Balancing Fairness and Efficiency in Tiered Storage Systems with Bottleneck-Aware Allocation

Hui Wang, Peter Varman
Rice University

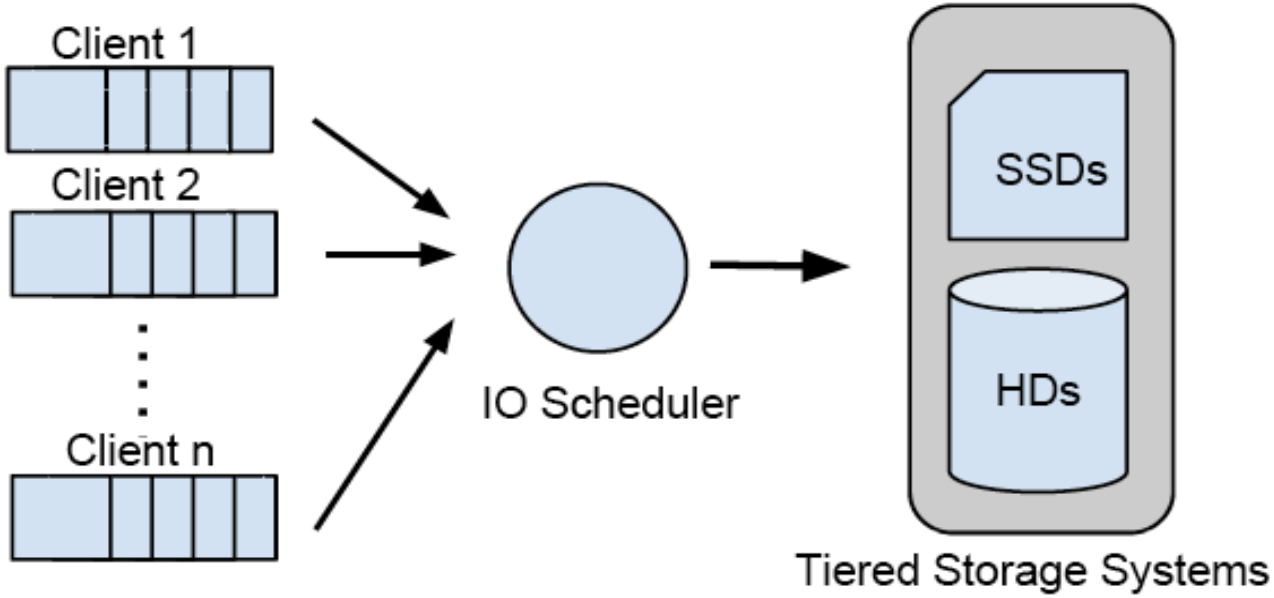
FAST'14, Feb 2014



Tiered Storage

- Tiered storage: HDs and SSDs
  - Advantages:
    - Performance
    - Cost
  - Challenges:
    - Fair resource allocation
    - High system efficiency
      - Variable system throughput

Tiered Storage Model



- **Clients**: Make requests to SSD (hit) and HD (miss) in certain ratio
- **Scheduler**: Aware of the request target, dispatches requests to storage
- **Storage**: SSD and HD independent, without frequent data migrations

Fairness and Efficiency in Tiered Storage

- How do we define fairness?
  - How to define fairness for multiple resources?
  - Fair allocation may cause low efficiency
- How to improve efficiency of both devices?
  - Only focusing on efficiency may cause unfairness

Existing Solutions for QoS Scheduling

- Proportional sharing in storage / IO scheduling
  - Extended from networks and CPU scheduling
  - Additional Reservation and Limit controls
  - *All of them are designed for a single resource!*

- Dominant Resource Fairness Model (DRF) [NSDI'11]
  - Designed for allocating multiple resources
  - *DRF does not explicitly address system utilization*

Talk Outline

- Motivation
- Bottleneck-Aware Allocation (BAA)
- Evaluation
- Conclusions and future work

Example: Single Device Type

- Configuration:
  - Single HD with capacity 100 IOPS;
  - Two clients with equal weights
    - Fully backlogged, Work-conserving
  - Proportional sharing



- Results:
  - Each gets 50 IOPS
  - Utilization 100%

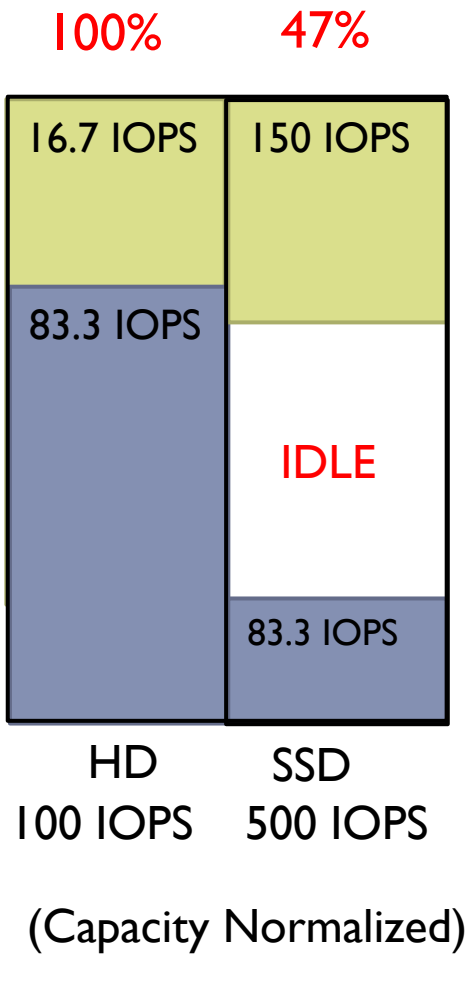
- *Device can be fully utilized for any allocation ratio*

RICE

What if there are multiple resources?

