Caching Policies with Partial Future Knowledge: the case of Spark

Supervisor
Name: Giovanni Neglia Mail: giovanni.neglia@inria.fr
Telephone: +33 (0) 4 92 38 7906
Web page: http://www-sop.inria.fr/members/Giovanni.Neglia/

Location
Place of the project: Inria Sophia-Antipolis Méditerranée
Address: 2004 route des Lucioles, Sophia Antipolis
Team: NEO
Web page: https://team.inria.fr/neo/

Description
Spark [1] is an in-memory distributed processing framework, that is experiencing rapid growth and adoption due to its use in big data analytics. A crucial reason for its success is its ability to persist intermediate data in memory between computation tasks, which eliminates significant amount of disk I/Os and reduces data processing time.
Crucial to Spark performance is then the capability to maintain in cache the "right data," i.e. the information that is likely to be requested again in the near future.
To this purpose, it has been recently proposed to take advantage of the application execution flow to determine which content to cache [2,3,4]. In particular, Spark represents the execution flow as a Directed Acyclic Graph (DAG), that captures data dependencies across multiple stages.
This application presents important challenges from an algorithmic point of view. When the pattern of future requests is known, the optimal caching policy is Bélády's algorithm: if retrieval of a new content requires to free some space in the cache, then discard the information that will not be needed for the longest time in the future. For the rest, past research has mainly focused on the following scenarios: i) when the content request pattern exhibits some statistical regularity, and ii) when the request process is unpredictable. For example, the Independent Reference Model (IRM) is a classic stochastic model, which assumes content popularities are constant over time and contents requests are drawn independently according to a given probability distribution. The second scenario is usually studied through competitive analysis [5]. The request process in Spark does not fit any of the models indicated above: there is a *partial* knowledge of future requests. Indeed, while the DAG establishes some temporal priorities among different tasks, there is high level of parallelism so that it is not possible to predict the exact execution order.

During the internship the student should
- study the relevant papers indicated below
- propose other caching policies exploiting the DAG information
- evaluate their performance analytically or by experiments with Spark

We think this research topic has the potential to bring to a publication by the end of the internship.

Candidate's profile

We are interested in two possible profiles:
1) a candidate with strong theoretical background on algorithms and probability;
2) a candidate with strong knowledge of Java and hands-on approach to distributed systems. Knowledge of Scala language would definitely be an important plus.

About Inria and NEO

Public science and technology institution established in 1967, Inria is the only French public research body fully dedicated to computational sciences. The Inria Sophia Antipolis-Méditerranée research centre (http://www.inria.fr/en/centre/sophia) is one of the 8 Inria centers, localized in a pine tree forest inside the technopole of Sophia Antipolis (http://www.sophia-antipolis.org/), the first technopole in Europe. It has more than 500 scientists including 200 PhD belonging to 37 different teams. It has strong collaborations with Greece including an agreement with The University of Athens.
NEO team is positioned at the intersection of Operations Research and Network Science. By using the tools of Stochastic Operations Research, we model situations arising in several application domains, involving networking in one way or the other. The aim is to understand the rules and the effects in order to influence and control them so as to engineer the creation and the evolution of complex networks.

Starting date and duration

The 6-month internship can start between September 2017 and January 2018 (the earliest the better).

[1] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica, Spark: Cluster computing with working sets. In Proc. 2nd USENIX HotCloud, 2010.
[2] L. Xu, M. Li, L. Zhang, A. R. Butt, Y. Wang, and Z. Z. Hu, MEM- TUNE: Dynamic memory management for in-memory data analytic platforms, in Proc. IEEE International Parallel and Distrib. Process. Symposium and Design (IPDPS), 2016.
[3] Y. Yu, W. Wang, J. Zhang, and K. B. Letaief, LRC: DAG-Aware Cache Management in Distributed Data Analytics Systems, IEEE Infocom, 2017
[4] Mingxing Duan, Kenli Li, Zhuo Tang, Guoqing Xiao, and Keqin Li, "Selection and replacement algorithms for memory performance improvement in Spark," Concurrency and Computation: Practice and Experience (Special Issue on WPBA 2014), vol. 28, no. 8, pp. 2473-2486, June 2016.
[5] A. Fiat, R. M. Karp, M. Luby, L. A. McGeoch, D. D. Sleator, and N. E. Young, "Competitive paging algorithms," Journal of Algorithms, vol. 12, pp. 685–699, 1991.