**Self Tuning Data-stores for Geo-distributed Cloud Applications**
Shankaranarayanan P N, Ashiwan Sivakumar, Sanjay Rao, Mohit Tawarmalani
Purdue University

**Interactive Online Applications**
- Latency, availability and consistency are ALL critical
- Solutions:
  - **Latency** → higher replication (e.g., CDNs)
  - **Availability** → geo-redundancy (e.g., geo-distributed DBs)
  - **Consistency** → quorum protocols (e.g., Dynamo, Cassandra)

**Challenges**
- Can I achieve all the three for my application?
- Will my application meet client's SLA constraints?
- How do I optimally balance the contrasting requirements?
- Is there a feasible solution for a given set of constraints?
- Can this complex decision making be automated?

**System Design**

![System Design Diagram]

- **Read/Write Access Patterns**
- **Optimization and Configuration Engine (OCE)**
- **Consistency constraints** (R+W > N)
- **Availability constraints** (e.g., N ≥ 3)
- **Latency constraints** (e.g., Tr ≤ 50 ms)

**Additional inputs**
- Percentiles of requests that needs to be optimized
- Priority of reads and writes

**Experimental evaluation of our models on EC2 test-bed**
- Evaluation on a Cassandra cluster with 27 nodes (DCs) using real world application traces - Twitter, Wikipedia
- Location of DCs similar to AWS edge locations
- Inter DC delays measured using planet-lab nodes and emulated using dummynet
- Failure of DCs emulated by shutting down nodes in the Cassandra cluster

**Conclusions**
- Our optimization models
  - evaluate limits on achievable performance given application constraints and a given workload
  - show the importance of choosing different replication strategies across different buckets
  - highlight the significant benefits of optimizing for the optimal latency percentiles
  - show that explicit modeling of performance under failure is critical for good performance
  - are embarrassingly parallel, flexible and easily extensible – replica configuration can be automated at scale