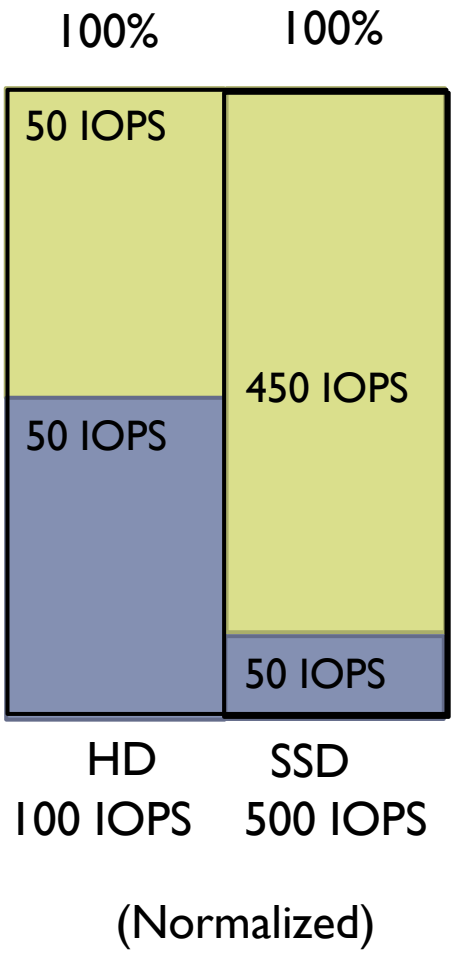
Example: Multiple Devices (Fairness)

- Natural policy: Weighted Fair Queuing
- Configuration:
  - HD capacity 100 IOPS, SSD 500 IOPS;
  - Two clients: h1 = 0.9, h2 = 0.5;
  - Conventional WFQ 1:1
- Results:
  - Each gets 167 IOPS
  - Utilization of HD = 100%, but SSD only 47%
- *Simply transferring WFQ to multiple resources will have efficiency problem!*



Example: Multiple Devices (Efficiency)

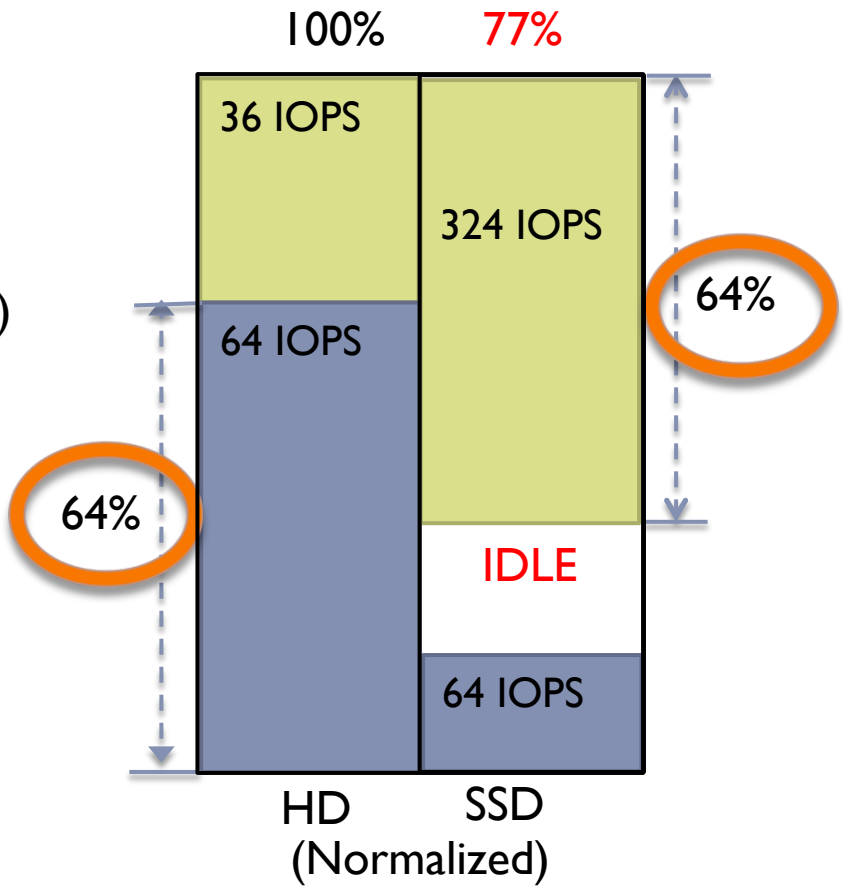
- Configuration:
  - HD capacity 100 IOPS, SSD 500 IOPS;
  - Two clients h1 = 0.9, h2 = 0.5;
- Results:
  - Utilization 100%
  - Client 1 gets 500 IOPS
  - Client 2 gets 100 IOPS
- *It is not possible to precisely assign both the relative allocations (fairness) and the system utilization (efficiency).*



RICE

DRF (Dominant Resource Fairness)

