



Technical Report No. 2012-594

Comprehensive (Depth) Exam –Preparing Q&A¹

Managing Data-Intensive Workloads in a Cloud

Student Name: Rizwan Mian

Student Number: 6001533

School of Computing

Queen's University

Kingston, Ontario, Canada

August 16, 2012

¹ Acknowledgements: I acknowledge research support from my supervisor (Dr. Patrick Martin), Queen's University and National Science and Engineering Research Council of Canada (NSERC).

Preface

I defended my PhD candidacy or comprehensive exam in Fall 11. In preparation for this exam, I prepared an extensive list of possible Q&As. Some of these questions came from my peer's comprehensive exam, while others arise from my mock defense before my research group. I also brainstormed and added more questions to the list. More importantly, they also include the questions that I was asked in my comprehensive exam. Some of these questions are generic, while others are specific to my area of research.

The questions are loosely structured according to the layout of my comprehensive report or the depth paper [14] needed for the comprehensive exams. Some of my peers find the Q&As useful in preparation for their comprehensive exam. Therefore, I put them in a technical report for the benefit of a wider audience. The language used is colloquial, and some of the answers are dated. Nonetheless, I hope they would help students in preparing for the comprehensive exams, and to a general reader in gaining knowledge in the area.

The curious reader is invited to look at my publications [15-18] to find out further developments in my research.

Keywords: comprehensive exam, Q&A preparation, anticipated questions.

Contents

1	General	3
2	Introduction/Background.....	5
3	Workload Management Taxonomy	7
4	Survey.....	11
4.1	Scheduling	12
4.2	Provisioning.....	13
5	Conclusions	16
6	Future work	17
7	Mock Defense.....	18
8	Report Layout.....	19
9	References	19

1 General

Q1. How workload management in cloud differ from any other workload management?

Q2. Is workload management in cloud different to workload management in [computing] grids?

Q3. why is workload management different for clouds to workload management in grids?

There are some similarities and differences that arise due to organization of grids and clouds. Would you like to talk about some organizational differences?

Q4. how is a cloud different to a grid?

Foster *et al.* have studied the similarities and differences in detail [8]. Let me point out the key differences you can ask :

- Grids usually expose physical machines to its users, where as clouds expose virtual machines. This has a control implication for users and providers.
- The grid servers may not be dedicated to its users. Virtual machines in clouds are dedicated to its users.
- The grid participant sites vary in scale. Grids usually consist of many small to medium sized data centers. Clouds usually consist of a few very large data centers. This has implications on the availability of resources.
- Grids don't have a pay-as-you-go model that went main stream, whereas clouds do.
- Grids usually contain multiple administrative domains, which complicate workload execution. Clouds are usually under one administrative domain.
- Most existing grid middleware, which enables access to Grid resources, is widely perceived as being too heavyweight. The heavyweight nature of the middleware, especially on the user-side, has severely restricted the uptake of grids by users. On the other hand, the cloud providers manage complexity on their side and offer interfaces of varying abstraction to their users.

Q5. Why do you compare your work to grids?

Given the similarities between grids and clouds, the successes achieved and lessons learnt in grids can be useful in clouds. Hence, I discuss relevant grid work.

Q6. How is cloud similar to grid?

- There is a common need to be able to manage large facilities;
- to define methods by which consumers discover, request, and use resources provided by the central facilities; and to
- implement the often highly parallel computations that execute on those resources.

Q7. Why is workload management in cloud different to dbms?

Some similarities and some differences arise due to the organization of dbms and clouds. The organizational difference include: dbms usually operate over fixed resource set, while resources in a cloud can be varied during workload execution; dbms usually operate over physical resources, while workload executes over virtual resources in clouds. Dbms, and especially (shared-nothing) parallel dbms, operate over dedicated, homogenous and tightly coupled servers. This limits the scalability of parallel dbms into 100s. The applications workloads may execute may over thousands of cloud resources. A parallel dbms operates with finer level of granularity, parts of a single query may be executed simultaneously on many resources and the results shipped to one destination. In contrast, workload execution in clouds has to operate with coarser granularity in order to avoid network becoming a bottleneck. The organizational differences would result in some differences in workload management in dbms and clouds.

Q8. What is the difference between dbms workload and cloud workloads?

“Workloads in a cloud can range from the ones consisting of relational queries with complex data accesses to the others involving highly parallel MapReduce tasks with simple data accesses. The workloads in clouds can also differ from the dbms workloads with respect to granularity of the requests, that is, the amount of data accessed and processed by a request. To reduce the data movement across the network, data can be processed at the same resource but this reduces parallelism. Increasing parallelism results in increased data movement. We see a tradeoff between amount of (processed) data movement and amount of parallelism. Coarse grain granularity attempts to strike the right balance. It seems that cloud workloads must operate at coarse-grain granularity to be highly scalable to thousands of resources.” p13 [14].

Q9. Why is cloud workload taxonomy different to dbms workload taxonomy?

It is different because the workloads in cloud and dbms have some differences.

Q10. Why don't you discuss scheduling algorithms in the suggested architectures?

That's a good question. Workload management frameworks differ considerably in their architectures at times, and so would the algorithms associated with them. Therefore, for my depth work, my focus was to look at the architectures instead of specific algorithms and policies. Now, I see many algorithms surfacing even for one system. For example, the built-in scheduler in hadoop employs First-In-First-Out (FIFO) variant. There is a fair-share scheduler for hadoop employing delay scheduling and copy-compute technique. Likely, there are other techniques that could also be employed. Personally, I was interested in scheduling algorithms and wanted to operate at that level. I would like to extend my depth work with the specific scheduling algorithms and policies.

Q11. Whose side are you taking?

Q12. Are you taking the provider's side?

I realize that the interests of users and providers may conflict. Their interests are captured by the objective functions.

Q13. Whose side is your taxonomy built for?

That's a good question. The taxonomy is built for “workload” management, and the core is “workload”. I define workload as a “set of requests accessing and processing data under some constraints”. I find this makes the taxonomy user-centered. Admittedly, this may not be clear from the paper since I was under the impression that my taxonomy is relevant for both users and providers equally.

Let me give you examples for why was I under this impression. In case of provisioning taxonomy, a cloud provider usually provides virtual processing and storage resources, and some trigger mechanisms. A cloud user exploits the mechanisms exposed to execute his application. For the scheduling part, I try and capture possibly conflicting interests of users and providers using objective functions.

