实验4 图的三角形计数

# 实验设计说明

## 主要设计思路

本次实验分为两个MapReduce Job完成。其中AdjList的输入为有向边的列表，输出为图的邻接矩阵；TriangleCount的输入为图的邻接矩阵，输出为图的三角形个数。

## AdjList

### Mapper

AdjListMapper将输入的Text拆分为节点对<v1, v2>，并发射<v1, v2>和<v2, v1>两个键值对。针对自指的边，Mapper通过判断v1与v2是否相等做了过滤。

|  |
| --- |
| **public** **static** **class** **AdjListMapper**  **extends** Mapper<Object, Text, Text, Text> {  **private** Text[] vertexPair = **new** Text[2];  **protected** void map(Object key, Text value, Context context) {  String[] valStrPair = value.toString().split(" ");  vertexPair[0].set(valStrPair[0]);  vertexPair[1].set(valStrPair[1]);  **if** (!vertexPair[0].equals(vertexPair[1])) {  context.write(vertexPair[0], vertexPair[1]);  context.write(vertexPair[1], vertexPair[0]);  }  }  } |

### UndirectedReducer

UndirectedReducer通过vertexSet对values进行去重，其输出为邻接表中的一行。由于版面限制，代码作了精简，省略了边界条件的判断。

|  |
| --- |
| **public** **static** **class** **AdjListUndirectedReducer**  **extends** Reducer<Text, Text, Text, Text> {  **private** Set<String> vertexSet = **new** HashSet<>();  **protected** void reduce(Text key, Iterable<Text> values, Context context) {  **for** (Text vertex : values)  vertexSet.add(vertex.toString());  StringBuilder builder = **new** StringBuilder();  **for** (String vertex : vertexSet) {  builder.append(vertex);  builder.append(' ');  }  context.write(key, **new** Text(builder.toString()));  vertexSet.clear();  }  } |

### DirectedReducer

选做部分中对有向边到无向边的转换有新的定义，当且仅当图中存在<v1, v2>和<v2, v1>这两条边时才认为v1与v2是相连接的。通过使用DirectedReducer就可以做到这点。

我们利用vertexSet判断当前的value是否在之前出现过，由于Mapper对每条有向边都发射了两组相反的键值对，因此values中出现次数大于1的那些节点才与key在两个方向上都有边。因此只有当set中已经存在该value时，才认为当前的value与key相互连接，并将value写入邻接表。由于图中不存在多重边，因此能够保证不会重复计数。由于版面限制，代码作了精简，省略了边界条件的判断。

|  |
| --- |
| **public** **static** **class** **AdjListDirectedReducer**  **extends** Reducer<Text, Text, Text, Text> {  **private** Set<String> vertexSet = **new** HashSet<>();  **protected** void reduce(Text key, Iterable<Text> values, Context context) {  StringBuilder builder = **new** StringBuilder();  **for** (Text vertex : values) {  **if** (!vertexSet.contains(vertex.toString()))  vertexSet.add(vertex.toString());  **else** {  builder.append(vertex);  builder.append(' ');  }  }  context.write(key, **new** Text(builder.toString()));  vertexSet.clear();  }  } |

## TriangleCount

### Mapper

### Reducer