Client-Side Resource Management on the Cloud: Challenges, Solutions and Future Directions

Article · October 2015

3 authors:

Marc Frincu
West University of Timisoara
111 PUBLICATIONS **661** CITATIONS
SEE PROFILE

Stéphane Genaud
University of Strasbourg
52 PUBLICATIONS **520** CITATIONS
SEE PROFILE

Julien Gossa
University of Strasbourg
29 PUBLICATIONS **200** CITATIONS
SEE PROFILE

Client-Side Resource Management on the Cloud: Survey and Future Directions

Marc E. Frincu

Viterbi School of Engineering,
University of Southern California,
EEB - 300A, McClintock Avenue, Los Angeles, CA, 90089, USA
E-mail: frincu@usc.edu

Stéphane Genaud

ICube, University of Strasbourg, CNRS,
Pôle API, 300 Blvd S. Brant, 67400 Illkirch France
E-mail: genaud@unistra.fr

Julien Gossa

ICube, University of Strasbourg, CNRS,
Pôle API, 300 Blvd S. Brant, 67400 Illkirch France
E-mail: gossa@unistra.fr

Abstract: More and more attention is focused on cloud computing and on bridging the gap between various providers. Clouds have the benefit of offering pay-per-use on-demand virtualized resources. In this context efficiently scheduling tasks on resources that are heterogeneous in terms of characteristics, diverse in what service level agreements are concerned and elastic based on user demand is of extreme importance. The state of the art for this field has been continuously growing for the last years and has now reached a point in which a comprehensive overview indicating current solutions and ongoing challenges is of extreme importance for researchers trying to address the problem.

This paper aims to offer this analysis from a client-side scheduling perspective in which emphasis is not put on physical resource selection but on task to virtual machine mappings and virtual machine allocation. It provides one taxonomy for the current state of the art and one unified model concerning the various metrics and goals used throughout literature. This model is designed to be sufficiently generic and comprehensive to support most of the future work in the field while being extensible to new needs. Based on the current state we then identify several promising research directions and describe some of the challenges they face.

Keywords: Scheduling; Cloud Computing; Intercloud; Taxonomy; Survey.

Biographical notes: Marc Eduard Frincu has received his B.Sc. and M.Sc. in 2006 and 2008 respectively, from the West University of Timisoara Romania. In 2011 he earned his Ph.D. from the same university in the field of Distributed Systems with the topic "Adaptive Scheduling for Distributed Systems". His main

interests are distributed systems with emphasis on scheduling algorithms for grids and clouds. These are backed by more than 30 papers in conference proceedings and journals. During his Ph.D. thesis he also worked as a junior researcher at the e-Austria Research Institute. He was involved in several European and national research projects for which he designed various scheduling policies. He also visited INRIA centers in Nancy and Lille as an intern and visiting researcher during 2007 and 2010, working on various topics such as modelling grid resources and adaptive scheduling. From 2012–2013 he was a a postdoc inside ICPS team part of University of Strasbourg, France, where he investigated workflow scheduling for cloud systems. He is currently a postdoctoral research associate at the Viterbi School of Engineering at University of Southern California, USA, where he is pursuing research in the fields of cloud computing, big data and smart grids.

Stephane Genaud is an Associate Professor at the engineering school in computer science ENSIIE (France). He received a Ph.D. in Computer Science from Strasbourg University (France) in 1997. From 1998 to 2007 he was an associate professor at Strasbourg University, then was a research scientist in INRIA Nancy - Grand Est center with the AlGorille team-project until 2009. His current research interests are in the resource management and simulation of large-scale distributed and parallel systems.

Julien Gossa received a Ph.D. in Computer Science from the INSA-Lyon (France) in 2007. Since September 2008, he is an Associate Professor at the University of Strasbourg (France), a member of the ICube Laboratory, and associated to the AlGorille team-project. His main interests are related to scientific computation in the cloud, with focus on resource management regarding rent costs.

