Towards Better Understanding of Black-box Auto-Tuning: A Comparative Analysis for Storage Systems

2018 USENIX Annual Technical Conference

Zhen Cao[1], Vasily Tarasov[2], Sachin Tiwari[1], and Erez Zadok[1]

[1]Stony Brook University; [2]IBM Research – Almaden;

Outline

- **Introduction**
- Background
- Experiment Settings
- Evaluations
- Related Work
- Conclusions & Future Work

Motivation

- Why tuning storage systems?
  - Slow storage impacts I/O bound workloads
  - Default settings are sub-optimal
  - Tuning can provide significant gains
    - 9× [FAST'10]
- Manual tuning is intractable
- Auto-tuning storage systems
  - Black-box optimization is promising
  - Lack of comparison of techniques
  - Lack of understanding

Contributions

- First comparative study on auto-tuning storage systems
  - 5 techniques
- Various aspects
  - Cumulative & instantaneous throughput
  - Impacts of hyper-parameters
- Explanations on evaluation results
  - From storage perspective
- **Future Goal**: complete solution to tune storage systems

Outline

- Introduction
- **Background**
- Experiment Settings
- Evaluations
- Related Work
- Conclusions & Future Work

Concepts

- Storage system
  - File system, underlying storage hardware and any layers between them
- Parameters
  - Configurable options
  - *E.g., file-system block size*
- Parameter values
  - *E.g., 1K, 2K, 4K* (Ext4 block size)
- Configuration
  - Combination of parameter values
  - *E.g., [Ext4, 4K, data=ordered]*
- Parameter space
  - All possible configurations
- Hyper-parameter

Challenges

- Vast parameter space
  - Ext4: 59 parameters, 10^{37} configs
  - Devices, Layers
  - Distributed, large-scale
- Discrete and non-numeric
  - Linux I/O scheduler: noop, cfq, deadline
- Non-linearity
- Sensitivity to environment
  - Hardware & workloads

Manual Tuning
Inefficient

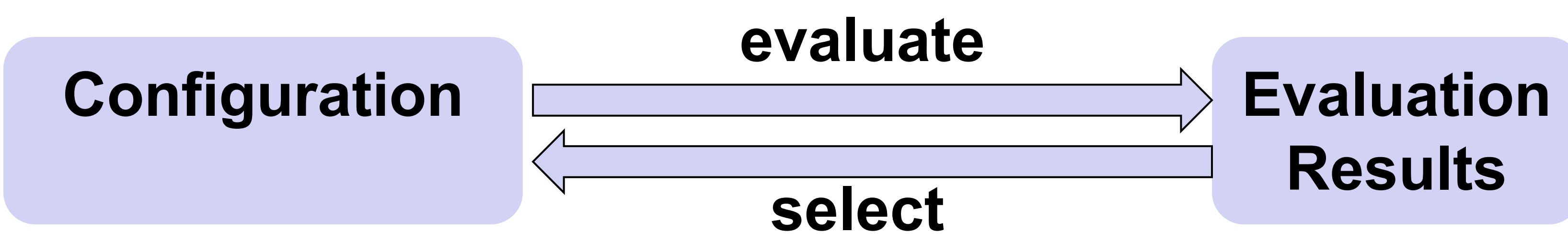
Gradient
Unavailable

Inapplicable Methods

- **Control Theory** ✖
  - Unstable in controlling non-linear systems
- **Supervised Machine Learning** ✖
  - Long training phase
  - High-quality training data
- Inapplicable or inefficient to serve as the core auto-tuning algorithm
  - Could be helpful as a supplement

Black-box Optimization

- Successfully applied in auto-tuning system configurations
- Examples
  - Genetic Algorithms (GA)
  - Simulated Annealing (SA)
  - Bayesian Optimization (BO)
- Obliviousness to system's internals



Key Factors

- **Fitness**: optimization objective(s)
  - Throughput, latency, energy, …
  - Complex cost functions
- **Exploration**
  - Search the unvisited area (e.g., randomly)
- **Exploitation**
  - Utilize neighborhood or history
- **History**
  - How much past data is kept and used for exploration/exploitation

Applied Methods

- Simulated Annealing (SA)
- Genetic Algorithms (GA)
- Deep Q-Network (DQN)
- Bayesian Optimization (BO)
- Random Search (RS)
  - Random selection without replacement

Genetic Algorithms

- Inspired by natural evolution
- Concepts
  - **Gene**: file system, block size, …
  - **Allele**: Ext4, XFS, Btrfs, …
  - **Chromosome**: configuration
  - **Population**: set of configurations

