D-Tunes: Self Tuning Datastores for Geo-distributed Interactive Applications

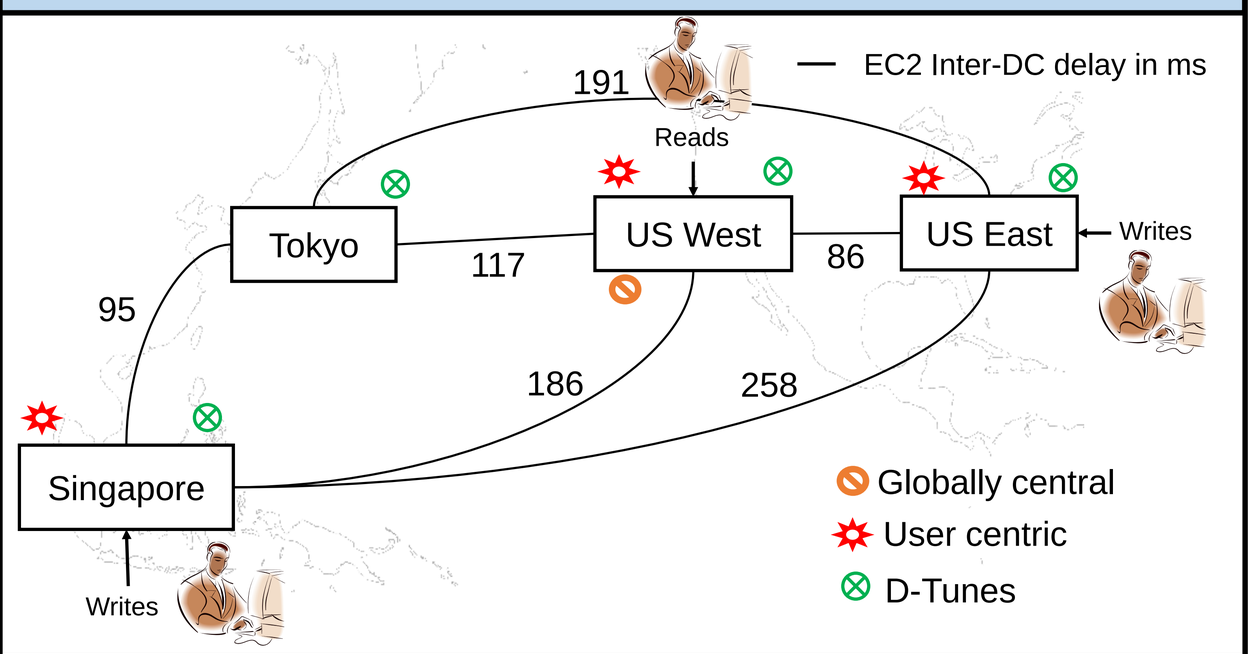
Shankaranarayanan P N, Ashiwan Sivakumar, Sanjay Rao, Mohit Tawarmalani

Purdue University

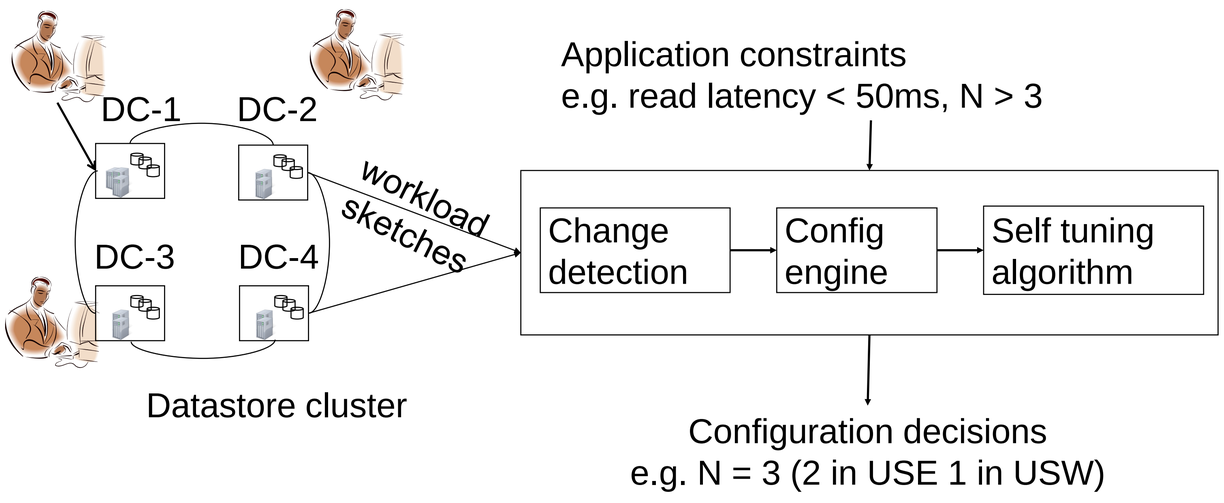
Motivation and Challenges

- Online interactive applications Google docs WIKIPEDIA
- Low latency – data close to users (e.g. < 100ms)
- High availability – DC or server failures, network partitions
- Strong consistency – All reads see the latest write
- Geo-distributed datastores (e.g. Spanner, Cassandra)
- Configuring datastores is challenging
  - Many parameters – location, # of replicas, quorum sizes
  - Judiciously tradeoff consistency, latency and availability
  - Heterogeneity across data items (e.g. location of access)
  - Scale of the data – millions of users (e.g. Twitter)

