Performance Regulation for Competing I/O Workloads toward Shared Storage

by

Young Jin Nam
Department of Computer Science and Engineering
Pohang University of Science and Technology

(in cooperation with Spencer Ng, David Chambliss)
Talk Outline

- Introduction
- Static Performance Regulation
- Dynamic Performance Regulation
- Performance Evaluations
- Conclusions and Future Work
Performance Racing Problem

- Different applications demand different I/O characteristics (storage)
- Storage itself has no QoS feature

How to Regulate Competing I/O workloads for Shared Storage?
Nomenclature

- A workload set
  \[ WS_m = \{ W_1^m, W_2^m, \ldots, W_n^m \} \]

- Performance demand of \( W_k^m \)
  \{ \text{iops}_k^m, \text{size}_k^m, r_{t_k}^m \} 
  - iops_k^m: contract I/O rate 
  - size_k^m: observed request size 
  - rt_k^m: target response time 

- IOPS_m^T = \sum iops_k^m 
- RT_m^T = \min\{rt_1^m, rt_2^m, \ldots, rt_n^m\}
Introduction

- Framework for Performance Regulation

\[ WS_1 : W_1^1 = \{iops_1^1, size_1^1, rt_1^1\}, W_2^1 = \{iops_2^1, size_2^1, rt_2^1\} \]
\[ \ldots W_n^1 = \{iops_n^1, size_n^1, rt_n^1\} \]

**Research Objectives**

1. Devising a decision scheme which determines if a given workload set can be mapped to underlying storage?
2. Devising efficient throttling algorithms
Static Performance Regulation

Decision Maker for Mapping I/O Workloads

Generate I/O requests mixed by $W_k^m$ in $WS_m$ with a ratio of $iops_k^m / IOPS_m^T$

Obtain IOPS vs. RT chart

Find $RT_m^E$, the 95th percentile of all RTs associated with $IOPS_m^T \pm 5\%$

Compute a deliverable response time $RT_m^D = 120\%$ of $RT_m^E$

If $RT_m^D \leq RT_m^T$, then $WS_m$ can be mappable to underlying storage
Dynamic Performance Regulation

q TA1 - Credits with Periodic Replenishments

- Works like a leaky bucket
- Parameters
  - $\gamma$: replenishing frequency
  - $\rho_i$: credits to be replenished
  - $\sigma_i$: maximum credits to be accumulated ($= w^*\rho_i$)

- Drawbacks
  - why throttling I/O workloads if they behave well?
  - underutilizing storage resources
Dynamic Performance Regulation

- **TA2 - Turning on/off**
  Throttling using RT

  - Not throttling as long as current RT is good
  - Parameters
    - \( \gamma \), \( \rho \), \( \sigma \)
    - threshold values (HWM/LWM)
  - Drawbacks
    - why a well-behaving I/O workload should be throttled along with a mis-behaving I/O workload?
Dynamic Performance Regulation

- **TA3 - Turning on/off**
  Throttling using RT+IOPS

  - Not throttling as long as current RT or IOPS is good
  - Parameters
    - $\gamma$, $\rho$, $\sigma_i$
    - threshold values (HWM/LWM)
    - contract/agreed-on IOPS
Performance Evaluations

Simulation Environments(1)

- Simulator specification
  - porting Gateway Performance Management library(C) to gateway simulator(C++)
  - implementing the proposed algorithms – TA1, TA2 & TA3 – in simulator

- Generic throttling parameters
  - replenishing frequency ($\gamma$) = 1000 times per sec (1ms)
  - $\rho_k^m = \text{iops}_k^m / \gamma$
  - $\sigma_k^m = 1.5 \cdot \rho_k^m$
  - decision frequency = 100 times per sec (10ms)
  - HWM/LWM = 95/90%, 85/80%, 75/70% of $RT_m^E$
Performance Evaluations

- **Simulation Environments(2)**

  - Shared storage specification
    - RAID level 0 (striped), |SU| = 128 blk, no cache
    - # of disks = 8 (10,000 RPM)
    - host interface = 2 FC’s (2x100MBPS)
    - backend I/O interface = 4 SCSI’s (4*80MBPS)

  - Two I/O workload sets

<table>
<thead>
<tr>
<th>parameter set</th>
<th>WS_A</th>
<th>WS_B</th>
</tr>
</thead>
<tbody>
<tr>
<td>size_k^m</td>
<td>4KB</td>
<td>4KB</td>
</tr>
<tr>
<td>iops_k^m</td>
<td>3500</td>
<td>1500</td>
</tr>
<tr>
<td>rt_k^m</td>
<td>41ms</td>
<td>41ms</td>
</tr>
</tbody>
</table>

- Nearly open-loop : avg. think time = 1 sec
Performance Evaluations

Sketch of Performance Evaluations

- Perform static performance regulation with a given storage and $WS_m$
- Define figures of merit (FOM) to evaluate dynamic performance regulation schemes (throttling algorithms)
- Analysis of I/O throughput, RT, and throttling behavior in time domain
- Analysis of FOM of algorithms
- Analysis of effects of leaky bucket size
Mapping I/O Workloads to Shared Storage