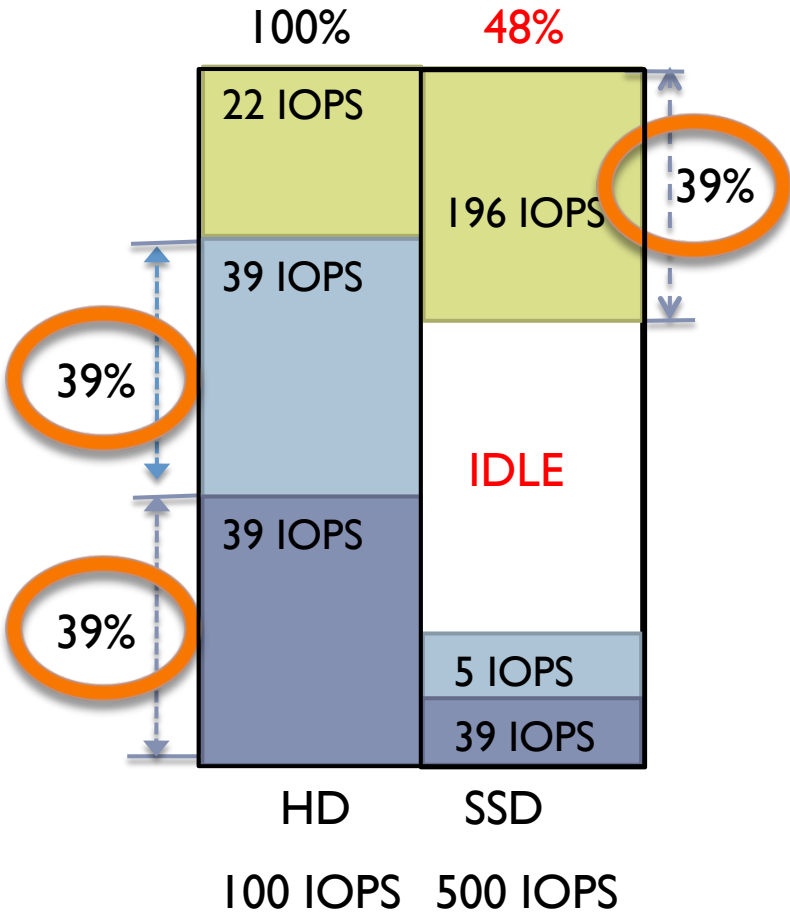
- Configuration:
  - HD 100 IOPS
  - SSD 500 IOPS
  - Two clients
    - h1 = 0.9 (dominant resource SSD)
    - h2 = 0.5 (dominant resource HD)
- What will DRF do?
  - Equalize dominant shares



RICE

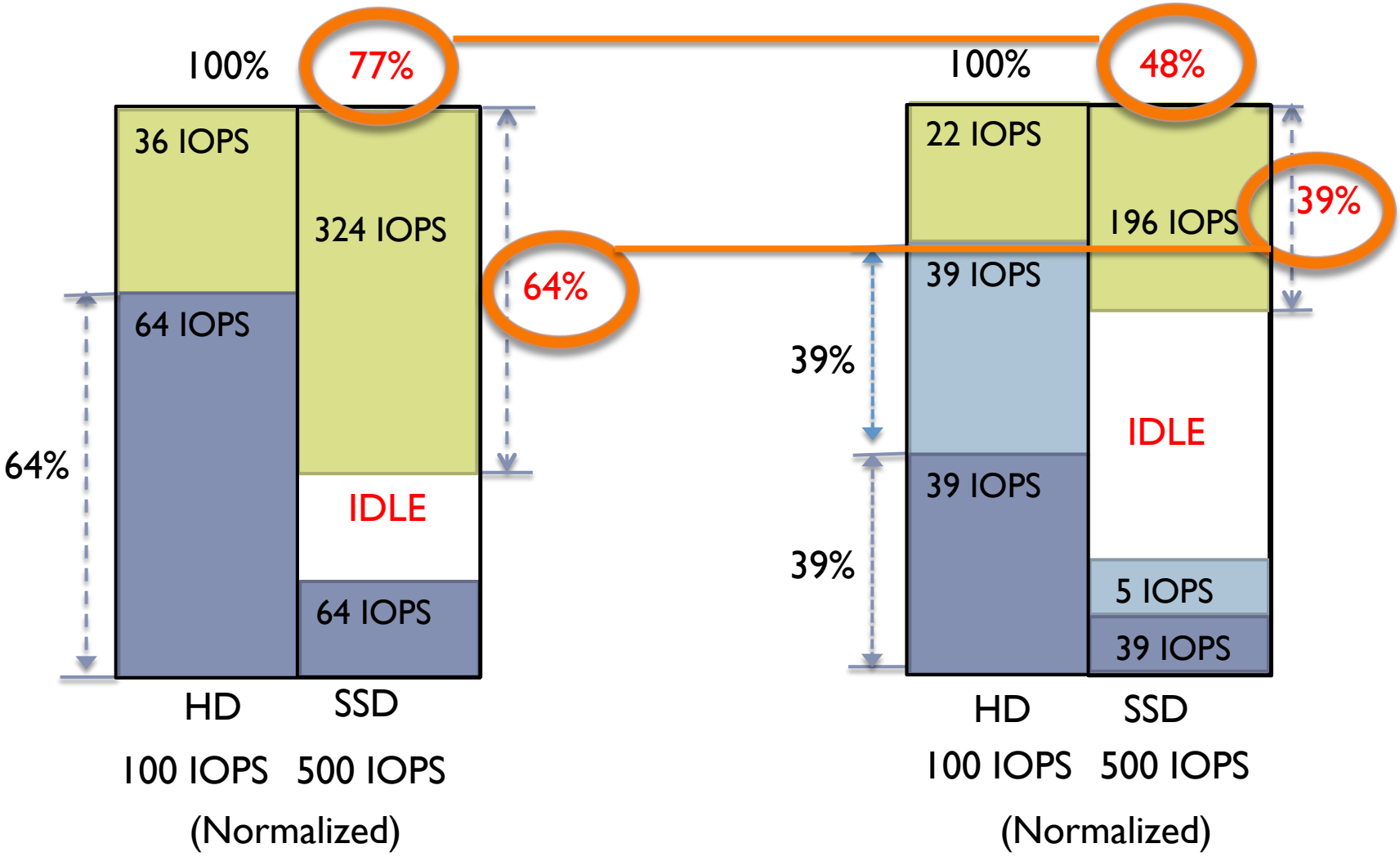
DRF

- Not addressing efficiency
  - Add a third client h3 = 0.1
  - Utilization further reduced to 48%
  - Worse if more clients bottlenecked on HD