Q14. Is there something fundamentally different about workload execution in a cloud?

If I may answer your question by comparing the workload management in clouds with workload management with dbms. I see some similarities between workload management for dbms and clouds. However, I see some fundamental differences. In cloud, the computing and storage resources are variable, while in dbms they are usually fixed. In cloud, the dollar-cost is associated with workload execution on a per-unit time basis – it is integral, while in dbms the dollar-cost is usually not associated with workload execution in the research literature I have seen – although it may be measured in terms of capital and maintenance cost.

Q15. How do people process large datasets now?

Many of the systems in my scheduling survey use cluster of commodity machines to process large data sets.

Q16. How did people process large datasets before clusters?

About 10 years ago, hundreds of megabytes to multi-terabytes was considered large. The database market is dominated by transactional database, where one terabyte is considered large. High-end servers can satisfy this need.

Q17. Why do you think there has not been much work on resource provisioning for workload execution?

Previously, resources had been pre-provisioned. That is a set of servers were bought and setup before hand before executing workload. The resource set was fixed. Hence, there was little need to discuss provisioning mechanisms. This has changed with the advent of cloud computing where resource provisioning is fluid and can be done in minutes. Therefore, I feel there is scope for research here.

Q18. Have you read all the papers?

I have looked at papers in varying details, as required. For example, I have looked at MapReduce papers in detail, but not looked at Flynn's taxonomy for the term single-program-multiple-data in any detail.

Q19. Why bother with workload management; why not buy more hardware?

That's a good basic question. Workload management allows us to associate objective functions such as deadline, dollar-cost, prioritization with workload execution, where as buying more hardware may only reduce the makespan of the workload. What's the point of buying more hardware, if they are not used effectively.

Q20. Is scheduling from OS relevant? Why or why not?

I guess I can point some key differences between OS scheduling and scheduling in clouds. First, resources in clouds are variable, whereas for an OS for a system it is fixed (e.g. some processors and some disks). The level of granularity of a work-unit is different: an OS deals with fine-grain work-units such as processes and possibly threads, while cloud deals with coarse-grain work-units such as workflow or a task. The workload in a cloud has to use coarse-grain work-units in order for it to be scalable.

2 Introduction/Background

Q1. What is the motivation for this work?

The primary objective of the work is to provide a systematic study of workload management of data-intensive workloads in clouds. The contributions are:

- A taxonomy of workload management techniques used in clouds.
- A classification of existing techniques for workload management based on taxonomy.
- An analysis of the current state-of-the-art for workload management systems in clouds.
- A discussion of possible directions for future research in the area.

Q2. What does data management include?

Data management is a generic term that may include: storage, data movement, archiving, backing up, serving requests of data efficiently.

Q3. What is data-intensive computation?

A data-intensive computation is an arbitrary computation on data where data access (read or write) is a significant portion of execution time.

Q4. What is large data?

I give real examples in the paper p7,8 [14]. Experiments at CERN will generate petabytes of data. Digital sky has terabytes of footprint. Analytical systems are approaching petabytes. There are other datasets which are gigabyte. What is large data is time sensitive. A decade ago hundreds of mega-bytes to multi-terabytes was considered large [20]. Now, commodity hard disks are approaching terabytes. Our understanding of large data is likely become outdated in five years time.

Q5. Are large data and data-intensive different?

Yes. Large data indicates the big data size (also see Q.4). Usually where there is large data, the computation becomes data-intensive. But there are examples where this is not the case. Consider SETI@home. The data recording-rate is 0.5MB/s or 2Gb/hour according to the published results of '97². However, the screensaver at home works for 12 hours on half a megabyte of input – this is compute intensive [9].

Q6. What is a request?

A request is a generic unit of work. It could be a task, workflow, query, transaction, job etc.

Q7. What is a resource?

In my case, it is usually a computing or a data node. Researchers also use it for other things such as networks, ports.

Q8. What is a data resource or a data node?

A machine that hosts some data. Let me share some examples. A dbms server is a data resource. A file server is a data resource. A ftp server is a data resource. I use data resource as a generic way of addressing all these.

Q9. What is the size of extremely large-scale commodity-compute data centers?

Google and Microsoft operate data centers which contain hundreds of thousands of machine [1].

Q10. What is the size of medium sized data center?

Thousands to tens of thousands of machines.

Q11. What is a data-center?

A data center is a facility used to house computer systems and associated components, such as telecommunications and storage systems.³

Q12. Wow would flash memory improve performance unpredictability?

I/O sharing by VMs is problematic, and flash memory will decrease I/O interference [2].

Q13. What is gang scheduling?

Gang scheduling is a scheduling algorithm for parallel systems that schedules related threads or processes to run simultaneously on different processors. Usually these will be threads all belonging to the same process, but they may also be from different processes, for example when

² http://setiathome.berkeley.edu/sah_papers/woody.php

³ Wikipedia, Data center, http://en.wikipedia.org/wiki/Data_center

the processes have a producer-consumer relationship, or when they all come from the same MPI program.⁴

Q14. What is a gang schedule for a VM?

“Most parallel computing today is done in large clusters using the message-passing interface (MPI). The problem is that many HPC applications need to ensure that all the threads of a program are running simultaneously, and today’s virtual machines and operating systems do not provide a programmer-visible way to ensure this. Thus, the opportunity to overcome this obstacle is to offer something like *gang scheduling* for Cloud Computing” [2].

Q15. What is a workflow, and how does it differ from workload?

In my paper, I define workload as a set of requests that access and process data under some constraints. I use workflow as a unit of work in my scheduling taxonomy. Niu et al. [21] use them interchangeably.

Q16. What are the basic barriers for enabling data-intensive distributed computing?

For general distributed computing the fundamental problems I see are: (a) bandwidth is small compared to data sizes we are dealing with, and (b) I/O is expensive. For cloud computing, I discuss suitability of data-intensive applications and highlight issues such as complex distributed commit protocols, engineering decisions behind parallel dbms’s p9-11 [14].

3 Workload Management Taxonomy

Q1. How would you compare your taxonomy and survey to other taxonomies and surveys?

Thanks for bringing that up. In preparation for my depth exam I revisited related work, and realized that other taxonomies and surveys should be a part of my depth paper. I would include that after the exam.

