**GROUP NUMBER: 008**

**GROUP MEMBERS: Riccardo Bellan, Davide Cosma, Riccardo Forzan**

**AVAILABLE INPUTS:** All input files are available in hdfs://data/BDC2223/. The file Orkut117M represents the Orkut social network and it has 117185083 edges and 3072441 nodes. The files OrkutXM with X in {1,2,4,8,16,32,64} are subsets of the previous with X millions edges. (See details here: https://snap.stanford.edu/data/com-Orkut.html)

**TEST 1:** The goal of this test is to assess the scalability of the exact and approximate algorithms with respect to the number of executors. You must fill in the following table.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **SCALABILITY WITH RESPECT TO NUMBER OF EXECUTORS** | | | | |
| **Number of executors** | **Exact algorithm through Node Coloring**  **C=8 colors, R=3 runs, file: orkut4M.txt** | | **Approximation through Node Coloring**  **C=16 colors, R=3 runs, file: orkut4M.txt** | |
| Exact number of triangles | Total running time in seconds (mean of 3 runs) | Approx. number of triangles (median of 3 runs) | Total running time in seconds (mean of 3 runs) |
| 2 | 12184731 | 157563 ms | 15327232 | 8566 ms |
| 4 | 12184731 | 81197 ms | 11370752 | 5049 ms |
| 8 | 12184731 | 44150 ms | 11066112 | 3491 ms |
| 16 | 12184731 | 29804 ms | 11225088 | 1900 ms |

**TEST 2:** The goal of this test is to assess how the approximation algorithm scales with respect to the input size and to show that it can efficiently handle large inputs. To this purpose you will use the orkutXM.txt datasets for increasing values of X. Fill in the following table, stopping at the largest dataset that your algorithm is able to handle in at most 300 seconds (5 mins) per run (average over 3 runs). Hopefully, your algorithm will be able to handle the largest size (X=117) within this time.

|  |  |  |
| --- | --- | --- |
| **SCALABILITY WITH RESPECT TO INPUT SIZE** | | |
| **Dataset** | **Approximation through Node Coloring**  **C=8 colors, R=3 runs, 8 executors** | |
| Approx. number of triangles (median of 3 runs) | Total running time in seconds (mean of 3 runs) |
| Orkut1M | 3080448 | 1452 ms |
| Orkut4M | 11653568 | 3006 ms |
| Orkut16M | 53177088 | 8898 ms |
| Orkut64M | 292978880 | 37063 ms |
| Orkut117M | 619280000 | 71633 ms |

**GENERAL HINTS:**

* The RDD of the input file in each experiment should be divided into 32 partitions and cached.
* Do not include the reading of the input in your running times
* In your program, after defining the Spark Configuration variable “conf”, add the line

conf.set("spark.locality.wait", "0s");

which should force Spark to use all required executors even for small datasets.