Belief-based Storage Systems*

Dusan Ramljak, and Krishna Kant
Temple University

*This research was conducted as a part of our CRIS IUCRC, and is supported by Hewlett Packard Enterprise

TEMPLE UNIVERSITY®

Outline

- Motivation
- Goals
- BeliefCache as basis for belief-based storage systems
- BeliefCache extensions
- Belief-based storage systems

Motivation

- Intelligent management of local storage is crucial
  - Rapid rise of remote storage (e.g. cloud storage)
  - Continuing increase in stored data
- Extend the current BeliefCache Framework and preserve
  - All operations data driven
    - including access patterns and provenance information
  - Avoiding user settable parameters as far as possible
  - Ability to self-adjust

Goals

- Design belief-based storage systems where:
  - Belief is an estimate of the probability that the storage entity will be needed
    - Entity might be file, block, segment, object, etc.
    - Multiple aspects of the need might be covered including time, space, and cost
      - Time: how soon will the entity be needed (cannot be too soon or too late)
      - Space: how is the entity location related to others with similar probability and time?
  - Enable as many I/O optimizations as possible while addressing any imposed requirement
  - Intelligent mechanisms should be
    - Largely free of user-settable parameters
    - Robust with respect to workload characteristics

BeliefCache as Basis for Belief-based Storage Systems

- We have already looked at storage issues related to
  - Caching
  - Pre-fetching

- We developed BeliefCache – framework that is
  - Flexible
  - Adaptive
  - Data-driven and can include additional information

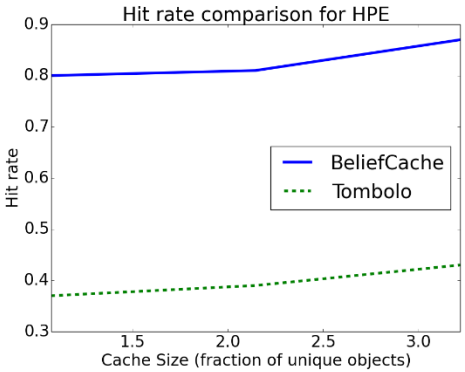
- Looking for ways to solve the issues identified in caching and pre-fetching led us to the idea of belief based storage systems

BeliefCache Properties

- Scheme is designed for relatively large data entities
- Incorporates Bayesian reasoning
  - Belief is an estimate of the probability that data entity is needed
  - Based on relationships gauged from accesses to data entities
- Integrated use of belief guides both pre-fetch and eviction
  - Pre-fetch entities with highest belief values
  - Evict entities with lowest belief values
- Regularly updated beliefs under the current conditions
  - Prevents replacing more useful data from the cache
  - Allows controlling the trade-off of expected latency and related data movement
- Adjusts parameters offline
- Improves the belief online

BeliefCache Results on Examined Workloads

- Complex non-sequential patterns captured better than state-of-the-art framework for optimizing cloud storage gateways (Tombolo[1])



- Our framework is able to adjust to the variations of the workload 2 times faster than Tombolo
  - The results of the adjustment largely dependent on the internal parameters
  - For the perturbations of the same workload
    - We want to inherit parameters not the belief values
  - When we face a phase change (variations that make a different workload)
    - We want to inherit neither belief values nor parameters, but also to identify
      - When the change comes
      - How to improve the speed and quality of adaptation

[1] Yang, Suli, et al. "Tombolo: Performance Enhancements for Cloud Storage Gateways." Proceedings of the 32nd International Conference on Massive Storage Systems and Technology (MSST 2016)

BeliefCache Extensions

- Leverage local vs. global information
- Explore multiple belief sets (Hierarchical belief system)
- Leverage provenance information
  - Provenance could be seen as information about structural relationships
- Leverage I/O slicing
  - All of the instructions in the program that are in some way relevant to deciding what I/Os are issued by the program
  - Like any slice, I/O slice is an actual executable except that it does only the I/Os that are required to decide what I/O to do next

Belief-based Storage Subsystems and Environments

Benefit from belief-based framework that exposes and leverages structural and behavioral relationships

- Belief based data placement (Natural extension to caching)
  - Intelligent data movement mechanism across different types of devices
    - Based on device and access characteristics
- Belief based redundancy removal (Generalization of caching)
  - What representation of data should be stored?
- Belief could also guide
  - Intelligent control of a trade-off between resilience and performance
  - Energy management
  - Metadata management
  - Settings in workflow environment

I am looking forward to
discussing the work during poster session.
Please, ask tough questions ☺
Thank You!