Pandey et al. [22] undertake a classification of management techniques for data-intensive application workflows, while I explore workload management mechanisms. Their target platforms are clusters, grids and clouds; however, their survey primarily consists of grid workflow management systems. My work is focused on clouds. They do not treat provisioning, workload characterization, and monitoring as workload management mechanisms, while I do. They treat data locality at a finer level by clustering tasks, files or workers. I treat data locality at a course level. Compared to my scheduling/replication coupling node, they treat data placement and replication as a leaf in their taxonomy, and do not explore its relationship with locality. They present restructuring of workflow as an optimization of workflow, and this can be loosely mapped to prediction-revision mapping in my taxonomy. They discuss various data transfer approaches, which I do not. They seem to cover more systems in their survey; however, in doing so map them to the parts of the taxonomy. In contrast, I cover fewer systems and map each system to the taxonomy, and summarize the survey in a table. I claim more cohesive taxonomy and survey.

Dong [6] presents a taxonomy and a survey of task scheduling algorithms in the grid. I present a workload management taxonomy for clouds. I see his work is focused on scheduling algorithms and as a result his work does not consider provisioning, workload characterization and monitoring, while mine studies workload management mechanisms. Nonetheless, his scheduling taxonomy and my scheduling sub-taxonomy can be compared directly. He explores scheduling algorithms at a finer detail, and especially explores dependent task set scheduling in some depth.

⁴ Wikipedia, Gang scheduling, http://en.wikipedia.org/wiki/Gang_scheduling

In contrast, I abstract the architectures of the entire data-intensive systems. His focus is on different scheduling approaches that may deal with some data. In contrast, my focus is on data-centered scheduling approaches, which are highlighted in the architectures. He covers more systems in his survey; however, in doing so maps them to the parts of the taxonomy. In contrast, I cover fewer systems and map each system to the taxonomy, and summarize the survey in a table. I claim more cohesive taxonomy and survey. He depicts the taxonomy in a figure, and then discusses different nodes and the systems in them. I present taxonomy and survey separately.

My scheduling portion of workload management builds on previous studies of scheduling in grids including [7]. As a consequence there are some similarities between studies, but nonetheless some important differences. Dong *et al.* (06) study scheduling algorithms in grids, our domain is clouds. They generalize the scheduling process in the Grid into three steps: (1) resource discovering and filtering, (2) resource selecting and scheduling algorithms according to a certain objective, and (3) job submission. Selection of resources can be compared to provisioning which is the process of allocation resources for workload execution. Provisioning in our case has a wider scope since it includes techniques such as migration. They draw a distinction between scheduling (a mapping plan of tasks to resources) and job submission (the process of submitting tasks for execution), and they focus on the second step in their study. I consider job submission and progress part of workload management mechanisms, and hence identify relevant mapping schemes such as just-in-time. They study scheduling algorithms in grids in some detail, and especially explores dependent task set scheduling in some depth. In contrast, I view the scheduling approaches at a coarser-level and abstract the architectures of the data-intensive systems. They study dependencies between tasks that can be static or dynamic, and algorithms associated with them. In my case, tasks have a static dependency (workflow) or no dependency (task), and I highlight their usage in scheduling approaches without providing algorithmic details. They explore some decoupled and combined strategies for computation and data scheduling. I also discuss scheduling/replication approaches, and in addition explore their relationship with data and process locality. He covers more systems in his survey; however, in doing so maps them to the parts of the taxonomy. In contrast, I cover fewer systems and map each system to the taxonomy, and summarize the survey in a table. I claim more cohesive and coupled taxonomy and survey. They combine taxonomy and survey, while I present taxonomy and survey separately.

Q2. Your way of building a taxonomy and conducting a survey is unusual, why?

Thanks for bringing that up. In preparation for my depth exam I revisited related work, and note that my peers [6; 22; 28]: (a) do not present taxonomy and survey separately, and more importantly (b) are able to build a bigger taxonomy and cover more systems by mapping systems to the parts of the taxonomy. In contrast, I feel I have invested a great deal of effort building a concise and cohesive taxonomy, against which I compare each systems in the survey. I feel there is more gain in their approach since they cover more breadth. I am inclined towards their approach if I do a similar project in future.

Q3. How do you defend your taxonomy style?

Q4. Why do you choose this style?

When I started my depth work, I had access to some grid work in managing and analyzing large datasets. Grid literature had a large repository of systems. In contrast, I did not find much literature in cloud computing for analyzing large data sets. However, I did find some recent work for analyzing large datasets in clusters. And I argue that these work can be exported to clouds with little effort. Therefore, I opted to look deeper at these systems.

Q5. How does your work meet the criteria for systematic review?

I came across this term after my depth work. Nonetheless, I still have some comments. My understanding of systematic review is that it has been proposed for healthcare, while I work in computer science. Systemic review suggests defining an appropriate health question prior to literature search, while my research question evolved as I looked through the literature. Systematic-reviews are wider in scope: same level of rigour to reviewing research literature because, for example, to establish clinical and cost-effectiveness of an intervention drug – people’s lives depend on it. The scope of my depth work is to establish that I deeply understand my research. The scope of a study has many consequences: the systematic-review demands some expertise and experience in the area, while I am yet developing both; the systematic-review may be done by a group, while I did most work for my depth paper; the systematic-review is aggressively reviewed by peers, while none of my peer-reviews are experts in cloud computing.

Q6. Why are there only four mechanisms in your workload taxonomy?

I started by looking at workload taxonomy for dbms. That had four functions at the top level. Now, my taxonomy is different to workload dbms taxonomy except one branch i.e. workload characterization. The taxonomy is different because dbms workload and cloud workload can be different. Anyway, what I’m trying to say is the number four is not unusual for workload taxonomy.

Q7. You claim that recent systems can be exported to clouds with little effort. Why? Substantiate your claim?

That’s a basic question, I’m glad you asked it. Answering it well will help convince my audience that my work is indeed relevant for clouds. A cluster is a set of physical servers. Each server has a set of devices such as processors, memory, hard-disks etc., and is connected to a high-speed inter-connect network. Clouds, specifically IaaS, export their computing resources as virtual machines. In Amazon EC2, a virtual machine or an instance uses para-virtualization technology. Para-virtualization allows many OS instances to run concurrently on a single physical machine with high performance, providing better use of physical resources and isolating individual OS instances [3]. For all practical purposes, it acts like a physical server. If a technique can work on a cluster of physical machines, why can’t they work on a work on a cluster of VMs. HadoopDB and Elastic MapReduce are examples of my claim.

Q8. Are there any practical difficulties in exporting the scheduling architecture to clouds?