1 Introduction

Cloud computing (Furht and Escalante, 2010) has revealed to be one of the trends with the biggest impact in the information technology field. It delivers *on-demand* access to almost any application that can be exposed as a service by using *virtualization* technologies. Although this kind of access is not new, the attractiveness of clouds is their *pay-per-use* policy in which users pay as much as they use. This form of utility computing has been also envisioned by earlier technologies such as Grid computing (Foster and Kesselman, 1999).

Seen from a certain perspective clouds are a natural extension of grids in which a fixed infrastructure is running virtual resources on an on-demand pay-per-use basis and users are not interested in where their services are executing as long as they get results. It is perhaps this economical prospect that made them appealing to the market in the first place. The Grid heritage comes not only from the lessons learned by using and managing computing grids but also from the difficulty of porting applications to this new environment and to properly handle virtual resource allocation in order to fit the user needs.

The first type of applications that were ported to clouds were web applications. The choice is not random as the increasingly use of the Internet combined with the fluctuating demand inflicted on websites required an elastic allocation mechanism in order to adjust the load on the machines. This property became known as *elasticity* and represents a key aspect in cloud based provisioning. Elasticity does not come for free and many of its issues fall in the area of application *scheduling* in which *Infrastructure as a Service* (IaaS) providers

need to come up with smarter mappings of *Virtual Machines* (VM) on Physical Machines (PM). These VMs are then used by the upper layer of *Platform as a Service* (PaaS) providers to map applications based on the users' requirements. The entire process is a complex choreography in which the needs of *clients* – i.e., either PaaS/SaaS (Software as a Service) providers or end users – become tangled with the IaaS *providers*' goals. Clients for instance, need their applications to be highly available and efficient while spending the least amount of money, whereas providers desire to maximize profits by attracting as many clients as possible through competitive prices and reliable resources. We distinguish here two types of scheduling, one concerning the provider side with the mapping of VMs to PMs and the second one on the client side with (1) the mapping of tasks on VMs and (2) VM provisioning. Clients usually have several objectives in mind when deploying an application or submitting a job. These include execution, cost, and availability constraints, which usually need to be achieved simultaneously. To make matters even more complicated the *intercloud* paradigm as described by Bernstein et al. (2009) recently came into attention. This concept allows users to span their applications across several clouds in order to increase the application availability or to safeguard their data and logic. It also allows clients to select the "best" and most adapted resources and billing model.

The intercloud is also appealing to smaller cloud providers that do not have the resources of larger players and do need to federate in order to reach a similar level of competitiveness. This new kind of cloud has its own challenges like: defining Service Level Agreements (SLA), security, negotiation, brokering, interoperability, and last but not least, standardization of the previously mentioned. Being aware of the advantages, restrictions and (current) problems clouds pose is a must for any Resource Management System (RMS) engineer if user satisfiability is to be met within a certain level of confidence.

Much work has been done, with ongoing results appearing at a steady pace in the field of cloud scheduling. We therefore consider that there is a sufficient basis for trying to offer a relevant state of the art in order to help researchers identify relevant topics and trends that we believe are of importance for the future research in cloud resource management.

Consequently, the main contribution of this paper is one thorough overview of the main issues and solutions regarding client-side resource management. To the best of our knowledge, this is the first survey work to exclusively focus on the client-side with emphasis on the emerging intercloud. Handling the intercloud various resources is becoming a necessity with research papers already considering multiple billing models and heterogeneous factors such as SLA, latency, and different IaaS/PaaS capabilities across providers. As such the timing is perfect to outline the main achievements in client side resource management from the perspective of the intercloud and to outline the existing challenges and possible research directions. For a broader survey on cloud resource management, we refer the reader to the two following surveys. Manvi and Shyam (2014) browses the state-of-the-art concerned with resource management in IaaS, almost only on the provider side, while Galante and Bona (2012) present a survey of methods in use to provide the elasticity (and related scaling) property in clouds, both on the provider and client-sides. This study overlaps our results on the aspects of scaling although their work present for this field many more details and solutions both from industry and academia.

