2022 PhD topic - Consistency trade-offs in edge computing

Keywords

Edge computing; Replication and consistency; Causal consistency; Hybrid consistency models; distributed programming

Background

Decentralised systems are becoming crucial across many areas of computing. Current trends are pushing applications towards the edge, close to clients and to the source of information.   This *edge-first approach* places data at the network edge; shared data is replicated, possibly in many locations. This improves response time and availability, tolerates high network latency, and enables disconnected operation, while potentially decreasing resource consumption by off-loading data centres.[1]

Strong consistency of the distributed data is desirable, as it is familiar and supports arbitrary application logic. However, it conflicts with availability and response time, because every update requires coordination across the whole system. In contrast, weak consistency models are error-prone for applications and provide lesser guarantees.

A good trade-off is provided by *transactional causal consistency* (TCC), which supports high concurrency and performance, but ensures that updates are atomic and there are no ordering anomalies (i.e., if update v depends on u, all users will observe u before v). TCC requires attaching ordering *metadata* to each update, in order to detect whether an event received might depend on one not yet received. It has been proved that the size of the metadata, in the worst case, is proportional to the number of processes, an unacceptable burden in large-scale edge systems.[2]   Another problem is that metadata tends to accumulate over time and is difficult to garbage-collect safely.

Many techniques have been explored to reduce the size of this metadata, but each one inevitably has drawbacks. For instance, some metadata may be arbitrarily omitted, trading memory for precision.[3]   In specific communication topologies (e.g. a tree) metadata reduces to a scalar, but a single fault may render a whole subtree unreachable.[4] Metadata garbage collection typically requires full system knowledge or strong consistency.

Accordingly, modern systems make trade-offs, using hybrid consistency models that mix (for instance) TCC overall with strong consistency in specific sub-networks, or leverage specific communication topologies.[5][6]. This hybrid approach aims both to mitigate the growth of metadata, and to provide stronger guarantees to the application when or where necessary.

Research

So far, these trade-offs were made in an ad-hoc manner. In contrast, the aim of this thesis will be to study them in a systematic manner. The research has both a fundamental and an applied aspect and aims for practical results.

There are different metadata mechanisms, such as explicit dependency (graph), vector clocks (message numbering), or hash chaining. Metadata serves a number of distinct purposes, such as: to order events (causality), to compare events (concurrency), to detect and resolve conflicts, to optimise network traffic (anti-entropy), for detecting distributed stability, etc. Trade-offs have different effects on each of these aspects, which can be measured or evaluated qualitatively.

Based on a literature search and on a study of existing systems, the PhD student will study the different facets of causal and stronger consistency, in order to isolate, classify, and formalise the trade-offs.   In addition, the student will perform an experimental study (most likely in simulation) of some interesting points in the design space, along metrics such as response time, availability, propagation delay, network cost, loss of parallelism, memory and computation resources, etc.

Ultimately, the goal of the research is to design, implement and evaluate an efficient hybrid consistency model for edge-first systems.

Requirements and application

Candidates should hold a Masters in Computer Science/Informatics or a related field. They should have an excellent academic record and be motivated by the theory and practice of distributed systems, distributed databases, and/or distributed algorithms. Bonus to good development skills, to prior experience with distributed systems and/or experimenting with large-scale software, or to a strong mathematical background.

To apply, please provide the following information:

- A resume or Curriculum Vitæ.
- A list of courses and grades of the last two years of study (an informal transcript is OK).
- A list of your top publications and/or open-source developments.
- Names and contact details of three references (people who can recommend you), whom we will contact directly.

Contact the advisors, Marc Shapiro and Pierre Sens. The research takes place within the The research takes place within the Delys group, a partnership between Sorbonne-Université and Inria. group, a partnership between Sorbonne-Université and Inria..

References

1. Local-first software: you own your data, in spite of the cloud. Martin Kleppmann, Adam Wiggins, Peter van Hardenberg, Mark McGranaghan. *Onward! 2019*, Int. Symp. on New Ideas, New Paradigms, and Reflections on Programming and Software, Athens Greece, October 2019. ↵

2. Concerning the size of logical clocks in distributed systems. Bernadette Charron-Bost. Information Processing Letters 39(1), July 1991. ↵

3. Plausible clocks: constant size logical clocks for distributed systems. Francisco J. Torres-Rojas, Mustaque Ahamad. Distributed Computing volume 12, pages 179–195 (1999). ↵

4. Saturn: a Distributed Metadata Service for Causal Consistency. Manuel Bravo, Luís Rodrigues, Peter Van Roy. EuroSys '17: Euro. Conf. on Computer Systems. Belgrade, Serbia, April 2017. ↵

5. Highly-available and consistent group collaboration at the edge with Colony. Ilyas Toumlilt, Pierre Sutra, Marc Shapiro. Middleware '21: Int. Middleware Conf. Dec. 2021, Québec (Canada), online. ↵

6. A scalable causal broadcast that tolerates dynamics of mobile networks. Daniel Wilhelm, Luciana Arantes, Pierre Sens. ICDCN 2022: Int. Conf. on Distributed Computing and Networking. Dehli, India, January 2022. ↵