A Resource Management Model for VM-based Virtual Workspaces

Master's Presentation

January 3, 2007

Borja Sotomayor

Advisor: Prof.Ian Foster (ANL/UC)
Committee: Dr.Kate Keahey (ANL/UC), Prof.Anne Rogers (UC)

Introduction

- Context of this work: Virtual Workspaces
- Virtual Machines (VMs) are a promising vehicle for realizing virtual workspaces.
- However, resource management models focused on the job abstraction do not schedule VM-based virtual workspaces efficiently.
  - They do not take into account the overhead of using VMs.
  - They do not leverage the features of virtualization.
- In this work we present a resource management model that allows us to accurately and efficiently provision and instantiate the virtual resources required by virtual machines deployed in a Grid.

Index

- Background: Virtual Workspaces
- Problem
- Modeling Virtual Resources
- Design and Implementation
- Experiments
- Related Work
- Conclusions

Index

- Background: Virtual Workspaces
- Problem
- Modeling Virtual Resources
- Design and Implementation
- Experiments
- Related Work
- Conclusions

What is a workspace?

- A virtual workspace is an execution environment that can be deployed dynamically and securely on the Grid.
- VM technology is a promising vehicle to achieve higher quality workspaces.
  - Quality of Life: Users get exactly the software environment they need.
  - Quality of Service: VMs allow for fine-grained resource allocation, with enforceable isolation.
- Workspaces can be encapsulated in VM images, and dynamically deployed on VMM-enabled sites.

GT4 Workspace Service

- The GT4 Virtual Workspace Service (VWS) is a VM-based workspace implementation.
  - http://workspace.globus.org/
- The VWS manages a pool of physical nodes where VMs can be deployed. An *image node* stores the VM images.
- Users request deployment of virtual workspaces through a remote interface.
- A VM image is not provided in the request. The user specifies its location in the image node, or provides an URI to the image (on a third-party site)

Use cases

- Our work is motivated by several use cases that stand to benefit from virtual workspaces:
  - Virtual labs
  - Event-driven applications
  - Batch jobs with strict software requirements
- These use cases present resource management scenarios such as best-effort scheduling, advance reservations, or a mix of both.

Index

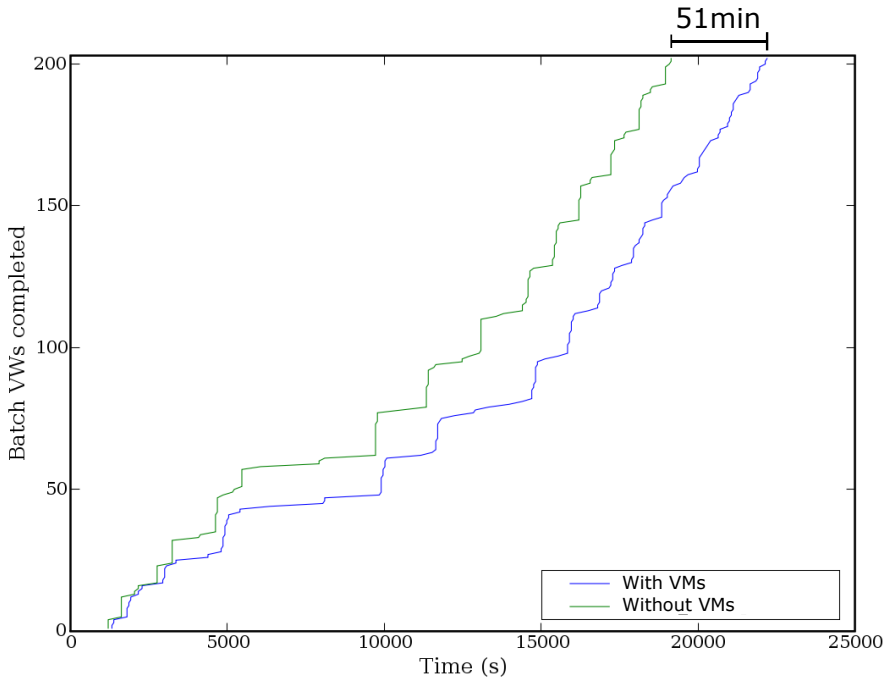
Problem

- VM-based workspaces have appealing features, but they involve a cost:
  - Potentially large VM images have to be deployed before we can start a virtual workspace.
  - Running inside a VM is slower than running directly on physical hardware.
- However, VMs also provide us with resource management mechanisms:
  - Suspend/Resume
  - Live Migration

To VM or not to VM?

- What happens if we try to schedule virtual workspaces in the same way we schedule jobs?
  - Preview of our results (we will explain them in more detail in "Experiments")
  - Simulation of running a 4h submission trace with many serial batch requests (75% of requested time), and some resource-hungry advance reservations (25% of requested time), assuming jobs in a VM take 10% longer to run.
  - We compare using VMs vs. not using VMs.
  - Both configurations use backfilling before an advance reservation.
  - Graph shows rate at which batch requests are completed.

To VM or not to VM?



To VM or not to VM?

Batch VW Duration	Resources requested by AR (%)	Proportion (Batch%-AR%)	Effect on Performance
long	000-025	25-75	**-8.88%**
long	000-025	50-50	**-5.10%**
long	000-025	75-25	**-13.28%**
long	025-050	25-75	**-2.97%**
long	025-050	50-50	**-8.45%**
long	025-050	75-25	**-7.80%**
long	050-075	25-75	**-3.94%**
long	050-075	50-50	**-10.29%**
long	050-075	75-25	**-7.40%**
long	075-100	25-75	**-4.76%**
long	075-100	50-50	**-16.22%**
long	075-100	75-25	**-15.99%**

Batch VW Duration	Resources requested by AR (%)	Proportion (Batch%-AR%)	Effect on Performance
medium	000-025	25-75	**-10.99%**
medium	000-025	50-50	**-9.68%**
medium	000-025	75-25	**-10.39%**
medium	025-050	25-75	**-3.88%**
medium	025-050	50-50	**-7.78%**
medium	025-050	75-25	**-8.74%**
medium	050-075	25-75	**-1.17%**
medium	050-075	50-50	**-11.28%**
medium	050-075	75-25	**-10.69%**
medium	075-100	25-75	**-6.50%**
medium	075-100	50-50	**-20.57%**
medium	075-100	75-25	**-12.42%**

Batch VW Duration	Resources requested by AR (%)	Proportion (Batch%-AR%)	Effect on Performance
short	000-025	25-75	**-5.56%**
short	000-025	50-50	**-18.57%**
short	000-025	75-25	**-20.00%**
short	025-050	25-75	**-8.92%**
short	025-050	50-50	**-11.27%**
short	025-050	75-25	**-19.93%**
short	050-075	25-75	**-0.78%**
short	050-075	50-50	**-21.07%**
short	050-075	75-25	**-23.68%**
short	075-100	25-75	**-10.84%**
short	075-100	50-50	**-26.28%**
short	075-100	75-25	**-31.58%**

VMs result in worse performance across the board!

To VM or not to VM?

- Performance in VM case is worse because:
  - We need to deploy VM images to the physical nodes before a virtual workspace can start.
  - The runtime overhead of using VMs.
- To make virtual workspaces cost-effective, we need to improve performance in the VM case.
- We need a resource management model that:
  - Manages the overhead of using VMs (with the goal of minimizing its effect on performance)
  - Leverages features of virtualization which could potentially increase utilization of physical resources.

Following slides

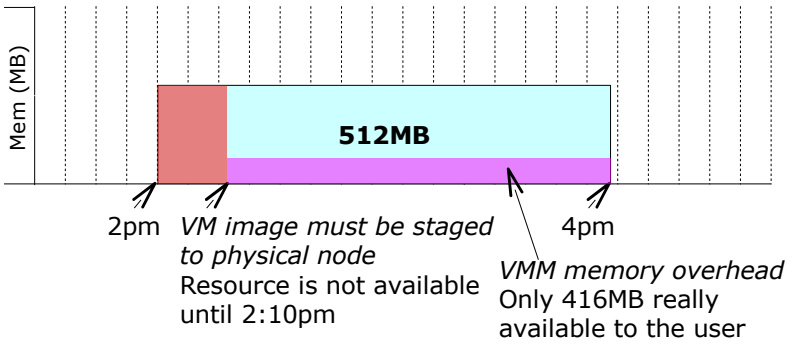
- In the following slides we will explain:
  - Our model for virtual resource management
  - The techniques we use, within this model, to improve performance of the VM case.
  - Present experimental results showing the effect our model and techniques have on performance.

