Programmable Storage for Datasets

Team: Aaron Chu, Ivo Jimenez, Jeff LeFevre, Carlos Maltzahn
Institution: UC Santa Cruz

Data movement costs
CPU servers copy data to network, clients copy data from network
Memory: clients store received data
Storage: clients store received data that won’t fit into memory
Network: bandwidth provisioning
- Move computational context to data instead of data to computational context.

Computational context
Completeness: the actual computation has complete input
Sensibility: input is meaningful
Orchestration: data and operations can be mapped to and coordinated within a distributed service, trading off between locality and parallelism.
Reliability: operations can program in spite of hardware and software failures
- Partition data along semantic boundaries so that operations are mostly local

Future: Extreme heterogeneity
Disruption: due to end of Dennard Scaling and Moore’s Law, new devices that undermine traditional software architectures arrive more frequently
Abstraction: software layers isolate applications from disruption but have to integrate assumptions about devices in order to survive
Correctness: correct software takes time to build – incorrect storage systems lose data.

Performance: software layers have to be able to both hide latency of slow devices and hide bandwidth of fast devices, and transform workloads for diverse device performance profiles for a multi-tiered storage hierarchy spanning multiple orders of magnitude performance differences
- Offload device-sensitive operations to storage systems that manage these devices.

Asynchronous data layout and indexing optimization
- SkyhookDM is using fast serialization libraries like FlatBuffers and Apache Arrow to store structured data.
- Structured data can be stored in different physical designs (e.g. with/without indexing, row-oriented vs column-oriented) optimizing for particular access patterns and particular storage devices.
- Physical Design Management is the carefully scheduled orchestration of transformations of physical designs (e.g. indexing, transforming datasets from row to column) balancing beneficial and adverse impacts on production workloads.
- [SWJ+ 17]

References
[LB+ 18] CROSS: The UC Santa Cruz Center for Research in Open Source Software funds part of SkyhookDM development and community work.

Ongoing Integration Effort
- Platforms X Service Clients, Service X
- Method execution in storage server
- Method execution in partner method calls
- Degradation clients中有持久化存储

SkyhookDM Services
- New storage services by individual objects
- Scope of method call:
- Local object
- Custom methods using Ceph-CIS (aka custom object classes)
- RADOS Storage Service
- RAIDOS Storage Service
- SkyhookDM

SkyhookDM Architecture
- “Reducers”
- “Mappers”
- RADOS Storage Service
- SkyhookDM

Contact
Jeff LeFevre
Assistant Adjunct Professor
users.ucsc.edu/~jlefevre
Carlos Maltzahn
Adjunct Professor
people.ucsc.edu/cmaltzahn

This project is supported by National Science Foundation under Cooperative Agreement OAC-1836850. Any opinions, findings, conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.