给giraph程序添加一个功能——计算每次迭代中参与计算的顶点数目。首先我们需要找到调用compute()的代码。通过多次单步调试，我们发现了名为computeCallable的类，于是推测compute()必定与其有某种联系，最终在该类中找到了调用compute()的位置。根据giraph在每次迭代中的运行原理，我们的设计思路是在compute()后面对计数加一，然后在日志中输出。 基于giraph的属性，我们首先需要在每一个图的划分的状态中统计compute()计算的次数，然后将统计的最终数据传递到全局的状态中。在思路设计成型之后，我们分别在computeCallable、partitionStats和globalStats三个类中添加相应的统计代码，具体添加的代码如下：

1. org.apache.giraph.partition. PartitionStats 类

添加变量和方法，用来统计每个Partition在每个超步中参与计算的顶点数目。添加的变量和方法如下：

1. /\*\* computed vertices in this partition \*/

2. private long computedVertexCount**=**0**;**

3.

4. /\*\*

5. \* Increment the computed vertex count by one.

6. \*/

7. public void incrComputedVertexCount**()** **{**

8. **++** computedVertexCount**;**

9. **}**

10.

11. /\*\*

12. \* @return the computedVertexCount

13. \*/

14. public long getComputedVertexCount**()** **{**

15. **return** computedVertexCount**;**

16. **}**

修改readFields()和write()方法。当每个划分计算完成后，会把自己的computedVertexCount发送给全局状态，全局状态再读取汇总。

1. @Override

2. public void readFields**(**DataInput input**)** throws IOException **{**

3. partitionId **=** input**.**readInt**();**

4. vertexCount **=** input**.**readLong**();**

5. finishedVertexCount **=** input**.**readLong**();**

6. edgeCount **=** input**.**readLong**();**

7. messagesSentCount **=** input**.**readLong**();**

8. //添加下条语句

9. computedVertexCount**=**input**.**readLong**();**

10. **}**

11.

12. @Override

13. public void write**(**DataOutput output**)** throws IOException **{**

14. output**.**writeInt**(**partitionId**);**

15. output**.**writeLong**(**vertexCount**);**

16. output**.**writeLong**(**finishedVertexCount**);**

17. output**.**writeLong**(**edgeCount**);**

18. output**.**writeLong**(**messagesSentCount**);**

19. //添加下条语句

20. output**.**writeLong**(**computedVertexCount**);**

21. **}**

2. org.apache.giraph.graph. GlobalStats 类

1. /\*\* computed vertices in this partition \*/

2. private long computedVertexCount**=**0**;**

3.

4. /\*\*

5. \* Increment the computed vertex count by one.

6. \*/

7. public void incrComputedVertexCount**()** **{**

8. **++** computedVertexCount**;**

9. **}**

10.

11. /\*\*

12. \* @return the computedVertexCount

13. \*/

14. public long getComputedVertexCount**()** **{**

15. **return** computedVertexCount**;**

16. **}**

添加变量和方法，用来统计每个迭代中参与计算的顶点总数目，包含每个Worker上的所有分割。

1. /\*\* computed vertices in this partition

2. \* Add by BaiSong

3. \*/

4. private long computedVertexCount**=**0**;**

5. /\*\*

6. \* @return the computedVertexCount

7. \*/

8. public long getComputedVertexCount**()** **{**

9. **return** computedVertexCount**;**

10. **}**

修改addPartitionStats()方法，增加统计computedVertexCount功能。

1. /\*\*

2. \* Add the stats of a partition to the global stats.

3. \*

4. \* @param partitionStats Partition stats to be added.

