The Nearest Replica Can Be Farther Than You Think

Kirill Bogdanov
KTH Royal Institute of Technology
kirillb@kth.se

Miguel Peón-Quirós *
University Complutense of Madrid
mikepeon@gmail.com

Gerald Q. Maguire Jr.
Dejan Kostić
KTH Royal Institute of Technology
maguire@kth.se dmk@kth.se

Cloud services implemented on top of third party cloud environments comprise many useful services and serve hundreds of millions of users spread across the globe. Ensuring low latency when serving user requests is highly important, as it has become one of the differentiating features of most popular services. However, this is a difficult problem as depending on replication policies, the consistency model of a service, and the current network state, clients have to choose which replica or set of replicas they will access, using so-called *replica selection algorithms*. These algorithms are expected to consistently make excellent choices in an unstable environment of geo-distributed systems spread across the wide-area.

The replica selection process is inherently hard. The service's clients conduct passive and active measurements of the latency for served requests, and use this history to drive future choices of the replicas to use. Unfortunately, Internet traffic is bursty and routing frequently changes, hence latency varies.

Errors in replica selection algorithms are extremely hard to find. Such errors usually do not result in critical system failures and it is hard to determine the optimal behavior in the absence of up-to-date, full global knowledge. Moreover, debugging replica selection algorithms is difficult due to the number of potential causes for bugs, such as sampling problems, problems in math calculations, selection logic problems, etc. In this work we propose GeoPerf, a systematic testing tool for replica selection algorithms.

Approach At the core of our approach is a novel technique that combines symbolic execution and lightweight modeling to generate a set of inputs that can expose weaknesses in replica selection. We apply symbolic execution, because it systematically uses the code *itself* to identify test inputs that can cause the code under test to examine all branches in the code and ultimately traverse all possible code paths. In our case, the inputs are the latencies that could be observed by the replica selection algorithm under test, while code paths correspond to different replica choices made by the algorithm.

The core of the tool is based on our own discrete event-based simulator developed as part of GeoPerf. The setup simulates: (i) a set of geo-distributed nodes connected via wide-area network paths, (ii) arrival of incoming client requests and (iii) a replica selection module that periodically chooses a subset of nodes to serve these requests. We create two instantiations of the simulator, one using the reference replica selection algorithm and the other the algorithm under test. Both instances run in parallel in identical environments (using synchronized clocks and deterministic synchronized pseudo-random number generators). A symbolic execution engine is used to drive the exploration of the code paths generating a set of symbolic latencies (i.e., inputs) that characterize the network paths among the nodes. The target of the exploration is to find a sequence of network states that exposes potential weaknesses (bugs) of one of the algorithms by repeatedly demonstrating inferior performance (choices) in the simulated environment.

Contributions We make the following contributions:

1. We conduct thorough round-trip time measurements across *all* geo-distributed datacenters belonging to one cloud provider (Amazon EC2)[1], for several weeks. Using this data, we show that the replica orderings change up to several tens of times per day, from any given datacenter's viewpoint.
2. We propose, design, and implement techniques that overcome challenges in applying symbolic execution to testing replica selection algorithms.
3. Using GeoPerf we found bugs in the replica selection algorithm of two popular geo-distributed data stores, Cassandra[3] and MongoDB[2].
4. In addition, we quantify the impact of the bugs that GeoPerf found. Specifically, we replay the trace of the latencies we collected across Amazon EC2 using GeoPerfs event simulator, and compute the median time that is wasted due to the bugs. In the case of Cassandra, the median wasted time for 10% of all requests is above 50 ms.

Acknowledgments Work funded by ERC project 259110.

References

[1] Amazon ec2. `http://aws.amazon.com/ec2/`.

[2] Mongodb. `http://www.mongodb.org/`.

[3] APACHE. Cassandra. `http://cassandra.apache.org/`.

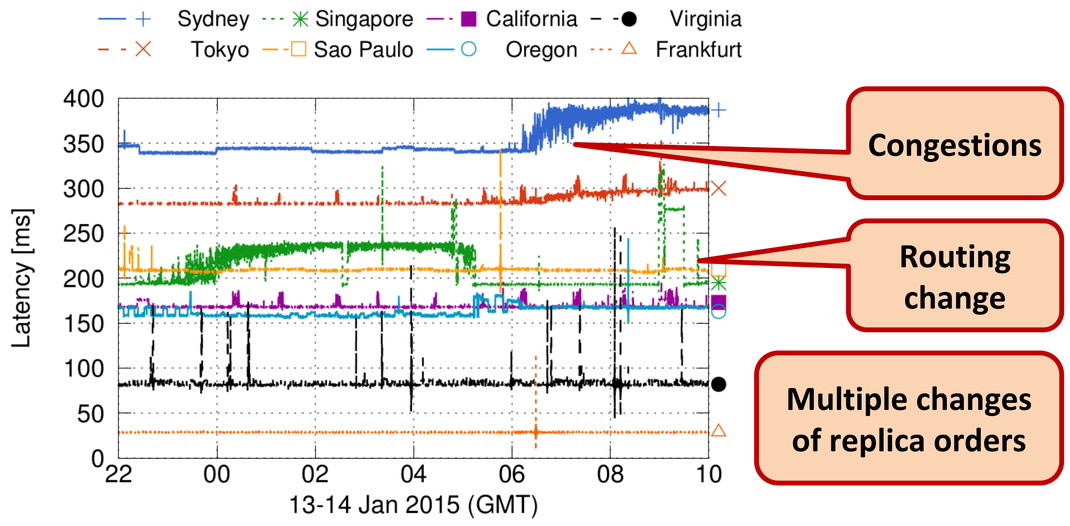
* Work done while the author was at IMDEA Networks Institute.