History

- Selection
- Genetic operators
  - Crossover
  - Mutation

Exploitation vs. Exploration

Outline

- Introduction
- Background
- **Experiment Settings**
- Evaluations
- Related Work
- Conclusions & Future Work

Experimental Setup

- Hardware
  - M1: 2 Intel Xeon single-core 2.8GHz CPU, 2G RAM, 73GB Seagate SCSI drive
  - M2: 1 Intel Xeon quad-core 2.4GHz CPU, 24G RAM, 4 drives (SAS-HDD 500GB, SAS-HDD 146GB, 1 SATA-HDD, SSD)
- Filebench
  - Macro-workloads: fileserver, mailserver, webserver, dbserver
  - Default *working set size*

Experiment Setup (cont.)

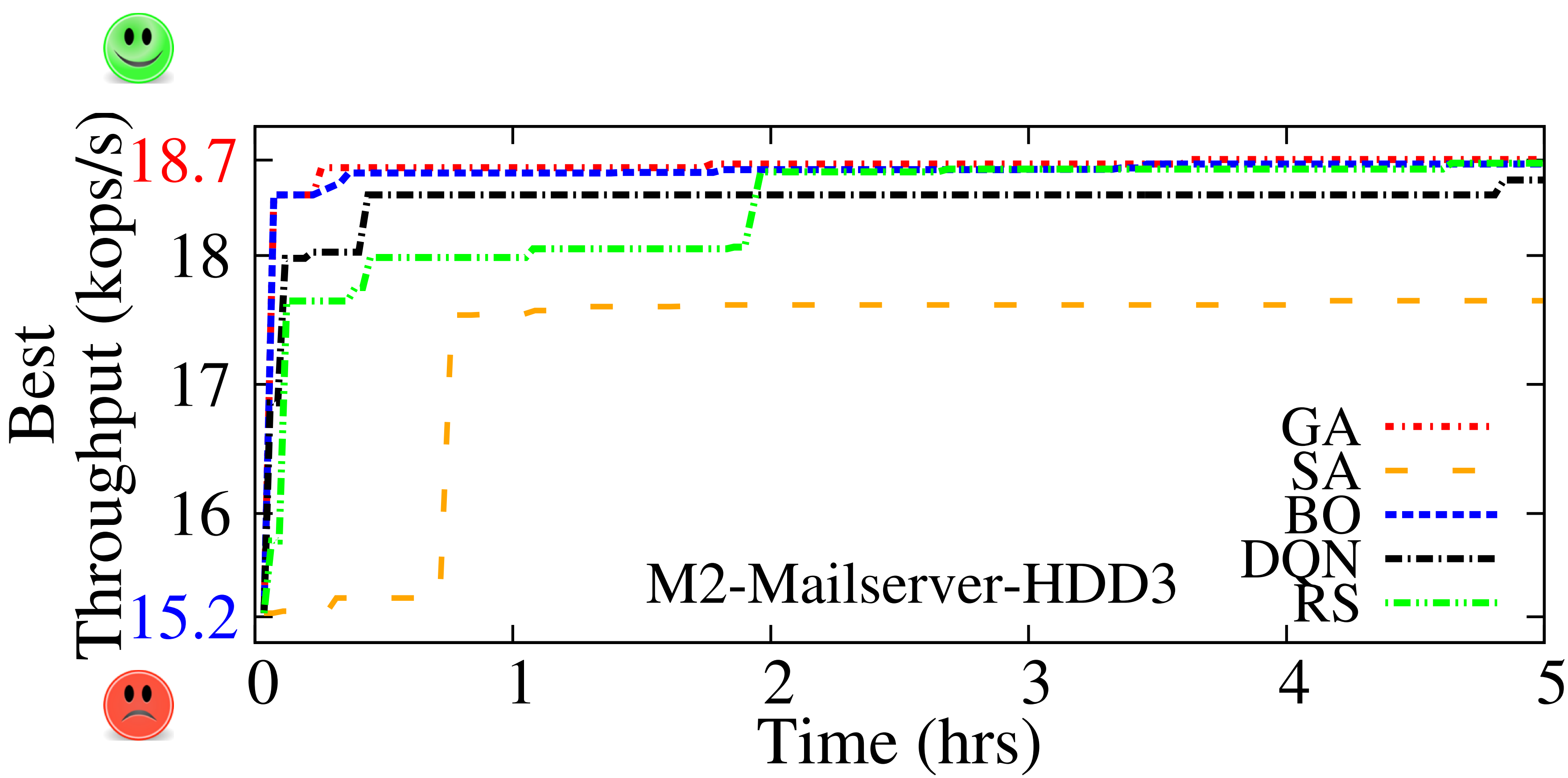
- Search spaces
  - *Storage V1*
    - File system, inode size, block size, block group, journal options, mount options, special options
  - *Storage V2*
    - V1 + I/O scheduler
    - 6,222 configurations
- Methodology
  - Exhaustive Search
    - Storage V2: 4 workloads $\times$ 4 devices
    - 3+ runs for each configuration
    - **Collected over 2+ years**
  - Simulate auto-tuning algorithms

