2021 级

《大数据存储与管理》课程

实 验 报 告

姓　名　匡昱恒

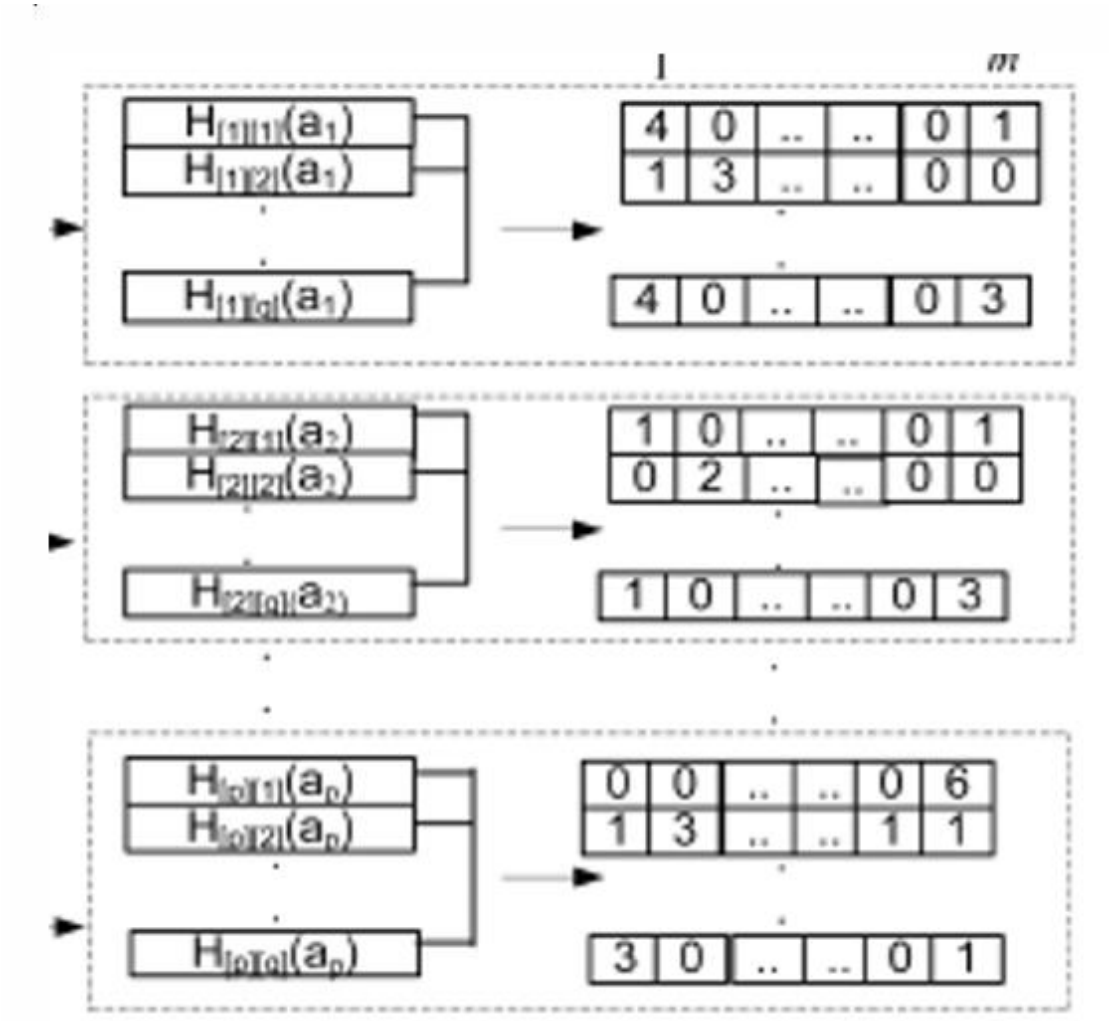
学　号　U202115337

班　号　CS2101

日　期　2024.4.22

一、选题

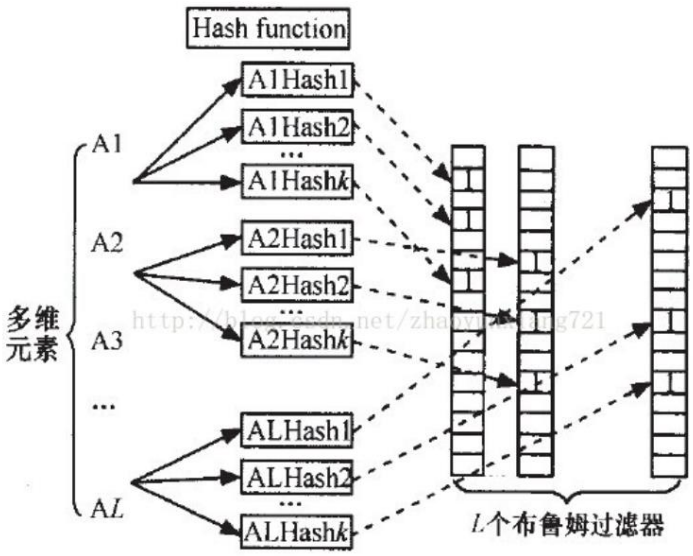
选题 1： 基于 Bloom Filter 的多维数据属性表示和索引



二、实验结构

2.1 数据结构的设计

在处理多维数据时，采用基于 Bloom Filter 的数据结构，会用到多维布鲁姆过滤器。多维布鲁姆过滤器采用多个 Bloom Filter 组成，其个数等于所需存储数据的维数。在元素查询过程中，通过多维元素查询各属性值是否存在相应过滤器中。结构如图所示。



2.2 算法的设计

采用 C++语言来实现。首先定义 Bloom Filter 的类。使用 C++标准库中的 bitset 存储位数组，哈希函数用 C++标准库的 std::hash 计算哈希值。为了处理多维数据，需要创建一个包含多个 Bloom Filter 的结构（多维布鲁姆过滤器），其中每个维度都有一个对应的 Bloom Filter。最后再额外使用一个 Bloom Filter 用于存储属性的联合值。其代码如下图所示：

```
class Bloom_Filter
{
private:
    std::bitset<1024>bits;
    std::hash<std::string>hash_f1;
    std::hash<std::string>hash_f2;
    /* data */
public:
    void add(sonst std::string& item)
    {
        auto hash1=hash_f1(item)%bits.size();
        auto hash2=hash_f2(item)%bits.size();
        bits.set(hash1);
        bits.set(hash2);
