Data Mining 26.11.2017 – Piotr Mrowczynski – Assignment 3

The objective of this assignment was to allow triangle counting of huge graphs, where edges are being streamed continuously. The algorithm also cares about memory requirement, and uses probabilistic methods to store only M edges in memory and still allow accurate triangle count predictions.

Project structure

```
stream-graph-proc
├── src/main/java/mrow4a/spark/java/
│   ├── TrianleCountJob.java          :  Job counting triangles for streamed edges
│   │
│   └── alg
│       ├── TrianleCount.java         :  Algorithm of Triest Base implementation
│       ├── BiasedCoint.java          :  Biased coin algorithm implementation
│       ├── EdgeParser.java           :  Helper lib to parse text into edges
│       ├── ReservoirSampling.java    :  Reservoior sampling algorithm based on biased coin
│       ├── Edge.java                 :  Helper lib wrapping edges
│       ├── Vertex.java               :  Helper lib wrapping vertexes
│       ├── TriestState.java          :  Helper lib wrapping seen edges and triangle counts
│       └── ReservoirGraph.java       :  Implementation of graph with memory restrictions
│                                        based on reservior sampling
├── build.sh                          : Building project
│
├── run.sh                            : Running project
│
└── StreamGraphProc.odt
```

Building (Docker required for maven single assembly)

chmod +x stream-graph-proc/build.sh
chmod +x stream-graph-proc/run.sh
stream-graph-proc/build.sh

Running

stream-graph-proc/run.sh

Dataset: 16,000,000 edges
Triangle count exact: 7,515,023

Results:
Memory limit: 2,000,000 edges
Current triangle count estimate: 7,594,782 at t=16,518,947

Memory limit: 1,000,000 edges
Current triangle count estimate: 7,626,910 at t=16,518,947

Memory limit: 100,000 edges
Current triangle count estimate: 9,015,504 at t=16,518,947