IBM Research Stony Brook University

Outline

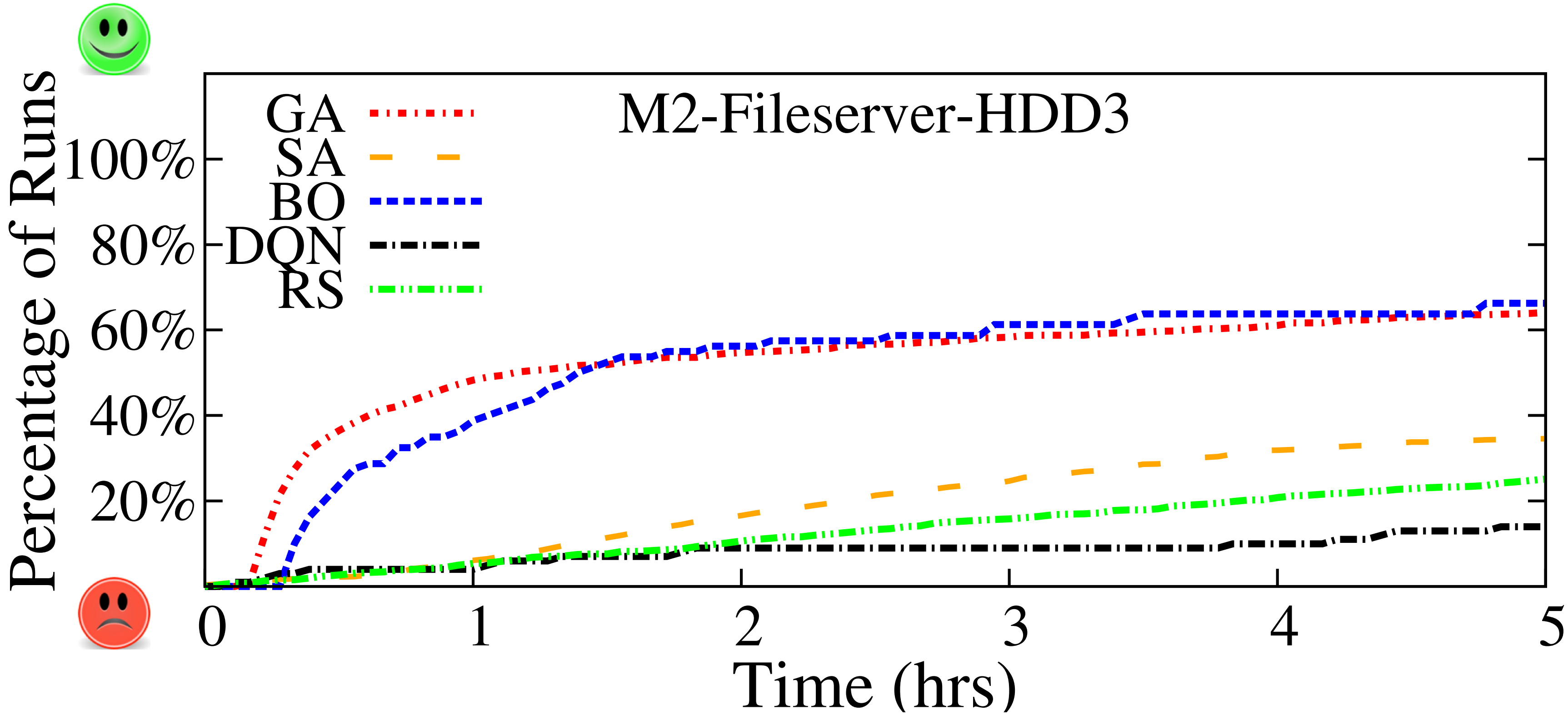
- Introduction
- Background
- Experiment Settings
- **Evaluations**
- Related Work
- Conclusions & Future Work