In Amazon EC2 cloud, instances are viewed as temporary objects. You get them, you use them, you let them go. As a result, any persistent data needs to be stored elsewhere e.g. S3 or EBS. Architectures like MapReduce assume the data resides on a server. Therefore, when MapReduce is exported to the clouds, it has to deal with the fact that instances do not keep data locally, and data has to be fetched from persistent storage. Fortunately, an instance experiences roughly the same access rate for EBS and local storage.

Q9. How do you build your taxonomy?

My taxonomy is based on published literature. I use a number of key features that can be used to differentiate among various approaches. Scheduling part of taxonomy is inspired by the scheduling taxonomy for grids. Meanwhile, I build provisioning taxonomy myself by abstracting commonalities between systems that are related to provisioning.

Q10. What do 1st level taxonomy do?

The top layer of the taxonomy contains the four main functions performed as part of workload management.

Q11. Do 1st level taxonomy have different objectives? i.e. orthogonal or not mutually exclusive?

Yes, they are loosely orthogonal. They serve different purpose and can co-exist. I see each one of them playing an important part in an effective workload management system.

Q12. Do 2nd level taxonomy have the same objectives? i.e. orthogonal or not mutually exclusive?

The 2nd level of taxonomy is loosely orthogonal. That is why I discuss a system against all 2nd level nodes. Only leaves can be mutually exclusive. For example, Hadoop does not employ both decoupled and combined scheduling/replication coupling.

Q13. Do you think the taxonomy is too coarse?

That's a good question. It appears so. I define the purpose of taxonomy to classify and evaluate current workload management mechanisms and systems. Let us take the example of scheduling taxonomy. When I look at the taxonomy I see many nodes in the taxonomy. However, when I look at the summary table, I see systems group into a few dense clusters. For example, I would consider Pig Latin and HadoopDB to be considerably different Hybrid MapReduce+DB systems, but my taxonomy puts them into the same cluster. Pig Latin integrates MapReduce and database constructs at the interface level, where as HadoopDB integrates MapReduce and database constructs at the system level. The integration at different levels have different implications. Therefore, I believe they should be classified differently.

Q14. What do arrows and boxes mean in your taxonomy?

My taxonomy fans out. The level of abstraction decreases as we move towards the leaves of taxonomy.

Q15. What do leaves represent in your taxonomy?

The leaves represent concrete implementations of the approaches in surveyed systems.

Q16. Why is the workload management taxonomy so small?

Cloud Computing is a fairly young field about five years or younger. Taxonomy is based on existing work and published literature. As a consequence, workload management in clouds, unlike dbms, is in infancy. Hence, the taxonomy is so brief. In fact, the systems in scheduling survey are primarily cluster based systems that could be exposed to clouds with effort.

Q17. Why is provisioning part of workload management?

Clouds offer the illusion of "infinite resources". Unlike clusters, adding new resources to existing resource set has a lead of minutes rather than weeks. With no upfront capital cost paired with short-term usage, provisioning can be varied during the execution of a workload and deserves to be an integral part of workload management in clouds.

Q18. Is workload characterization simply prioritization?

Workload characterization may include prioritization amongst other things, such as average execution time, average processor or disk utilization.

Q19. What is the difference between technique, architecture and a system?

A technique is a concept, an abstraction, without much details. A system is an real implementation. Architecture lies in between. I see architecture as a technique with blue prints of the implementation, details of the design decisions and choices.

Q20. Why isn't workload characterization/monitoring explored in detail? Couldn't you have used existing literature on workload characterization similar to what you have done with scheduling?

I see workload characterization and monitoring as an important pillars of workload management. I also feel that it should be discussed in some detail for a complete study. I saw much literature on scheduling and provisioning, and I want to pursue research in this area. Therefore, I felt I should explore them first. I was also limited by space of the depth paper and time. I look forward to exploring them for my doctoral thesis.

Q21. Where would caching fit in taxonomy?

One place where caching fits nicely is when combined-scheduling is used with process locality. In exploiting process locality, if the scheduler creates a cache or a replica for one work-unit then subsequent work-units requiring the same data can be scheduled on that host or in its neighborhood. We are dealing with work-units that operate at a coarse-level. Over here, I would consider certain types of replicas as caches.

Q22. Is your scheduling taxonomy comprehensive?

I can see how I could extend the scheduling part of taxonomy. I could add a 2nd level node, called layers with two leaves as interface and system. Nonetheless, I argue it is cohesive as can be seen by reading the survey part of the section. Each system maps to one of the leaves.

Q23. Does the taxonomy contain all the techniques?

No, I do not make that claim. There could be more.

Q24. Why don't you present architectures for the provisioning part of the taxonomy?

I discuss some commercial clouds in the provisioning part, whose architectural details are not available. I could have extracted architectures for open source systems instead of presenting techniques. That's a valid point. Back of my mind, there was always the length of depth paper.

Q25. Are the techniques under provisioning orthogonal or inter-related?

I consider orthogonal as "loosely to mean mutually independent or separated." So, are the provisioning techniques mutually independent? Loosely speaking yes. It is possible to have workload manager that uses scaling and migration in a hybrid-cloud setup.

4 Survey

Q1. What are the grid systems that jointly employ scheduling and provisioning?

- The Falcon scheduler [24] triggers a provisioner component for host increase/decrease. This host variation has also been explored during the execution of a workload hence providing dynamic provisioning.
- The MyCluster project [29] similarly allows Condor or SGE clusters to be overlaid on top of TeraGrid resources to provide users with personal clusters. Various provisioning policies with different trade-offs are explored including dynamic provisioning. The underlying motivation is to minimize the wastage of resources.

Q2. Why can't the grid systems that jointly employ scheduling and provisioning be ported across to clouds?

Let us take examples of grid systems that jointly employ scheduling and provisioning:

- Shortcoming in Falcon: Presently, tasks stage data from a shared data repository. Since this can become a bottleneck as data scales, scheduling exploiting data locality is suggested as a solution.

- Shortcoming in MyCluster: MyCluster is aimed at compute-intensive tasks.

Q3. What was the criteria for selecting a system for your survey?

I target some depth and some breadth in my depth paper. Depth of systems which were most popular and discussed in the mainstream research. I also discuss systems which are not in the lime light to demonstrate breadth of my work. You would see architectures such as stream-processing, which don't have many systems in it. There are systems which deserve attention but are not included in my work such as map-reduce-merge due to lack of time and space in the paper.

