TASK-AWARE VIRTUAL MACHINE SCHEDULING FOR I/O PERFORMANCE

Hwanju Kim, Hyeontaek Lim, Jinkyu Jeong, Heeseung Jo  
(Korea Advanced Institute of Science and Technology)  
Joonwon Lee (Sungkyunkwan Univ.)  
VEE 2009 March 13
Virtual Machine Consolidation

- Centralized various computing environments
  - Virtual desktop infrastructure
    - VMware, Sun, HP, MS
  - Cloud computing
    - Amazon EC2
Virtual Machine Consolidation

- Centralized various computing environments
  - Virtual desktop infrastructure
    - VMware, Sun, HP, MS
  - Cloud computing
    - Amazon EC2

Unpredictable workloads due to the diversity
Virtual Machine Consolidation

- Performance enhancement
  - Paravirtualization
  - Hardware-assisted techniques
    - Intel VT, AMD SVM
  - Optimization
Virtual Machine Consolidation

- Performance enhancement
  - Paravirtualization
  - Hardware-assisted techniques
    - Intel VT, AMD SVM
  - Optimization

High degree of consolidation
Virtual Machine Consolidation

- Performance enhancement
  - Paravirtualization
  - Hardware-assisted techniques
    - Intel VT, AMD SVM
  - Optimization

Unpredictable workloads

High degree of consolidation

Intelligent CPU management can improve the performance
Background

- **A semantic gap** between the VMM and a guest OS
  - VMM’s lack of knowledge of VM internal
  - No tracking characteristics of guest-level tasks
  - Internal workload-agnostic scheduling
  - Poor decision about “**when**” to schedule a VM
- Simple design of the VMM

---

OS awareness

Low overheads
- Low TCB

Efficient resource management
Task-unawareness leading to poor responsiveness

Run queue sorted based on CPU fairness
Task-unawareness leading to poor responsiveness

- Run queue sorted based on CPU fairness

- VM (VCPU)
- VM (VCPU)
- VM (VCPU)

- I/O-bound task
- CPU-bound task
- Mixed task

- VMM
- I/O event
Background

Task-unawareness leading to poor responsiveness

Run queue sorted based on CPU fairness

That event is mine!
Background

Task-unawareness leading to poor responsiveness

Run queue sorted based on CPU fairness

That event is mine!

I have no idea this incoming event is yours and your VM has low priority now. Sorry not to schedule you..

Responsiveness VS. Fairness
Background

Task-unawareness leading to poor responsiveness

Run queue sorted based on CPU fairness

I have no idea this incoming event is yours and your VM has low priority now. Sorry not to schedule you.

That event is mine!

Responsiveness VS. Fairness
Background

The worst case example for 6 domains consolidated

Workloads
- 1 dom: Server & CPU-bound task
- 5doms: CPU-bound task
Background

The worst case example for 6 domains consolidated

Workloads
1 dom: Server & CPU-bound task
5 doms: CPU-bound task
Background

The worst case example for 6 domains consolidated

By boosting mechanism of Xen Credit scheduler

Boosting mechanism realizes I/O boundness with only VCPU-level

Workloads
1 dom: Server & CPU-bound task
5doms: CPU-bound task
Main Goals

- Improve responsiveness of an I/O-bound task
  - Priority boosting with task-level granularity
    - “Partial boosting”
- CPU fairness guarantee
- Transparency
- Low management overheads
Issues

- How to identify an I/O-bound task
- How to know an incoming event is for the I/O-bound task
Approach

- Non-intrusive approach
  - No guest OS modification
    - No explicit interface to inform I/O-bound task and event data
  - Pros.
    - No additional engineering cost for different OSes
    - Strong trustworthiness
  - Cons.
    - False decision
Tracking I/O-bound Tasks

- Observable information at the VMM
  - Task switching
  - Monitoring address space changes (Antfarm USENIX’06)
- CPU time usage
- Running time of a task

**Example (x86)**
Tracking I/O-bound Tasks

- Inference based on common \textit{gray-box} knowledge
- Kernel policy to improve responsiveness of I/O-bound tasks
  - An I/O-bound task is preemptively scheduled in response to its incoming event
- Characteristic of I/O-bound tasks
  - Short running time
  - Threshold to decide a short running time: \textit{I0threshold}
Tracking I/O-bound Tasks

- Three disjoint observation classes based on two gray-box criteria
  - **Positive evidence** supports I/O-boundness
  - **Negative evidence** supports non-I/O-boundness
  - **Ambiguity** No evidence