<table>
<thead>
<tr>
<th>Parameter</th>
<th>IOPS&lt;sub&gt;m&lt;/sub&gt;</th>
<th>RT&lt;sub&gt;m&lt;/sub&gt;</th>
<th>RT&lt;sub&gt;E&lt;/sub&gt;</th>
<th>RT&lt;sub&gt;D&lt;/sub&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>WS&lt;sub&gt;A&lt;/sub&gt;</td>
<td>5000</td>
<td>41ms</td>
<td>33.42ms</td>
<td>40.10ms</td>
</tr>
<tr>
<td>WS&lt;sub&gt;B&lt;/sub&gt;</td>
<td>3500</td>
<td>40ms</td>
<td>32.72ms</td>
<td>39.26ms</td>
</tr>
</tbody>
</table>
Performance Evaluations

- **Performance Metrics – Figures of Merit**
  - $\text{FOM}^{\text{TP}}(\text{WS}_m)$
    - for a given observation period, an average IOPS of $\text{WS}_m$
  - $\text{FOM}^{\text{RT}}(W_k^m)$
    - for a given observation period, a percentage of I/O requests of $W_k^m$ which can meet target $r_t_k^m$ when its demanded IOPS is equal to or less than $\text{iops}_k^m$

- **Varying Demanding I/O Workload Levels**

![Graph showing varying demanding I/O workload levels with combinations and IOPS_A^T=5000]
Analysis of I/O T-put & RT with $WS_A(1)$
Performance Evaluations

- Analysis of I/O T-put & RT with $W_{SA}(2)$

Blue uses more TA2 (75/70%)
Red uses more
Blue’s RT gets better
Red’s RT gets little worse

Red’s RT gets better
Blue’s RT gets little worse

Throttling of both I/O workloads are turning on/off together!
Performance Evaluations

- Analysis of I/O T-put & RT with WS_A (3)

**TA2** (75/70%)

**TA3** (75/70%)

Well-behaving Red

Well-behaving Blue

OFF
Performance Evaluations

Analysis of Figures of Merit

I/O T-put Improvement by TA2/TA3

FOM\textsubscript{RT}(W_{S_A})

FOM\textsubscript{RT}(W_{1A})

FOM\textsubscript{RT}(W_{2A})

100% Target RT desirable

TA3 works better

TA3 works better

TA2/TA3 Target RT desirable
Analysis of Effects of Leaky Bucket Size

- Small bucket size disallows even small bursty I/O traffic to pass through!
- With increase of LB size, TA2 and TA3 become similar
- Now $\sigma_k^m = 200 \rho_k^m$ (was $\sigma_k^m = 1.5 \rho_k^m$)
Conclusions and Future Work

Summary

- A framework for performance regulation has been introduced
- We proposed a static performance regulation scheme:
  - decide if an I/O workload set can be mapped to underlying storage
- We devised and evaluated three dynamic performance regulation (throttling) algorithms:
  - $FOM_{TP}(WS_m)$ showed that TA2 and TA3 utilizes underlying storage resource more than TA1
  - $FOM_{RT}(W_{k,m})$ showed that TA3 works almost as good as TA1
  - TA3 adaptively turning on/off throttling based on current RT/IOPS works better than other algorithms TA1 and TA2
  - TA2 and TA3 worked similarly with large bucket size
Conclusions and Future Work

Where Can Performance Regulation be Fit Into?

- IDC Gateway (IBM ARC)
  - embedded in PM module
  - directly applicable (using GWPM lib used in PM)

- Large-scale RAID
  - regulating multiple LUNs sharing HBAs and physical drives
  - idea/algorithms can be applicable
Conclusions and Future Work

Future Work

- **Near term**
  - devising additional FOM presenting average RT
  - experiments with different access patterns (sequential) & arrival behaviors (burstness)
  - new throttling algorithm adjusting contract iops rather than turning off/on throttling

- **Long term**
  - configuring throttling parameters \((\gamma, \rho, \sigma, \text{threshold values})\) based on monitored performance data
  - allowing complex performance requirements

\[ W_1 = \{(\text{iops}_{1,1}, \text{size}_{1,1}, \text{rt}_{1,1}), \ldots, (\text{iops}_{1,p}, \text{size}_{1,p}, \text{rt}_{1,p})\} \]
Thank you for listening
Any questions & suggestions are welcome!
Backup Materials
Performance Evaluations

- Analysis of Figures of Merits

![Bar Chart: FOM\textsuperscript{RT}(WS\textsubscript{B})](chart.png)

- Combinations
- TA0
- TA1
- TA2(75/70)
- TA2(85/80)
- TA2(95/90)
- TA3(75/70)
- TA3(85/80)
- TA3(95/90)
Performance Evaluations

- Analysis of Figures of Merits

![Graphs showing FOM<sup>RT</sup>(W<sub>1</sub>,<sup>B</sup>) and FOM<sup>RT</sup>(W<sub>2</sub>,<sup>B</sup>) for different combinations.](image)