scc: Cluster Storage Provisioning Informed by Application Characteristics and SLAs

Harsha V. Madhyastha, John C. McCullough, George Porter, Rishi Kapoor, Stefan Savage, Alex C. Snoeren, and Amin Vahdat
UC Riverside and UC San Diego

Cluster Storage Provisioning

- Diverse requirements in cluster applications:
  - Sequential throughput for bulk data analysis
  - IOPS for key-value storage
  - Mix of the two in other scenarios
- Currently, requirements roughly characterized
- Configurations typically selected by rule of thumb, often homogeneous

- Homogeneous configurations result in hard-to-utilize resources and unnecessary increase in cost

Challenge

- Diverse storage options:
  - size: MB/s, IOPS, Cost
  - 7.2k-rpm: 500GB, 90/90, 125/125, $213
  - 15k-rpm: 146GB, 150/150, 285/285, $296
  - SSD: 32GB, 250/80, 2500/1000, $296
  - DRAM: 1GB, 13K/13K, 1.6B/1.6B, $35

- Best storage choice:
  - Coupled to workload
  - May change with growth
- Storage choice constrained to chassis capacity and bus capacity:

  ![Storage Options Diagram]

- Need CPU to drive storage and application
- Must understand relationship between application, workload, and storage

Overview of scc

- Storage Configuration Compiler
- Automates cluster configuration
- Combine:
  - Formal application specification
  - Hardware properties
  - Workload specification

Specifying Apps

- Application broken into tasks and datasets
- Tasks parametrized by cpu-time, write count and size for each dataset, and dependencies
- Datasets parametrized with overall size
- SLA is throughput of tasks per unit-time

Algorithm

- Model for photo sharing application

Heterogeneous beats scale-out

scc achieves 2 − 4.5 × cost average savings vs. scaling fixed server configurations

Storage regimes

- Parametrized by:
  - Capacity
  - Sequential throughput
  - Operation Gap (e.g., seeks, erasures)
- Gap captures more than IOPS
- Operation latency = \( \frac{\text{CPU}}{\text{IOPS}} + \text{gap} \)

Validation

Cheaper configurations miss the SLA, and simple scale-out solution requires much higher cost

Robustness

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Range with same architecture</th>
</tr>
</thead>
<tbody>
<tr>
<td>Photo size</td>
<td>30 KB ← 200 KB → 850 KB</td>
</tr>
<tr>
<td>Thumbnail size</td>
<td>1 KB ← 4 KB → 30 KB</td>
</tr>
<tr>
<td>SSD unit price</td>
<td>$200 ← $450 → $900</td>
</tr>
</tbody>
</table>

scc can estimate the robustness of its output