Preemptively scheduling in response to an event &
Short running time ($< IO\text{threshold}$)
Tracking I/O-bound Tasks

Example

Event pending to VCPU₁

Time

<table>
<thead>
<tr>
<th>Positive</th>
<th>Negative</th>
<th>Ambiguity</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

$IO_{threshold}$
Tracking I/O-bound Tasks

Example

Event pending to VCPU₁

Scheduling VCPU₁

$T₁$

$IO_{threshold}$

<table>
<thead>
<tr>
<th>Positive</th>
<th>Negative</th>
<th>Ambiguity</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Tracking I/O-bound Tasks

Example

Event pending to VCPU1

Scheduling VCPU1

T1

T2

Time

IOthreshold

<table>
<thead>
<tr>
<th>Positive</th>
<th>Negative</th>
<th>Ambiguity</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Tracking I/O-bound Tasks

Example

<table>
<thead>
<tr>
<th>Positive</th>
<th>Negative</th>
<th>Ambiguity</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>$T_1$</td>
</tr>
</tbody>
</table>
Tracking I/O-bound Tasks

Example

Event pending to VCPU₁

Scheduling VCPU₁

T₁ T₂ T₃

IOthreshold

<table>
<thead>
<tr>
<th>Positive</th>
<th>Negative</th>
<th>Ambiguity</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>T₁</td>
</tr>
</tbody>
</table>
Tracking I/O-bound Tasks

Example

Event pending to VCPU1

Scheduling VCPU1

Positive | Negative | Ambiguity
---|---|---
T2 | | T1
Tracking I/O-bound Tasks

Example

Event pending to VCPU₁

Scheduling VCPU₁

Time

<table>
<thead>
<tr>
<th>Positive</th>
<th>Negative</th>
<th>Ambiguity</th>
</tr>
</thead>
<tbody>
<tr>
<td>T₂</td>
<td></td>
<td>T₁</td>
</tr>
</tbody>
</table>
Tracking I/O-bound Tasks

Example

Event pending to VCPU1

Scheduling VCPU1

Time

Positive | Negative | Ambiguity
---|---|---
T2 | T3 | T1
Tracking I/O-bound Tasks

**Example**

Event pending to VCPU1

Scheduling VCPU1

<table>
<thead>
<tr>
<th>Positive</th>
<th>Negative</th>
<th>Ambiguity</th>
</tr>
</thead>
<tbody>
<tr>
<td>T2</td>
<td>T3</td>
<td>T1</td>
</tr>
</tbody>
</table>

IOthreshold
Tracking I/O-bound Tasks

Example

Event pending to VCPU1

Scheduling VCPU1

$IO_{threshold}$

<table>
<thead>
<tr>
<th>Positive</th>
<th>Negative</th>
<th>Ambiguity</th>
</tr>
</thead>
<tbody>
<tr>
<td>T2</td>
<td>T3</td>
<td>T4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>T1</td>
</tr>
</tbody>
</table>
Tracking I/O-bound Tasks

Example

Event pending to VCPU1

Scheduling VCPU1

<table>
<thead>
<tr>
<th>Positive</th>
<th>Negative</th>
<th>Ambiguity</th>
</tr>
</thead>
<tbody>
<tr>
<td>T2</td>
<td>T3</td>
<td>T4</td>
</tr>
<tr>
<td>T1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Tracking I/O-bound Tasks

Example

<table>
<thead>
<tr>
<th>Positive</th>
<th>Negative</th>
<th>Ambiguity</th>
</tr>
</thead>
<tbody>
<tr>
<td>T2</td>
<td>T3</td>
<td>T4</td>
</tr>
<tr>
<td>T1</td>
<td>T5</td>
<td></td>
</tr>
</tbody>
</table>
Tracking I/O-bound Tasks

**Example**

Event pending to VCPU₁

Scheduling VCPU₁

Descheduling VCPU₁

<table>
<thead>
<tr>
<th>Positive</th>
<th>Negative</th>
<th>Ambiguity</th>
</tr>
</thead>
<tbody>
<tr>
<td>T₂</td>
<td>T₃</td>
<td>T₄</td>
</tr>
<tr>
<td>T₁</td>
<td>T₅</td>
<td></td>
</tr>
</tbody>
</table>
Tracking I/O-bound Tasks