4.1 Scheduling

Q1. Why didn't you discuss IBM cloud?

I have requested IBM for some details twice, once on a CASCON and once here at Queen's to a visiting IBM guest but no success so far. I guess, I could have produced something from whatever information is available on the web.

Q2. Do systems surveyed in scheduling exist in clouds?

Most of them have been evaluated in clusters. These systems can be exposed to clouds with little effort. Therefore, I include them in the scheduling survey.

Q3. For dryad, why can't there be a static schedule mapped on top of run-time engine?

Yes, there could be. The dryad paper discusses various dynamic refinements, which is good. However, I see benefits in prediction-revision scheme.

Q4. What are the benefits of prediction-revision scheme?

Q5. You claimed in the description of Dryad that dynamic refinement is often more efficient than attempting a static schedule in advance, so why is SCOPE relevant in SCOPE/Cosmos setup?

That's a good question. I would like draw analogies to compile time optimizations such as dead code elimination and loop unfolding. So, if a workflow has tasks whose results are never used and thrown away that could be eliminated pre-execution, instead of dynamic approach where these optimizations are possibly unexploited.

Q6. What is the difference between application and a Sphere client?

The Sphere is a set of APIs that developers can use in their applications to make use of the Sphere cluster.

Q7. Can you give me an example for an application that uses Sphere?

The paper [10] uses an example application to find brown dwarfs (stellar objects) in 1 billion astronomical images of the Universe from the SDSS.

Q8. what is a nested data model in Pig Latin?

Let me demonstrate this with a simple example. Suppose we want to capture information about the positional occurrences of terms in a collection of documents. We can represent a *term* and a *position* object like:

Term: (term_id, term_string)

Position: (character index)

Position_info: (term_id, document_id, position) one for each occurrence of the term

These are flat objects. A nested object would something like:

Map<document_id, Set<position_info>>
A nested data model is built of nested objects.

Q9. How would you quantitatively compare architectures?

I have come across some work that compare systems in various architectures [4; 5; 23; 27]. My approach would follow their footsteps.

Q10. How do researchers compare architectures?

I would like to talk more about one specific work. For example, Pavlo *et al.* [23] compare Hadoop, and Vertica and dbms-X. The later two are parallel dbms's. They compare them both analytically and qualitatively. They setup these systems on a cluster of 100 nodes. The systems are compared against a series of benchmarks. The first benchmark is the one that Google published for their MapReduce implementation i.e. a distributed grep against a terabyte of data. They measure the data loading time as well as benchmark execution times for each benchmark. They also discuss system-level aspects, such as task start-up times and value of compression, and user-level aspects, such as ease of use and additional tools.

Q11. In the summary table of scheduling, why is there cloud objective function missing?

I have identified one cloud objective from reading the literature i.e. load-balancing. I have put that objective in the summary table, where the system either explicitly claims it, or it is obvious that system performs load balancing.

Parallel dbms's are likely to suffer from execution and data skew in normal execution. So, it is likely some resources are not used well; hence no claim for load-balancing.

For Clustera, clients are seeking work. So, the scheduler cannot ensure the cluster is load-balanced.

Dryad does not claim load-balancing but it may be doing so because of runtime optimization and just-in-time scheduling. But I feel this "may" is not enough to earn a place in load-balancing.

Condor claims load-balancing [12]. For DAGMan/Condor, DAGMan is an external service that only ensures that tasks are executed in the right order. That is, the specified DAG or workflow does not have a mapping of a task to an execution resource. This would allow Condor scheduler to load balance the system.

Typically, Map and Reduce tasks are higher than the number of resources. And MapReduce based systems discussed start redundant tasks as soon as some tasks are finished, so the MapReduce systems are load-balanced.

Q12. Why didn't you discuss map-reduce-merge?

Map-Reduce-Merge is an extension to MapReduce for relational data processing. It adds to MapReduce a Merge phase that can efficiently merge data already partitioned and sorted (or hashed) by map and reduce functions. It is likely to fit under Hybrid MapReduce+DB part of the survey. I should have discussed it.

4.2 Provisioning

Q1. Why is your provisioning part thin?

When I talk about provisioning, I mean resource provisioning just before workload execution or during execution. The literature for this type of provisioning is scarce, especially for data-intensive applications. While I found some sample taxonomies on scheduling, I didn't have much luck finding taxonomy for resource provisioning for workload execution. Since my depth paper is based on existing literature, this is reflective in my depth paper.

Q2. Is there any literature on provisioning?

On provisioning yes and I list some in my survey, but not much on taxonomies for resource provisioning for workload execution. There is some work on taxonomies for resource provisioning for network protocols.

Q3. Isn't surge-computing a special form of scaling?

Scaling is a process of increasing or decreasing the amount of resources allocated during workload execution. While surge computing is a provisioning technique that augments resources of a private cloud with public cloud in times of load spikes. There appears to be some similarity. However, public cloud is under single administrative domain and hybrid cloud is not. Public clouds usually deals with virtual resources, where private cloud may deal with either virtual or physical resources. I restrict the use of the term "scaling" to public clouds only. It is possible to have a workload manager that uses scaling and migration in a hybrid-cloud setup. Therefore, scaling and surge-computing are orthogonal for my purposes.

Q4. Does Amazon EC2 allows you to view underlying hardware?

At the time of writing the paper, I was under the impression that this information is not available to the users. Now, I have come across some literature that indicates that some underlying information may be available such as processor type [26].

Q5. Where is data stored in S3?

Amazon has not published details on the implementation of S3. I think the buckets, storage units for S3, are virtual units of storage that are mapped down to physical media such as hard disks.

Q6. How easy it to compare different techniques?

Q7. If you invent a new technique, how would you compare it to existing techniques?

Q8. How would you compare techniques?

I believe that my provisioning techniques are loosely orthogonal or mutually independent. It is possible to have workload manager that uses scaling and migration in a hybrid-cloud setup. I could compare their "performance" on a dollar-cost/benefit ratio, ease-of-setup, ease-of-use and so on. Would you like me to talk about how would I compare systems within one technique (also see question below)?

Q9. How would you test a technique?

Q10. How would compare systems within one technique?

For scaling, my criteria would be questions like: how quick I can scale-in or out resources, does the technique support data re-balancing, how long does it take for data-rebalancing, and what impact does scaling has over executing workload.

