HamaWhite 原创，转载请注明出处！欢迎大家加入**Giraph 技术交流群**： 228591158

欢迎访问： 西北工业大学 - 大数据与知识管理研究室 （Northwestern Polytechnical University - BigData and Knowledge Management Lab），链接：<http://wowbigdata.cn/>，<http://wowbigdata.net.cn/>，[http://wowbigdata.com.cn](http://wowbigdata.com.cn/)。

目的：近日因实验要求，需分析Giraph每个超步发送的消息量。Giraph自带能统计每个超步的执行时间，而无消息统计功能。本文通过添加和修改Giraph源码来实现此功能。

1. 添加类，把每个超步发送的消息量大小写入Hadoop的Counter中。在org.apache.giraph.counters包下新建GiraphMessages类，来统计消息量。源代码如下：

**[java]** [view plaincopyprint?](http://blog.csdn.net/xin_jmail/article/details/24411891)[在CODE上查看代码片](https://code.csdn.net/snippets/309739)

1. **package** org.apache.giraph.counters;
3. **import** java.util.Iterator;
4. **import** java.util.Map;
6. **import** org.apache.hadoop.mapreduce.Mapper.Context;
7. **import** com.google.common.collect.Maps;
9. /\*\*
10. \* Hadoop Counters in group "Giraph Messages" for counting every superstep
11. \* message count.
12. \*/
13. **public** **class** GiraphMessages **extends** HadoopCountersBase {
14. /\*\* Counter group name for the giraph Messages \*/
15. **public** **static** **final** String GROUP\_NAME = "Giraph Messages";
17. /\*\* Singleton instance for everyone to use \*/
18. **private** **static** GiraphMessages INSTANCE;
20. /\*\* superstep time in msec \*/
21. **private** **final** Map<Long, GiraphHadoopCounter> superstepMessages;
23. **private** GiraphMessages(Context context) {
24. **super**(context, GROUP\_NAME);
25. superstepMessages = Maps.newHashMap();
26. }
28. /\*\*
29. \* Instantiate with Hadoop Context.
30. \*
31. \* @param context
32. \*            Hadoop Context to use.
33. \*/
34. **public** **static** **void** init(Context context) {
35. INSTANCE = **new** GiraphMessages(context);
36. }
38. /\*\*
39. \* Get singleton instance.
40. \*
41. \* @return singleton GiraphTimers instance.
42. \*/
43. **public** **static** GiraphMessages getInstance() {
44. **return** INSTANCE;
45. }
47. /\*\*
48. \* Get counter for superstep messages
49. \*
50. \* @param superstep
51. \* @return
52. \*/
53. **public** GiraphHadoopCounter getSuperstepMessages(**long** superstep) {
54. GiraphHadoopCounter counter = superstepMessages.get(superstep);
55. **if** (counter == **null**) {
56. String counterPrefix = "Superstep- " + superstep+" ";
57. counter = getCounter(counterPrefix);
58. superstepMessages.put(superstep, counter);
59. }
60. **return** counter;
61. }
63. @Override
64. **public** Iterator<GiraphHadoopCounter> iterator() {
65. **return** superstepMessages.values().iterator();
66. }
67. }

2. 在BspServiceMaster类中添加统计功能。Master在每次同步时候，会聚集每个Worker发送的消息量大小（求和），存储于GlobalStats中。因此只需要在每次同步后，从GlobalStats对象中取出总的通信量大小，然后写入GiraphMessages中。格式为<SuperStep-Number，TotalMessagesCount>，实际存储于上步GiraphMessages类中定义的Map<Long, GiraphHadoopCounter> superstepMessages 对象中。

    在BspServiceMaster的构造方法中，最后面追加一行代码，对GiraphMessages进行初始化。

**[java]** [view plaincopyprint?](http://blog.csdn.net/xin_jmail/article/details/24411891)[在CODE上查看代码片](https://code.csdn.net/snippets/309739)

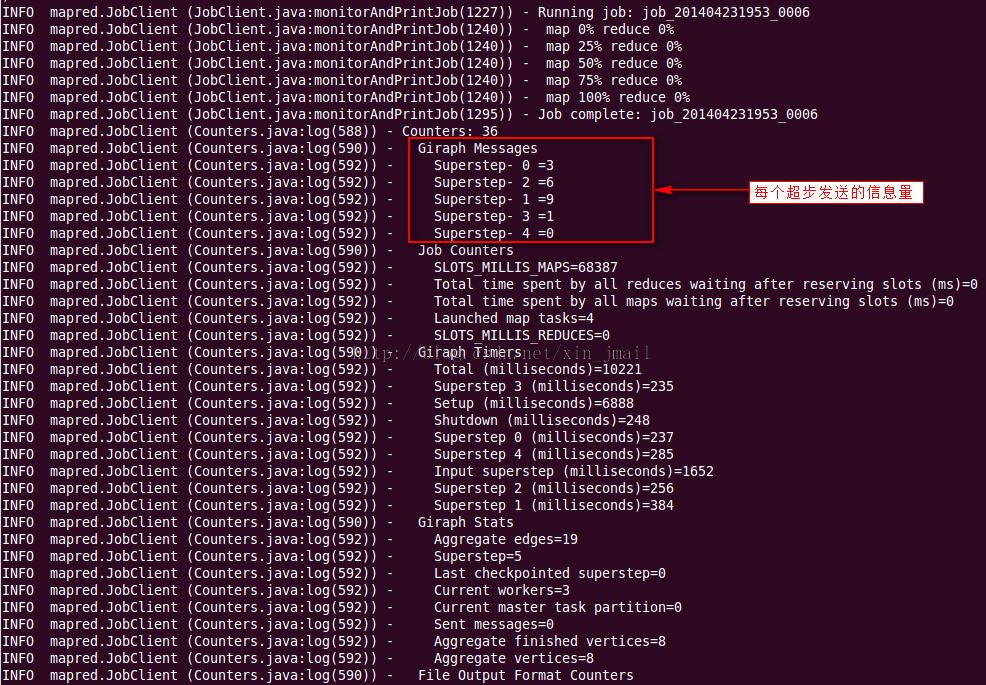
1. GiraphMessages.init(context);

   在BspServiceMaster类的SuperstepState coordinateSuperstep()方法中，添加记录功能。片段代码如下：

**[java]** [view plaincopyprint?](http://blog.csdn.net/xin_jmail/article/details/24411891)[在CODE上查看代码片](https://code.csdn.net/snippets/309739)

1. ……
2. // If the master is halted or all the vertices voted to halt and there
3. // are no more messages in the system, stop the computation
4. GlobalStats globalStats = aggregateWorkerStats(getSuperstep());
6. LOG.info("D-globalStats: "+globalStats+"\n\n");
7. //添加下面语句。从第0个超步起开始记录。
8. **if**(getSuperstep() != INPUT\_SUPERSTEP) {
9. GiraphMessages.getInstance().getSuperstepMessages(getSuperstep()).increment(globalStats.getMessageCount());
10. }
11. ……

3. 实验结果如下：



   完！

   本人原创，转载请注明出处！ 本人QQ：530422429，欢迎大家指正、讨论。

版权声明：本文为博主原创文章，未经博主允许不得转载。