In our paper, we proceed by presenting the actual scheduling approaches taken by research in the cloud community. Section 2 introduces the main concepts and results found in the literature related to cloud scheduling, which serve as a basis to build a taxonomy of current solutions. We then give in Sect. 3 one model for classifying existing goals used in cloud scheduling. These are later used in Sect. 4 to uniformly represent and unify the

various objectives and metrics used in scheduling. Finally Sect. 5 takes all that has been presented and uses it as a foundation for presenting some of the main future directions we believe are of great value for the client-side resource management.

2 Taxonomy Criteria

The objective of this section is to explain which elements found in the literature guided the construction of the taxonomy we propose, depicted on Fig. 1. The taxonomy, presented in details in the next section, has been devised after a careful analysis of more than one hundred papers related to cloud scheduling. From these, we eliminated those addressing the provider side resource management (unless they only rely on information available on client side) and kept 48 relevant papers. The papers have been evenly published in conferences or journals, in the period 2008 – 2013. We have noticed a constantly increasing interest in client-side resource management over this period, as witnessed by the number of papers targeting the client side: 2008:2, 2009:3, 2010:9, 2011:12, 2012:14,2013:4 (at the time of writing, figures for 2013 were not yet complete) whose starting point seems to be in somewhere around 2010. Correlated with the advent of Amazon EC2 in mid 2006, and Eucalyptus and OpenNebula in 2008 the research community interest had a late start. This can be nonetheless linked to the momentum grid computing had on the community and the time needed by the cloud paradigm to prove itself a success.

In all these papers, we identified the key elements and assumptions made by the authors. Although these elements are different in nature (for instance the type of application, the information known to the scheduling process, or the resolution method used) they form a set of criteria that enables to classify the recent research work in this field. Rather than the criteria themselves, the challenge was more on finding a coherent manner to order those criteria so that it becomes easy for the readers to use the taxonomy.

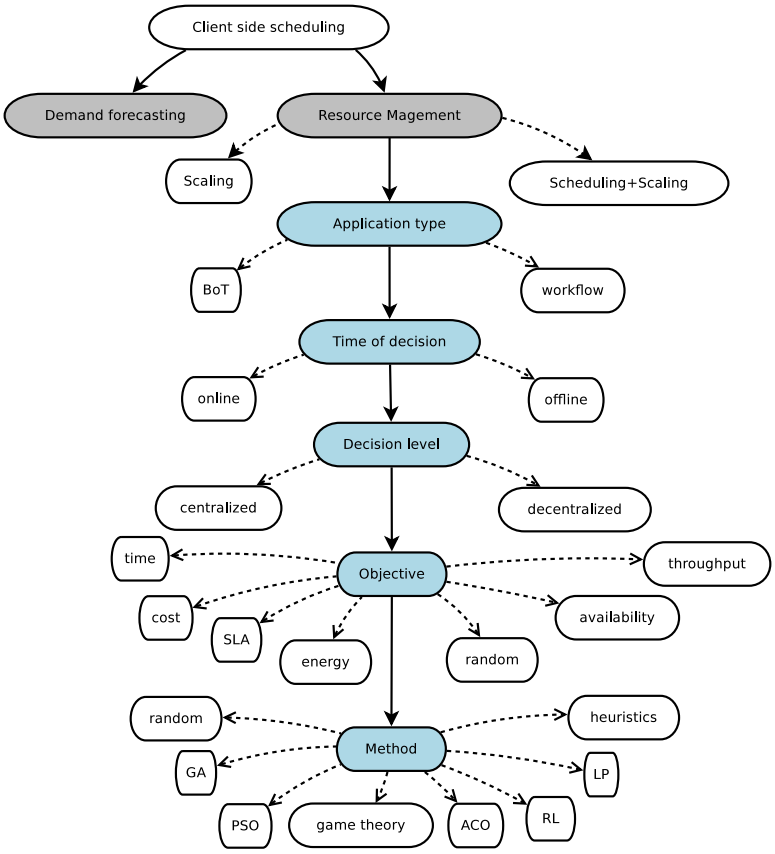
2.1 Demand Forecasting vs. Resource Management

