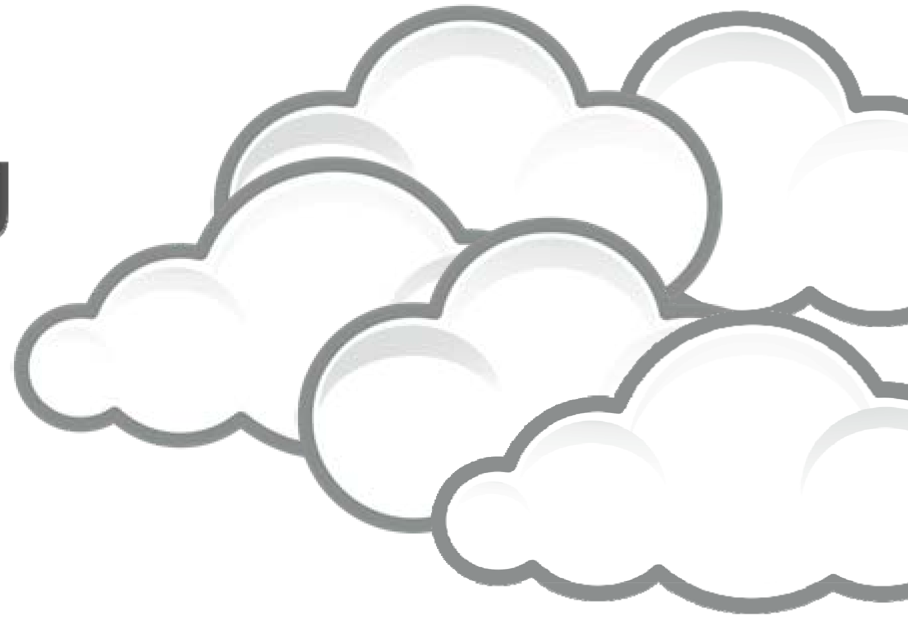
Evaluating the Cost-Benefit of Using Cloud Computing to Extend the Capacity of Clusters

Marcos Dias de Assunção[1,2], Alexandre di Costanzo[1]
and Rajkumar Buyya[1]

[1] Department of Computer Science and Software Engineering
[2] National ICT Australia (NICTA)
Victoria Research Laboratory
The University of Melbourne

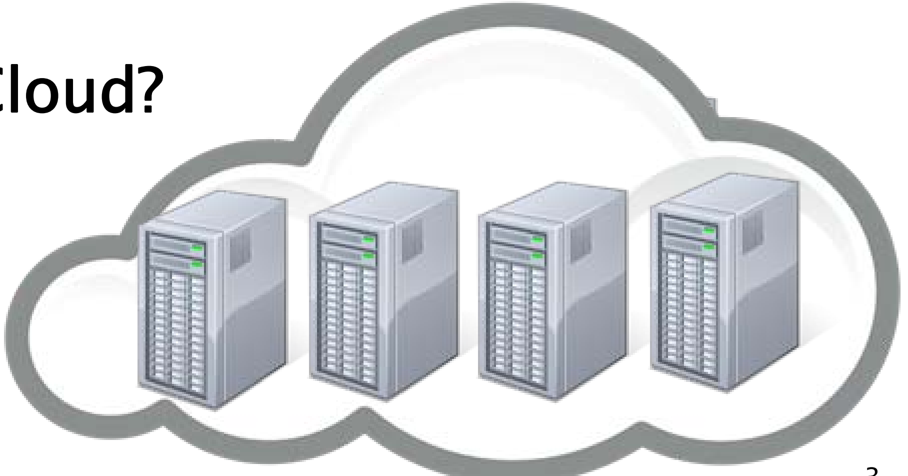
Cloud Computing



- Maturity of **virtual machines, virtualised storage** and **Web technologies**
- Software, Platform and **Infrastructure**
- Emergence of **commercial infrastructure** managed by virtual machine technologies
  - Amazon EC2