    }

    bool seek(const std::string& item) const
    {
        auto hash1=hash_f1(item)%bits.size();
        auto hash2=hash_f2(item)%bits.size();
        return bits.test(hash1)&&bits.test(hash2);
    }
```

主函数用来测试，具体代码如下：

```
int main()
{
    MulDimenBloomFilter mdbf;
    mdbf.add({ "a","b","c", });
    mdbf.add({ "d","e","f", });
    std::cout << " TEST 'a','d','b':"
        << (mdbf, seek({ "a","d","b" })) ? "Found" : "Not Found" << std : endl
        std::cout << " TEST 'a','d','e':"
        << (mdbf, seek({ "a","d","e" })) ? "Found" : "Not Found" << std : endl
        std::cout << " TEST 'a','b','c':"
        << (mdbf, seek({ "a","b","c" })) ? "Found" : "Not Found" << std : endl
    }
```

测试结果为

TEST ‘a’,’d’,’b’: NOT FOUND

TEST ‘a’,’d’,’e’: NOT FOUND

TEST ‘a’,’b’,’c’: FOUND

结果符合预期。

三、实验总结体会

本次实验主要完成了对多维联合 Bloom Filter 的具体代码实现，通过将数据对象的多维属性映射到 Bloom Filter 中，实现快速的数据定位查询。