UNIVERSITÀ DEGLI STUDI DI SALERNO
DIPARTIMENTO DI INFORMATICA

DOTTORATO DI RICERCA IN INFORMATICA
(CICLO XIV - NUOVA SERIE)

Tesi di Dottorato in Informatica

The Role of Distributed Computing in Big Data Science: Case Studies in Forensics and Bioinformatics

Abstract

Candidato:
Gianluca ROSCIGNO
Matr.: 8880900108

Tutor:
Prof. Giuseppe CATTANEO

Coordinatore: Prof. Gennaro COSTAGLIOLA

A.A. 2014/2015

Abstract

The era of Big Data is leading the generation of large amounts of data, which require storage and analysis capabilities that can be only addressed by distributed computing systems. To facilitate large-scale distributed computing, many programming paradigms and frameworks have been proposed, such as MapReduce and Apache Hadoop, which transparently address some issues of distributed systems and hide most of their technical details.

Hadoop is currently the most popular and mature framework supporting the MapReduce paradigm, and it is widely used to store and process Big Data using a cluster of computers. The solutions such as Hadoop are attractive, since they simplify the "transformation" of an application from non-parallel to the distributed one by means of general utilities and without many skills. However, without any algorithm engineering activity, some target applications are not altogether fast and efficient, and they can suffer from several problems and drawbacks when are executed on a distributed system. In fact, a distributed implementation is a necessary but not sufficient condition to obtain remarkable performance with respect to a non-parallel counterpart. Therefore, it is required to assess how distributed solutions are run on a Hadoop cluster, and/or how their performance can be improved to reduce resources consumption and completion times.

In this dissertation, we will show how Hadoop-based implementations can be enhanced by using carefully algorithm engineering activity, tuning, profiling and code improvements. It is also analyzed how to achieve these goals by working on some *critical points*, such as: data local computation, input split size, number and granularity of tasks, cluster configuration, input/output representation, etc.

In particular, to address these issues, we choose some case studies coming from two research areas where the amount of data is rapidly increasing, namely, *Digital Image Forensics* and *Bioinformatics*. We mainly describe full-fledged implementations to show how to design, engineer, improve and evaluate Hadoop-based solutions for *Source Camera Identification* problem, *i.e.*, recognizing the camera used for taking a given digital image, adopting the algorithm by *Fridrich et al.*, and for two of the main problems in Bioinformatics, *i.e.*, *alignment-free sequence comparison* and extraction of *k-mer cumulative or local statistics*.

The results achieved by our improved implementations show that they are substantially faster than the non-parallel counterparts, and remarkably faster than the corresponding Hadoop-based naive implementations. In some cases, for example, our solution for k-mer statistics is approximately $30\times$ faster than our Hadoop-based naive implementation, and about $40\times$ faster than an analogous tool build on Hadoop. In addition, our applications are also *scalable*, *i.e.*, execution times are (approximately) halved by doubling the computing units. Indeed, algorithm engineering activities based on the implementation of smart improvements and supported by careful profiling and tuning may lead to a much better experimental performance avoiding potential problems.

We also highlight how the proposed solutions, tips, tricks and insights can be used in other research areas and problems.

Although Hadoop simplifies some tasks of the distributed environments, we must thoroughly know it to achieve remarkable performance. It is not enough to be an expert of the application domain to build Hadop-based implementations, indeed, in order to achieve good performance, an expert of distributed systems, algorithm engineering, tuning, profiling, etc. is also required. Therefore, the best performance depend heavily on the cooperation degree between the domain expert and the distributed algorithm engineer.