Infrastructure as a Service

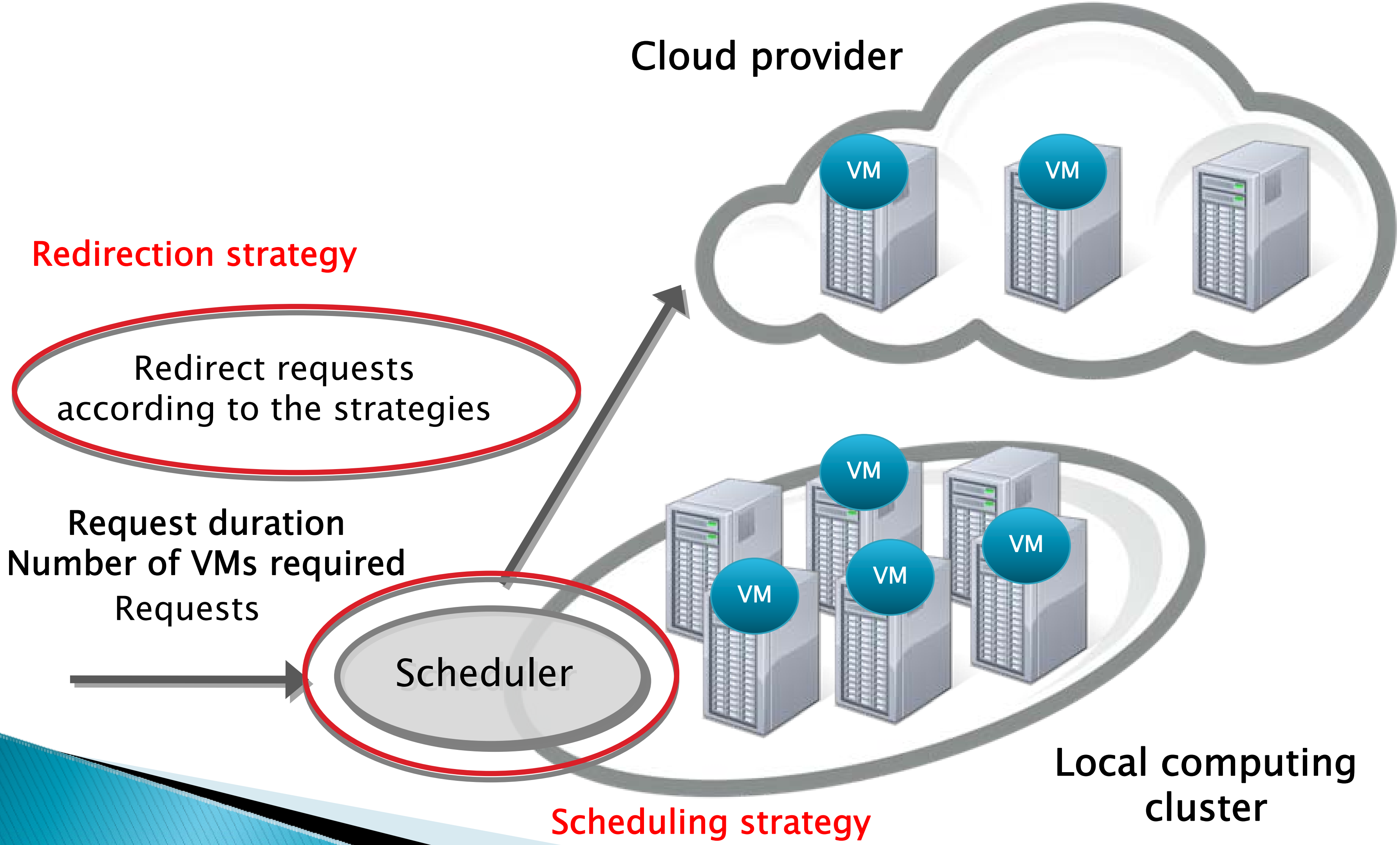
- Use of resources in a **pay as you go** manner
- Web Services APIs and command line tools
- Environments can scale on demand
- Start-ups can avoid initial outlays for computing capacity
- Organisations may have **existing computing infrastructure**
    - How to scale out to the Cloud?