The Nearest Replica Can Be Farther Than You Think

Kirill Bogdanov, Miguel Peón-Quirós, Gerald Q. Maguire Jr., Dejan Kostić

Problem

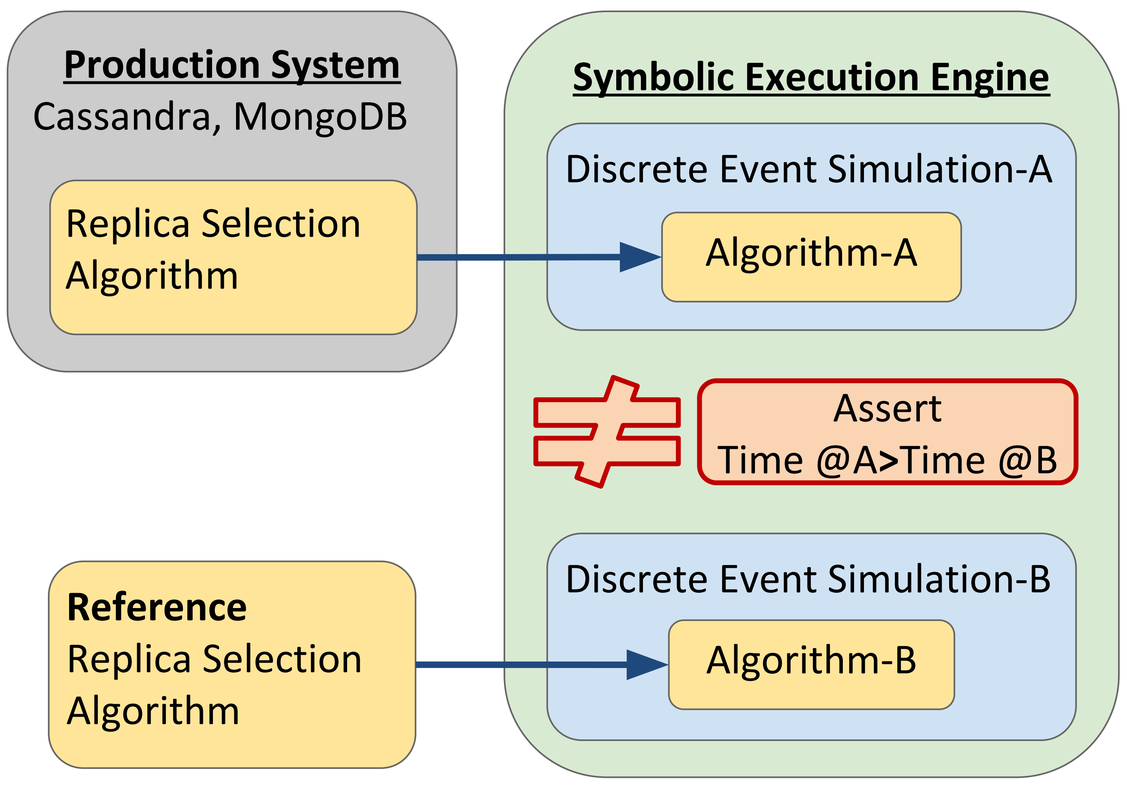
- Geo-Distributed systems are spread across the globe
- Clients need to choose which set of replicas to utilize
- Constantly changing network conditions pose significant problem in determining *closest* replicas



- **Suboptimal replica choices can result in increased latency and can drive a significant fraction of the customers away**

Our Approach (GeoPerf)

1. Extract replica selection algorithm from a production system
2. Instantiate 2 simulations; configure one to use the target algorithm and configure the other to use a reference algorithm
3. Systematically examine code paths in an effort to produce a case in which one algorithm under test performs worse than the reference one
   a. Evaluate decisions using simulation output
   b. Generate concrete test cases to reproduce



Work supported by the European Research Council
Project PROPHET, 259110 (http://prophet.ssvl.kth.se)

Why is it difficult?