Index

- Background: Virtual Workspaces
- Problem
- Modeling Virtual Resources
- Design and Implementation
- Experiments
- Related Work
- Conclusions

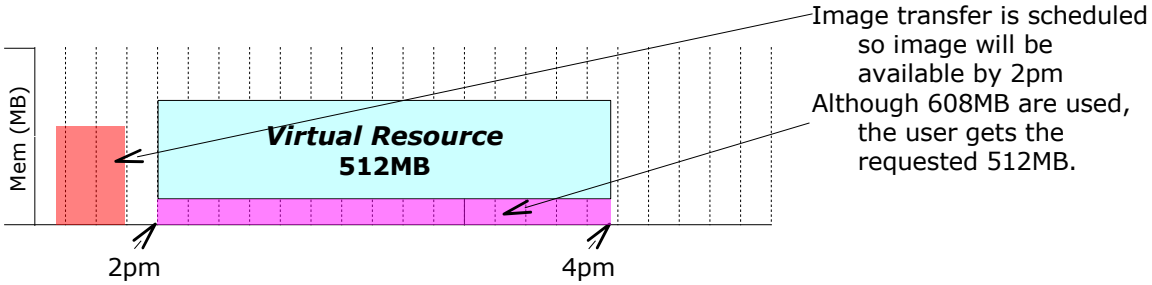
"Job Style" Workspace Management

- Existing schedulers will treat workspaces like a job, mapping a user's requested resource allocation directly to the physical hardware.
- This is not adequate for scheduling VMs.
- e.g. A user requires a VM with 512MB of memory from 2PM to 4PM (for this example, let's consider memory and time as the only resources)
- Various parts of this allocation will be "invaded" by overhead associated with preparing and managing a VM.



Modeling Virtual Resources

- We argue in favor of a *virtual resource management model*, where users get *exactly* the resources they requested.
- Overhead is managed separately from the user's resource allocation.
  - More work for the scheduler (must schedule the virtual resource *and* the overhead), but clients get exactly the resources they requested.
  - Two types of overhead: Preparation overhead (staging images, setting up a software environment, etc.) and runtime overhead (resulting from managing the VMs themselves)



"Job Style" vs. Virtual Resources

- "Job Style" affects accuracy of VM deployments.
  - VM does not start at agreed time, even in the presence of AR-capable schedulers.
  - VM does not have the agreed resource allocation.
- It burdens the user with having to factor in overhead into resource requests. This is not easy:
  - Users cannot accurately predict the time to stage the VM image (they are unaware of the network traffic conditions on the site).
  - Users have no control over how much of the VMM's overhead will be deducted from their allocation (e.g. if several VM's are deployed on a single node, the deduction might be shared)
- Virtual resources put a greater burden on the scheduler, but they enable the accurate and efficient creation of VM-based virtual workspaces.

Virtual Resource Dimensions

- Managing virtual resources involves challenges along several dimensions:
  - *Time*: Assuring that resources are available at the agreed time (and rejecting requests that are deemed infeasible because it would not be possible to set up the required environment on time)
  - *Memory*: Part of the available physical memory is assigned to the VMM. This dimension is trivial, since VMM memory usage is generally constant.
  - *Networking*: Network bandwidth is shared by all the VMs on a node *and* with preparation overhead (such as image staging). Furthermore, network usage can affect CPU usage in the VMM.
  - *CPU*: The CPU share required by the VMM can vary over time, depending on the resource usage of the VMs it is managing.
- We currently focus on the resource dimension of *Time*, and assume that VMs produce no network activity (i.e. the Networking dimension does not affect the Time dimension)

Index

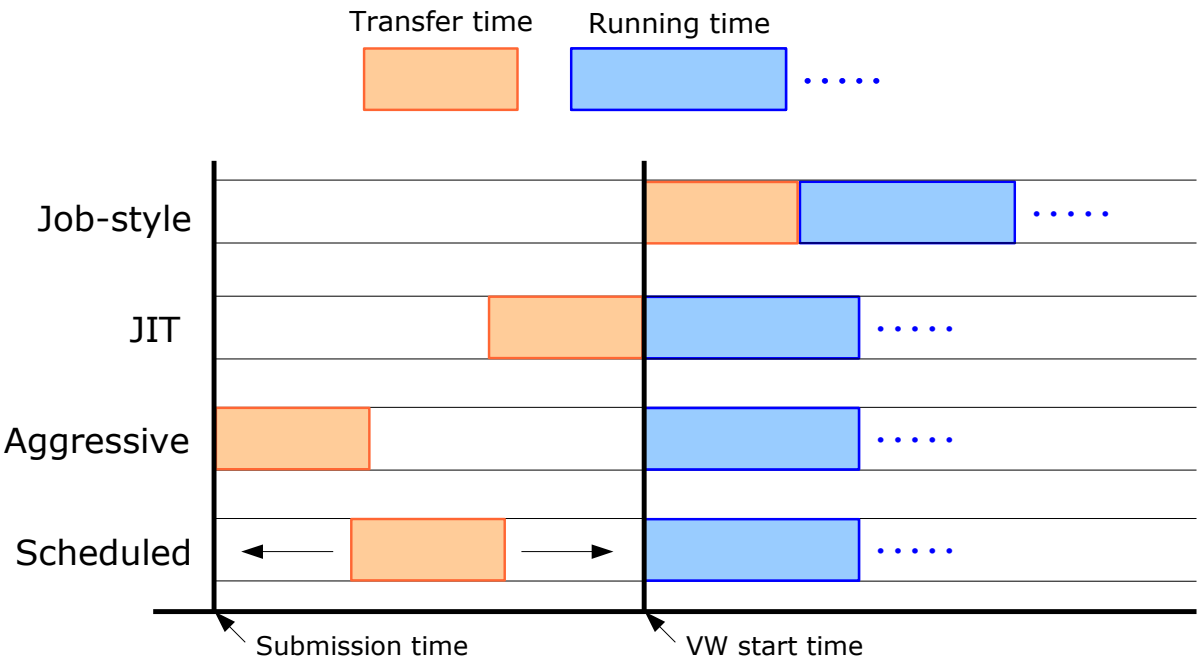
Accuracy and Efficiency

- To improve performance of VM-based virtual workspaces, we need to use the virtual resource model to achieve:
  - Accuracy: If a client requests a virtual workspace to start at time t, it is satisfied at that time. We achieve this by scheduling overhead.
    - Deadline-driven file staging
  - Efficiency: Minimize overhead, and maximize utilization.
    - Image caches
    - VM Suspend/Resume

Design: File Staging

- VM images are costly to transfer
- "Job Style" file staging strategies, commonly used by job schedulers without application-specific knowledge of what they are scheduling, are inadequate for time-sensitive VWs.
- Solution: Provide the scheduler with VW-specific knowledge, so it will *schedule* the overhead of the image transfer, instead of using a naïve strategy.

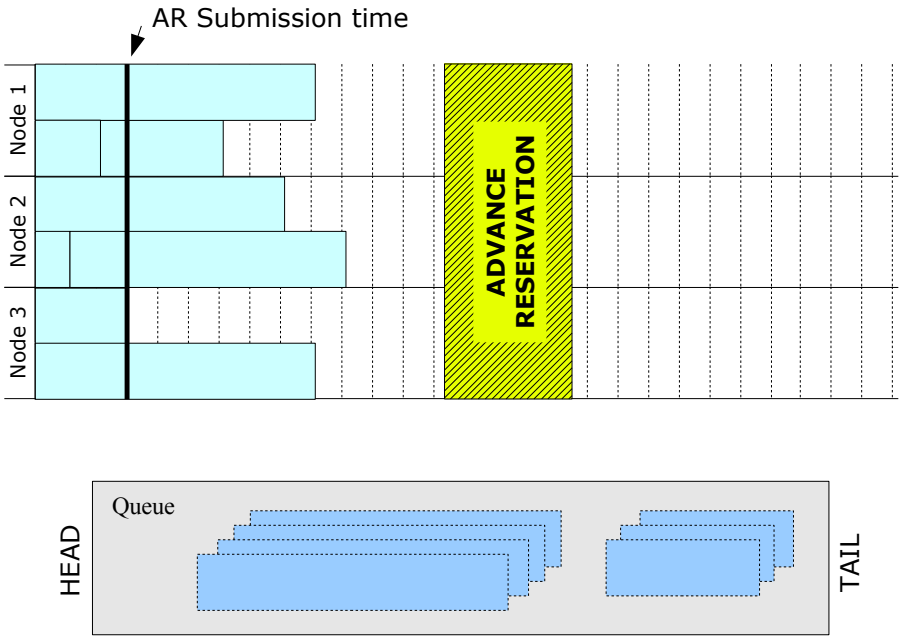
File Staging



Design: Image caches

- Our virtual workspace model allows the use of generic VM images that can be reused in the same node.
  - Make a local copy of the image and *bind* it to a specific metadata file and deployment request.
- This enables the use of image caches on a resource provider's physical nodes.
  - Reduce number of transfers
  - Avoid redundant transfers