RICE

One More HD-bound Client



RICE

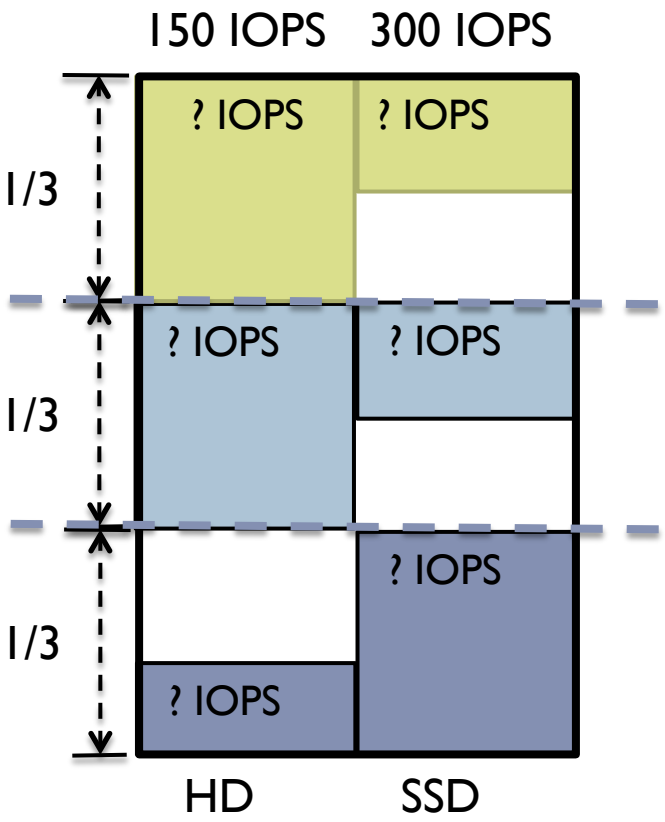
Talk Outline

- Motivation
- Bottleneck-Aware Allocation (BAA)
- Evaluation
- Conclusions and future work

Fair Shares

- Fair Share of a client
  - IOPS it would get if each resource was partitioned equally among the clients

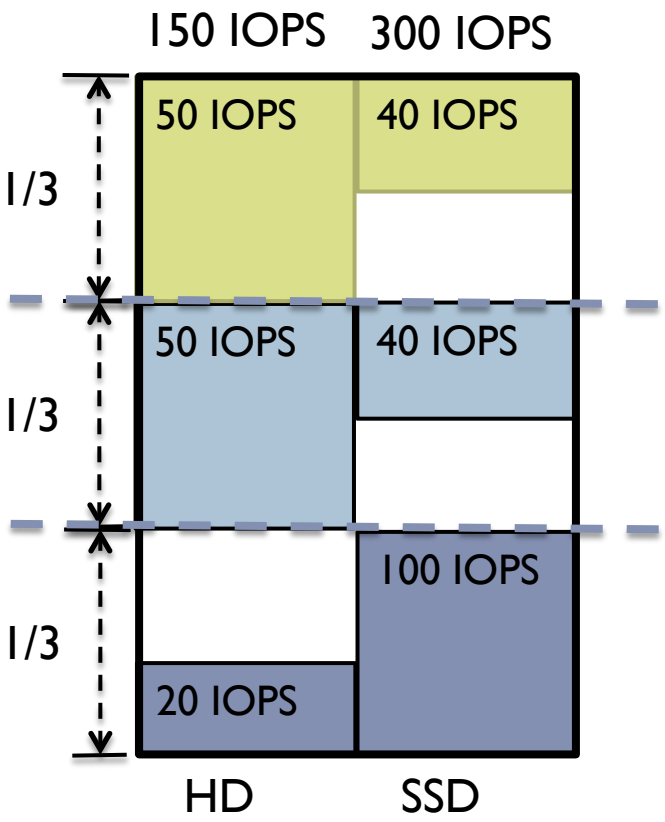
- Two devices (150 IOPS and 300 IOPS)
  - Client 1: h1 = 4/9
  - Client 2: h2 = 4/9
  - Client 3: h3 = 5/6



Fair Shares

- Client 1: h1 = 4/9
- Client 2: h2 = 4/9
- Client 3: h3 = 5/6

- Fair share (f_i):
  - Client 1: 90 IOPS
  - Client 2: 90 IOPS
  - Client 3: 120 IOPS
  - Depends only on client's hit ratio and capacities of the devices



RICE

Fairness Policy

- Allocate in the ratio of fair shares ?
  - Fair share reflects what a client would get if running alone
- Problem
  - Throttling across devices similar to DRF example
- Solution
  - Bottleneck-aware allocation

Bottleneck-Aware Allocation

- Bottleneck Sets
  - Define load-balancing point $h_{bal} = C_s / (C_s + C_d)$
  - If $h_i \leq h_{bal}$: in HD-bottleneck Set (*D*)
  - If $h_i > h_{bal}$: in SSD-bottleneck Set (*S*)