First of all, we consider that the one of the biggest challenges of cloud scheduling is the introduction of scaling besides scheduling. Hence, we argue that the taxonomy should first evidence the objective of the work in terms of scaling only, or simultaneous scaling and scheduling. We term the latter type of objective *Demand Forecasting*, while the former is termed *Resource Management*.

Demand Forecasting

It must be noted that we treat the optional step of *forecasting the demand* as an independent step from scaling. The reason is that we think that the scaling itself consists mainly of strategies for (de)allocating (provisioning) VMs and not the process of forecasting usage demand. This last acts as a trigger for the scaling process and can be substituted by proactive methods such as those used by Rightscale (2014). In the category of demand forecasting, a variety of approaches have been used to try to predict the demand: Kupferman et al. (2009) propose linear regression, Caron et al. (2010) string pattern based matching, Finger et al. (2010) unsupervised learning, and Islam et al. (2012) use neural networks. A comparative study has shown that their efficiency is highly dependent on the trace's characteristics and on the user's objectives as reported in Frincu and Spataru (2012). A decentralized clustering based method for pro-actively provision VMs is depicted in Quiroz et al. (2009).

Figure 1 A taxonomy of client side cloud scheduling



In general, proactive methods rely on patterns in the trace, which can prove difficult to catch unpredictable activities.

Resource Management

We include in this category two approaches. The first one solely focuses on the *scaling* (which in our context is used interchangeable with *provisioning*) of the platform. In a cloud environment scalability also refers to scaling up (i.e. add more processing and disk capabilities to a resource) and scaling out (i.e. provision more resources). The objective of scaling is thus to evaluate what is the configuration and which is the right amount of resources needed to deal with the workload at a given moment. The second approach combines scaling and *scheduling*. Given the inherent elasticity of clouds, scheduling algorithms proposed for clouds must consider different schedules obtained when adding or retrieving resources from the initial platform and select the optimal schedule-platform couple. For this reason, we have omitted a few works considering only a fixed-size set of resources.

2.2 Criteria for Resource Management Solutions

This section categorizes the related literature regarding the colored bullets derived from *Resource Management* (cf. Fig. 1). Our break-down is mostly governed by the top three criteria. The first one discriminates between scheduling-only / scaling-only / simultaneous scheduling and scaling. The second criterion is the application type. We define this based on the task data and not temporal interdependencies. As such we consider two criteria here, bag-of-tasks (BoTs) and workflows depending on whether or not data dependencies exist between tasks. We argue next that most applications can be classified as being either of these and leave out the specific case of message-passing applications, e.g MPI programs for HPC, whose processes start simultaneously and hence scheduling does not apply. While web applications could be disputed as a third category, we consider these to be part of either one or another as explained next. BoTs are applications formed of independent tasks which can run in parallel. To this extent they can be made up of scientific parameter sweep tasks, as in Tordsson et al. (2012), or multiple instances of a single web service running at the same time in order to accommodate a high number of requests or to provide High Availability (HA), as targeted by Bonvin et al. (2011). Contrary to BoTs, workflows are formed of tasks that can be chained to form various execution cyclic or acyclic (DAGs) data flows. These too can be made either of scientific workflows for processing large datasets, as in Caron et al. (2012), or component based web applications (Frincu, 2014). Further the link between web applications and workflows/BoTs is made stronger by (1) standards such as WS-BPEL 2.0 OASIS (2014) which allow the design and deployment of web based workflows (e.g., on-line banking systems), and (2) self-contained web services that process requests independently and simultaneously (e.g., weather forecasting service). When dealing with large input rates and bursts of requests both cases elastically scale and need to be scheduled on the right resources together with the web server container they run on.

