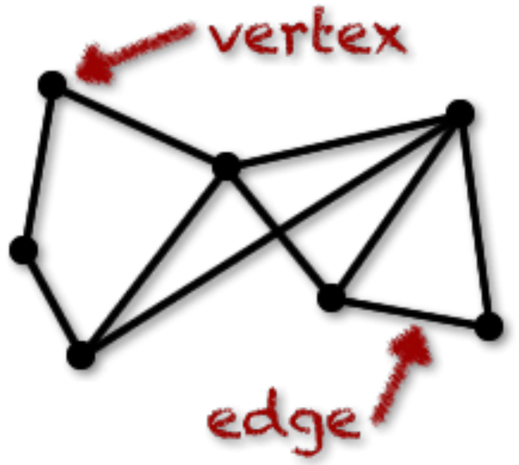
Giovanni Stilo
stilo@di.uniroma1.it



G

Basic principles of G - Graph Library

G

- Low Level **Fast** and **Memory Efficent Graph** Libary in Java.
- Manage **Weighted** Direct / Undirect **Graph** also with weigths on **Vertices**.
- Many algorithms ready to use **PageRank**, **HITS**, many others.
- Multithread **concurrent** algorithm model **configurable** by call.

How to Get G (not on mvn repo)

- GIT Clone or Download :
  - https://github.com/giovanni-stilo/G.git
  - Uncompress the archive (if it is needed) and go to G-master directory;
  - Install using maven (mvn install)
  - Include the dependency in your project:

```
<dependency>
        <groupId> it.stilo </groupId>
        <artifactId> G </artifactId>
        <version> 1.0.0-PUBLIC </version>
</dependency>
```

Main Concepts
Create Graph

- The **core** of G is based on simple **Arrays**; for this reason is very **fast**; the **drawbacks** of this approach is that you must specify the **biggest vertex id +1** at graph **creation** time.
- G is **memory efficient** even if the graph is very sparse and some index are not used.

-
  ```
  WeightedDirectedGraph g = new
  WeightedDirectedGraph(4+1);
  ```

```
    g.testAndAdd(0, 1, 0.1);
    g.testAndAdd(0, 2, 0.2);
    g.testAndAdd(0, 3, 0.3);
    g.testAndAdd(0, 4, 0.4);
```

Main Concepts
Load Graph

- Create the graph and read the graph from a file with:
  - `GraphReader.readGraph`
- The file must contain a list of edge (one per line) **tab** separated with weight in text compressed gzip form:
  - ***id_node_A***<TAB>***id_node_B***<TAB>***edge_weight***<CR>

```
WeightedUndirectedGraph g = new
WeightedUndirectedGraph(3000);

GraphReader.readGraph(g, ”myGraph.gz",
true);
```

Algorithms

Algorithms in G are implemented

externally from the Graph Class; and it is possible to selected number of used cores.

To generate the well know zachary's network for example:

WeightedDirectedGraph g = new WeightedDirectedGraph(**ZacharyNetwork.VERTEX**);

ZacharyNetwork.**generate**(g, **numberOfThreads**);

Print Arrays

3 important arrays with direct access:

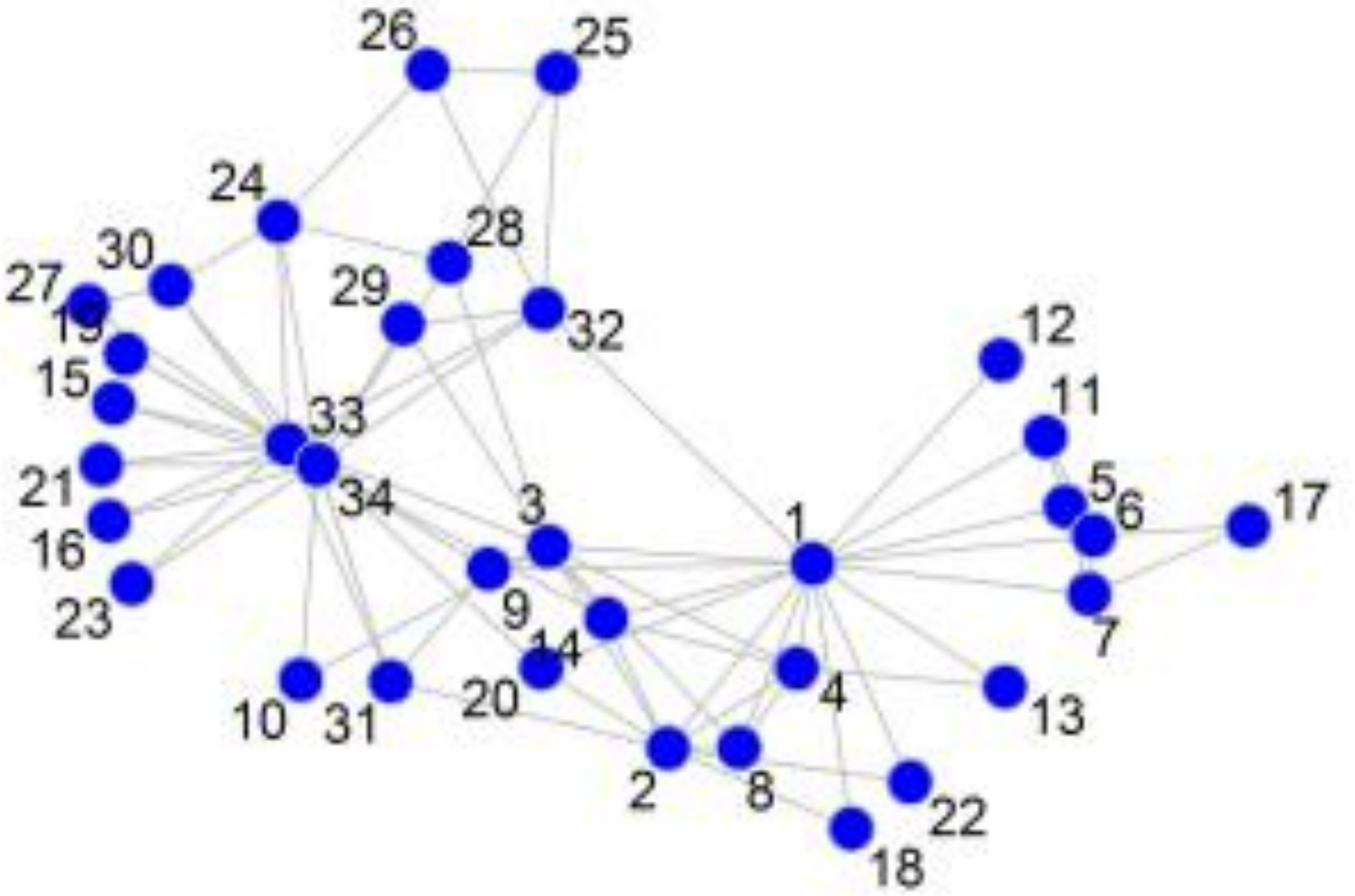
g.**out** { is the outlink of everynodes}

g.**weights** { is the weigths of outlink of everynodes}

g.**in**{ is the inlink of everynodes}

System.out.println(Arrays.deepToString(g.out))

Zachary



26
25
24
28
30
27
29
32
19
15
33
12
11
21
34
5
6
17
3
1
16
9
7
23
14
4
20
13
10
31
2
8
22
18

HITS

```
ArrayList<ArrayList<DoubleValues>> list;

list = HubnessAuthority.compute(g, 0.00001, worker);

        for (int i = 0; i < list.size(); i++) {

            ArrayList<DoubleValues> score = list.get(i);

            String x = "";

            if (i == 0) { x = "Auth ";

            } else { x = "Hub ";}

            for (int j = 0; j < score.size(); j++) {

                System.out.println( x + score.get(j).value +
                              ":\t\t" + score.get(j).index);

            }

        }
```