Fairness Requirements of BAA

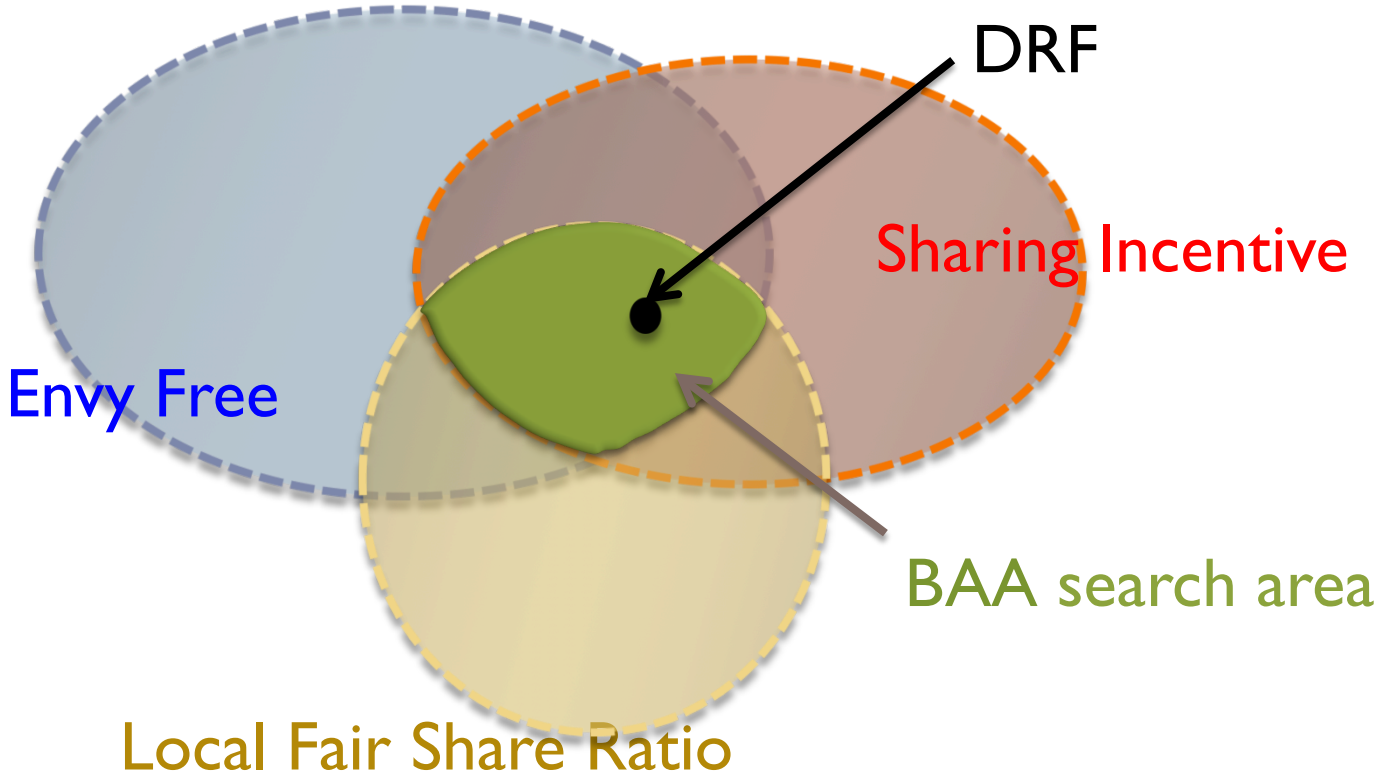
- Sharing Incentive (SI)
  - No client gets less IOPS than it would from equally partitioning each resource
- Envy-Freedom (EF)
  - Clients prefer their own allocation over the allocation of any other client
- Local Fair Share Ratio
  - Clients belong to the same bottleneck set get IOPS in proportion to their fair shares

Bottleneck-Aware Allocation

- Maximize system throughput
- Satisfy fairness requirements

Solution Space Satisfying All Properties

- BAA will match SI and EF of DRF
- Get better or same utilization than DRF



RICE

Fairness Constraints of BAA

- *Fairness between clients in D:*

$$\forall i, j \in D,\ \frac{A_i}{A_j} = \frac{f_i}{f_j}. \quad \text{Define } \rho_d = \frac{A_i}{f_i}$$

- *Fairness between clients in S:*

$$\forall i, j \in S,\ \frac{A_i}{A_j} = \frac{f_i}{f_j}. \quad \text{Define } \rho_s = \frac{A_j}{f_j}$$

- *Fairness between a client in D and a client in S:*

$$\forall i \in D, j \in S: \quad \frac{h_j}{h_i} \geq \frac{A_i}{A_j} \geq \frac{m_j}{m_i}.$$

  - constraints $O(n^2)$

RICE

Optimization for Allocation (2-variable LP)

Maximize $$\rho_d \sum_{i \in D} f_i + \rho_s \sum_{j \in S} f_j \qquad (1)$$

subject to:

$$\rho_d \sum_{i \in D} f_i m_i + \rho_s \sum_{j \in S} f_j m_j \leq C_d \qquad (2)$$

$$\rho_d \sum_{i \in D} f_i h_i + \rho_s \sum_{j \in S} f_j h_j \leq C_s \qquad (3)$$

$$\beta \geq \frac{\rho_d}{\rho_s} \geq \alpha \qquad (4)$$

$$\alpha = max_{i,j} \left\{ \frac{m_j f_j}{m_i f_i} \right\} \qquad \beta = min_{i,j} \left\{ \frac{h_j f_j}{h_i f_i} \right\}$$

RICE

Talk Outline

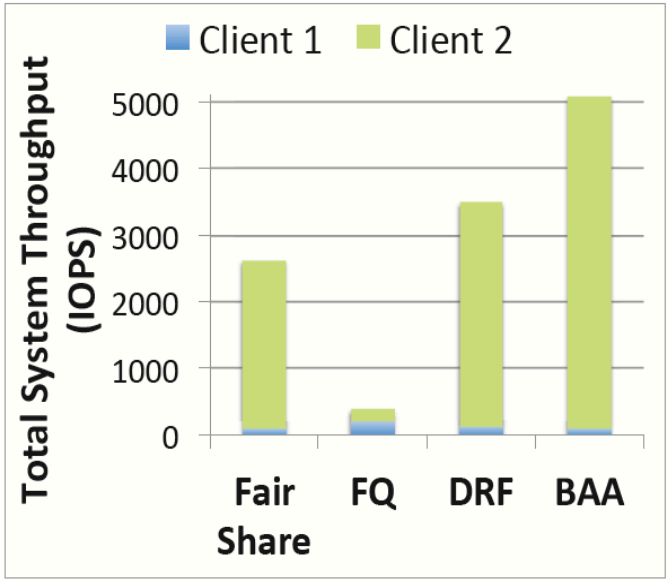
- Motivation
- Bottleneck-Aware Allocation (BAA)
- Evaluation
- Conclusions and future work