Best Throughput



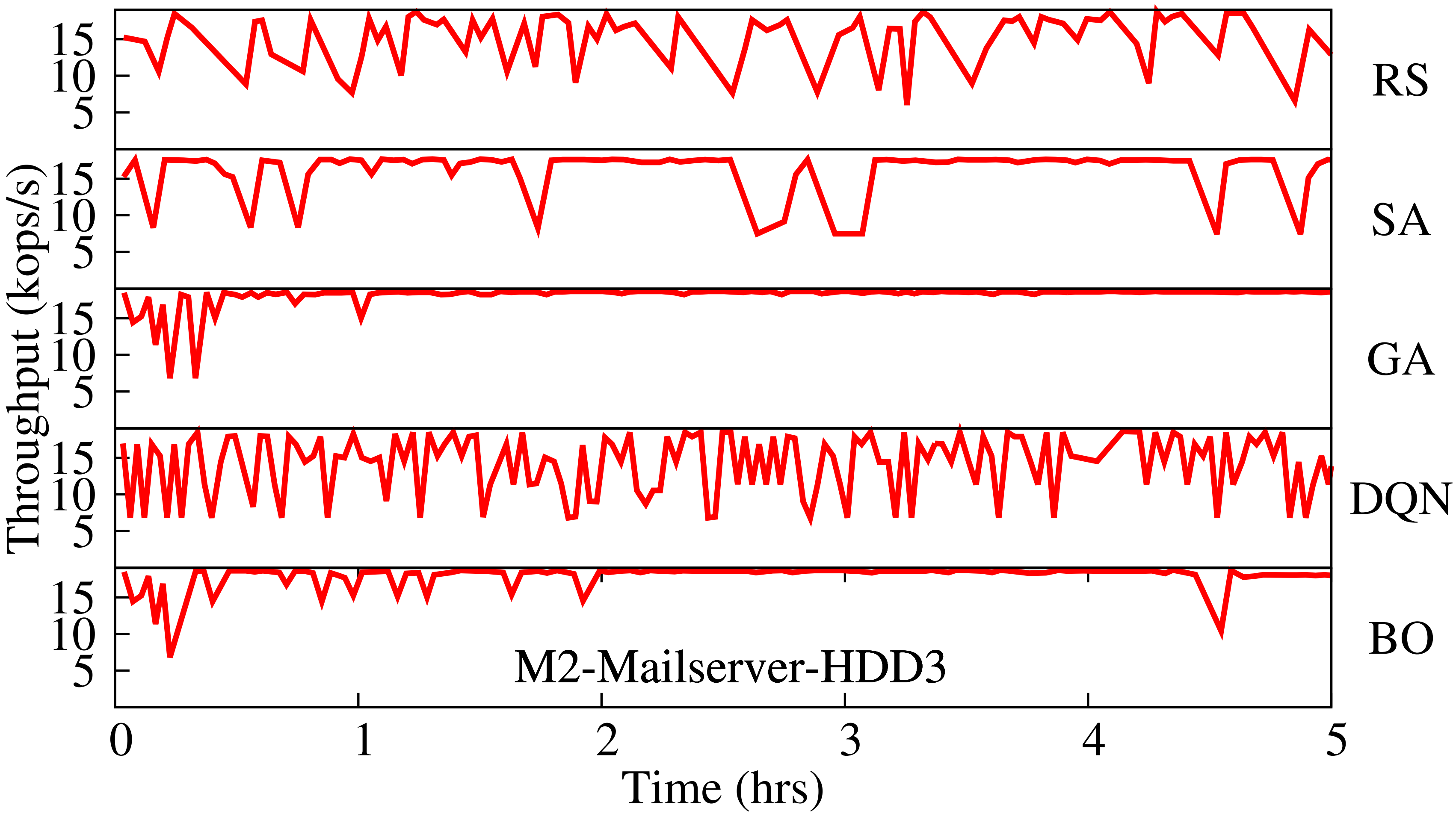
Success rate for finding near-optimal configurations

Near-optimal configuration: one with throughput higher than 99% of the global optimal value.