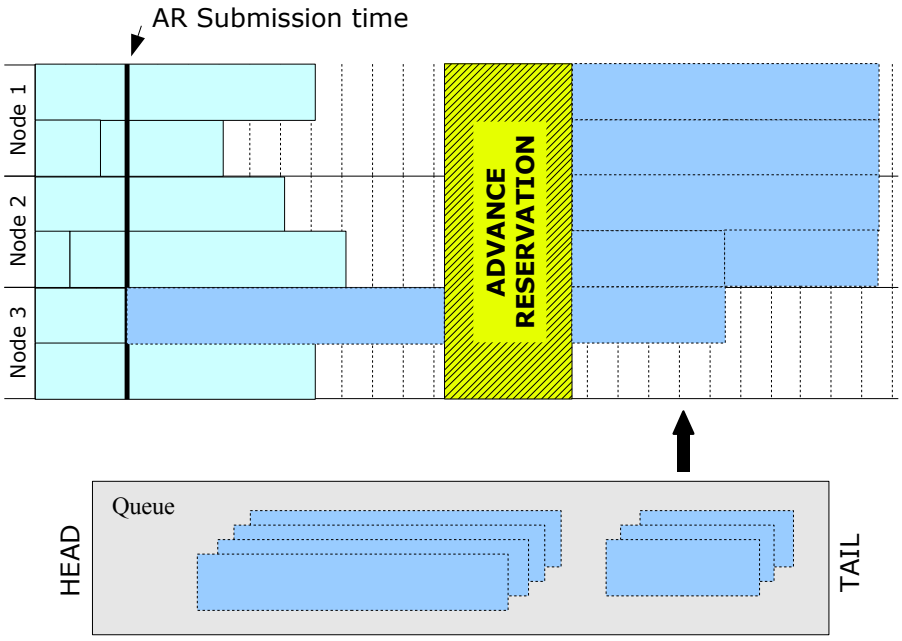
Mixing AR and Batch



Mixing AR and Batch

- A batch scheduler will process requests in a specific order (determined by priorities, local policies, etc.)
- However, physical nodes must be "drained" of batch jobs before a reservation can begin.
  - i.e. We want to avoid a batch job running at the time an AR starts, as this will mean it will have to be canceled.
- So, before an AR, it might be impossible to schedule the request at the head of the queue.
- One solution is to wait for resources to become available (after the reservation)

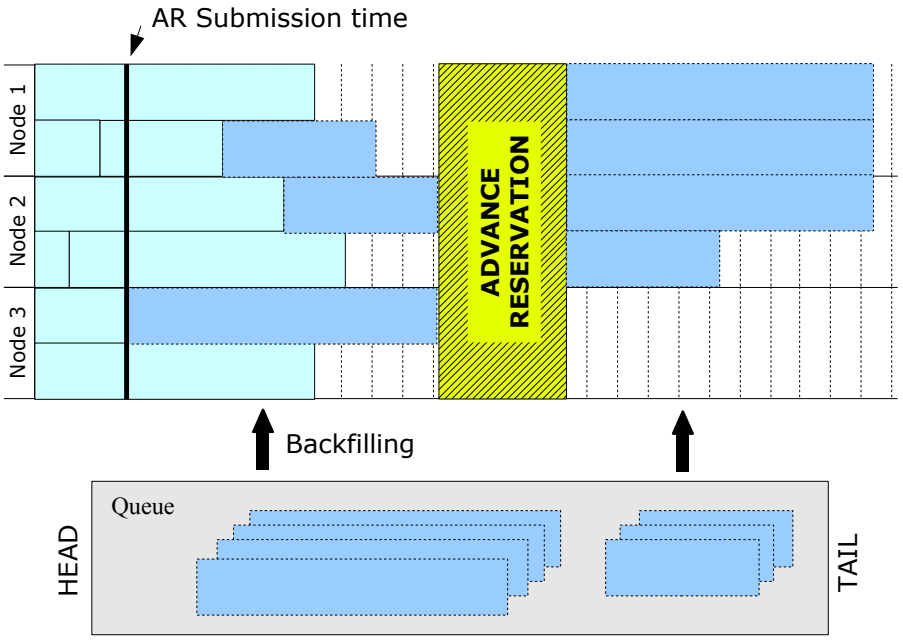
Draining



Backfilling

- A better solution is *backfilling*
- Instead of waiting for resources to become available (after the reservation), the scheduler traverses the queue in search of jobs which could run up to their maximum allowed time before the AR.
  - The selection can be done according to different algorithms. We use a first-fit algorithm.
- This results in lower-priority jobs running before the jobs at the head of the queue. However, it does not affect the starting time of the skipped jobs, since these will still run after the reservation.

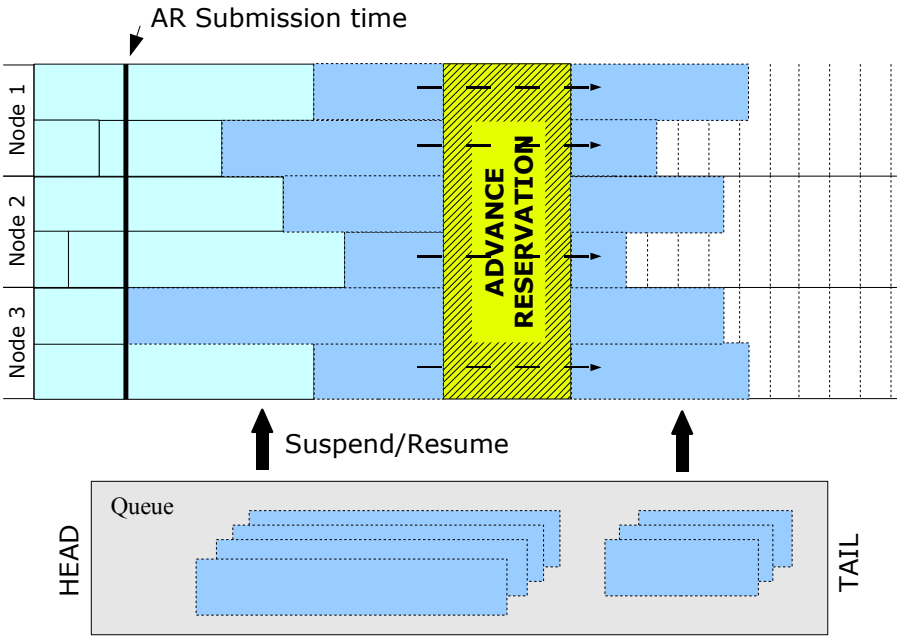
Backfilling



Suspend/Resume

- Backfilling is still not ideal, since there might still not be jobs short enough to backfill the time before the AR.
- A better solution is to use suspend/resume:
  - Select job at the head of the queue, even if we cannot run it to completion before the AR.
  - Suspend execution of the job before the AR, and resume it after the AR.
  - Also called checkpointing or preempt/resume

Suspend/Resume



Suspend/Resume

- Suspend/Resume is not a new technique.
  - Many existing schedulers support it: SGE, Condor, LSF, ...
  - However, the job executable must support suspend/resume, usually by linking with the scheduler's checkpointing libraries, or the kernel must support checkpointing.
- With VMs, we can use suspend/resume, regardless of what software is running inside the VM.

Implementation

- Two implementations:
  - **SGE-based**: We use Sun Grid Engine (SGE) as a local resource manager backend, extending it so it will take into account VW-specific information, enabling it to prestage images to nodes, and take into account the state of image caches in its scheduling decisions.
    - However, SGE does not support advance reservations. This implementation is used to test image prestaging and caching techniques on a real cluster.
  - **Prototype scheduler**: A scheduler developed by us which can co-schedule batch and AR requests, with support for image prestaging and caching, and suspend/resume.
    - Currently does not interact with a real testbed and runs in simulation.

Index

Experiments

- Three groups of experiments:
  - *#1 Measuring Accuracy*: Investigates effect of scheduling image transfers on accuracy, using in AR-only deployments.
  - *#2 Measuring Efficiency*: Investigates effect of image caches on efficiency, using Batch-only deployments.
  - *#3 Batch and AR*: Investigates what types of mixed workloads (Batch and AR) stand to benefit from using VMs (by adequately managing overhead and using suspend/resume)
- #1 and #2 use a physical testbed with our SGE-based implementation
- #3 uses a simulated testbed with our prototype scheduler.

