Research Project Proposal: Towards a unifying model for data-intensive applications

Nicolò Felicioni, nicolo.felicioni@mail.polimi.it

1. Introduction to the problem

In recent years, the need for software applications that can handle large amounts of rapidly varying and heterogeneous data has become greater and greater. We refer to these applications as **data-intensive**. They require a distributed software system to store and process the large amount of data, in order to exploit the resources of many interconnected computers. The research areas involved in this topic are mainly the database research area and the distributed systems and algorithms one. The former is the area that deals with storing and managing data developing databases and database management systems (DBMS), while the latter has the main focus on the creation of distributed platforms and algorithms to process and analyze large quantity of data.

Data-intensive applications are applications that have as their primary challenge the management of large amounts of data, that are rapidly changing and that are highly heterogenous[4], as opposed to compute-intensive applications, where the CPU is the bottleneck.

The development of this kind of applications is fundamental to sustain the increase of volume, production velocity and heterogeneity of data that will arise in the next future [1]. For example, the Internet of Things (IoT) advent is bringing billions of smart devices that will produce enormous quantity of data [5]. Data produced by IoT devices are also rapidly changing and often have to be analyzed in real-time. Another trending sector is the one of autonomous vehicles [3], that contributes to this phenomenum of "data flooding" making cars continuously produce data. Apart from machine-created data, there are also human-created data, like social networks' data, that billions of users created and continue to create, or, more in general, all the data created while surfing the web. These data are usually collected for advertising and for training recommendation systems of various websites [6]. This "information overload" made business strategies data-driven, i.e. taking into account data collected by customers to make strategies and take business decisions. For these reasons, creating software systems that are scalable and capable of handling large quantities of data is relevant in modern computer science research.

While distributed data systems offer effective programming interfaces to solve specific data processing and management tasks, data-intensive applications present heterogeneous requirements that cannot be satisfied by any of these data systems alone. Because of that, developers in practice build complex architectures that combine multiple systems and then implement application logic in order to orchestrate their interaction. The problem is that, in doing so, they move out of the disciplined programming paradigms of individual systems and lose their benefits in terms of guarantees on the data and transparent deployment and communication. In addition, integrating data systems together necessitates a deep understanding of their semantics, workload assumptions, performance characteristics, deployment strategies, and configuration opportunities. For this reason, the development of a formal *modeling framework*, which defines a high-level programming interface and captures the functionalities and characteristics of data systems, is necessary. Also, this kind of systems usually present intersections among them, therefore a unifying model can be useful to better understand the semantics of the converging concepts of different systems.

2. Main related work

In the data management area, a new breed of systems called NoSQL was born from the desire to overcome the scalability issues of the classical relational databases, at the cost of relaxing some constraints on strong data consistency and giving up some data guarantees. Indeed, it is almost always the case that a NoSQL system doesn't have transaction with fully ACID[1] guarantees. The NoSQL family of databases is difficult to define precisely, because in literature the definition can be found with disparate meanings. Usually, the term is used to indicate a database that is non-relational, can be distributed and it is horizontally scalable. NoSQL systems are valuable tools, but their biggest flaw is that they usually do not support full ACID transactions. However, nowadays OLTP scenarios with great quantity of data are a matter of growing relevance and researchers tried to find a new solution for an efficient distribution of the classical relational database model, with full support for transactions and their guarantees. These types of solutions are called NewSQL, since they try to make scalable as much as possible the traditional relational systems, while preserving all their guarantees.

In the data processing domain, the increasing size of data motivated the development of a new kind of systems explicitly designed for distributed processing in large-scale compute infrastructures. These systems all trace their root to the **MapReduce** paradigm [2]. MapReduce is a programming model introduced by Google in 2004 that enables application programs to be written in terms of high-level operations (Map and Reduce) on immutable data, while the runtime system controls scheduling, load balancing, communication and fault tolerance. The first open-source implementation of the MR[2] paradigm was Hadoop in 2006. From there on, data processing systems evolved to overcome the limitations of the paradigm (e.g. being constrained to use only Map and Reduce operations) and resulted in the development of Spark (for batch data processing) and Flink (for streaming data processing).

