华东师范大学数据科学与工程学院实验报告

课程名称： 分布式模型与编程	**年级：** 2016 级	**上机实践成绩：**
指导教师： 徐辰	**姓名：** 杜云滔	
上机实践名称 Giraph 配置与使用	**学号：** 10153903105	**上机实践日期：** 2018.12.25
上机实践编号： 实验 15	**组号：**	**上机实践时间：** 2018.12.25

一、 **实验目的**

熟悉 Giraph 环境配置与编程模型，并运行示例代码

二、 **实验任务**

1. 单机部署 Giraph 并运行 Giraph 最短路径程序 （参考 http://giraph.apache.org/quick_start.html ）

2. 熟悉例子程序代码 https://github.com/apache/giraph/tree/trunk/giraph-examples/src/main/java/org/apache/giraph/examples

三、 **使用环境**

Ubuntu
Hadoop2.5.1

四、 **实验过程**

环境配置

Hadoop 配置

这里使用 Hadoop2.5.1 进行配置，可以参考这里。

查看版本号

```
hadoop@candice:/usr/local$ ./hadoop-2.5.1/bin/hadoop version
Hadoop 2.5.1
Subversion https://git-wip-us.apache.org/repos/asf/hadoop.git -r 2e18d179e4a806
5b6a9f29cf2de9451891265cce
Compiled by jenkins on 2014-09-05T23:11Z
Compiled with protoc 2.5.0
From source with checksum 6424fcab95bfff8337780a181ad7c78
This command was run using /usr/local/hadoop-2.5.1/share/hadoop/common/hadoop-c
ommon-2.5.1.jar
```

运行 Hadoop

```
./hadoop-2.5.1/sbin/start-dfs.sh
```

查看是否成功

```
hadoop@scott:/usr/local$ jps
8164 StandaloneSessionClusterEntrypoint
94457 SecondaryNameNode
95256 Jps
8618 TaskManagerRunner
94251 DataNode
94079 NameNode
```

Giraph 配置

下载 Giraph

```
cd /usr/local
sudo git clone https://github.com/apache/giraph.git
sudo chown -R hadoop:hadoop giraph
```

编译

```
cd giraph
mvn -Phadoop_2 -Dhadoop.version=2.5.1 -DskipTests clean package
```

编译成功：

```
[INFO] Building zip: /usr/local/giraph/giraph-dist/target/giraph-1.3.0-SNAPSHOT-
for-hadoop-2.5.1-src.zip
[INFO] ------------------------------------------------------------------------
[INFO] Reactor Summary:
[INFO]
[INFO] Apache Giraph Parent ............................... SUCCESS [ 36.184 s]
[INFO] Apache Giraph Core ................................. SUCCESS [01:22 min]
[INFO] Apache Giraph Blocks Framework ..................... SUCCESS [ 22.288 s]
[INFO] Apache Giraph Examples ............................. SUCCESS [ 11.648 s]
[INFO] Apache Giraph Accumulo I/O ......................... SUCCESS [ 22.956 s]
[INFO] Apache Giraph HBase I/O ............................ SUCCESS [ 29.306 s]
[INFO] Apache Giraph HCatalog I/O ......................... SUCCESS [ 36.291 s]
[INFO] Apache Giraph Gora I/O ............................. SUCCESS [ 26.227 s]
[INFO] Apache Giraph Distribution ......................... SUCCESS [ 11.895 s]
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 04:39 min
[INFO] Finished at: 2018-12-25T08:27:57+08:00
[INFO] Final Memory: 70M/1033M
[INFO] ------------------------------------------------------------------------
```

运行

执行最短路径程序

输入

创建/tmp/tiny_graph.txt，输入:

```
[0,0,[[1,1],[3,3]]]
[1,0,[[0,1],[2,2],[3,1]]]
[2,0,[[1,2],[4,4]]]
[3,0,[[0,3],[1,1],[4,4]]]
[4,0,[[3,4],[2,4]]]
```