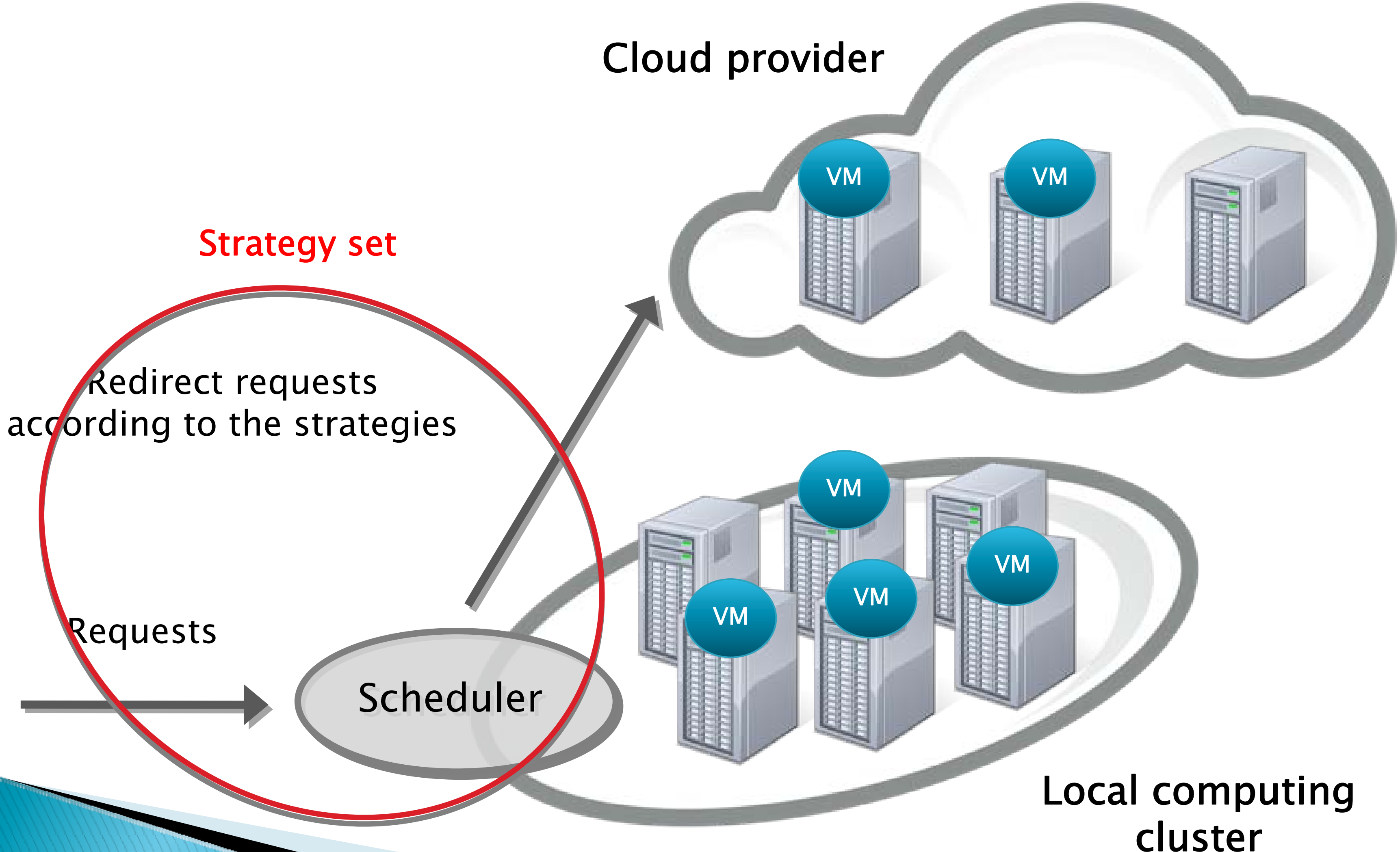
Challenge and Scenario

- Evaluation of using a **commercial provider to extend the capacity of a local cluster**
- **Different provisioning strategies** may yield **different ratios** of **performance improvement to money spent** using resources from the Cloud