For migration, my criteria would be questions like: how quickly does it migrate, does the technique support data migration, how long data transfer take, and how migration impacts executing workload.

For surge-computing, my criteria is based on speed and accuracy of the overload detection, and questions like: how long does it take before the public cloud starts up and helps with surge. My criteria is also based on the speed and accuracy of the normal load detection, and questions like how long does it take before the resources in public cloud are relinquished or what happens to data in the public cloud.

Q11. How researchers compare existing techniques?

I believe I have advocated provisioning as part of workload management. I have developed a provisioning taxonomy. I plead ignorance if there is any work that does that, for example, compare scaling against migration. Would you like me to talk about how I would compare implementations of a technique?

Q12. What challenges do you anticipate in testing a technique?

Q13. How difficult is it to test a technique?

Recently, I developed a Queuing Network Model (QNM) to predict response times of a workload.

Due to some assumptions, I am having difficulty validating that model on a dbms because many assumptions are undermined, and many factors inter-play. That is, it is difficult to isolate the execution environment for validation. Factors include changing caching affects, really small device utilization by workload such that system utilization cannot be ignored.

Recently, Pavlo *et al.* compare a MapReduce and parallel dbms. They find that setting up the systems require some trial-and-error, and on occasions input from vendors.

Q13. What are reasonable assumptions?

For example, CPU and disk utilization by OS activities can be ignored. Caching affects for a request remains the same whether the request is executed independently, or along with other requests.

Q14. What is a QNM?

A queuing model is used to approximate a real queuing situation or system, so the queuing behaviour can be analyzed mathematically.

Q15. What are some of the difficulties in process-level migration?

There are ‘residual dependencies’ in which the original host machine must remain available and network-accessible in order to service certain system calls or even memory accesses on behalf of migrated processes. For example, local host-dependent calls, calls related to the file system (in the absence of a distributed file system) [19].

Q16. What is application-level restart? Can you give some examples?

In application-level restart, the application state and client connection are lost. Take the example of streaming media server, an application restart would require clients to reconnect and re-submit their viewing requests.

Q17. What is the “right time”, when Xen suspends the VM to copy to the destination the CPU state and the remaining memory pages?

Right time can be defined in terms of some threshold. Clark *et al.* [3] binds the number of rounds of pre-copying by analyzing writable working set (WWS), which is some set of pages that are updated very frequently. Luo *et al.* [13] perform whole VM migration. They follow a similar approach but they also define the maximum number of iterations to avoid endless migration, and if the dirty rate is higher than the transfer rate, the storage pre-copy must be stopped proactively.

Q18. What is a storage cloud?

A cloud where storage is available as a utility. That is, “illusion” of infinite storage on a pay-as-you-go basis, with no upfront commitment.

Q19. Why didn’t you discuss Eucalyptus?

That’s a valid point. I should have discussed it. By the way, eucalyptus is a software, and needs to be installed on machines to get a private cloud. Its not out there like Amazon or Google clouds. If I am to extend my work, I would include it.

Q20. Where does eucalyptus fit in your taxonomy?

Eucalyptus emerged from University of San Diego. It exports a user-facing interface that is compatible with the Amazon EC2 and S3 services. So, I think some discussion on Amazon cloud in the paper would be relevant for Eucalyptus.

5 Conclusions

Q1. What is the difference between data processing and data analysis?

Usually, they are inter-changeable. If used together, data analysis may mean extracting useful information out of data, while data processing may mean transforming data into more useful form. For example, transforming the raw data into structured data for storage in databases, for example.

Q2. What are inherent costs in large data-sets?

Large data-sets costs in: (a) time for data movement and remote access, (b) storage and network resources, and (c) becoming a bottleneck when it swamps the network,

Q3. What are the major discoveries as part of this work?

Q4. What critical commentary would you offer in the systems surveyed?

- I see that elastic systems and large-scale data processing systems are disjoint at present.
- Most of the current work on provisioning in clouds involves scaling and is applied to web applications. These do not involve large-scale data processing.
- Almost all the schedulers for data-intensive systems use a decoupled approach and try to place tasks close to data. However, they do not consider the need to create replicas in the face of increased workload demand and so may overload data resources [25].
- Almost all data-intensive system use just-in-time mapping, where as there are potential benefits in prediction-revision mapping.
- I didn't see many objective functions in my survey. QoS is discussed in the literature, but none of the surveyed systems use it as an objective function.
- None of elastic systems surveyed use a predictive trigger.

Q5. What are the open problems in the cloud?

- Armbrust *et al.* [1] point out that there is a need to create a storage system that could harness the advantage of elastic resources provided by a cloud while meeting existing storage systems expectations in terms of data consistency, data persistence and performance.
- Maintaining data consistency for read/write operations between sites in a hybrid cloud is still an open problem.

Q6. What are the research opportunities?

- Integrate dollar-cost in workload management
- Develop workload management methods that integrate scheduling and provisioning
- Executing data-intensive workloads given an objective function while harnessing elasticity
- Exploring different replication strategies that are independent (decoupled) and that work in concert (combined) with the scheduler.
- Makespan is the prevalent objective function in the survey. Clouds are competitive and dynamic market systems in which users and providers have their own objectives. We therefore believe that objective functions related to cost and revenue, or participants' utilities, are appropriate and require further study. Because the economic cost and revenue are considered by the cloud users and cloud providers, respectively, objective functions and scheduling policies based on them need to be developed.

- Most of the data-intensive platforms use just-in-time scheduling. A system could benefit from prediction-revision mapping techniques that incorporate some pre-execution planning, workflow optimization, heuristics or history analysis. This additional analysis could help in creating an appropriate number of replicas or determining an appropriate amount of resources required for a computation.
- Given the similarities between grids and clouds, the joint techniques for scheduling and provisioning in these systems and related work are worth exploring for their relevance in clouds.
- Study the multiplex/scale-out tradeoff: either multiplex the increased workload across existing resources or resize the resource pool to increase processing capacity.
- Estimating a system capacity plays an important role in the workload management process in dbms, as all controls imposed on the users' requests are based on the system state. It is yet unclear how estimating a system capacity will turn out for a workload execution in a cloud with the resource pool scaling into hundreds and possibly more.
- It is clear that there are several issues that need to be explored and addressed for a data-intensive workload management system in a cloud. It becomes more important when (a) systems need to automatically choose and apply appropriate techniques to manage users' workloads during execution, (b) dynamically estimating available system capacity and executing progress of a running workload, and (c) reducing the complexity of a workload management system's operation and maintenance.

