# GridGraph实验报告

姓名：-

学号：-

电话：-

## 硬件环境

1. OS：Ubuntu16.04
2. GCC: 5.4.0
3. CPU: 3.6GHZ, 4 cores (2 threads per core)
4. Memory: 16GB
5. Disk: 3TB

## PageRank delta

### 理论

Basic Pagerank:

其中，P[i]指节点i的Pagerank值，D[i]指节点i的出度, damping常数，一般设置为0.85.

Delta Pagerank:

其中，指上一层迭代中，节点j的Pagerank值的变化。

### 实现

在Pagerank的迭代计算过程中，往往只有一部分点的rank值变化较大，因此，我们可以通过给Delta设置阈值，只有当delta大于阈值才会传播给邻居。

基于Basic Pagerank实现，本人实现了Delta Pagerank：

1. 加载图，初始化
2. 定义四个Bigvector：degree, pagerank, sum, delta. Degree初始为0，通过在所有边上做计算统计每个节点的出度。Pagerank初始为0.15。sum初始为0。delta初始为0.15/degree，如果degree为0，delta初始为0.
3. 定义Bitmap： active\_vertices。初始全1，在迭代过程中，会有很多节点收敛，这样的点在active\_vertices中对应位置置成0。
4. 开始迭代过程，对于每轮迭代：
   1. 对于每条边（边出发点inactive），更新目的点的sum值
   2. 对于每个节点，计算delta值，更新pagerank值。判断delta值与阈值大小关系，如果delta小于阈值，则置active\_vertices对应位为0，否则为1.并统计收敛节点的个数。

详细过程请参考代码。

### 实验结果

本人测试了内存为8G、16GB情况下，Basic Pagerank和Delta Pagerank每次迭代的平均时间如下表所示：

Twitter数据集：

|  |  |  |
| --- | --- | --- |
|  | 8G | 16G |
| Basic Pagerank | 63.3s(10iter) | 9.78s(46iter) |
| Delta Pagerank | 59.03s(46iter) | 4.82s(46iter) |

UK数据集：

|  |  |  |
| --- | --- | --- |
|  | 8G | 16G |
| Basic Pagerank | 249s | 238s |
| Delta Pagerank | 221s | 215s |

Twitter数据集上：

在delta pagerank中，当delta\_threshold设置为0.01时，46次迭代后算法收敛。而Basic Pagerank的收敛需要的迭代次数没有测试。

下图是delta pagerank算法46次迭代过程中，active vertices变化情况。可以看到，绝大部分点再前几轮中已经收敛，只有一小部分点在之后参与了更新过程。

在Twitter数据集上，从上表中可以看出，Delta pagerank在性能上优于Basic pagerank，原因在于：delta pagerank中不会计算已经收敛的点，大大降低了计算的开销。

## Conductance

### 理论

给定割，图分为S1,S2两部分，conductance值计算如下：

是图的邻接矩阵。

### 实现

算法实现过程非常简单，只需要统计出三个数据即可：

1. 出发点在S1中边的个数
2. 出发点在S2中边的个数
3. 一个点在S1中，另一个点在S2中边的个数

### 结果

Twitter数据集：

|  |  |  |
| --- | --- | --- |
|  | 8G | 16G |
| Time | 58.99s | 48.14s |
| Conductance | 1.011366 | 1.011366 |

UK数据集：

|  |  |  |
| --- | --- | --- |
|  | 8G | 16G |
| Time | 151.81s | 150.45s |
| Conductance | 1.004442 | 1.004442 |