Hadoop案例之二度人脉与好友推荐

参考：

https://my.oschina.net/u/176897/blog/99761

# 1.实例描述

社交网站上的各个用户以及用户之间的相互关注可以抽象为一个图。以下图为例：

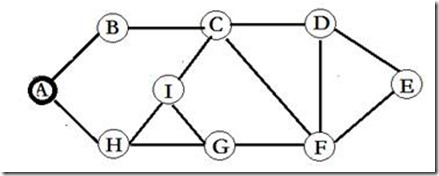
[](http://static.oschina.net/uploads/img/201301/03220045_TLFw.jpg)

图1

顶点A、B、C到I分别是社交网站的用户，两顶点之间的边表示两顶点代表的用户之间相互关注。那么如何根据用户之间相互关注所构成的图，来向每个用户推荐好友呢？

现在我们以上图为例，介绍下如何利用用户之间相互关注所构成的图，来向每个用户推荐好友。首先我们不得不假设的是如果两用户之间相互关注，那么我们认为他们认识或者说是现实中的好友，至少应该认识。假设我们现在需要向用户I推荐好友，我们发现用户I的好友有H、G、C。其中H的好友还有A，G的好友还有F，C的好友还有B、F。那么用户I、H、G、C、A、B、F极有可能是同一个圈子里的人。我们应该把用户A、B、F推荐给用户I认识。进一步的想，用户F跟两位I的好友C、G是好友，而用户A、B都分别只跟一位I的好友是好友，那么相对于A、B来说，F当然更应该推荐给用户I认识。

可能你会发现，在上面的分析中，我们使用了用户I的二度人脉作为他的推荐好友，而且我们对用户I的每个二度人脉进行了投票处理，选举出最优推荐。其实，我觉得，**二度人脉的结果只能看看某个用户的在社交网站上的人际关系链，而基于投票选举产生的二度人脉才是好友推荐功能中所需要的好友**。

# 2.设计思路

我们的输入是deg2friend.txt，保存用户之间相互关注的信息。每行有两个用户ID，以逗号分割，表示这两个用户之间相互关注即认识。

A,B

B,C

C,D

D,E

E,F

F,D

F,C

F,G

G,I

G,H

H,I

I,C

H,A

二度好友的计算需要两轮的MapReduce。第一轮MapReduce的Map中，如果输入是“H，I”，我们的输出是key=H，value=“H，I”跟key=I，value=“H，I”两条结果。前者表示I可以通过H去发现他的二度好友，后者表示H可以通过I去发现他的二度好友。

根据第一轮MapReduce的Map，第一轮MapReduce的Reduce 的输入是例如key =I，value={“H，I”、“C，I”、“G，I”} 。**其实Reduce 的输入是所有与Key代表的结点相互关注的人**。如果H、C、G是与I相互关注的好友，那么H、C、G就可能是二度好友的关系，如果他们之间不是相互关注的。对应最上面的图，H与C是二度好友，G与C是二度好友，但G与H不是二度好友，因为他们是相互关注的。第一轮MapReduce的Reduce的处理就是把相互关注的好友对标记为一度好友（“deg1friend”）并输出，把有可能是二度好友的好友对标记为二度好友（“deg2friend”）并输出。

第二轮MapReduce则需要根据第一轮MapReduce的输出，即每个好友对之间是否是一度好友（“deg1friend”），是否有可能是二度好友（“deg2friend”）的关系，确认他们之间是不是真正的二度好友关系。如果他们有deg1friend的标签，那么不可能是二度好友的关系；如果有deg2friend的标签、没有deg1friend的标签，那么他们就是二度好友的关系。另外，特别可以利用的是，**某好友对deg2friend标签的个数就是他们成为二度好友的支持数**，即他们之间可以通过多少个都相互关注的好友认识。

