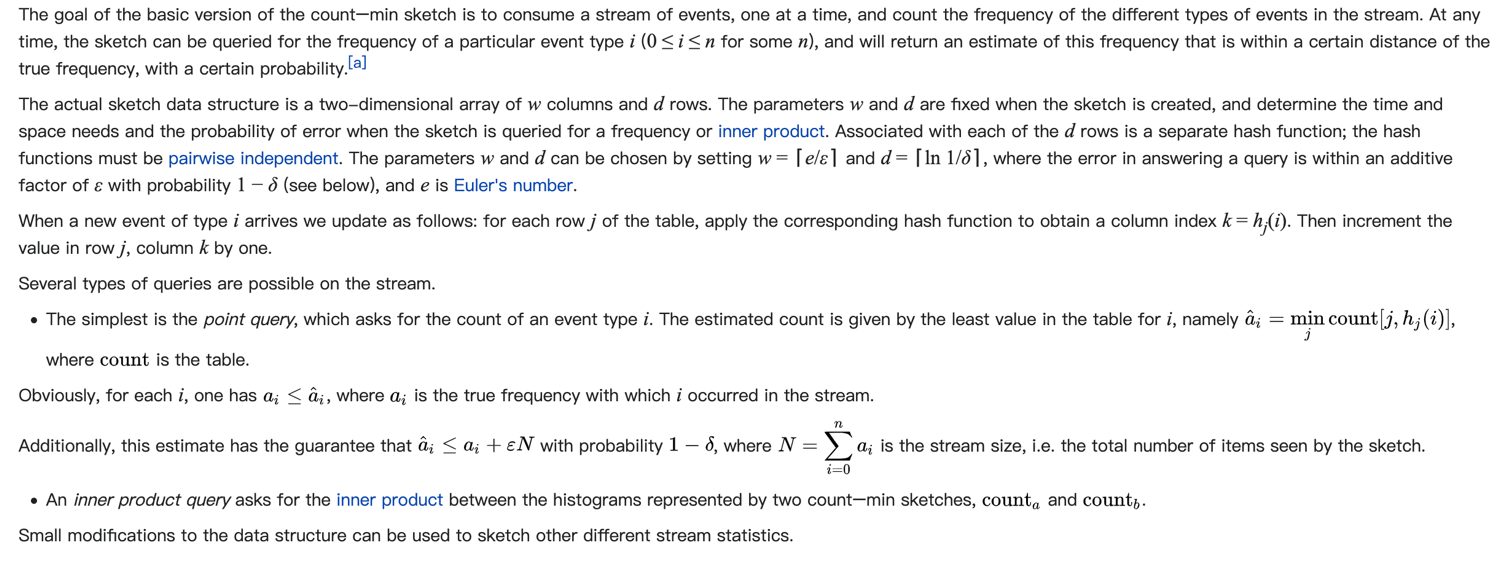
|  |  |
| --- | --- |
| 姓名： | 思子华 |
| 学号： | 2018202181 |
| 学院： | 信息学院 |
| 日期： | 2020.5.24 |

# 一、题目：

实现Count-Min Sketch算法

# 二、算法思路：

具体描述算法的设计思想

****

# 三、程序设计框架：

## 1. 核心函数

Count-min sketch函数主体

* 使用哈希函数将元素i映射到槽中
* 一共使用5个独立的哈希函数对应5个哈希表

1. **void** count\_min\_sketch(**int** index)
2. {
3. **int** size = node.size();
4. **for** (**int** i = 0; i < size; i++)
5. {
6. **for** (**int** j = 0; j < 5; j++)
7. {
8. **int** place = (a[index][j] \* node[i] + b[index][j]) % p[index] % m[index];
9. hash\_tables[j][place] += change[i];
10. }
11. }
12. }

## 哈希函数的设计

全域哈希

其中，p 为一个足够大的质数，使得每一个可能的关键字 key 都落在 0 到 p - 1 的范围内。m 为哈希表中槽位数。任意 a∈{1,2,3,…,p-1}，b∈{0,1,2,…,p-1}。

* P选取的是大于元素种类数的质数
* 每一个哈希表的哈希函数都是独立的

1. **int** p[3] = {11, 1009, 10007};                                                           //质数p
2. **int** a[3][5] = {{2, 3, 5, 7, 9}, {101, 157, 211, 269, 457}, {773, 853, 919, 911, 983}};  //a属于{1,p-1}
3. **int** b[3][5] = {{6, 8, 10, 3, 2}, {31, 37, 307, 367, 431}, {613, 673, 1031, 1091, 1601}}; //b属于{0,p-1}
4. **int** m[3] = {8, 512, 5023};
5. **int** nums[3] = {10, 1000, 10000};

