MPI key-value 系统阶段性报告

一、 程序演示

见小视频

二、 实现细节

1、 准备工作

在个人 PC 上下载了微软的 MPI 相关软件，用 vs 2017 配置 MPI 运行环境，用一个进程模拟一个计算节点。

2、 程序结构

主从结构。以 10 个程序为例，进程 0 为主进程，进程 1-9 为从进程。主进程负责处理与用户的交互与对 MPI 系统发出增删改查命令(主进程不存储数据)，从进程响应主进程的命令并负责 key-value 对的增删改查。

程序实现了增删改查的功能，函数定义如下：

```
int addPair(int dest, string key_value_pair);
string queryPair(int dest, string key);
int deletePair(int dest, string key);
int modifyPair(int dest, string key_value_pair);
```

3、 主从进程交互过程

以添加键值对为例，当住进程接收到用户添加键值对的请求时，**首先会通过哈希函数计算 key 对应的哈希值，用所得到的哈希值对 9 求余数后再加一，此结果决定了该键值对存储在哪个节点（以 10 个进程为例）**。决定存储节点后，主进程将会把存储节点的编号以及操作对应的 tag(以添加键值对为例，添加键值对的 tag 为 1)广播给所有的从进程，这里我定义了一个 Info 结构体用于传递对应信息：

```
struct Info
{
    int tag;
    int dest;
};
```

从进程接收到主进程的广播，若发现自己是目标存储节点，则会先与主进程交流得知本次所传输数据大小，并在本地申请对应的内存空间，做好接收数据的准备后接收主进程传递过来的数据，并将对数据的操作结果返回给主进程，由主进程显示给用户。在这里，从进程使用的存储结构是 C++ STL 库中的 map，增删很方便，查找的效率也很快（C++的 map 使用了红黑树的相关知识，查找的时间复杂度为 logN）。