# 3.程序代码

**package** Hadoop\_Deg2friend;

**import** java.io.IOException;

**import** java.util.Vector;

**import** org.apache.hadoop.conf.Configuration;

**import** org.apache.hadoop.fs.Path;

**import** org.apache.hadoop.io.Text;

**import** org.apache.hadoop.mapreduce.Job;

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

**import** org.apache.hadoop.mapreduce.Reducer;

**import** org.apache.hadoop.mapreduce.lib.input.FileInputFormat;

**import** org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

**import** org.apache.hadoop.util.GenericOptionsParser;

**public** **class** Deg2friend {

//map1

**public** **static** **class** Map1 **extends** Mapper<Object, Text, Text, Text>

{

**private** Text map1\_key = **new** Text();

**private** Text map1\_value = **new** Text();

@Override

**protected** **void** map(Object **key**, Text **value**, **Context** **context**)

**throws** IOException, InterruptedException {

String[] **eachterm** = value.toString().split(",");

**if** (eachterm.length != 2) {

**return**;

}

**if** (eachterm[0].compareTo(eachterm[1]) < 0) {

map1\_value.set(eachterm[0] + "\t" + eachterm[1]);

}

**else** **if** (eachterm[0].compareTo(eachterm[1]) > 0) {

map1\_value.set(eachterm[1] + "\t" + eachterm[0]);

}

map1\_key.set(eachterm[0]);

context.write(map1\_key, map1\_value);

map1\_key.set(eachterm[1]);

context.write(map1\_key, map1\_value);

}

}

//reduce1

**public** **static** **class** Reduce1 **extends** Reducer<Text, Text, Text, Text>

{

@Override

**protected** **void** reduce(Text **key**, Iterable<Text> **values**, **Context** **context**)

**throws** IOException, InterruptedException {

Vector<String> **hisFriends** = **new** Vector<String>();

**for**(Text **val** : values)

{

String[] **eachterm** = val.toString().split("\t");

**if** (eachterm[0].equals(key.toString())) {

hisFriends.add(eachterm[1]);

context.write(val, **new** Text("deg1friend"));

}

**if** (eachterm[1].equals(key.toString())) {

hisFriends.add(eachterm[0]);

context.write(val, **new** Text("deg1friend"));

}

}

**for**(**int** **i** = 0; i < hisFriends.size(); i++)

{

**for**(**int** **j** = 0; j < hisFriends.size(); j++)

{

**if** (hisFriends.elementAt(i).compareTo(hisFriends.elementAt(j)) < 0) {

Text **reduce\_key** = **new** Text(hisFriends.elementAt(i)+"\t"+hisFriends.elementAt(j));

context.write(reduce\_key, **new** Text("deg2friend"));

}

}

}

}

}

//map2

**public** **static** **class** Map2 **extends** Mapper<Object, Text, Text, Text>

{

@Override

**protected** **void** map(Object **key**, Text **value**, **Context** **context**)

**throws** IOException, InterruptedException {

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

**if** (line.length == 3) {

Text **map2\_key** = **new** Text(line[0]+"\t"+line[1]);

Text **map2\_value** = **new** Text(line[2]);

context.write(map2\_key, map2\_value);

}

}

}

//reduce2

**public** **static** **class** Reduce2 **extends** Reducer<Text, Text, Text, Text>

{

@Override

**protected** **void** reduce(Text **key**, Iterable<Text> **values**, **Context** **context**)

**throws** IOException, InterruptedException {

**boolean** **isdeg1** = **false**;

**boolean** **isdeg2** = **false**;

**int** **count** = 0;

**for**(Text **val** : values)

{

**if** (val.toString().compareTo("deg1friend") == 0) {

isdeg1 = **true**;

}

**if** (val.toString().compareTo("deg2friend") == 0) {

isdeg2 = **true**;

count++;

}

}

**if** ((!isdeg1) && isdeg2) {

context.write(**new** Text(String.*valueOf*(count)),key);

}

}

}

//main

**public** **static** **void** main(String[] **args**) **throws** Exception {

Configuration **conf** = **new** Configuration();

String[] **otherArgs** = **new** GenericOptionsParser(conf,args).getRemainingArgs();

**if** (otherArgs.length != 3) {

System.***err***.println("Usage: Deg2friend <in> <temp> <out>");

System.*exit*(2);