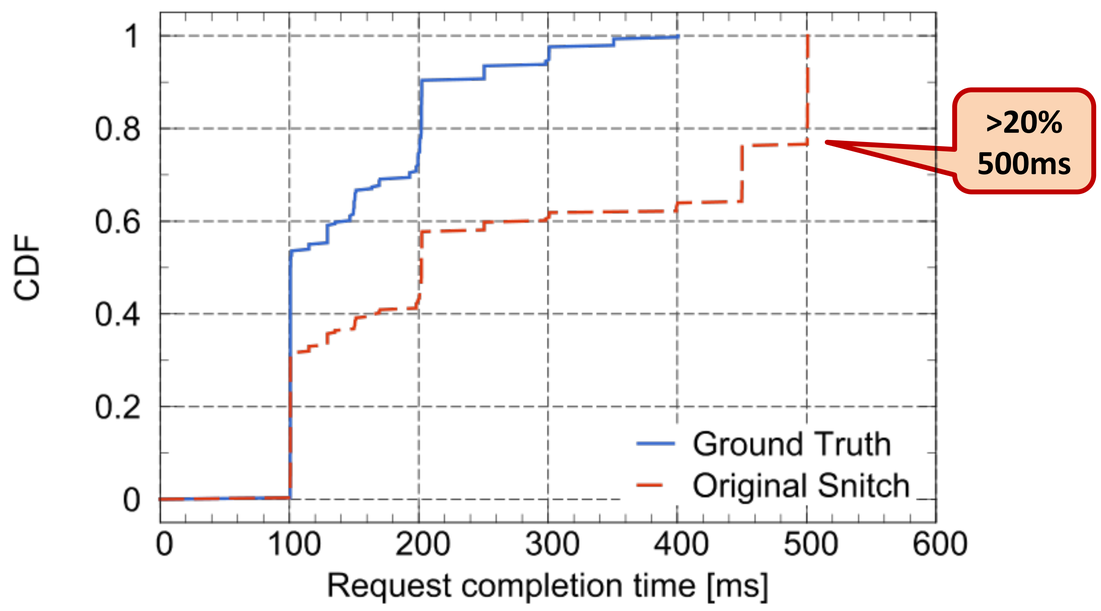
- **Errors** in replica selection algorithms do **not** lead to critical failures (hard to find) and it is hard to determine the correct behavior at a run time without global knowledge
- **Debugging** is complicated by many potential sources of problems
- A huge **testing** space is required to cover the many possible topologies, bandwidth, latency and loss rates

Tested Systems

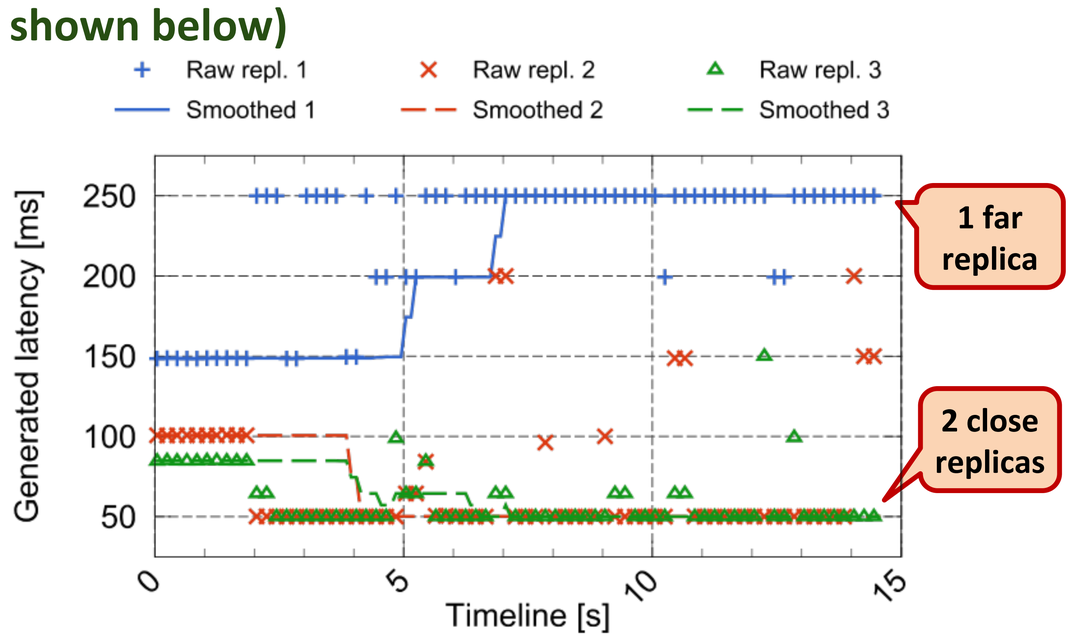
- We have examined 2 production level geo distributed storage systems: **Apache Cassandra** and **MongoDB** and found a bug in each system

Evaluating by Testing Cassandra

- Compared Cassandra's replica selection module with GeoPerf's ground truth model
- GeoPerf was able to generate a set of latency inputs that resulted in algorithms making different choices and as a result achieving different performance
- Performance difference indicate bugs in the code (as shown below)



- **Generated latency trace used to identify real bug (as shown below)**



KTH ROYAL INSTITUTE OF TECHNOLOGY

Kirill Bogdanov, Gerald Q. Maguire Jr., Dejan Kostić
KTH Royal Institute of Technology
School of Information and Communication Technology (ICT)
SE-164 40 Kista
Sweden

Miguel Peón-Quirós
University Complutense of Madrid
Avda. de Séneca, 2
Ciudad Universitaria
28040 Madrid

kirillb@kth.se
mikepeon@gmail.com
maguire@kth.se
dmk@kth.se