Evaluation

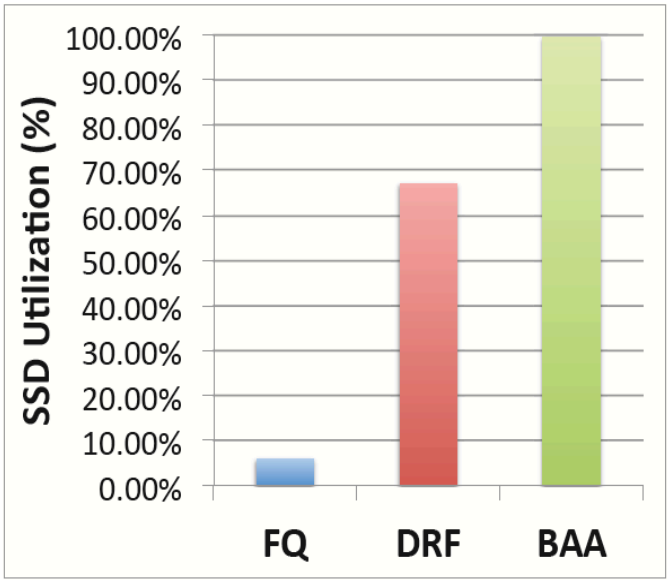
- Simulation
  - Evaluate BAA's efficiency
  - Evaluate BAA's dynamic behavior when workload changes

- Linux
  - Prototype by interposing BAA scheduler in the IO path
  - Evaluate BAA's efficiency, fairness (SI and EF)

Simulation (Efficiency - 2 clients)



(a) Throughputs



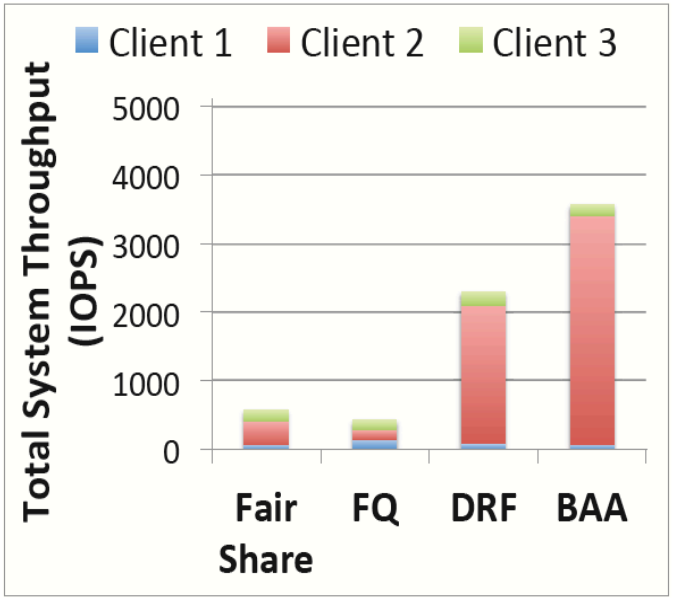
(b) Utilizations

- Two clients: h1 = 0.5; h2 = 0.95
- Two devices:
  - HD= 100 IOPS; SSD = 5000 IOPS

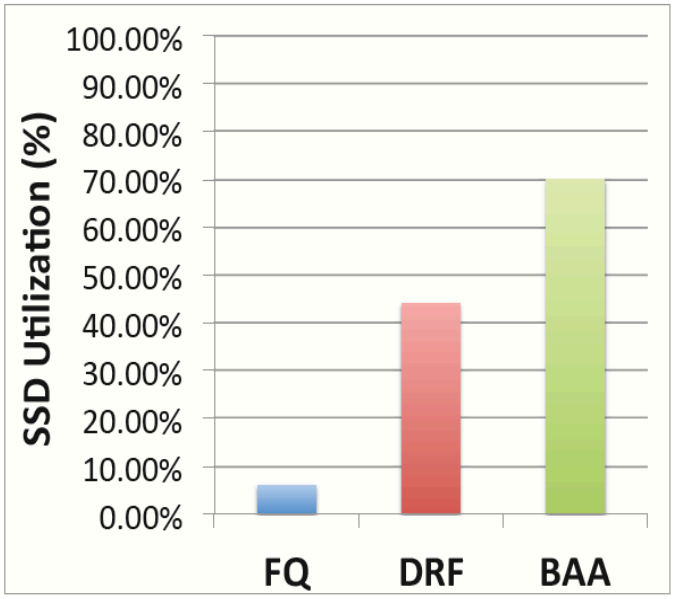
- SSD Utilization:
  - FQ: 7%
  - DRF: 65%
  - BAA: 100%

RICE

Simulation (Efficiency - 3 clients)



(a) Throughputs



(b) Utilizations

- A third client: h3 = 0.8
- SSD Utilization:
  - FQ: 6%
  - DRF: 45%
  - BAA: 71% (bounded by fairness)

RICE

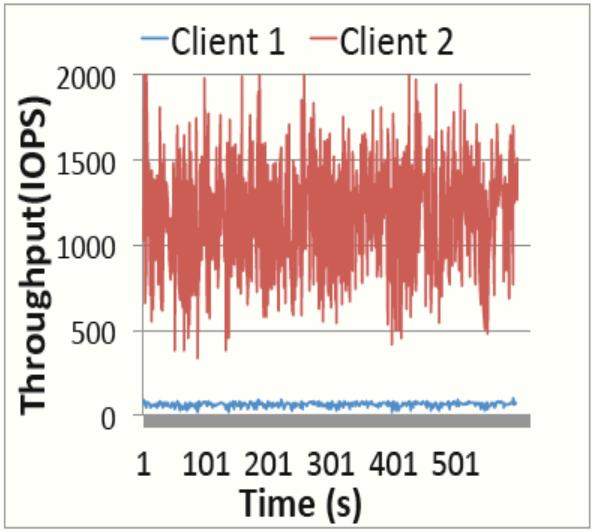
Simulation (Dynamic Behavior)



