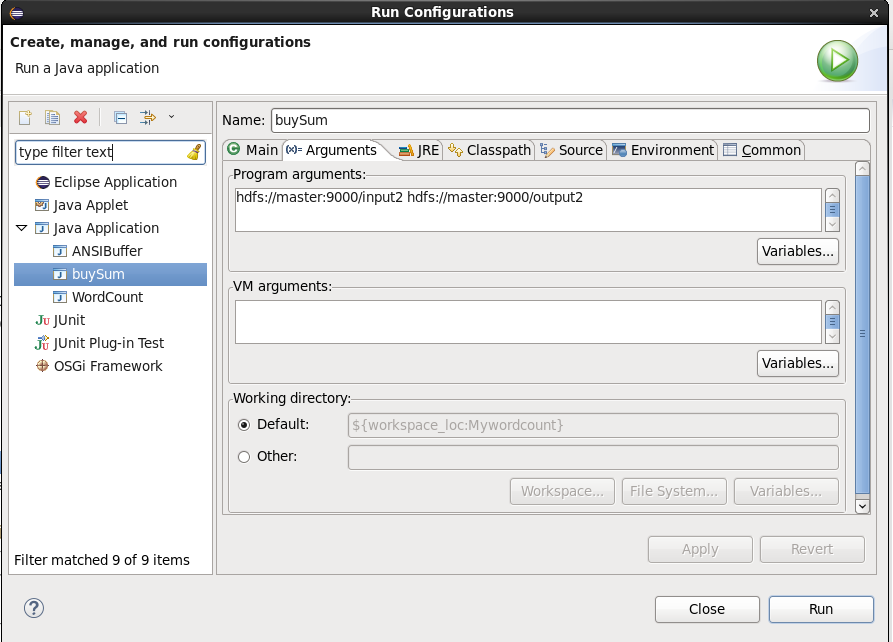
**实验二**

1. **实验要求**
2. 编写mapreduce代码统计每位用户收藏商品的数量。
3. 掌握eclipse如何运行mapreduce程序。
4. 进一步巩固对mapreduce的了解
5. **实验内容**
6. 待处理数据结构

表一 原始数据结构

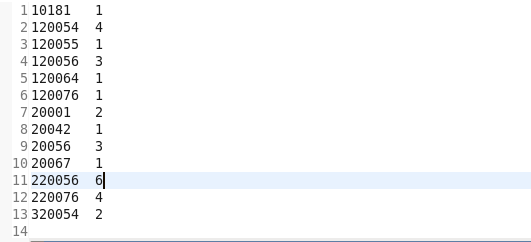
|  |  |  |
| --- | --- | --- |
| 买家ID | 商品ID | 收藏时间 |
| 10181 | 1000481 | 2010-04-04 16:54:31 |
| 20001 | 1001597 | 2010-04-07 15:07:52 |
| 20001 | 1001560 | 2010-04-07 15:08:27 |
| 20042 | 1001368 | 2010-04-08 08:20:30 |
| 20067 | 1002061 | 2010-04-08 16:45:33 |
| 20056 | 1003289 | 2010-04-12 10:50:55 |
| 20056 | 1003290 | 2010-04-12 11:57:35 |
| 20056 | 1003292 | 2010-04-12 12:05:29 |

1. 设置输入输出路径



图一 设置输入输出路径

1. 实验结果



图二 统计结果

1. 处理数据流程

在map函数中将每一个key、value对的value进行分割，分隔符为’\t’,然后提取提取分割后的列表中的第一项（即用户ID），然后发送一个key、value对。在reduce函数中将相同key值的value相加，即可得到用户收藏的商品总数。

1. 部分代码

Map函数

String[] words = value.toString().split("\t");

/\* 使用’\t’作为分隔符。得到切分列表 \*/

word.set(words[0]);

context.write(word, one);

/\* 发送一个key，value对，表示用户收藏了一项商品。 \*/

Reduce函数

int sum = 0;

for (IntWritable val : values) {

sum += val.get();

}

/\* 统计同一个用户收藏的商品数量 \*/

result.set(sum);

context.write(key, result);

/\* 发送一个key、value对 \*/

主函数：

job.setJarByClass(buSum.class);

/\* 通过class名字获取到class所在的jar包 \*/

job.setMapperClass(TokenizerMapper.class);

job.setCombinerClass(IntSumReducer.class);

job.setReducerClass(IntSumReducer.class);

/\* 设置map、reduce和conbiner的所在类 \*/

job.setOutputKeyClass(Text.class);

job.setOutputValueClass(IntWritable.class);

/\* 设置输入输出参数格式 \*/

FileInputFormat.addInputPath(job, new Path(otherArgs[0]));

FileOutputFormat.setOutputPath(job, new Path(otherArgs[1]));

/\* 设置输入输出路径 \*/

**三、 实验总结**

通过本次实验是在上一次WordCount的基础上进一步的掌握mapreduce编程，通过自己动手写代码可以对mapreduce的过程有更深层次的认识。同时也通过eclipse更方便的运行mapreduce程序。