5. \*/

6. public void addPartitionStats**(**PartitionStats partitionStats**)** **{**

7. this**.**vertexCount **+=** partitionStats**.**getVertexCount**();**

8. this**.**finishedVertexCount **+=** partitionStats**.**getFinishedVertexCount**();**

9. this**.**edgeCount **+=** partitionStats**.**getEdgeCount**();**

10. //Add by BaiSong，添加下条语句

11. this**.**computedVertexCount**+=**partitionStats**.**getComputedVertexCount**();**

12. **}**

当然为了Debug方便，也可以修改该类的toString()方法（可选），修改后的如下：

1. public String toString**()** **{**

2. **return** "(vtx=" **+** vertexCount **+** ", computedVertexCount="

3. **+** computedVertexCount **+** ",finVtx=" **+** finishedVertexCount

4. **+** ",edges=" **+** edgeCount **+** ",msgCount=" **+** messageCount

5. **+** ",haltComputation=" **+** haltComputation **+** ")"**;**

3. org.apache.giraph.graph. ComputeCallable<I,V,E,M>

添加统计功能。在computePartition()方法中添加

1. **if** **(!**vertex**.**isHalted**())** **{**

2. context**.**progress**();**

3. TimerContext computeOneTimerContext **=** computeOneTimer**.**time**();**

4. try **{**

5. vertex**.**compute**(**messages**);**

6. //添加下面一句，当顶点调用完compute()方法后，就把该Partition的computedVertexCount加1

7. partitionStats**.**incrComputedVertexCount**();**

8. **}** finally **{**

9. computeOneTimerContext**.**stop**();**

10. **}**

为了将统计结果输出在日志中，我们在org.apache.giraph.counters包下新建GiraphMessages类，以此来实现统计的输出。添加的代码如下：

1. package org**.**apache**.**giraph**.**counters**;**

2.

3. import java**.**util**.**Iterator**;**

4. import java**.**util**.**Map**;**

5.

6. import org**.**apache**.**hadoop**.**mapreduce**.**Mapper**.**Context**;**

7. import com**.**google**.**common**.**collect**.**Maps**;**

8.

9. /\*\*

10. \* Hadoop Counters in group "Giraph Messages" for counting every superstep

11. \* message count.

12. \*/

13.

14. public class GiraphComputedVertex extends HadoopCountersBase **{**

15. /\*\* Counter group name for the giraph Messages \*/

16. public static final String GROUP\_NAME **=** "Giraph Computed Vertex"**;**

17.

18. /\*\* Singleton instance for everyone to use \*/

19. private static GiraphComputedVertex INSTANCE**;**

20.

21. /\*\* superstep time in msec \*/

22. private final Map**<**Long**,** GiraphHadoopCounter**>** superstepVertexCount**;**

23.

24. private GiraphComputedVertex**(**Context context**)** **{**

25. super**(**context**,** GROUP\_NAME**);**

26. superstepVertexCount **=** Maps**.**newHashMap**();**

27. **}**

28.

29. /\*\*

30. \* Instantiate with Hadoop Context.

31. \*

32. \* @param context

33. \* Hadoop Context to use.

34. \*/

35. public static void init**(**Context context**)** **{**

36. INSTANCE **=** new GiraphComputedVertex**(**context**);**

37. **}**

38.

39. /\*\*

40. \* Get singleton instance.

41. \*

42. \* @return singleton GiraphTimers instance.

43. \*/

44. public static GiraphComputedVertex getInstance**()** **{**

45. **return** INSTANCE**;**

46. **}**

47.

48. /\*\*

49. \* Get counter for superstep messages

50. \*

51. \* @param superstep

52. \* @return

53. \*/

54. public GiraphHadoopCounter getSuperstepVertexCount**(**long superstep**)** **{**

55. GiraphHadoopCounter counter **=** superstepVertexCount**.**get**(**superstep**);**

56. **if** **(**counter **==** null**)** **{**

57. String counterPrefix **=** "Superstep: " **+** superstep**+**" "**;**

58. counter **=** getCounter**(**counterPrefix**);**

59. superstepVertexCount**.**put**(**superstep**,** counter**);**

60. **}**

61. **return** counter**;**

62. **}**

63.

64. @Override

65. public Iterator**<**GiraphHadoopCounter**>** iterator**()** **{**

66. **return** superstepVertexCount**.**values**().**iterator**();**

67. **}**

68. **}**