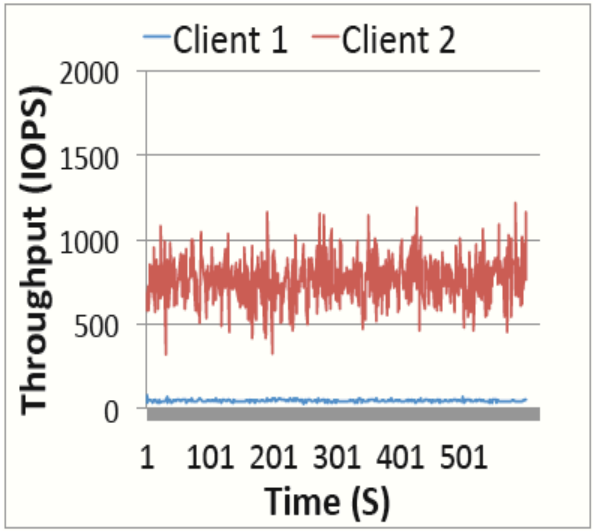
- Two clients
  - h1 = 0.45, 0.2 (after 510s)
  - h2 = 0.95
- Two devices:
  - HD= 200 IOPS
  - SSD = 3000 IOPS
- *The utilization is pulled back high after a short period*

RICE

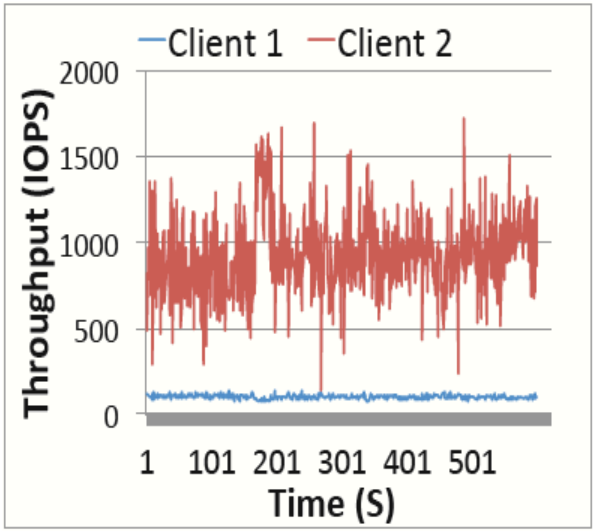
Linux (Efficiency-Throughput)



(a) BAA



(b) DRF

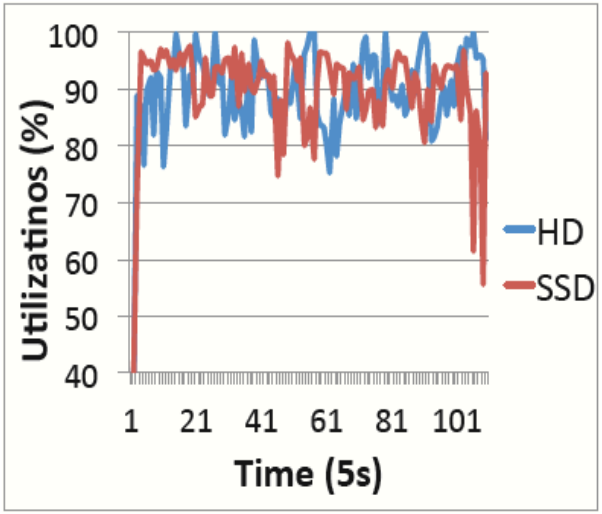


(c) CFQ

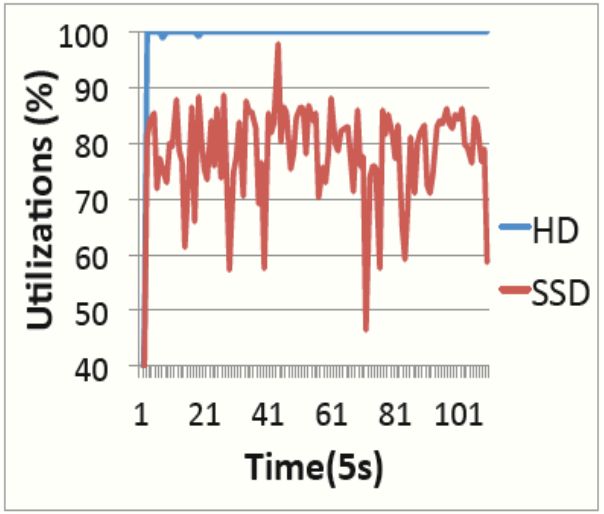
- Two clients:
  - Financial workload (h1= 0.3)
  - Exchange workload (h2 = 0.95)
- Total throughputs:
  - BAA: 1396 IOPS
  - DRF: 810 IOPS
  - CFQ: 1011 IOPS

RICE

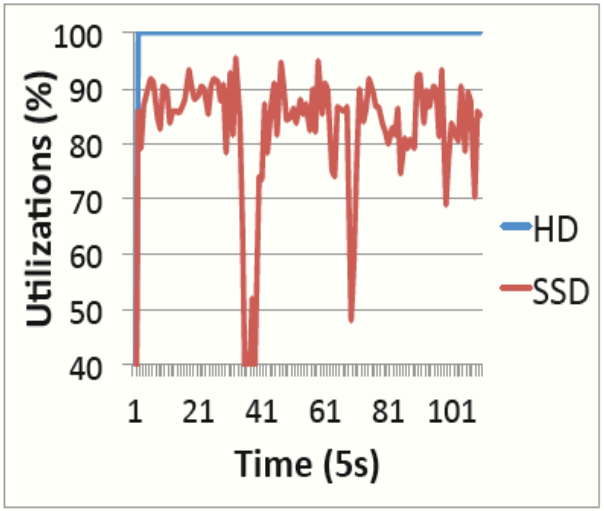
Linux (Efficiency-Utilization)