Experimental Setup
(Experiments #1 and #2)

- Testbed
  - 10 nodes in Chiba City @ ANL
  - Dual CPU Pentium III 500MHz, 512MB RAM, 9GB local disk
  - One head node, one image node, eight worker nodes.
- Real and artificial workloads
  - Real: Taken from Parallel Workloads Archive (see paper for details on how workload was adapted to our testbed)
  - Artificial: Using a trace generator developed by us.
- Assumptions
  - Each submission requests same amount of CPU and memory (allowing 2 VMs per physical node)
  - VW remains idle during runtime
  - No network traffic that would be shared with preparation overhead.

Measuring Accuracy

- This experiment measures the effect of scheduled image transfers.
- Clients make advance reservations for fixed amount of time.
- Artificial submission trace with a feasible schedule.
- We measure "client satisfaction":

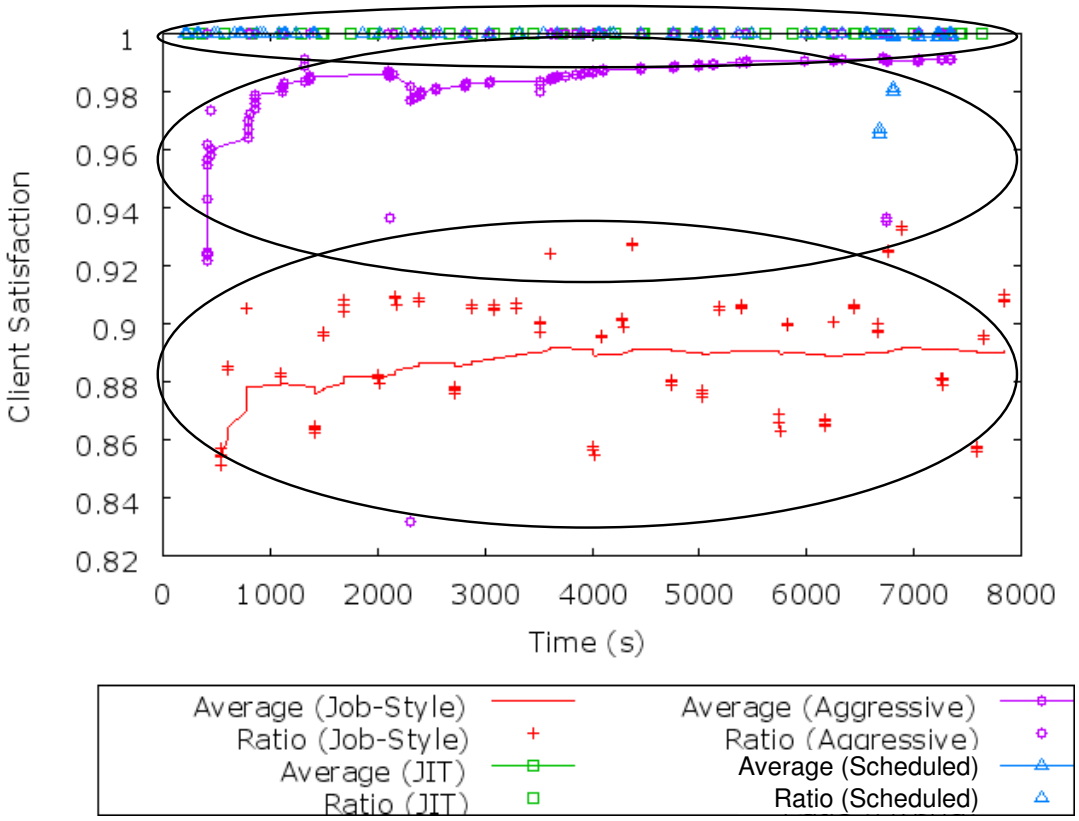
$$Client\ satisfaction = \frac{Real\ duration\ of\ workspace}{Duration\ requested\ by\ user}$$ (larger is better)

  - Note: We assume workspaces always end at their requested time. i.e. if a workspace started late, a user can't cheat client satisfaction by delaying the end time.

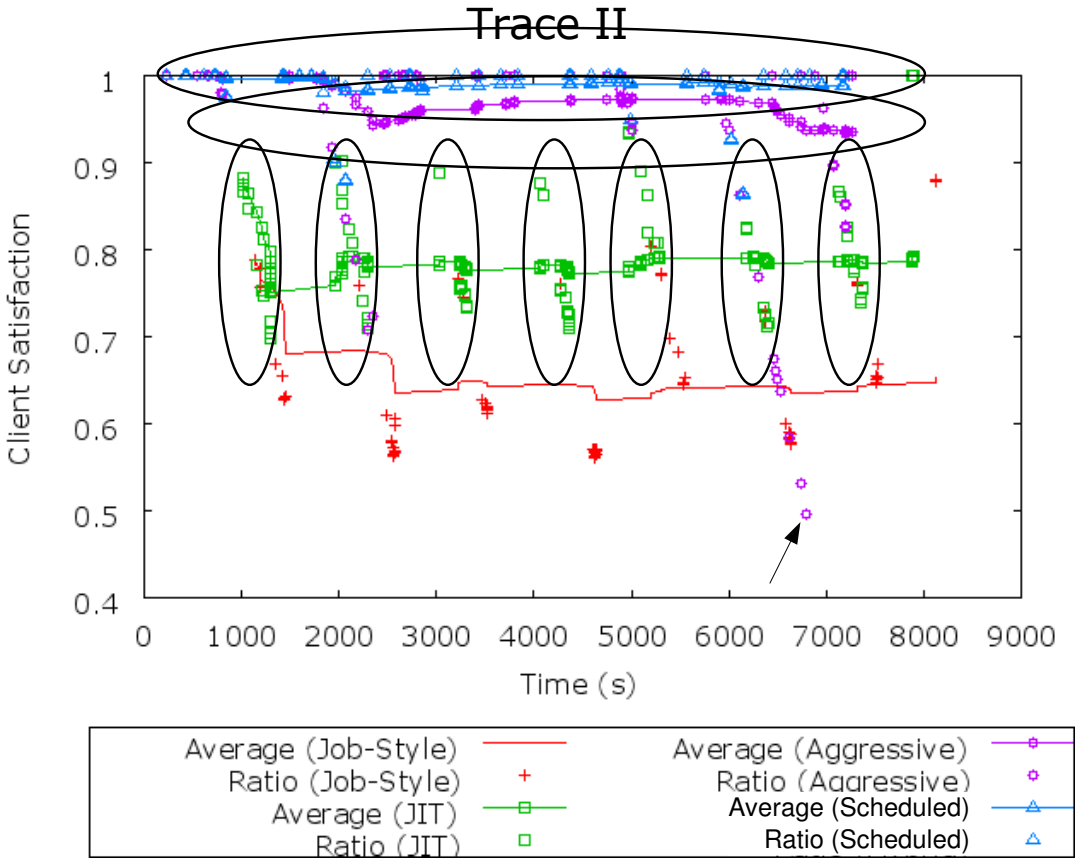
Trace Composition
(Measuring Accuracy)

- Traces I and II have a duration of 2h each, during which AR submissions are performed.
  - Each submitted VW has 2-4 nodes and must run for 30min.
  - The image for each VW is selected at random from a list of six possible images with size 600MB (selection is uniformly distributed)
  - More details about trace composition in paper.
- Traces I and II differ in the distribution of the VW starting times.
  - Trace I: uniformly distributed throughout the trace
  - Trace II: pooled in 100s windows (each occurring every 900s)

Trace I



Trace II



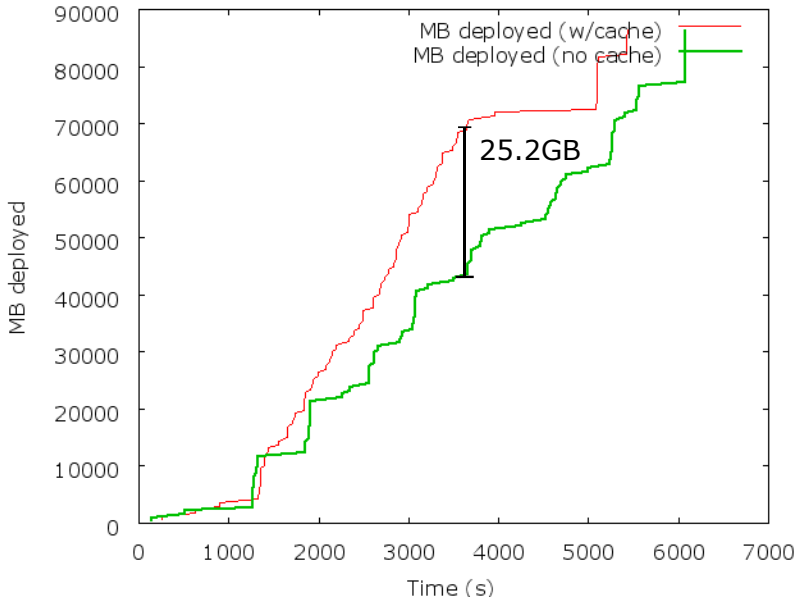
Measuring Efficiency

- Investigates effect of image caches.
- Clients make requests that can be scheduled on a best-effort basis
- Image caches are used to reduce preparation overhead.
- Real workload taken from the Parallel Workloads Archive
- We measure MB deployed (larger is better)