}

Job **job1** = **new** ~~Job~~(conf, "Deg2friend");

job1.setJarByClass(Deg2friend.**class**);

job1.setMapperClass(Map1.**class**);

job1.setReducerClass(Reduce1.**class**);

job1.setOutputKeyClass(Text.**class**);

job1.setOutputValueClass(Text.**class**);

**FileInputFormat**.*addInputPath*(job1, **new** Path(otherArgs[0]));

**FileOutputFormat**.*setOutputPath*(job1, **new** Path(otherArgs[1]));

**if** (job1.waitForCompletion(**true**)) {

Job **job2** = **new** ~~Job~~(conf, "Deg2friend");

job2.setJarByClass(Deg2friend.**class**);

job2.setMapperClass(Map2.**class**);

job2.setReducerClass(Reduce2.**class**);

job2.setOutputKeyClass(Text.**class**);

job2.setOutputValueClass(Text.**class**);

**FileInputFormat**.*addInputPath*(job2, **new** Path(otherArgs[1]));

**FileOutputFormat**.*setOutputPath*(job2, **new** Path(otherArgs[2]));

System.*exit*(job2.waitForCompletion(**true**)? 0 : 1);

}

System.*exit*(job1.waitForCompletion(**true**)? 0 : 1);

}

}

# 4. 程序执行

root@node1:/usr/local/hadoop/hadoop-2.5.2/myJar# hadoop jar Deg2friend.jar Hadoop\_Deg2friend.Deg2friend /usr/local/hadooptempdata/input/deg2 /usr/local/hadooptempdata/temp/deg2 /usr/local/hadooptempdata/output/deg2

16/12/30 23:35:36 INFO client.RMProxy: Connecting to ResourceManager at node1/192.168.233.129:8032

16/12/30 23:35:40 INFO input.FileInputFormat: Total input paths to process : 1

16/12/30 23:35:41 INFO mapreduce.JobSubmitter: number of splits:1

16/12/30 23:35:43 INFO mapreduce.JobSubmitter: Submitting tokens for job: job\_1483111826986\_0001

16/12/30 23:35:45 INFO impl.YarnClientImpl: Submitted application application\_1483111826986\_0001

16/12/30 23:35:45 INFO mapreduce.Job: The url to track the job: http://node1:8088/proxy/application\_1483111826986\_0001/

16/12/30 23:35:45 INFO mapreduce.Job: Running job: job\_1483111826986\_0001

16/12/30 23:36:32 INFO mapreduce.Job: Job job\_1483111826986\_0001 running in uber mode : false

16/12/30 23:36:32 INFO mapreduce.Job: map 0% reduce 0%

16/12/30 23:37:36 INFO mapreduce.Job: map 100% reduce 0%

16/12/30 23:38:21 INFO mapreduce.Job: map 100% reduce 100%

16/12/30 23:38:24 INFO mapreduce.Job: Job job\_1483111826986\_0001 completed successfully