Challenge and Scenario



Challenge and Scenario



Backfilling Policies

- Conservative and Aggressive
- Selective
  - Requests are given reservations if they have waited **long enough** in the queue
  - **Long enough** is determined by the **requests' eXpansion Factor**:
    - Xfactor = (wait time+runtime)/run time
  - The **threshold** is given by the average slowdown of previously completed requests
  - Use of **Adaptive-Selective-Backfilling***

* S. Srinivasan, R. Kettimuthu, V. Subramani and P. Sadayappan, Selective Reservation Strategies for Backfill Job Scheduling, 8th International Workshop on Job Scheduling Strategies for Parallel Processing (JSSPP '02), pp. 55-71, 2002

Strategy Sets

- **Naïve:**
  - **Use commercial provider** when **the request cannot start** immediately on local cluster
- Shortest Queue:
  - Aggressive backfilling
  - Compute **number of VMs required by requests** in the queue
  - **Redirect** request **if commercial provider's number is smaller**
- **Weighted Queue:**
  - Number of VMs that can be borrowed from commercial provider is the **number of VMs required by requests** minus **VMs in use**
- **Selective**
  - When the request's **xFactor exceeds the threshold,** the **scheduler makes a reservation at the place that yields the smallest slowdown**

Experiments

- Simulation of two-month-long periods
- SDSC Blue Horizon machine with 144 nodes
  - Number of VMs
- **Price** of a virtual machine **per hour**
  - Amazon EC2's small instance: US$0.10
  - Network and storage are not considered
- Values are averages of **5 simulation rounds**

Performance Metrics

- **Average Weighted Response Time** (AWRT) of site *k*:

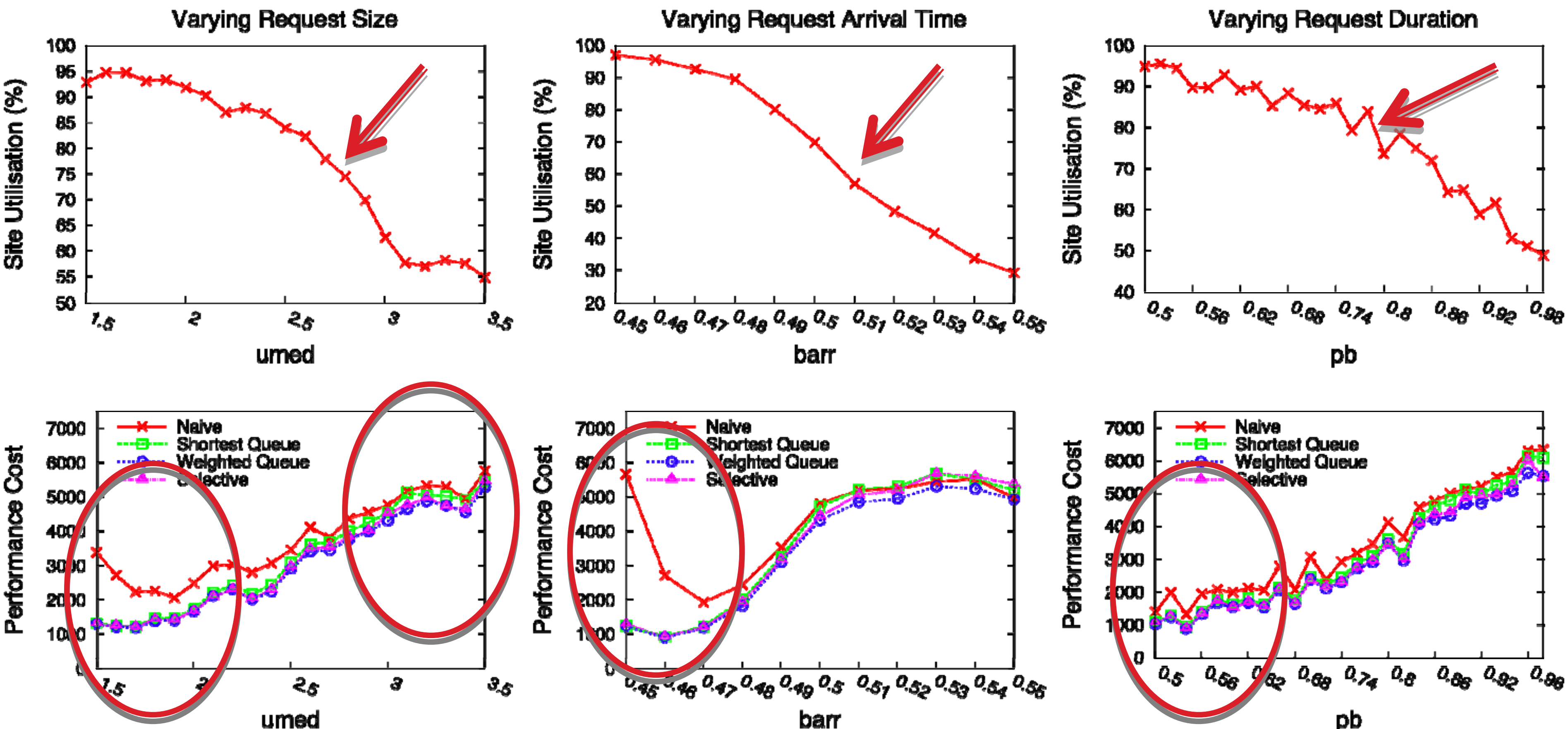
$$AWRT_k \quad \frac{{}_{j \qquad k} \quad p_j \quad m_j \quad ct_j \quad st_j}{{}_{j \qquad k} \quad p_j \quad m_j}$$