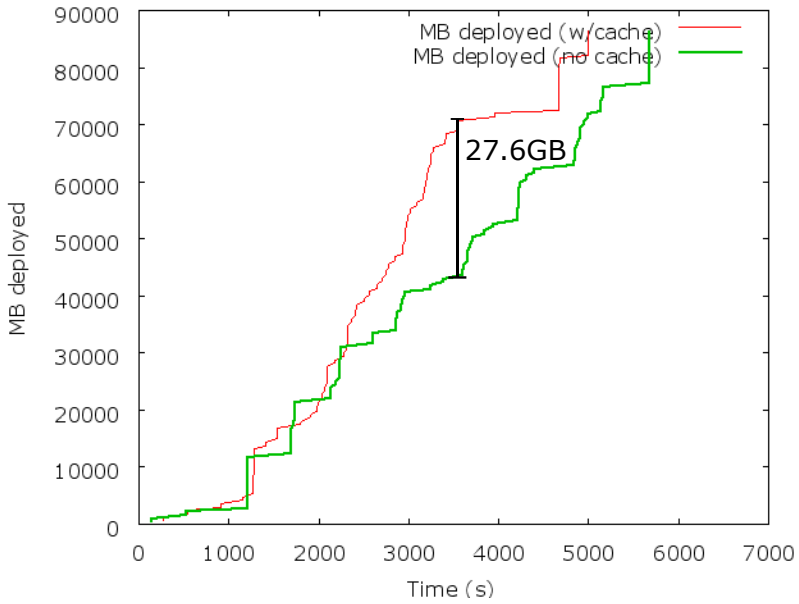
Trace Composition
(Measuring Efficiency)

- Traces III and IV
  - 80 minutes, with 62 submissions, extracted from SDSC DataStar express queue. This segment was selected specifically because it is a flurry of short-duration jobs.
- Images selected from a list of six possible 600MB images. Traces III and IV differ in the distribution of the images.
  - Trace III: Uniformly distributed
  - Trace IV: Two of the images (33%) account for 82% of the submissions.

Trace III



Trace IV



Experimental Setup
(Experiment #3)

- Simulated testbed
  - 8 nodes with two CPUs and 1024MB RAM.
  - 100Mbps network
- Artificial workloads
  - Using our trace generator
  - Mixes batch requests and AR requests
- Assumptions
  - Each submission requests same amount of CPU and memory (allowing 2 VMs per physical node)
  - No network traffic that would be shared with preparation overhead.

Batch and AR

- This experiment measures the effect of using our virtual resource model on mixed workloads.
  - In particular, we are interested in finding under what conditions VMs could provide better performance.
- Clients make both batch and AR requests.
- We use image prestaging and caching to reduce preparation overhead, and suspend/resume to improve utilization.
- We measure:
  - Time to complete all batch requests
  - Utilization of physical resources

Trace Composition
(Batch and AR)

- We use several traces where we modify three parameters:
  - Duration of batch requests
    - "short": Avg = 5min
    - "medium": Avg = 10min
    - "long": Avg = 15min
  - AR resource consumption
    - Up to 25% of available physical resources
    - 25% to 50%
    - 50% to 75%
    - 75% to 100%
  - Proportion of Batch/AR requests (measured in terms of total time requested)
    - 25% Batch, 75% AR
    - 50% Batch, 50% AR
    - 75% Batch, 25% AR

Trace Composition
(Batch and AR)

- 36 traces in total
- Each trace has a duration of 4 hours.
  - Workload is generated so that, even with 100% utilization, it would still take 4 hours to process all the requests.
- The image for each VW is selected at random from a list of six possible images with size 600MB (selection is uniformly distributed)
- We show the results of running each trace in 5 different configurations.

Configurations

	noVM_nosuspend	VM_nosuspend	VM_nosuspend_ov10	VM_suspend_ov10	VM_suspend_cache_ov10
	Without VMs	**With VMs (no suspend/resume)**	**With VMs (with runtime overhead)**	**With VMs (with suspend/resume)**	**With VMs (with image caching*)**
Suspend/Resume	No	No	No	Yes	Yes
Deploy images?	No	Yes	Yes	Yes	Yes
Image cache	No	No	No	No	Yes
Overhead	0.00%	0.00%	10.00%	10.00%	10.00%

We change only one parameter between each configuration

* Image cache in each node can contain 3 out of the 6 possible VM images.

Single Trace

- We start by showing the results of a single trace.
  - Duration of batch requests: "long"
  - AR resource consumption: 75% - 100%
  - Proportion of Batch/AR: 75%/25%
- This is the trace used in the results shown at the beginning of the presentation.