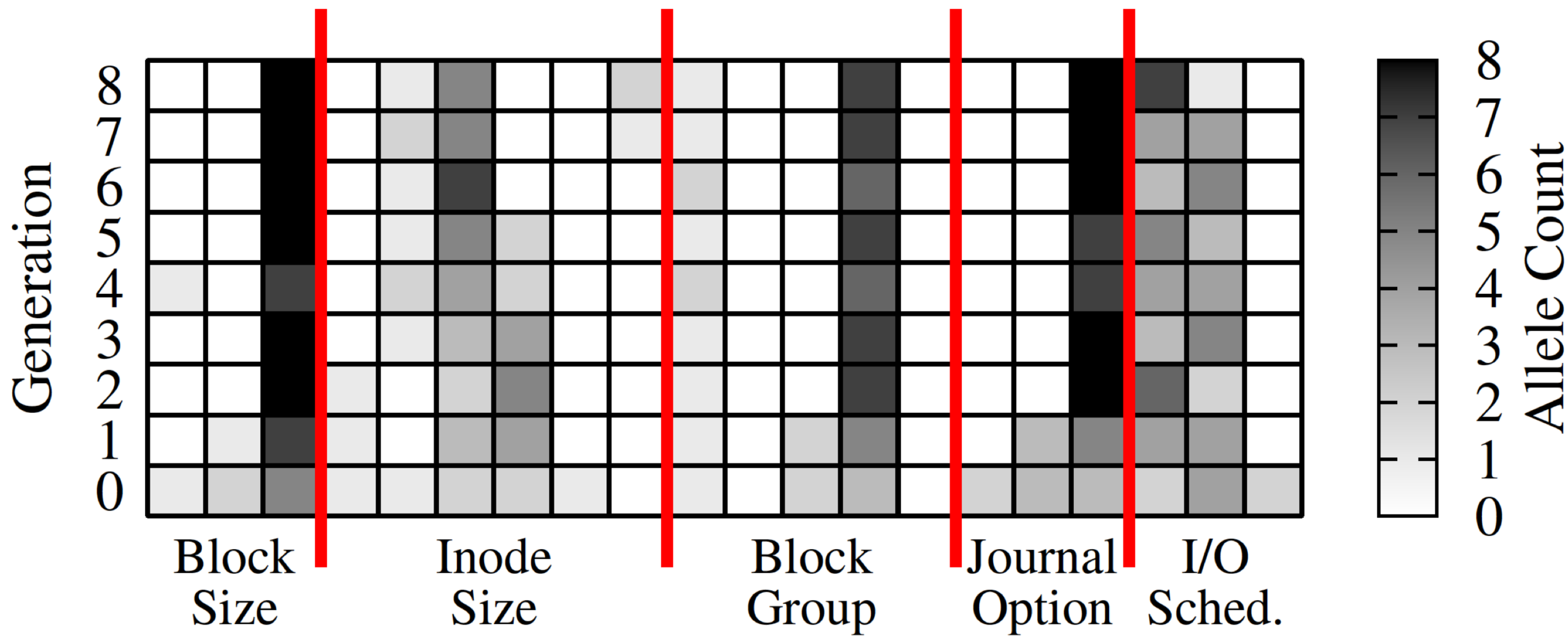
IBM Research Stony Brook University

Instant Throughput



IBM Research
Stony Brook University

Genetic Algorithm (GA) Diversity



Correlation Analysis

- Correlation analysis
  - Ordinary Least Squares (OLS)
  - Example: **block size** and **journal option** are the most correlated Ext4 parameter (Fileserver, SSD)
- Explanations on evaluation results
  - GA and BO can identify important parameters through "*history*"
  - SA keeps no *"history"*; thus performs poorly
  - DQN spends too much time on exploration
  - Random Search
    - Near-optimal configurations take up 4.5% of the whole search space (M2, Mailserver, HDD).

Outline

- Introduction
- Background
- Experiment Settings
- Evaluations
- **Related Work**
- Conclusions & Future Work

IBM Research Stony Brook University

Related Work

- Auto-tuning storage
  - Storage system design (bin-packing heuristics) [Alvarez et al.]
  - Data recovery scheduling (GA) [Keeton et al.]
  - HDF5 optimization (GA) [Behzad et al.]
  - Lustre optimization (DQN) [Li et al.]
- Auto-tuning other systems
  - Database [Alipourfard et al.]
  - Cloud VMs [Aken et al.]

Outline

- Introduction
- Background
- Experiment Settings
- Evaluations
- Related Work
- **Conclusions & Future Work**

IBM Research Stony Brook University

Conclusions & Contributions

- First comparative analysis on 5 techniques on auto-tuning storage systems
  - Efficiency on finding near-optimal configurations
  - Instant throughput
- Provide insights from storage perspective
  - Importance of parameters
    - E.g., impact of mutation rates on convergence
- Valuable datasets
  - Will release to public

Future Work

- More complex workloads and search spaces
- Hyper-parameter tuning
- More sophisticated auto-tuning
  - E.g., penalty functions to cope with costly parameter changes

Towards Better Understanding of Black-box Auto-Tuning: A Comparative Analysis for Storage Systems

Zhen Cao, Vasily Tarasov, Sachin Tiwari, and Erez Zadok

Thank You
Q&A

IBM Research Stony Brook University