  - τ_k : requests submitted to site *k*
  - p_j : the runtime of request *j*
  - m_j : the number of processors required by request *j*
  - ct_j : request *j*'s completion time
  - st_j : if the submission time of request *j*

- **Performance Improvement Cost** of a strategy set *st*:

$$PIC_{st} \quad \frac{Amount\_spent}{AWRT_{base} \quad AWRT_{st}} \quad AWRT_{st}$$

Performance Improvement Cost



U. Lublin and D. G. Feitelson, The Workload on Parallel Supercomputers: Modeling the Characteristics of Rigid Jobs, Journal of Parallel and Distributed Computing, Vol. 63, n. 11, pp. 1105-1122, 2003

Deadline Constrained Applications

- Users may have stringent requirements on **when the virtual machines are required**
- Deadline constrained requests have:
  - Ready time
  - Duration
  - Deadline
- **Cost of using Cloud resources** used to **meet requests' deadlines** and **decrease** the number of deadline **violations** and request **rejections**

Deadline Aware Strategies

- Conservative
  - Places a request where it achieves the **best start time**
  - If rejections are allowed and **deadline cannot be met**, reject the request
- Aggressive
  - **Builds the schedule** using aggressive backfilling* and Earliest Deadline First
  - **If request deadlines are broken** in the local cluster, **try the commercial provider**
  - **If rejections are allowed and deadlines are broken, reject the request**

*G. Singh, C. Kesselman and E. Deelman, Adaptive Pricing for Resource Reservations in Shared Environments, In 8th IEEE/ACM International Conference on Grid Computing (Grid 2007), pp. 74-80, Austin, 2007.