Q7. What is the difference between open problem and a research opportunity?

$P = NP$ is an open problem. People have looked at it, yet unable to determine conclusively whether $P = NP$ or not. However, I use open problem loosely and interchangeably with a research opportunity.

6 Future work

Q1. What critical commentary do you offer for your paper?

If I may rephrase the question to, how would I improve my paper.

Q2. How would you improve your depth paper?

- If I take a helicopter view, my taxonomy is imbalanced. Scheduling part of taxonomy is large, provisioning part has some detail but not much for workload characterization and monitoring. I would work on workload characterization and monitoring parts, building up their taxonomies from the published literature. I would also survey existing systems using the taxonomies and include relevant discussion.
- If I look at the table which summarizes scheduling survey, it can be collapsed. As a result, my existing scheduling taxonomy would appear coarse, since I have dense clusters of systems in the summary table. I would extend my scheduling and provisioning taxonomies, so the resulting clusters of systems in the survey is somewhat balanced.
- I can include more systems in the scheduling and partitioning surveys with the literature published after I have completed my depth paper. I may also have to expand my taxonomies, for example, new objective functions may have appeared.
- There are hardly any diagrams for provisioning survey. I could include diagrams there.
- I would improve layout: table of contents, figures, tables.

Q3. What scheduling algorithms are you considering to schedule work-units to resources? And why?

For now, I use simple scheduling approaches such as First-Come-First-Serve (FCFS). We use simple benchmark such as TPC-C, where the workload can be split into classes. Within each

class, the requests have similar execution time. All the requests need to be executed, so performance measures obtained from a QNM model will be the same regardless of FCFS, Last-Come-First-Serve (LCFS), Priority Scheduling (PS).

Q4. Can any other scheduling algorithms be used?

The book [11] claims that a large number of scheduling algorithms can be represented in QNM, but the above suffice for practical purposes.

7 Mock Defense

Q1. Do you have a test bed?

I don't have a test bed for my depth work since that was primarily a literature review. I am working on a test bed for my research as we speak.

Q2. Are you going to build your own environment or use commercial cloud?

I plan to use Amazon cloud, which is a commercial offering.

Q3. What do resources mean for cloud providers and cloud consumers?

Q4. What is a resource?

In my case, it is usually a computing or a data node. Researchers also use it for other things such as networks and ports.

Q5. What is data process locality?

Scheduling approaches employing data process locality map work-units to resources containing the replica of the data. For example, the MapReduce approach schedules map tasks to machines containing the data that map tasks need. So, the data is local, and no data is transferred over the network.

Q6. What is coupling?

Data may be replicated to improve opportunities for parallelization. Now, the replication process could approach may operate independently of scheduling, or scheduling process can invoke the replication process. The latter is the case when scheduling and replication are coupled.

Q7. Has predictive provisioning not been used in dbms?

Dbms usually have a fixed set of resources for workload execution. Sophisticated techniques have been developed and studied including prediction based, in how to use the "available" resources efficiently. Since the resource set is usually fixed for a workload execution, dbms do not exploit provisioning. Therefore, it is safe to say dbms do not employ any kind of dynamic provisioning or technique for it.

Q8. What does prediction mean for provisioning?

It means predicting the need for provisioning e.g. vary resources or to migrate. For example, if the workload is gradually increasing, prediction may forecast that there would be a need for more resources to maintain the service rate for workload execution.

Q9. What actually do you predict?

In case of provisioning, prediction points a need for provisioning, say, to increase the resource set.

Q10. What cloud layer do your provisioning techniques target?

My provisioning taxonomy and survey are primarily based on IaaS abstraction of the cloud.

Q11. Does grid support elasticity?

Yes it does. I mention two grid works that utilize elasticity. I believe that the successes achieved and lessons learnt in grids can be useful for clouds.

Q12. Do all dbms have a static resources?

Typically the resource set of dbms is fixed, and the workload management mechanisms there try to make the best use of available resource with techniques such as admission control and throttling.

Q13. How do your workload management technique would satisfy SLA?

Say there is an SLA requirement for bank that a transaction needs to be served within 30s. But we are experiencing a response time of 2m. One approach might be to move the bank database in the cloud to a more powerful VM.

8 Report Layout

Q1. Your diagrams are inconsistent in report. Why?

I know. Many diagrams are verbatim copies from the research papers. I copied-pasted to save time, but this would probably be unacceptable for a thesis. If you are unclear about a certain diagram, you can ask me questions.

Q2. Why is there no table of contents, figures, tables?

I know. I wanted to put these in, but I was already way passed the depth paper limit. So, I decided not to. Also, I did put them in to see how many paper looks, I ended up fighting with word, and my figures and text were all over the place, so I took them out.

Q3. Why don't you use latex?

I want to switch to latex, since I don't have a pleasant experience writing my master thesis in word. My supervisor has a strong preference for word. So, I am using word for now.

Q4. Why is your paper so lengthy?

The chapter based on the depth paper is about 25 pages. The depth paper goes in some detail discussing cloud computing and applications suitable for cloud computing that are not there in the chapter. The provisioning survey in the chapter was thin. That has been extended. Arguably, the provisioning part of the depth paper is still weaker than its scheduling counter part.

9 References

- [1] Armbrust, M., Fox, A., Griffith, R., Joseph, A.D., Katz, R., Konwinski, A., Lee, G., Patterson, D., Rabkin, A., Stoica, I., and Zaharia, M., 2010. A view of cloud computing. *Communications of the ACM* 53, 4, 50-58. DOI=<http://dx.doi.org/10.1145/1721654.1721672>.
- [2] Armbrust, M., Fox, A., Griffith, R., Joseph, A.D., Katz, R.H., Konwinski, A., Lee, G., Patterson, D.A., Rabkin, A., Stoica, I., and Zaharia, M., 2009. *Above the Clouds: A Berkeley View of Cloud Computing*. University of California at Berkeley. <http://www.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-28.html>
- [3] Clark, C., Fraser, K., Hand, S., Hansen, J.G., Jul, E., Limpach, C., Pratt, I., and Warfield, A., 2005. Live migration of virtual machines. In *USENIX Association Proceedings of the 2nd Symposium on Networked Systems Design & Implementation (NSDI)* Usenix Assoc, Berkeley, CA, USA, 273-286.