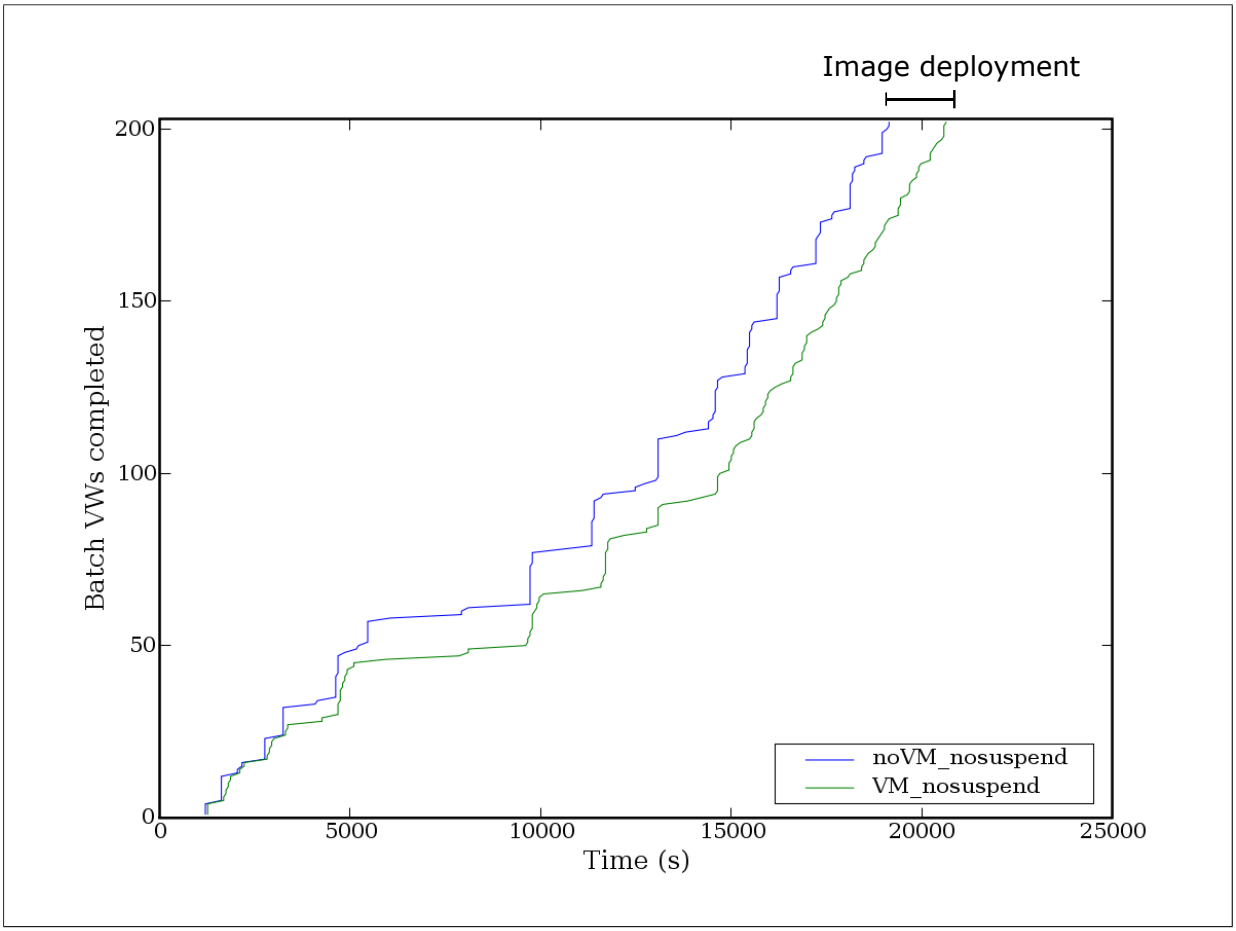
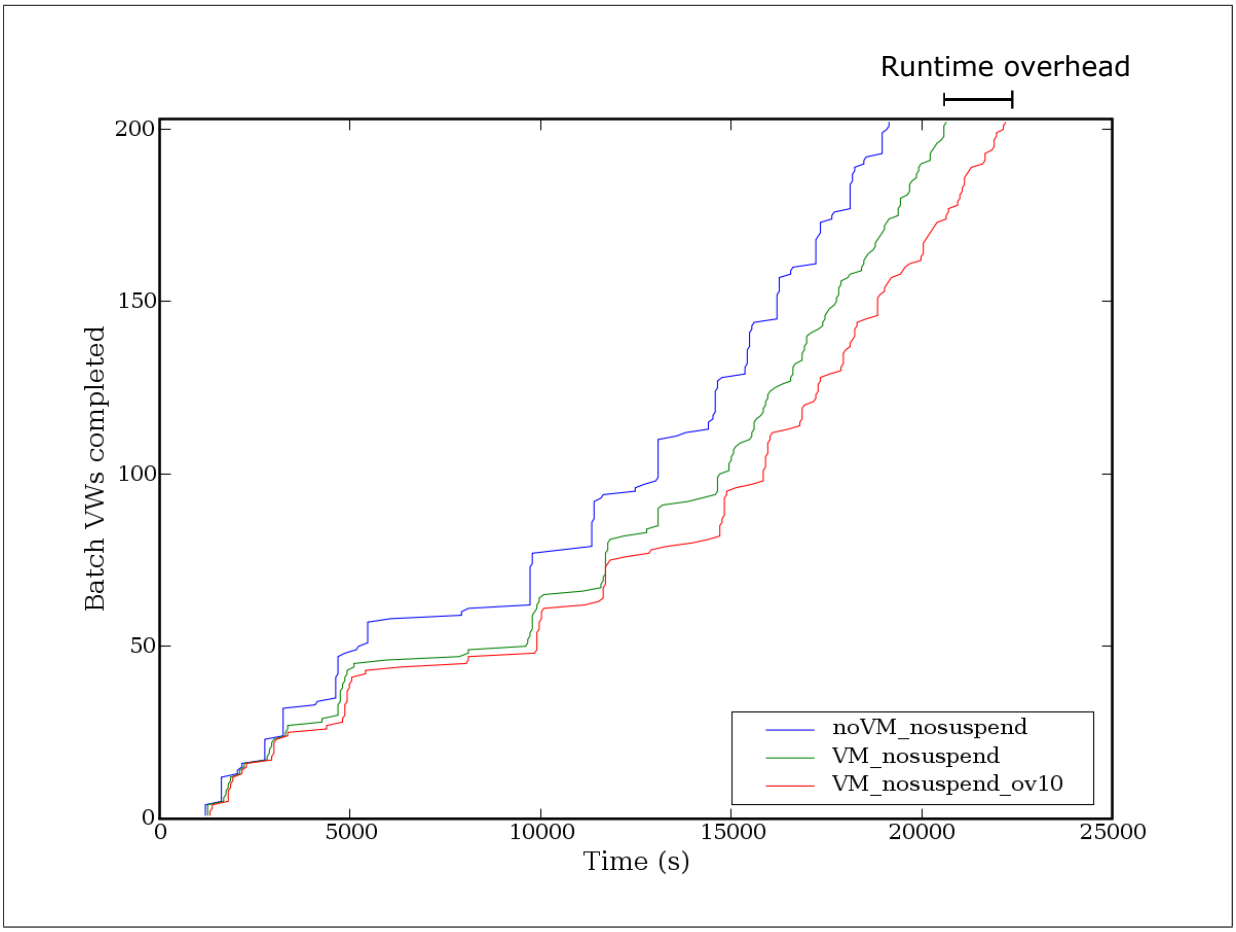
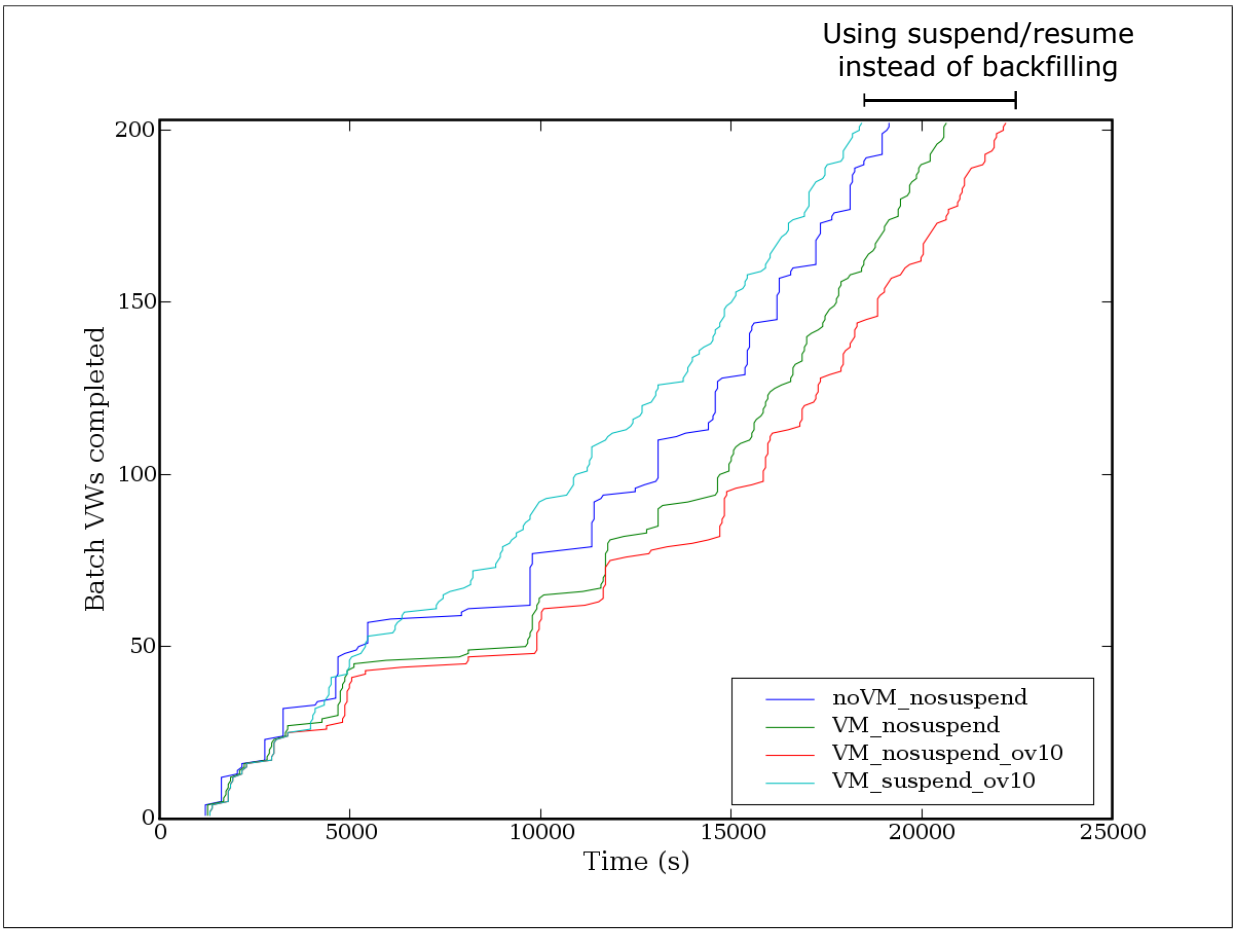


Image deployment
Batch VWs completed
Time (s)
noVM_nosuspend
VM_nosuspend
0
50
100
150
200
0
5000
10000
15000
20000
25000


Runtime overhead
Batch VWs completed
Time (s)
noVM_nosuspend
VM_nosuspend
VM_nosuspend_ov10
0
50
100
150
200
0
5000
10000
15000
20000
25000