(a) BAA



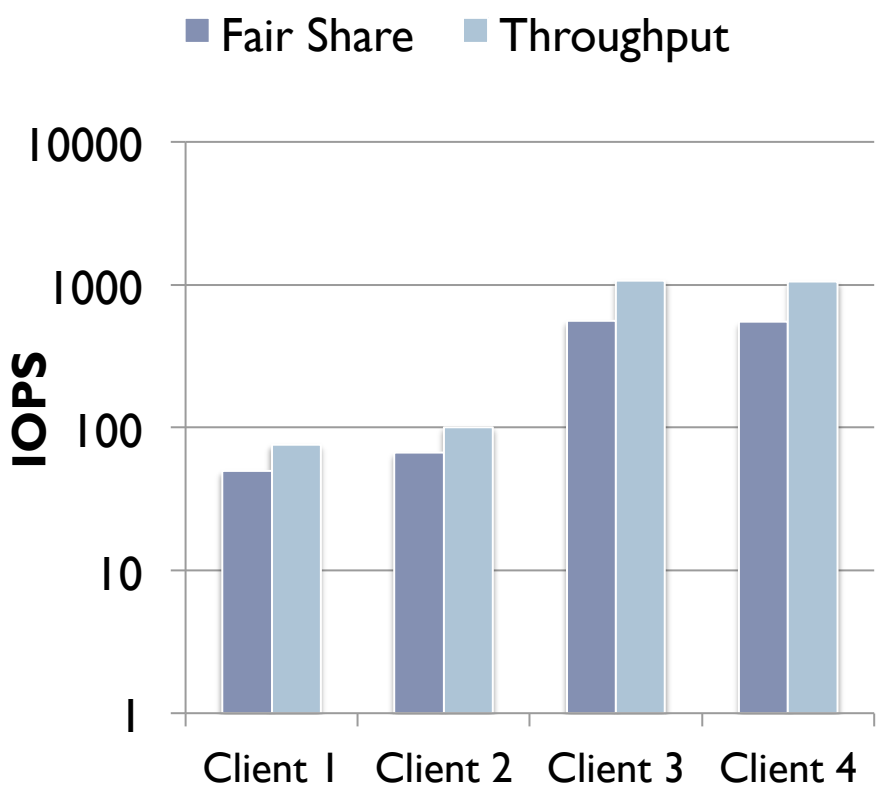
(b) DRF



(c) CFQ

- The average utilization:
- BAA (HD 94% and SSD 92%),
- DRF (HD 99% and SSD 78%), CFQ (HD 99.8% and SSD 83%)

Linux (Fairness – Sharing Incentive)

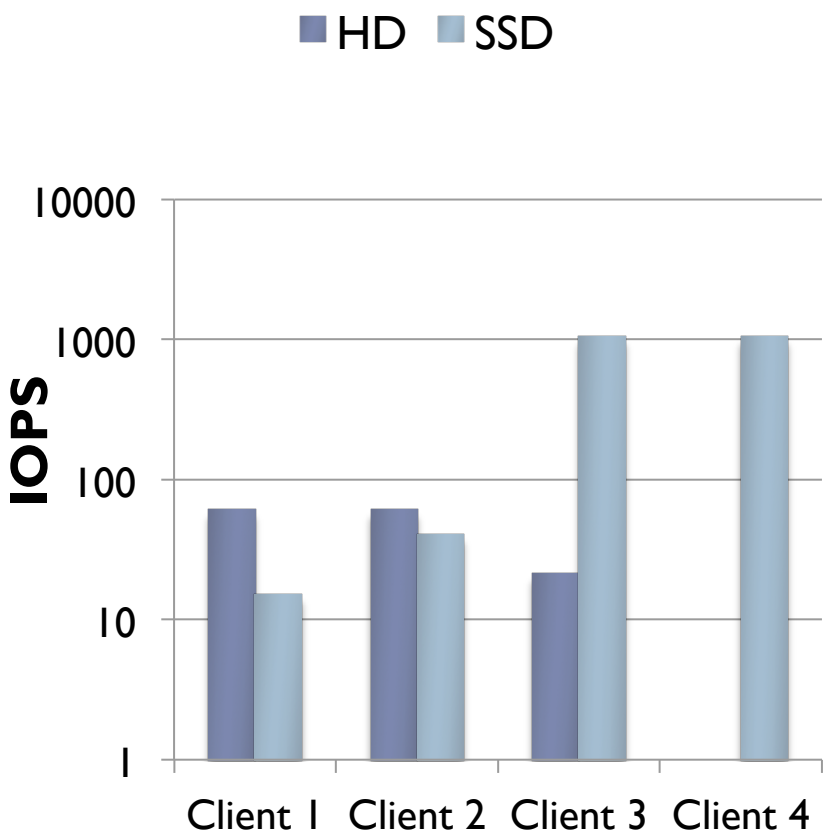


- Four financial clients
  - h1=0.2 (*D* Set)
  - h2=0.4 (*D* Set)
  - h3= 0.98 (*S* Set)
  - h4 =1.0 (*S* Set)

- *Every client receives at least its fair share.*
  - Proportional to fair share

RICE

Linux (Fairness – Envy freedom)



- *No one envies others' allocation*
  - No one get higher allocation on all devices
    - D set: Higher HD allocation
    - S set: Higher SSD allocation

RICE

Talk Outline

- Motivation
- Bottleneck-Aware Allocation (BAA)
- Evaluation
- Conclusions and future work

Conclusions and Future Work

- A new model (BAA) to balance *fairness* and *efficiency*
  - Fairness:
    - Sharing Incentive
    - Envy free
    - Local Fair Share
  - Efficiency:
    - Maximize utilization subject to fairness constraints

Ongoing Work

- Apply BAA for broader multi-resource allocation
  - CPU, Memory, Networks
- Other fairness policies
  - Cost, reservations
- Cache model
  - SSD as a cache of HD
  - Data migration

Thank you!