- [4] Dean, J. and Ghemawat, S., 2008. MapReduce: simplified data processing on large clusters. *Communications of the ACM* 51, 1, 107-113.
- [5] Dean, J. and Ghemawat, S., 2010. MapReduce: a flexible data processing tool. *Commun. ACM* 53, 1, 72-77. DOI= <http://dx.doi.org/10.1145/1629175.1629198>.
- [6] Dong, F., 2009. *Workflow Scheduling Algorithms in the Grid*. PhD Thesis Report. School of Computing, Queen's University. <http://hdl.handle.net/1974/1795>
- [7] Dong, F. and Akl, S.G., 2006. *Scheduling Algorithms for Grid Computing: State of the Art and Open Problems*. Queen's University. <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.69.9579&rep=rep1&type=pdf>
- [8] Foster, I., Yong, Z., Raicu, I., and Lu, S., 2008. Cloud Computing and Grid Computing 360-Degree Compared. In *Grid Computing Environments Workshop, 2008. GCE '08*, 1-10.
- [9] Gray, J., 2008. Distributed Computing Economics. *Queue* 6, 3, 63-68. DOI= <http://dx.doi.org/10.1145/1394127.1394131>.
- [10] Gu, Y. and Grossman, R.L., 2009. Sector and Sphere: the design and implementation of a high-performance data cloud. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 367, 1897, 2429-2445. DOI= <http://dx.doi.org/10.1098/rsta.2009.0053>.
- [11] Lazowska, E.D., Zahorjan, J., Graham, G.S., and Sevcik, K.C., 1984. *Quantitative system performance: computer system analysis using queueing network models*. Prentice-Hall, Inc., NJ.
- [12] Litzkow, M.J., Livny, M., and Mutka, M.W., 1988. Condor-a hunter of idle workstations. In *8th International Conference on Distributed Computing Systems.*, 104.
- [13] Luo, Y., Zhang, B., Wang, X., Wang, Z., Sun, Y., and Chen, H., 2008. Live and incremental whole-system migration of virtual machines using block-bitmap. In *Cluster Computing, IEEE International Conference on*, 99-106.
- [14] Mian, R., 2011. *Managing Data-Intensive Workloads in a Cloud (Ph.D. Depth Paper)*. School of Computing, Queen's University, <http://research.cs.queensu.ca/TechReports/Reports/2011-580.pdf>
- [15] Mian, R. and Martin, P., 2012. Executing data-intensive workloads in a Cloud. In *Cluster, Cloud and Grid Computing (CCGrid), 2012 12th IEEE/ACM International Symposium on*, Ottawa, Canada, 758-763.
- [16] Mian, R., Martin, P., Brown, A., and Zhang, M., 2011. Managing Data-Intensive Workloads in a Cloud. In *Grid and Cloud Database Management*, G. ALOISIO and S. FIORE Eds. Springer, Heidelberg.
- [17] Mian, R., Martin, P., and Vazquez-Poletti, J.L., 2012. Estimating Costs of Data-intensive Workload Execution in Public Clouds. In *10th International Workshop on Middleware for Grids, Clouds and e-Science - MGC 2012* ACM, Montreal, Quebec, Canada, submitted. DOI= <http://dx.doi.org/10.1016/j.future.2012.01.008>.
- [18] Mian, R., Martin, P., and Vazquez-Poletti, J.L., 2012. Provisioning data analytic workloads in a cloud. *Future Generation Computer Systems (FGCS)*, 0, in press. DOI= <http://dx.doi.org/10.1016/j.future.2012.01.008>.
- [19] Milojevic, D.S., Douglis, F., Paindaveine, Y., Wheeler, R., and Zhou, S., 2000. Process migration. *ACM Computing Surveys (CSUR)* 32, 3, 241-299.
- [20] Moore, R., Prince, T.A., and Ellisman, M., 1998. Data-intensive computing and digital libraries. *Commun. ACM* 41, 11, 56-62.
- [21] Niu, B., Martin, P., and Powley, W., 2009. Towards autonomic workload management in DBMSs. *Journal of Database Management* 20, 3, 1-17.

- [22] Pandey, S. and Buyya, R., 2010. Scheduling and Management Techniques for Data-Intensive Application Workflows. In *Data Intensive Distributed Computing: Challenges and Solutions for Large-scale Information Management*, T.K. (ED) Ed. IGI Global, USA.
- [23] Pavlo, A., Paulson, E., Alexander, R., Daniel, J.A., David, J.D., Samuel, M., and Michael, S., 2009. A comparison of approaches to large-scale data analysis. In *Proceedings of the 35th SIGMOD international conference on Management of data* ACM, Providence, Rhode Island, USA, 165-178. DOI=<http://dx.doi.org/10.1145/1559845.1559865>.
- [24] Raicu, I., Zhao, Y., Dumitrescu, C., Foster, I., and Wilde, M., 2007. Falkon: a Fast and Light-weight task execution framework. In *Proceedings of the 2007 ACM/IEEE conference on Supercomputing* ACM, Reno, Nevada, 43-55. DOI=<http://dx.doi.org/10.1145/1362622.1362680>.
- [25] Ranganathan, K. and Foster, I., 2002. Decoupling computation and data scheduling in distributed data-intensive applications IEEE Comput. Soc, Piscataway, NJ, USA, 352.
- [26] Schad, J., Dittrich, J., and Quiane-Ruiz, J.-A., 2010. Runtime measurements in the cloud: observing, analyzing, and reducing variance. *Proc. VLDB Endow.* 3, 1-2, 460-471.
- [27] Stonebraker, M., Abadi, D., DeWitt, D.J., Madden, S., Paulson, E., Pavlo, A., and Rasin, A., 2010. MapReduce and parallel DBMSs: friends or foes? *Commun. ACM* 53, 1, 64-71. DOI= <http://dx.doi.org/10.1145/1629175.1629197>.
- [28] Venugopal, S., Buyya, R., and Ramamohanarao, K., 2006. A taxonomy of Data Grids for distributed data sharing, management, and processing. *ACM Comput. Surv. (CSUR)* 38, 1, 123-175. DOI=<http://dx.doi.org/http://doi.acm.org/http://doi.acm.org/10.1145/1132952.1132955>.
- [29] Walker, E., Gardner, J.P., Litvin, V., and Turner, E.L., 2006. Creating personal adaptive clusters for managing scientific jobs in a distributed computing environment. In *Challenges of Large Applications in Distributed Environments, 2006 IEEE*, 95.