3. Research plan

3.1. Goal

The goal of the research is finding a formal model that defines high-level notions and structures, to unify the semantics of concepts, functionalities and characteristics that are recurrent in the various data systems. The purpose is twofold: first, the various data-intensive systems usually present intersections among them, therefore a unifying model can be useful to better understand the semantics of the converging concepts of different systems; secondly, this modeling framework can be a first fundamental step in the direction of a change of paradigm, that leads to a new approach for designing data-intensive application. Following this new paradigm, developers should define the application specifying for example the data to be stored, the guarantees and the performance requirements and a runtime system should determine the best strategies for data format, replication, partitioning and guarantees implementation. In this way, developers no more have to deal with trying to put different and indipendently developed systems together in a sort of "software collage", where the formal guarantees provided by the single systems could be lost.

The nature of the research is hybrid, since it is theory-based, but it needs various experiments in order to validate the hypothesis that will arise during the research period.

[1] Atomicity, Consistency, Isolation, Durability.

[2] Short for MapReduce.

3.2. Research activity

1 **Scope definition**

The first task is to define the scope of the research, i.e. reviewing the research literature to understand which areas are involved in the research topic

2 **Systems identification and classification**

Having identified the research areas of interest, it is necessary to define which systems and software tools will be studied and analyzed during the reserach activity.

3 **Preliminary study of the tools**

The preliminary study of the various tools that are matter of research interest is fundamental to better understand their functionalities, concepts and characteristics. In this phase the various systems will be compared. This study is preparatory for the creation of a first unifying model.

4 **First model**

After a deep dive into the various systems, it is necessary to create a first draft of the model, taking care of all the possible intersections and inconsistencies among them.

5 **Experiments and consolidated model (iterative task)**

Once a first model is created, experiments are needed to validate or invalidate it. Having collected the result of the experimantal phase, it is possiible to correct and consolidate the previous model to create a new model. This task is iterative, which means that the experiments and the creation of a consolidated model must be repeated until a final model is created and it is able to capture all recurring concepts of the systems.

6 **Writing**

This is the final step of the research activity in which the model and the experiments' results are written in the form of a scientific paper and a master thesis.

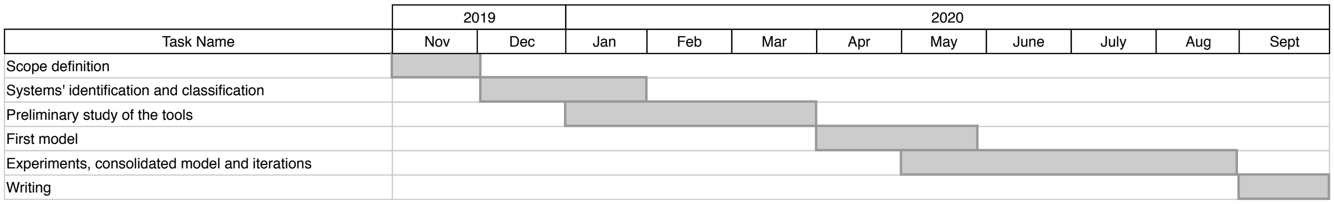

Figure 1: The Gantt chart of the research stages

References

[1] Cilloni, S. Towords a unifying modeling framework for data-intensive tools. Master's thesis, Politecnico di milano, 12 2019. Supervisor: Alessandro Margara.

[2] Dean, J., and Ghemawat, S. Mapreduce: Simplified data processing on large clusters.

[3] Gerla, M., Lee, E.-K., Pau, G., and Lee, U. Internet of vehicles: From intelligent grid to autonomous cars and vehicular clouds. In *2014 IEEE world forum on internet of things (WF-IoT)* (2014), IEEE, pp. 241–246.

[4] Kleppmann, M. *Designing data-intensive applications: The big ideas behind reliable, scalable, and maintainable systems*. " O'Reilly Media, Inc.", 2017.

[5] Nordrum, A. Popular internet of things forecast of 50 billion devices by 2020 is outdated (2016). *Dosegljivo: https://spectrum. ieee. org/tech-talk/telecom/internet/popular-internet-ofthings-forecast-of-50-billion-devices-by-2020-is-outdated.[Dostopano: 11. 8. 2017]* (2017).

[6] Ricci, F., Rokach, L., and Shapira, B. Introduction to recommender systems handbook. In *Recommender systems handbook*. Springer, 2011, pp. 1–35.