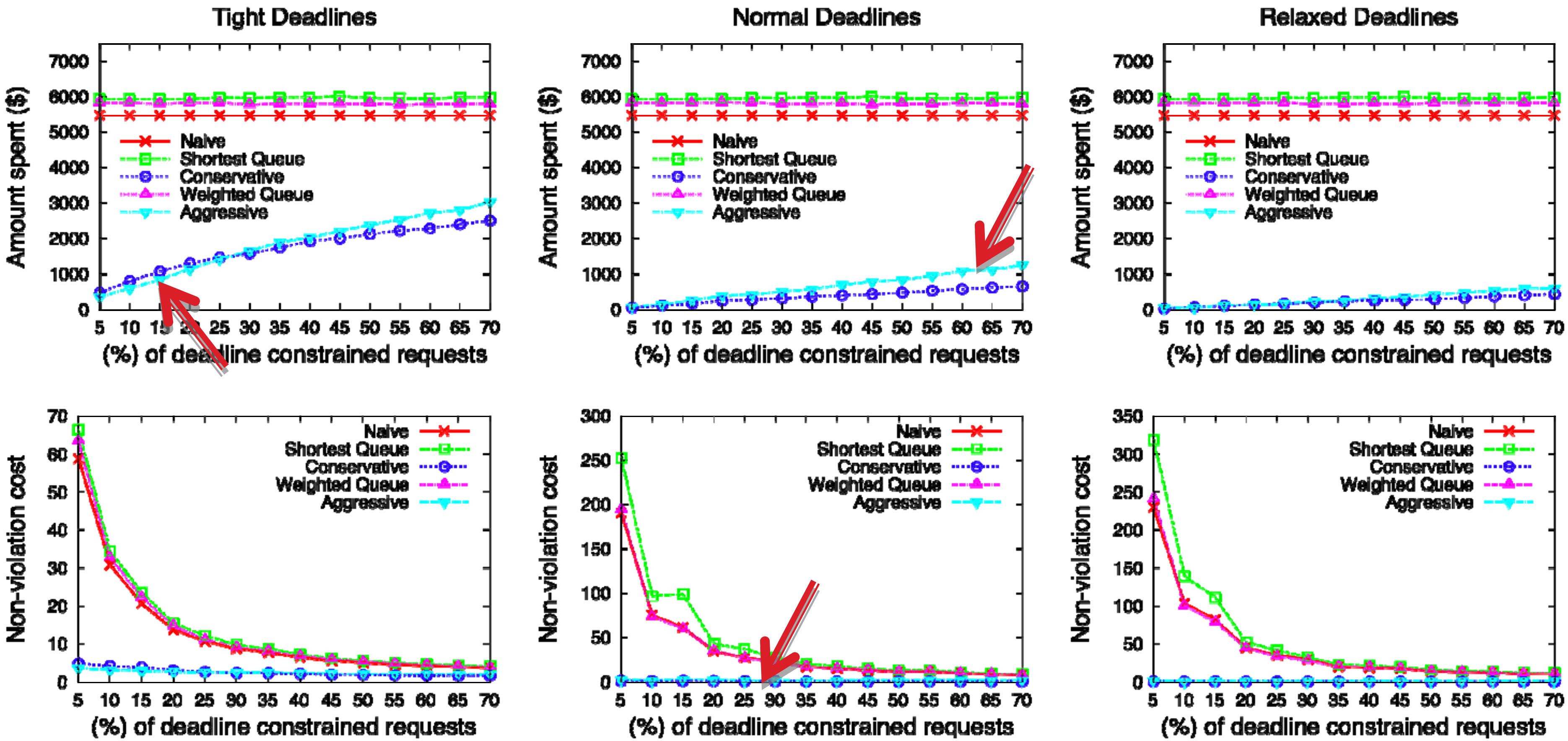
Cost of Reducing Deadline Violations

- The non-violation cost is given by:

$$non \quad violation\_cost_{st} \quad \frac{Amount\_spent_{st}}{viol_{base} \quad viol_{st}}$$