Motivating example – real world Twitter trace



D-Tunes design

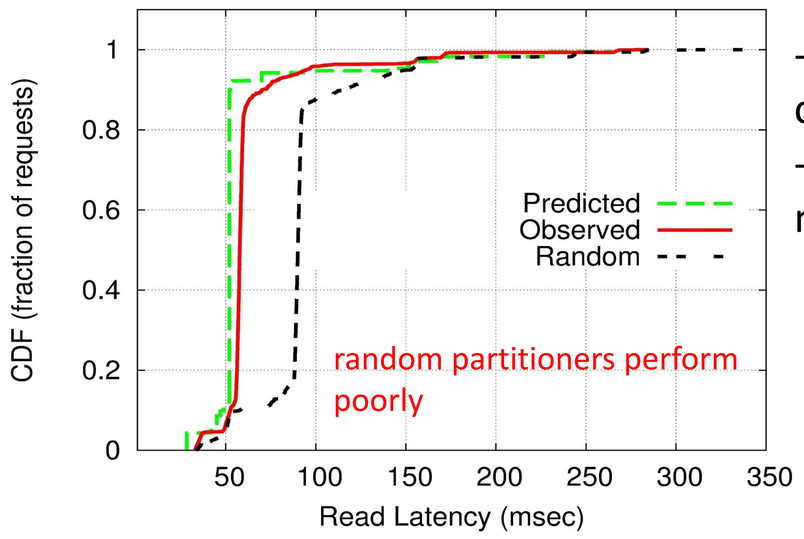


Modeling datastore performance

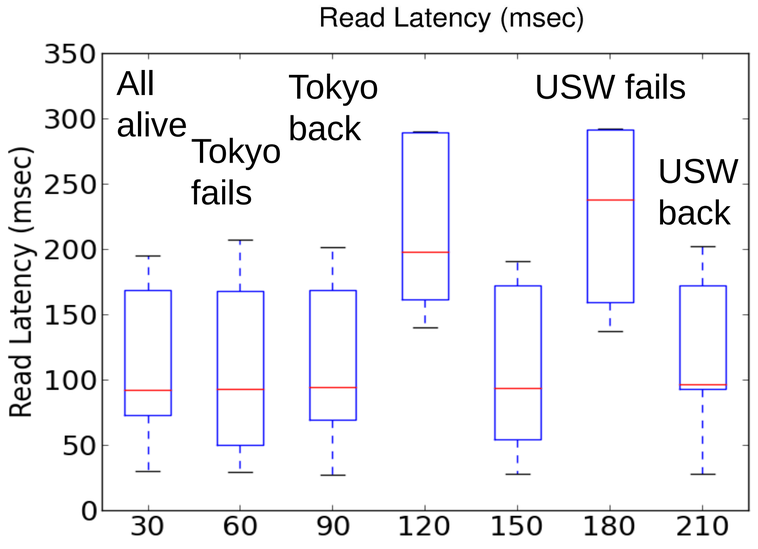
- Analytical models, solved as an optimization problem
- Explore limits on achievable latency given constraints
- Our initial focus - Quorum based systems e.g. Cassandra
  - more models in future – e.g. Paxos
- Novel aspects of our model:
  - Geographical distribution of accesses
  - Latency percentiles to be optimized - SLAs
  - Asymmetry between reads and writes
  - Latency under normal and failures conditions

Experimental validation on Amazon EC2

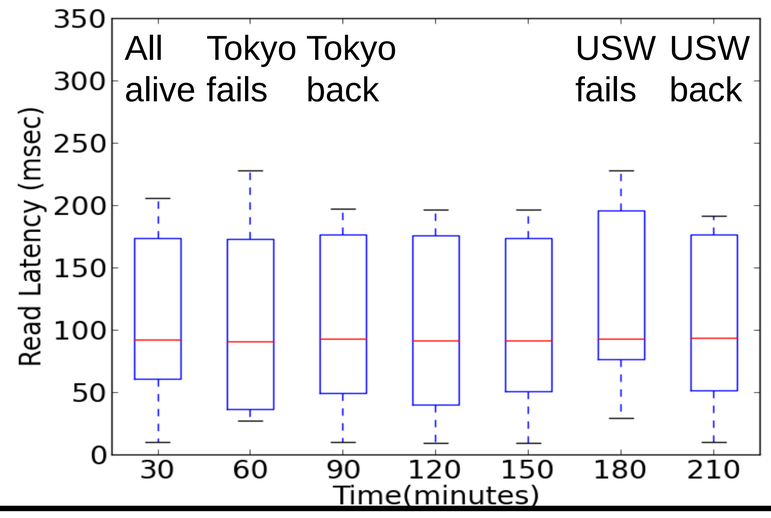
- Cassandra cluster on Amazon EC2
- Across 8 regions, 21 Availability Zones world-wide
- Real world application traces – Twitter, Wikipedia, Gowalla