Example

Event pending to VCPU1

Scheduling VCPU1

Descheduling VCPU1

IOthreshold

<table>
<thead>
<tr>
<th>Positive</th>
<th>Negative</th>
<th>Ambiguity</th>
</tr>
</thead>
<tbody>
<tr>
<td>T2</td>
<td>T3</td>
<td></td>
</tr>
<tr>
<td>T4</td>
<td>T6</td>
<td></td>
</tr>
<tr>
<td>T1</td>
<td>T5</td>
<td></td>
</tr>
</tbody>
</table>
Tracking I/O-bound Tasks

- Weighted evidence accumulation
  - The degree of belief to reinforce the inference
  - Weight of positive evidence < Weight of negative evidence
  - More penalize for negative evidence

At this time, this task is believed as an I/O-bound task
HOW TO KNOW AN INCOMING EVENT IS FOR AN I/O-BOUND TASK
Correlation Mechanism

To distinguish an incoming event for I/O-bound task

- **Block I/O**
  - Block read

- **Network I/O**
  - Packet reception
Correlation Mechanism
- Block I/O -

- Request/response style

If Tᵢ requests for reading Bᵢ and Tᵢ is I/O-bound
Completion event for Bᵢ is for I/O-bound task

- How to decide “Tᵢ read Bᵢ” at the VMM
Correlation Mechanism
- Block I/O -

Request/response style

If Tᵢ requests for reading Bᵢ and Tᵢ is I/O-bound
Completion event for Bᵢ is for I/O-bound task

How to decide “Tᵢ read Bᵢ” at the VMM

When the VMM observes a read event, it checks whether the current task is I/O-bound
Correlation Mechanism
- Block I/O -

- Request/response style

**If** T₁ requests for reading B₁ and T₁ is I/O-bound
Completion event for B₁ is for I/O-bound task

- How to decide “T₁ read B₁” at the VMM

  - When the VMM observes a read event, it checks whether the current task is I/O-bound

- But, how about “**delayed read event**”?
  - Guest OS dependent (e.g. block I/O scheduler)
Correlation Mechanism
- Block I/O -

**Inspection window**

If an I/O-bound task **in** inspection window
The actual read request is for I/O-bound task

**False positive VS. False negative**
Correlation Mechanism
- Network I/O -

- Event identification
  - Socket-like information is too heavy for the VMM
  - **Destination port number** for TCP/IP communication
    - Most specific to a recipient task
- Asynchronous packet reception
  - No prior information about incoming packets
  - **History-based prediction mechanism**
Correlation Mechanism

- Network I/O -

- History-based prediction mechanism

- Inference

  "If an incoming packet is for I/O-bound task, this packet makes the I/O-bound task to be preemptively scheduled"

- Monitoring the first woken task in response to an incoming packet
Correlation Mechanism
- Network I/O -

- History-based prediction mechanism (cont’)
  - **Portmap**
    - An entry for each destination port number
    - Each entry is an N-bit saturating counter

**Example (2-bit counter)**

- If the first woken task is I/O-bound
- Otherwise
Correlation Mechanism
- Network I/O -

- History-based prediction mechanism (cont’)

- **Portmap**
  - An entry for each destination port number
  - Each entry is an N-bit saturating counter

**Example (2-bit counter)**

- If the first woken task is I/O-bound
- Otherwise

If portmap counter’s MSB is set, this packet is for I/O-bound
PARTIAL BOOSTING
Partial Boosting

- Priority boosting with task-level granularity
  - Priority boosting lasts during the run of an I/O-bound task

Why?

- To prevent CPU-bound tasks in a boosted VCPU from compromising CPU fairness
Partial Boosting

Procedure

Run queue sorted based on CPU fairness

VM1 (VCPU)  VM2 (VCPU)  VM3

VMM

CPU-bound task  CPU-bound task

Head  Tail
Partial Boosting

Procedure

Run queue sorted based on CPU fairness

If this event is inferred for an I/O-bound task in VM3, do partial boosting for VM3
Partial Boosting

Procedure

- Run queue sorted based on CPU fairness

VM3
- CPU-bound task
- CPU-bound task

VM1 (VCPU)

VM2 (VCPU)

VMM

I/O event

If this event is inferred for an I/O-bound task in VM3, do partial boosting for VM3
Partial Boosting

Procedure

Run queue sorted based on CPU fairness

If this event is inferred for an I/O-bound task in VM3, do partial boosting for VM3
Partial Boosting

Procedure

Run queue sorted based on CPU fairness

Head

VM3

CPU-bound task

CPU-bound task

VM1 (VCPU)

VM2 (VCPU)

VMM

Tail

If this event is inferred for an I/O-bound task in VM3, do partial boosting for VM3
Partial Boosting