每一条线由[source_id,source_value,[[dest_id, edge_value],...]]构成。

并拷贝到 HDFS 中：

```
cd /usr/local
hadoop-2.5.1/bin/hadoop dfs -copyFromLocal /tmp/tiny_graph.txt /tiny_graph.txt
```

```
hadoop@candice:/usr/local$ hadoop-2.5.1/bin/hadoop dfs -ls /
DEPRECATED: Use of this script to execute hdfs command is deprecated.
Instead use the hdfs command for it.

Found 1 items
-rw-r--r--   1 hadoop supergroup        112 2018-12-25 09:05 /tiny_graph.txt
```

提交任务

```
/usr/local/hadoop-2.5.1/bin/hadoop jar /usr/local/giraph/giraph-examples/target/
giraph-examples-1.3.0-SNAPSHOT-for-hadoop-2.5.1-jar-with-dependencies.jar org.ap
ache.giraph.GiraphRunner org.apache.giraph.examples.SimpleShortestPathsComputati
on -vif org.apache.giraph.io.formats.JsonLongDoubleFloatDoubleVertexInputFormat
-vip /tiny_graph.txt -vof org.apache.giraph.io.formats.IdWithValueTextOutputForm
at -op /shortestpaths -w 1 -ca giraph.SplitMasterWorker=false
```

查看结果

```
hadoop@candice:~/Documents/Giraph$ /usr/local/hadoop-2.5.1/bin/hadoop dfs -cat /
shortestpaths/p* | less
```

每个点离点 1 的最短路径：

```
0       1.0
1       0.0
2       2.0
3       1.0
4       5.0
```

查看代码例子

```
package org.apache.giraph.examples;

import org.apache.giraph.graph.BasicComputation;
import org.apache.giraph.conf.LongConfOption;
import org.apache.giraph.edge.Edge;
import org.apache.giraph.graph.Vertex;
import org.apache.hadoop.io.DoubleWritable;
import org.apache.hadoop.io.FloatWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.log4j.Logger;

import java.io.IOException;

/**
 * Demonstrates the basic Pregel shortest paths implementation.
 */
@Algorithm(
    name = "Shortest paths",
    description = "Finds all shortest paths from a selected vertex"
)
public class SimpleShortestPathsComputation extends BasicComputation<
    LongWritable, DoubleWritable, FloatWritable, DoubleWritable> {
  /** The shortest paths id */
  public static final LongConfOption SOURCE_ID =
      new LongConfOption("SimpleShortestPathsVertex.sourceId", 1,
          "The shortest paths id");
  /** Class logger */
  private static final Logger LOG =
      Logger.getLogger(SimpleShortestPathsComputation.class);

  /**
   * Is this vertex the source id?
   *
   * @param vertex Vertex
   * @return True if the source id
   */
  private boolean isSource(Vertex<LongWritable, ?, ?> vertex) {
    return vertex.getId().get() == SOURCE_ID.get(getConf());
  }

  @Override
  public void compute(
      Vertex<LongWritable, DoubleWritable, FloatWritable> vertex,
      Iterable<DoubleWritable> messages) throws IOException {
    if (getSuperstep() == 0) {
      vertex.setValue(new DoubleWritable(Double.MAX_VALUE));
    }
    double minDist = isSource(vertex) ? 0d : Double.MAX_VALUE;
    for (DoubleWritable message : messages) {
      minDist = Math.min(minDist, message.get());
    }
    if (LOG.isDebugEnabled()) {
```

```
      LOG.debug("Vertex " + vertex.getId() + " got minDist = " + minDist +
          " vertex value = " + vertex.getValue());
    }
    if (minDist < vertex.getValue().get()) {
      vertex.setValue(new DoubleWritable(minDist));
      for (Edge<LongWritable, FloatWritable> edge : vertex.getEdges()) {
        double distance = minDist + edge.getValue().get();
        if (LOG.isDebugEnabled()) {
          LOG.debug("Vertex " + vertex.getId() + " sent to " +
              edge.getTargetVertexId() + " = " + distance);
        }
        sendMessage(edge.getTargetVertexId(), new DoubleWritable(distance));
      }
    }
    vertex.voteToHalt();
  }
}
```

主要就两个函数：

1. compute
   - 执行实际的计算
2. sendMessage
   - 发送消息

五、总结

本次实验我们熟悉了在 Hadoop 上搭建 Giraph 环境，并对其基本算子 compute、sendmessage 函数有所了解。