- observed latency close to predicted
- validates analytical models



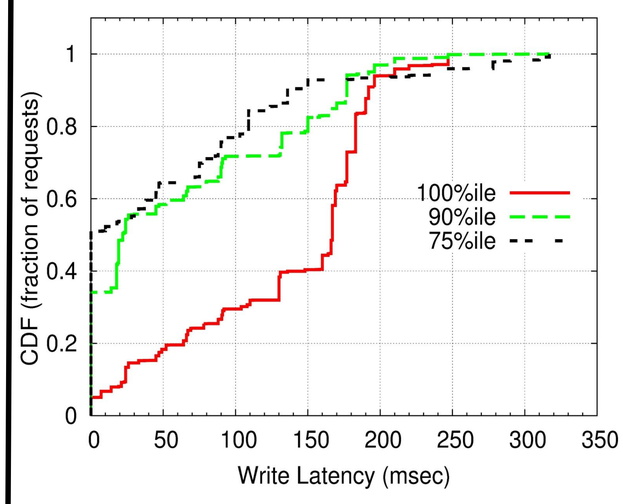
Basic Availability model
- guarantee availability under failures
- variable performance during failures



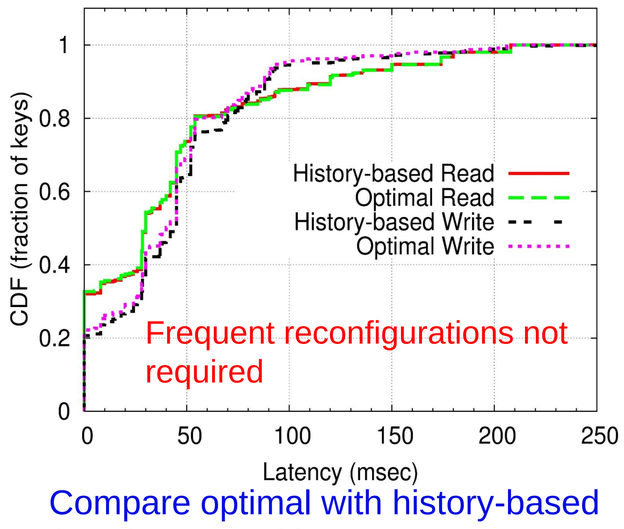
Failure resilient model
- guarantee availability
- Good performance even during failures, congestion events

Large scale experiments – trace driven simulation

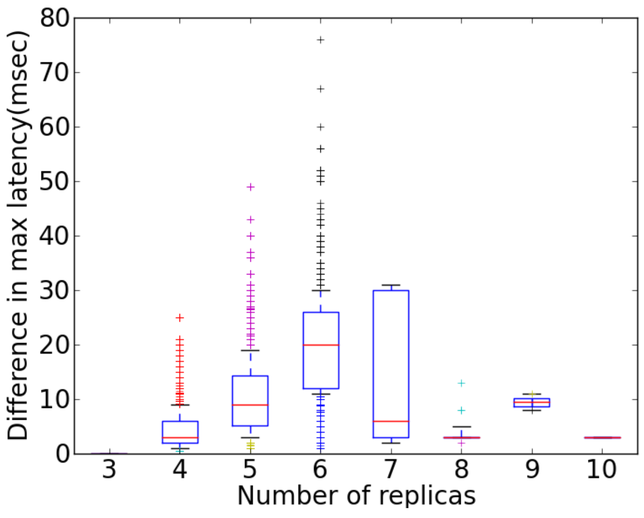
- Real world application traces
  - ✓ Twitter – 5 year trace, 3 million users
  - ✓ Wikipedia – 3 year trace, 4 million+ wiki articles
  - ✓ Gowalla – 2 year trace, 0.2 million users
- Lowers normal operation latency - as much as 40%
- Failure resilient – 55% better than failure agnostic model



Optimizing for the right percentile



Compare optimal with history-based (previous month)



Need for heterogeneous replica configuration:
- uniform replication policy leads to poor performance
- more than 15% of keys in Twitter needed heterogeneity
- benefits as high as 70ms

Acknowledgements

This work was supported in part by NSF grants 0953622 and 1162333

NSF PURDUE UNIVERSITY