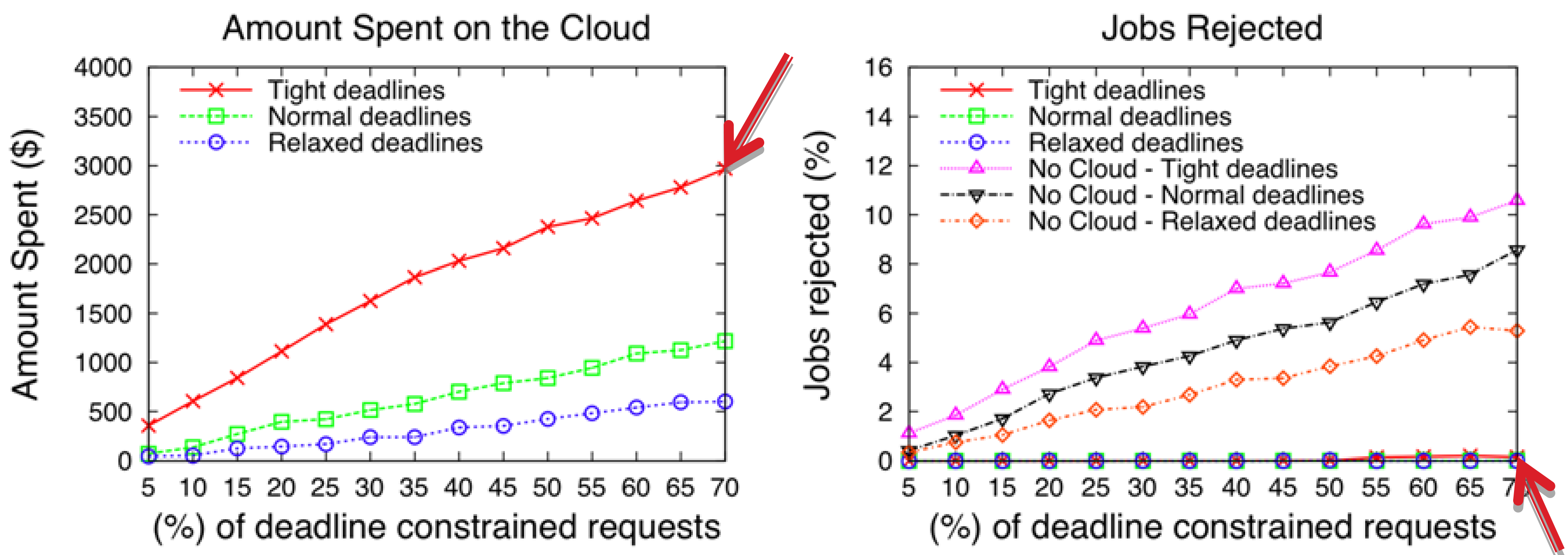
- Where:
  - $Amount\_spent_{st}$: amount spent with Cloud resources
  - $viol_{base}$: the number of deadline violations under the base strategy set
  - $viol_{st}$: the number of deadline violations under the evaluated strategy set

Cost of Reducing Deadline Violations



- SDSC Blue Horizon's trace divided into two-month-long intervals
- We vary the % of requests with deadlines
- Stringency factors of 0.9, 1.3 and 1.7

Cost to Reduce Job Rejections: Aggressive Strategy Set



- SDSC Blue Horizon's trace
- We vary the % of requests with deadlines
- Stringency factors of 0.9, 1.3 and 1.7

Expenditure vs. Job Slowdown

Metric	Naïve	Shortest Queue	Weighted Queue	Selective
Amount spent with VMs ($)	5478.54	5927.08	5855.04	4880.16
Number of VM/Hours	54785.40	59270.80	58550.40	48801.60
AWRT (improvement)	15036.77	15065.47	15435.11	14632.34
Req. slowdown (improvement)	38.29	37.65	38.42	39.70

- SDSC Blue Horizon's trace divided into two-month-long intervals

Conclusions

- Scheduling policies can yield different **ratios of performance improvement to money spent**
  - Naïve policy has a **higher performance improvement cost**
- **Selective policy provides a good ratio of money spent to job slowdown improvement**
- Using commercial provider to meet job deadlines
  - **Less than $3,000** were spent to **keep the number of rejections close to zero**

Future Work

- Scheduling strategy that strikes a **balance** between **money spent and performance** improvement
- Use of the **Cloud** to handle **peak demands**
- Experiments with the real system
  - Applications that can benefit from using local and remote resources
  - Consider other resources such as storage and network

Thank you for your attention and patience

Questions & Answers