**Procedure**

Run queue sorted based on CPU fairness

- VM1 (VCPU)
- VM2 (VCPU)
- VM3
  - CPU-bound task
  - CPU-bound task

VMM

If this event is inferred for an I/O-bound task in VM3, do partial boosting for VM3
IMPLEMENTATION & EVALUATION
Implementation

- Based on Credit scheduler in Xen 3.2.1
- Task information maintained by hash
  - Limited number of tasks maintained
  - Remove of a task with infrequent I/O
- Correlation
  - Block I/O : using grant table in Xen
  - Network I/O : supported by network backend driver
- No consideration of multiple VCPUs
Evaluation

Interactive workload

Packet request-response

Worst case scenario
1 dom: Server & CPU-bound task
5doms: CPU-bound task

Think time: 100 ~ 1000 ms
IO threshold = 0.5 ms
Evaluation

Interactive workload
Packet request-response

Worst case scenario
1 dom: Server & CPU-bound task
5doms: CPU-bound task

Think time: 100 ~ 1000 ms
IOthreshold = 0.5 ms
Correlation evaluation

Partial boosting hit ratio (PBHR)

\[
PBHHR \ (\%) = \frac{\sum h}{\text{The number of partial boostings}} \times 100
\]

where

\[
h = \begin{cases} 
1, & \text{if an I/O-bound task awakes during partial boosting.} \\
0, & \text{otherwise.}
\end{cases}
\]

defined as true positive ratio

False positive ratio = (100 - PBHR) \%
Evaluation

Correlation evaluation: Block I/O

PBHR (%)

Workloads
1 dom: 8 tasks
1 task: I/O-bound task
7 tasks: I/O+CPU task
(CPU usage 1-300ms between IOs)
5 doms: CPU-bound task

Inspection window size
Evaluation

Correlation evaluation: Block I/O

Throughput

Workloads
1 dom: 8 tasks
1 task: I/O-bound task
7 tasks: I/O+CPU task
(CPU usage 1-300ms between IOs)
5doms: CPU-bound task
Evaluation

Correlation evaluation: Network I/O

Workloads
1 dom: 8 tasks
1 task: I/O-bound task
7 tasks: I/O+CPU task
(CPU usage 1-300ms between IOs)
5 doms: CPU-bound task

2-bit is reasonable in terms of space overheads

<table>
<thead>
<tr>
<th>Bit-width of portmap counter</th>
<th>1</th>
<th>2</th>
<th>4</th>
<th>NC</th>
</tr>
</thead>
<tbody>
<tr>
<td>PBHR (%)</td>
<td>64</td>
<td>90</td>
<td>93</td>
<td>10</td>
</tr>
</tbody>
</table>
Evaluation

- Response time for text editing during CPU-intensive workload

![Graph 1: Text editing (Xen compilation)]

![Graph 2: Text editing (Web browsing with Flash animations)]
Evaluation

Execution time of I/O-bound tasks with CPU-intensive workloads

<table>
<thead>
<tr>
<th>Background workload</th>
<th>Compilation</th>
<th>Compression</th>
</tr>
</thead>
<tbody>
<tr>
<td>grep</td>
<td>Normal</td>
<td>TAVS</td>
</tr>
<tr>
<td>find</td>
<td></td>
<td></td>
</tr>
<tr>
<td>wget</td>
<td></td>
<td></td>
</tr>
<tr>
<td>smbcp: copy files from remote samba server</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Evaluation

Overheads

- Task tracking overheads: 0.06%
- No overhead for inspecting incoming packets
- Increased network throughput : decreased CPU throughput = 48 : 1
- Space overhead of N-bit portmap
  - N * 8KB for each VM
  - e.g. 2-bit portmaps for TCP and UDP: 32KB for each VM
Conclusions

- Task-aware VM scheduling
- Bridging the semantic gap in CPU management
- Transparency by VMM-level inference
- Gray-box technique
- Low overheads
Future Work

- Extension on multicore system
- Simulation-based analysis for more intelligent scheduling
- Evaluation for more various workloads
THANK YOU!
BACKUP SLIDES
Implementation

* Block correlation

![Diagram showing implementation details with VMM, IDD, Dom1, Dom2, and corresponding tables and drivers.]
Implementation

Network correlation

![Diagram showing network correlation]
Throughput

\[ \text{PBratio} = \frac{\text{Allowed CPU usage for partial boosting}}{\text{Total CPU usage}} \]
Degree of Belief

Degree of belief (grep, find + compilation)

Elapsed time (sec)

The degree of belief