Using suspend/resume
instead of backfilling
Batch VWs completed
Time (s)
0
50
100
150
200
0
5000
10000
15000
20000
25000
noVM_nosuspend
VM_nosuspend
VM_nosuspend_ov10
VM_suspend_ov10

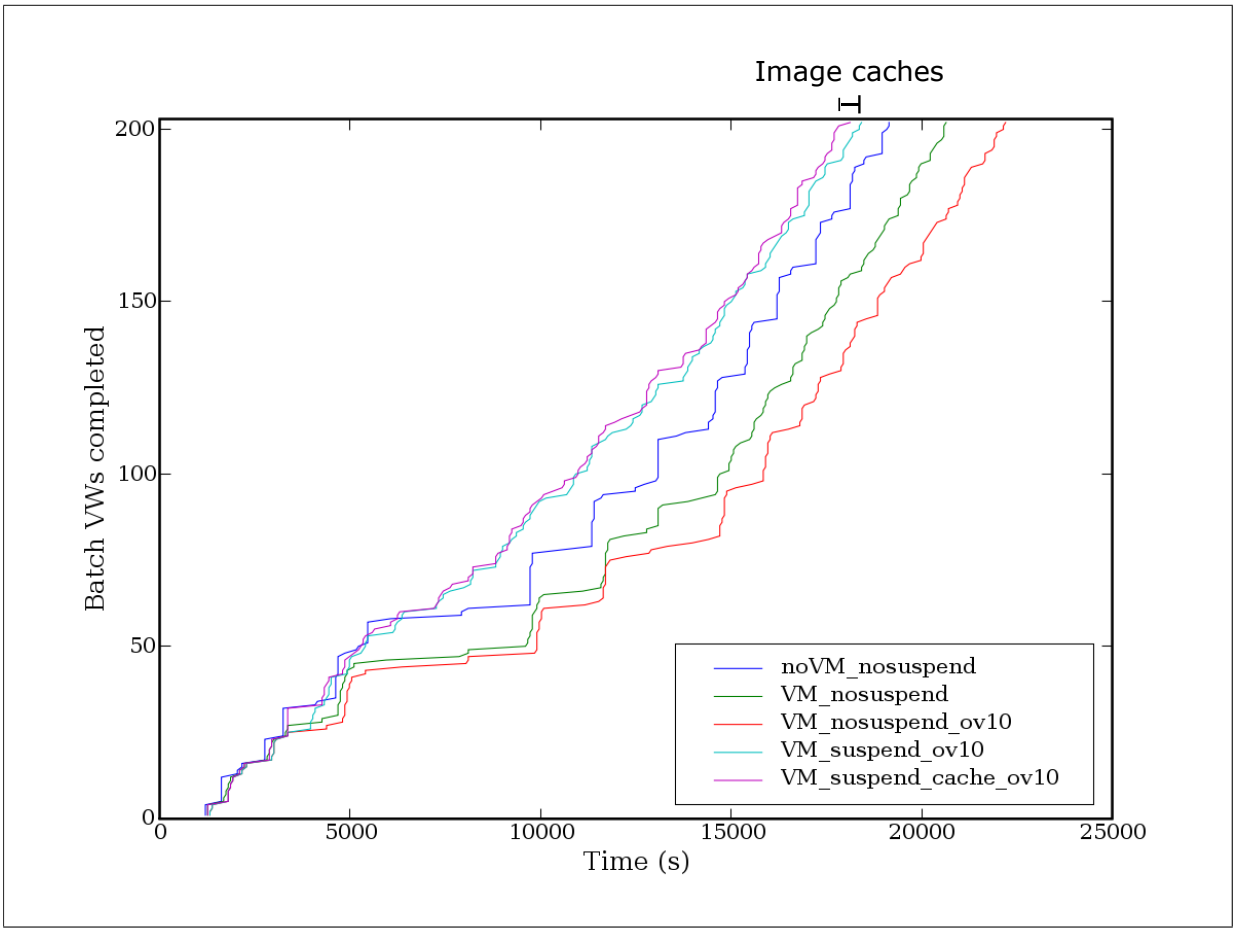


Image caches
Batch VWs completed
Time (s)
0
50
100
150
200
0
5000
10000
15000
20000
25000
noVM_nosuspend
VM_nosuspend
VM_nosuspend_ov10
VM_suspend_ov10
VM_suspend_cache_ov10

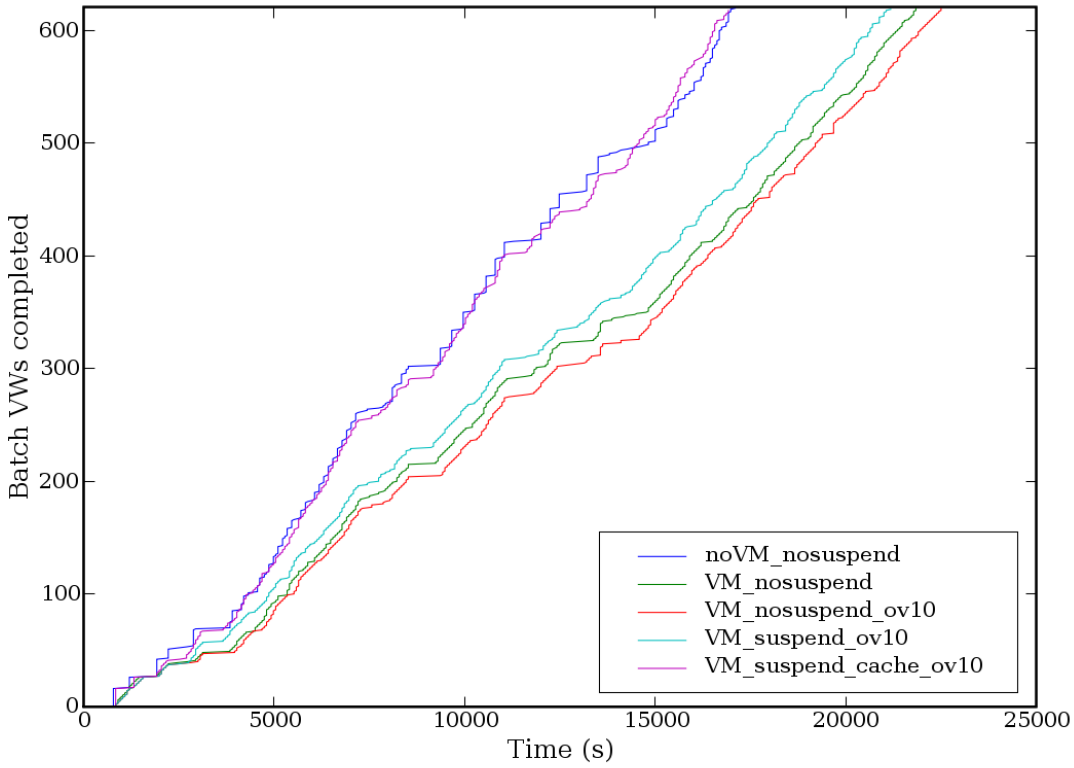
Utilization

Configuration	Utilization
noVM_nosuspend	76.00%
VM_nosuspend	70.00%
VM_nosuspend_ov10	72.00%
VM_suspend_ov10	87.00%
VM_suspend_cache_ov10	88.00%

- Utilization is measured as the percent of physical resources used (not idle) at a given time.
  - In our case, since all requests require the same amount of CPU and memory, allowing for 2 VMs to run on the same machine, measuring CPU and memory would yield the same utilization %.
  - The above is the average utilization at the end of the experiment.

Single Trace

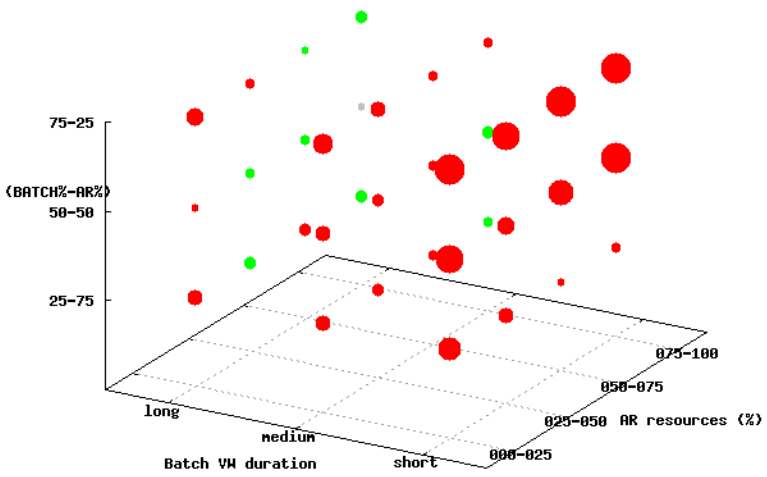
- In this trace, the presence of long batch requests do not allow the time before an AR to be backfilled with short requests.
- Using suspend/resume results in better utilization of resources before an AR, which makes up for the overhead of using VMs.
- What happens if we use short batch requests instead?