16/12/30 23:38:28 INFO mapreduce.Job: Counters: 49

File System Counters

FILE: Number of bytes read=214

FILE: Number of bytes written=197899

FILE: Number of read operations=0

FILE: Number of large read operations=0

FILE: Number of write operations=0

HDFS: Number of bytes read=178

HDFS: Number of bytes written=795

HDFS: Number of read operations=6

HDFS: Number of large read operations=0

HDFS: Number of write operations=2

Job Counters

Launched map tasks=1

Launched reduce tasks=1

Data-local map tasks=1

Total time spent by all maps in occupied slots (ms)=60503

Total time spent by all reduces in occupied slots (ms)=38314

Total time spent by all map tasks (ms)=60503

Total time spent by all reduce tasks (ms)=38314

Total vcore-seconds taken by all map tasks=60503

Total vcore-seconds taken by all reduce tasks=38314

Total megabyte-seconds taken by all map tasks=61955072

Total megabyte-seconds taken by all reduce tasks=39233536

Map-Reduce Framework

Map input records=13

Map output records=26

Map output bytes=156

Map output materialized bytes=214

Input split bytes=126

Combine input records=0

Combine output records=0

Reduce input groups=9

Reduce shuffle bytes=214

Reduce input records=26

Reduce output records=53

Spilled Records=52

Shuffled Maps =1

Failed Shuffles=0

Merged Map outputs=1

GC time elapsed (ms)=406

CPU time spent (ms)=2790

Physical memory (bytes) snapshot=290168832

Virtual memory (bytes) snapshot=3772538880

Total committed heap usage (bytes)=139837440

Shuffle Errors

BAD\_ID=0

CONNECTION=0

IO\_ERROR=0

WRONG\_LENGTH=0

WRONG\_MAP=0

WRONG\_REDUCE=0

File Input Format Counters

Bytes Read=52

File Output Format Counters

Bytes Written=795

16/12/30 23:38:29 INFO client.RMProxy: Connecting to ResourceManager at node1/192.168.233.129:8032

16/12/30 23:38:41 INFO input.FileInputFormat: Total input paths to process : 1

16/12/30 23:38:42 INFO mapreduce.JobSubmitter: number of splits:1

16/12/30 23:38:42 INFO mapreduce.JobSubmitter: Submitting tokens for job: job\_1483111826986\_0002

16/12/30 23:38:43 INFO impl.YarnClientImpl: Submitted application application\_1483111826986\_0002

16/12/30 23:38:43 INFO mapreduce.Job: The url to track the job: http://node1:8088/proxy/application\_1483111826986\_0002/

16/12/30 23:38:43 INFO mapreduce.Job: Running job: job\_1483111826986\_0002

16/12/30 23:39:26 INFO mapreduce.Job: Job job\_1483111826986\_0002 running in uber mode : false

16/12/30 23:39:26 INFO mapreduce.Job: map 0% reduce 0%

16/12/30 23:40:30 INFO mapreduce.Job: map 100% reduce 0%

16/12/30 23:40:59 INFO mapreduce.Job: map 100% reduce 100%

16/12/30 23:41:00 INFO mapreduce.Job: Job job\_1483111826986\_0002 completed successfully

16/12/30 23:41:01 INFO mapreduce.Job: Counters: 49

File System Counters

FILE: Number of bytes read=907

FILE: Number of bytes written=199287

FILE: Number of read operations=0

FILE: Number of large read operations=0

FILE: Number of write operations=0

HDFS: Number of bytes read=924

HDFS: Number of bytes written=90

HDFS: Number of read operations=6

HDFS: Number of large read operations=0

HDFS: Number of write operations=2

Job Counters

Launched map tasks=1

Launched reduce tasks=1

Data-local map tasks=1

Total time spent by all maps in occupied slots (ms)=47074

Total time spent by all reduces in occupied slots (ms)=36364

Total time spent by all map tasks (ms)=47074

Total time spent by all reduce tasks (ms)=36364

Total vcore-seconds taken by all map tasks=47074

Total vcore-seconds taken by all reduce tasks=36364

Total megabyte-seconds taken by all map tasks=48203776

Total megabyte-seconds taken by all reduce tasks=37236736

Map-Reduce Framework

Map input records=53

Map output records=53

Map output bytes=795

Map output materialized bytes=907

Input split bytes=129

Combine input records=0

Combine output records=0

Reduce input groups=28

Reduce shuffle bytes=907

Reduce input records=53

Reduce output records=15

Spilled Records=106

Shuffled Maps =1

Failed Shuffles=0

Merged Map outputs=1

GC time elapsed (ms)=268

CPU time spent (ms)=2570

Physical memory (bytes) snapshot=296046592

Virtual memory (bytes) snapshot=3772530688

Total committed heap usage (bytes)=140697600

Shuffle Errors

BAD\_ID=0

CONNECTION=0

IO\_ERROR=0

WRONG\_LENGTH=0

WRONG\_MAP=0

WRONG\_REDUCE=0

File Input Format Counters

Bytes Read=795

File Output Format Counters

Bytes Written=90

# 5.输出结果

root@node1:/usr/local/hadoop/hadoop-2.5.2/myJar# hdfs dfs -cat /usr/local/hadooptempdata/output/deg2/\*

1 A C

1 A G

1 A I

1 B D

1 B F

1 B H

1 B I

2 C E

2 C G

1 C H

1 D G

1 D I

1 E G

1 F H

2 F I