Exercise G

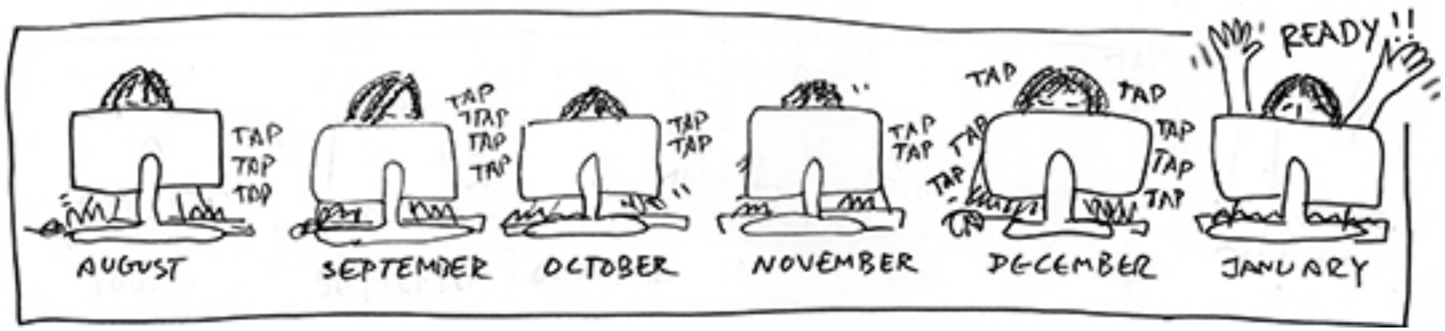
- Using G load Zachary network and use HITS on that.
- Try to remove subsequently the most important Authority

  (***g.remove(VERTEX_ID);***)

How the ranks are changed?

- Try to create your own graph on **file** and **load** it with G.

Let's Try?!?!


TAP TAP TAP
AUGUST
TAP TAP TAP TAP
SEPTEMBER
TAP TAP
OCTOBER
TAP TAP
NOVEMBER
TAP TAP TAP TAP TAP TAP TAP TAP
DECEMBER
READY!!
JANUARY

Dealing with String

```
NodesMapper<String> mapper = new NodesMapper<String>();
String stra = ″ANY_STRING_A";
String strb = "ANY_STRING_B";
int ida     = mapper.getId(stra);
int idb     = mapper.getId(strb);
System.out.println( mapper.getNode(ida) + ": ″ + ida );
System.out.println( mapper.getNode(idb) + ": ″ + idb );


WeightedDirectedGraph g = new WeightedDirectedGraph(2+1);
g.add( mapper.getId(stra), mapper.getId(strb), 1.0d);
```

Copy & SubGraph

```
WeightedUndirectedGraph g1     =
               UnionDisjoint.copy(g, numThreads);


WeightedDirectedGraph g1 =
               SubGraph.extract(g, {1,3,5},
numThreads);
```

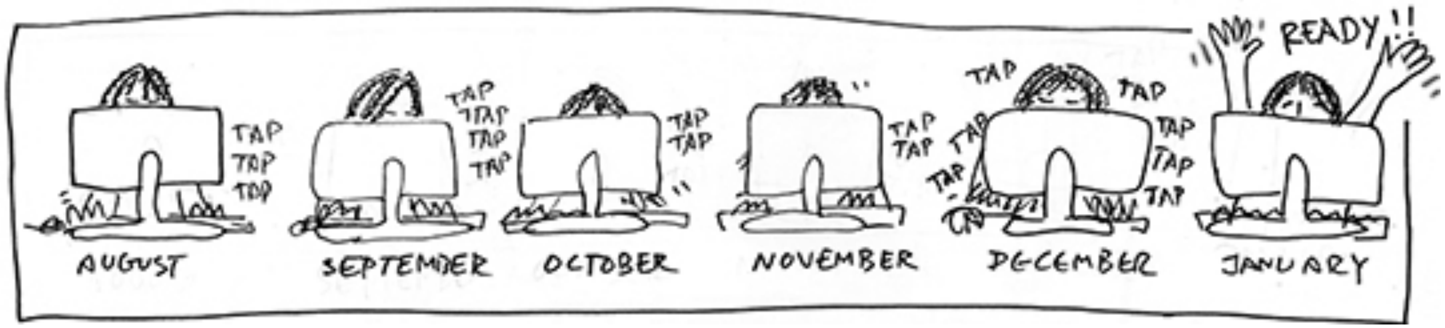
KPP-NEG

```
List<DoubleValues> brokers

             = KppNeg.searchBroker(g, g.getVertex(), worker);


for(DoubleValues b : brokers){

       System.out.println(

             ”Broker value: ” + b.value +

             “\tid: " + b.index);

}
```

Exercise

- Using Twitter API get the network for 15 users (Politics, Football Player).
- Twitter ID are long or more, so use NodeMapper to convert to int;
- Load the network in G and execute KPP-NEG on that.

Let's Try?!?!


TAP TAP TAP
AUGUST
TAP TAP TAP TAP
SEPTEMBER
TAP TAP
OCTOBER
TAP TAP
NOVEMBER
TAP TAP TAP TAP TAP TAP TAP
DECEMBER
READY!!
JANUARY