course: Big Data Computing — Master's Degree in Computer Science

Homework 1

instructor: Prof. Irene Finocchi — March 16, 2016

Deadline: April 15, 2016

Please, read carefully the following instructions:

- *You should hand in a report (as a pdf file) summarizing your solutions and, when implementation is required, your code in a compressed archive. Instructions to run the code must be contained in the report, besides a description of your solution.*
- *You are allowed to work in groups up to three persons: write in the report your names, student ID numbers, and the name of your Master's degree.*
- *Write legibly: English is preferable, but Italian is equally fine. The description and analysis of your algorithms and implementation details should be as clear as possible (which does not imply long).*
- *If you group is formed, e.g., by two students named "StudentA" and "StudentB", send me an email with subject: "BDC 2016 Homework 1: StudentA Student B". Similarly, your report must be named "Homework1-StudentA-StudentB.pdf".*

Problem 1: File Permutation

Implementation. Implement the *scan & sort* external memory permutation algorithm. You can exploit the C software machinery used in class for the generation of large files and for I/O-efficient sorting. Assume that your program is given two separate input files containing the data to be sorted and the permutation, respectively.

Experimental analysis. Experiment with your implementation, reporting in charts or tables the running time obtained by your implementation on files of increasing sizes.

Input data. Data to be permuted can be generated using `make-rand-file`. The permutation *should* be randomly chosen. If you do not use a random permutation, explain in your report how permutations used in the experiments have been generated.

Hint on generating random permutations *(not mandatory)*. Generating a random permutation of integers in $[1, n]$ in external memory is a problem on its own. You can use the following approach.

1. Generate a file of n pairs x y, where x is a random value chosen in some arbitrarily large interval and y is an increasing number. For instance, if $n = 4$ the file could be:

```
210743    1
17942     2
42687     3
135743    4
```

2. Sort the file based on x values. In the previous example you get:

```
17942     2
42687     3
135743    4
210743    1
```

3. Scan the file and remove the x values, obtaining the random permutation:

```
2
3
4
1
```

You can implement step 1 by changing `make-rand-file.c` and step 2 by setting the appropriate comparator and item type in the implementation of k-way mergesort.

Problem 2: Friendship in Social Networks

How would you solve the following problem about friendship in social networks in external memory?

> *You are given a social network dataset as a large file containing the edges of the network, one edge per line. As an example, the beginning of the file could be:*
>
> ```
> 100483    97158
> 307       11484
> 102687    17942
> 11484     307
> ...
> ```
>
> *Notice that the endpoints are not sorted. The "friend" relationship is often symmetric, meaning that if I am your friend, then you are my friend. You must check whether this property holds, and, if the network is not symmetric, generate a list of all non-symmetric friend relationships.*

Design an external memory algorithm to solve this problem. You can describe the algorithm in any combination of English/Italian and pseudocode. Analyze the number of I/O operations of the proposed algorithm, assuming that the graph has n nodes and m edges. You are not required to implement your solution to this problem.