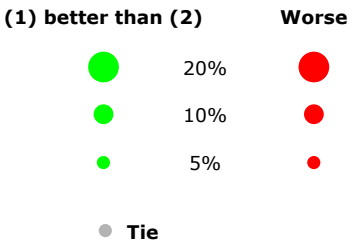
Single Trace

- The presence of short batch requests allows the backfilling algorithm to achieve very good performance in the non-VM case.
- Furthermore, many more images have to be deployed in this case, which increases the preparation overhead.
  - Without image caches: Using suspend/resume does not compensate for the preparation overhead.
  - With image caches: Preparation overhead is reduced enough to achieve slightly better performance than the non-VM case.

All Traces

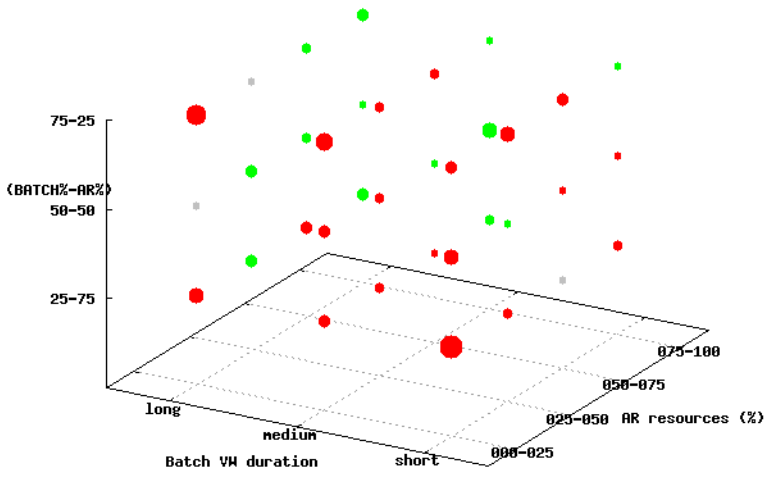


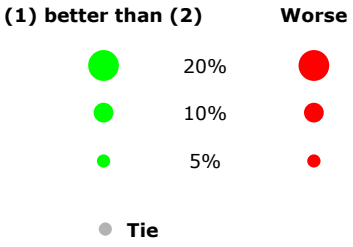
- This graph shows the results of running all the traces, comparing the time to complete all the batch requests:
  - (1) Using VMs, suspend/resume, and 10% overhead.
  - (2) No VMs, without suspend/resume.



All Traces

- Here we modify the VM configuration to use image caches.





All Traces

- Regardless of caching, VMs are specially advantageous in the absence of short jobs (which a scheduler could use to backfill the time before a reservation). In this case, the performance gained by suspend/resume compensates for the overhead of using VMs.
- On the other hand, when many short batch requests have to be processed, backfilling is effective enough that little is gained by suspend/resume.
- Adequate overhead management (such as using caches) allows us to get reasonable results all across the board (and, in some cases, switch from having worse performance to having better performance).

Index

Related Work

- Shirako (Duke University)
  - Allows virtual clusters to be deployed onto one or more physical clusters.
  - Resources can be leased (i.e. AR)
  - Does not take into account preparation overhead, which can affect accuracy.
  - References:
    - David Irwin, Jeff Chase, Laura Grit, Aydan Yumerefendi, David Becker, and Ken Yocum. Sharing Networked Resources with Brokered Leases. in USENIX Technical Conference, 2006.
    - COD: Cluster-On-Demand project: http://www.cs.duke.edu/nicl/cod/
    - Shirako (part of Cereus project): http://www.cs.duke.edu/nicl/cereus/shirako.html

Related Work

- Violin, VioCluster (Purdue)
  - Allows overlay of a virtual cluster over several physical clusters.
  - Assumes VM images are predeployed.
  - References
    - Paul Ruth, Junghwan Rhee, Dongyan Xu, Rick Kennell, Sebastien Goasguen, Autonomic Live Adaptation of Virtual Computational Environments in a Multi-Domain Infrastructure. in ICAC'06
    - Paul Ruth, Phil McGachey, Dongyan Xu. VioCluster: Virtualization for Dynamic Computational Domains. in Proceedings of the IEEE International Conference on Cluster Computing (Cluster'05), 2005.
    - Paul Ruth, Xuxian Jiang, Dongyan Xu, Sebastien Goasguen. "Virtual Distributed Environments in a Shared Infrastructure", IEEE Computer, Special Issue on Virtualization Technologies, May 2005.

Related Work

- Maestro-VC (NCSA/UIUC)
  - Optimizes scheduler for VMs, including image caching.
  - Focuses on long-term deployments, and so assumes that any preparation overhead will be acceptable (without attempting to guarantee accuracy)
  - References
    - Nadir Kiyanclar, Gregory A. Koenig, William Yurcik. "Maestro-VC: A Paravirtualized Execution Environment for Secure On-Demand Cluster Computing". CC-Grid 06.

Related Work

- Virtuoso and VSched (Northwestern)
  - Co-schedules interactive and batch workloads on individual machines, but assuming no deployment overhead.
  - References
    - B. Lin, and P. Dinda, VSched: Mixing Batch and Interactive Virtual Machines Using Periodic Real-time Scheduling, Proceedings of ACM/IEEE SC 2005 (Supercomputing), November, 2005.
    - Virtuoso: Resource Management and Prediction for Distributed Computing Using Virtual Machine. http://virtuoso.cs.northwestern.edu/

Related Work

- XGE (University of Marburg)
  - Extend SGE so it will use different VMs for serial batch requests and for parallel requests.
  - Improve cluster utilization by using suspend/resume.
  - Assume two VM images predeployed on all the cluster nodes.
  - References
    - Niels Fallenbeck, Hans-Joachim Picht, Matthew Smith, Bernd Freisleben. "Xen and the Art of Cluster Scheduling". VTDC 2006.

Index

Conclusions

- VM-based Virtual workspaces offer users with qualitative improvements, such as a custom software environment and enforceable resource allocations, but these come at a cost, caused by the overhead of using VMs.
- We propose a model for managing virtual resources which provides an *accurate representation of resources* for the user and an *efficient resource usage* for providers.
  - Our model separates resource use devoted to the overhead of VM deployment from resources available to the VM itself.
  - Uses VM mechanisms, such as suspend/resume.

Conclusions

- Results: Using metadata and overhead scheduling, in accordance with our model, results in improved accuracy and efficiency.
  - Judicious use of workspace metadata, provided to the scheduler, enables scheduling and optimizing of the workspace preparation.
  - Scheduling workspace preparation results in better adherence to requested availability time (scheduling image transfers).
  - Optimizing workspace preparation results in efficient resource usage (image caches).
  - Leveraging VM resource management mechanisms results in improved utilization of resources (suspend/resume)

Future Work
(Implementation)

- Use real submission traces for Batch+AR experiments.
- Run Batch+AR experiments on real hardware.
- Enhance scheduler with support for live migration of VMs.
- Integrate prototype scheduler into VWS
- Evaluate existing schedulers where our improvements could be integrated.

Future Work
(Research)

- Extending model to provide accurate and fine-grained use of other resources: CPU, memory, network bandwidth, and disk usage.
- The following present interesting questions:
  - **Network bandwidth**: Network traffic affects the CPU usage of Dom0 (Xen's management domain). Running network-intensive VMs requires ensuring that Dom0 always has enough CPU share to provide all the VMs with the network bandwidth they require. How can we dynamically compute the CPU share required by Dom0? How can we schedule network-intensive VMs guaranteeing that not only that bandwidth will be available, but also that Dom0 will not be strained for resources? How do we adapt to changing network loads?
  - **Disk**: VM images for ARs can be predeployed to nodes, but could potentially remain there for a long time waiting for the AR to begin, wasting disk space. What file prestaging strategies can maximize accuracy while minimizing disk usage over time?

Future Work
(Research)

- Supporting event-driven reservations
  - Resources must be available when an event arrives. Exact time is not known in advance, although the user could only send the event during an agreed-upon period of time
  - How do we keep all necessary virtual resources on standby without affecting all other virtual workspaces?

Questions?

Borja Sotomayor

Department of Computer Science – University of Chicago
borja@cs.uchicago.edu