## 2．输入输出的格式等（输入文件请写为相对路径）

给定的三个数据集分别对应编号0、1、2，运行文件后输入对应编号即可。

默认数据集和源代码在同一个目录下，如果需要修改可以在第八行的file\_name处自行修改。

# 四、实验结果说明：

## （1）实验结果截图

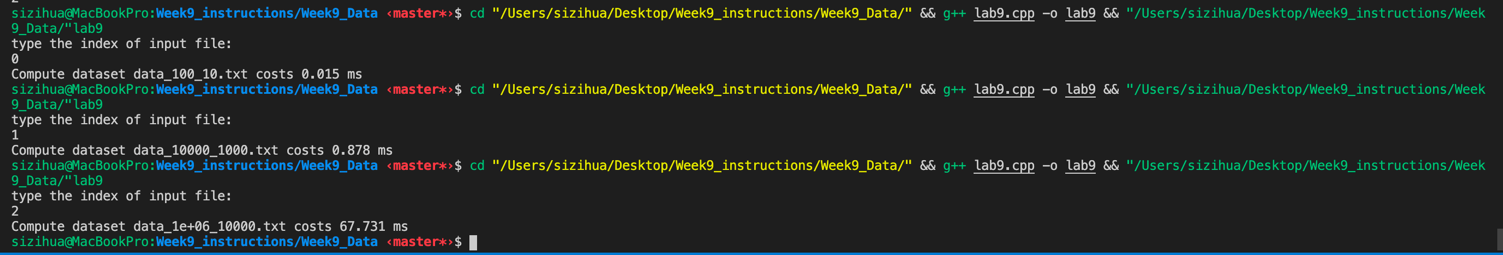


Figure ：运行时间



Figure ：data\_100\_10的运行结果

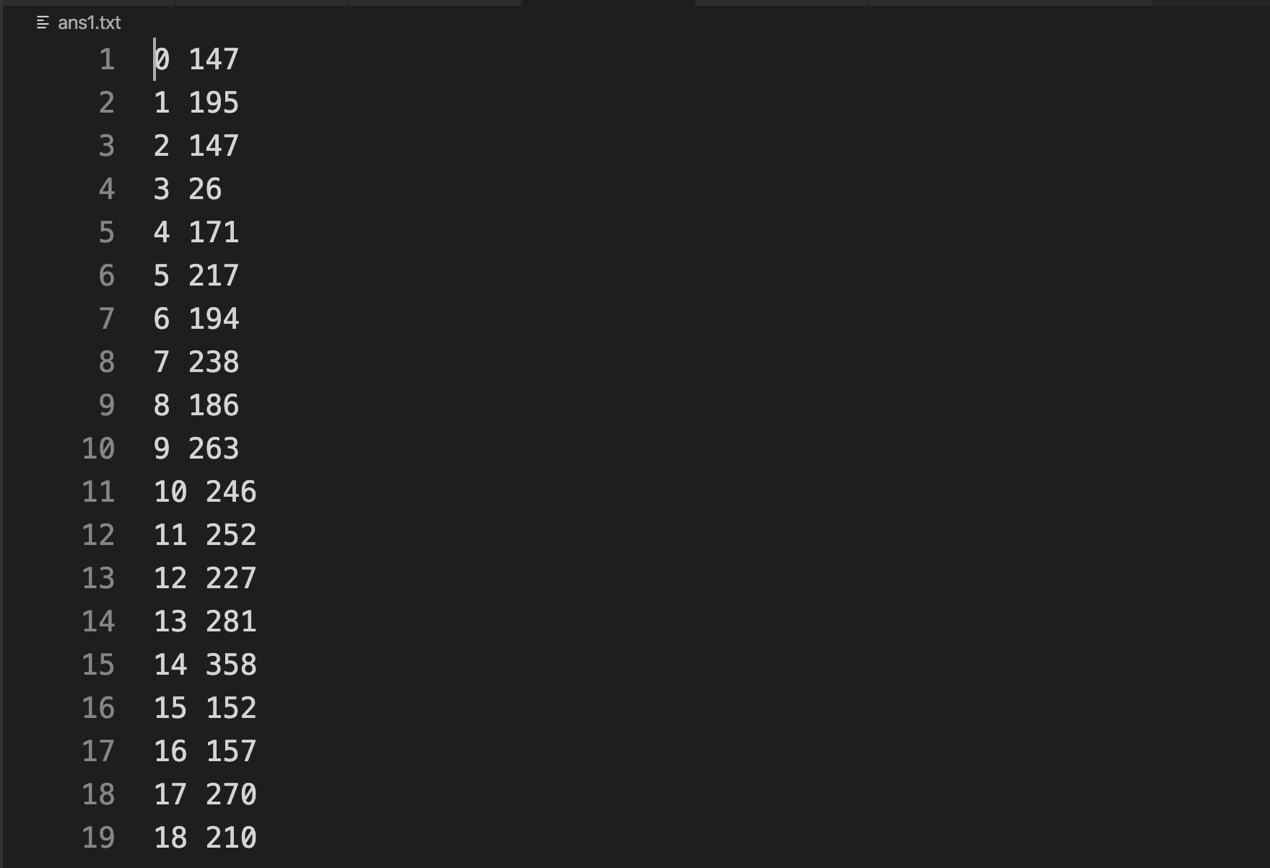


Figure ：data\_10000\_1000的运行结果

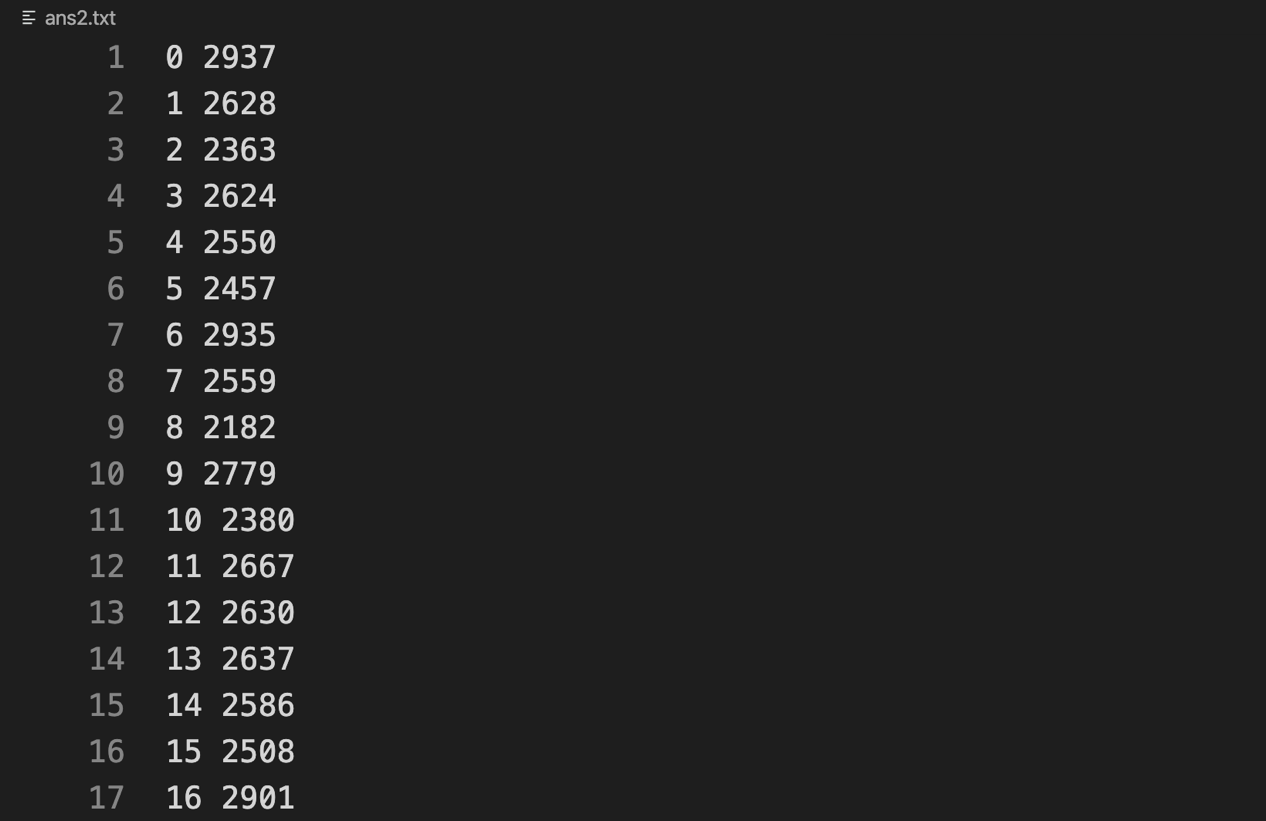


Figure ：data\_1e+06\_10000的运行结果

## （2）实验结果的分析

可以看到count-min sketch的速度非常快，准确率也比较高。误差均在以内。算法的时间复杂度为O(n), 空间复杂度为O(w) = O(