The third criterion is the *a priori* knowledge assumed about the system, which determines the approach adopted by the authors: when tasks and resources are known *a priori* the scheduling can be computed *offline* (i.e., statically), whereas the more realistic assumption that tasks and resources can vary dynamically involves the use *online* techniques.

The last criteria corresponds to the decision level, i.e. whether the decision process is centralized/decentralized.

Out of all objectives, client side energy management seemed to be the most difficult to classify in provider or client side since it was difficult to determine which information a client has access to regarding power consumption, CPU speed, etc. Cost and time seem to be the most widely used metrics when considering scheduling, while VM performance is widely used for scaling. Since many users are interested in cost and time efficient execution of their applications the general use of these metrics is only natural. Other metrics such as HA, SLA, and energy are also being used, especially in recent papers indicating an increasing interest in other aspects such as reliability, energy savings, and complex brokering based on SLAs.

3 A Taxonomy for Scaling and Scheduling in the Clouds

We now categorize the approaches followed in the published papers accordingly to the above taxonomy. To ease the reading, the objectives and methods described throughout the text of this section are summarized in two different tables. Table 1 classifies the works according

to the criteria listed in Figure 1, which essentially categorize the assumptions regarding the environment and the chosen method. We provide in the next section (Section 4) another view of the taxonomy which lists the objectives of the works from a quantitative point of view. This view is given in Table 2 and in its companion list of notations in Table 3.

Table 1 Classification of the literature derived from the taxonomy depicted in Fig 1

	References	App. type	Decision time	Decision level	Method
scaling	Calcavecchia et al. (2012)	BoT	online	decentralized	probability heuristics
	Dougherty et al. (2012); Maurer et al. (2013); Bonvin et al. (2011); Calheiros et al. (2012); Espadas et al. (2013)	BoT	online	centralized	greedy
	Quiroz et al. (2009)	BoT	online	decentralized	clustering
	Lucas-Simarro et al. (2013); Tordsson et al. (2012)	BoT	offline	centralized	LP
	Pawluk et al. (2012)	BoT	online	centralized	multi-criteria-objective fct
	Bonvin et al. (2010)	BoT	online	decentralized	greedy
	Xu et al. (2012)	BoT	online	centralized	reinforced learning
	Zaman and Grosu (2013)	BoT	online	centralized	greedy, LP
	Deelman et al. (2008)	workflows	offline	centralized	greedy
	Iqbal et al. (2011)	workflows	online	centralized	greedy
scaling & scheduling	Oprescu and Kielmann (2010)	BoT	online	centralized	dynamic programming
	den Bossche et al. (2010)	BoT	offline	centralized	binary LP
	Marshall et al. (2010); Duong et al. (2011); Genaud and Gossa (2011); Villegas et al. (2012); Li et al. (2012); Nathani et al. (2012); de Assunção et al. (2010); Michon et al. (2012); Vecchiola et al. (2012); Bossche et al. (2013)	BoT	online	centralized	greedy
	Gutiérrez-García and Sim (2013)	BoT	offline	decentralized	greedy
	Frincu (2014)	workflows	online	centralized	GA, LP
	Hwang and Kim (2012); Li et al. (2012); Deelman et al. (2008)	workflows	online	centralized	greedy
	Lin and Lu (2011); Mao and Humphrey (2011); Byun et al. (2011); Caron et al. (2012); Michelle Zhu et al. (2012)	workflows	offline	centralized	greedy
	Wu et al. (2013)	workflows	both	centralized	ACO, PSO, GA

3.1 Scaling

Scaling (i.e. provisioning) is essentially a dynamic process involving adaptation to the ongoing changes in the environment (i.e. users, tasks and resources). Thus online techniques are the most widely adopted solutions for dealing with it. However, a couple of solutions for optimal VM placements obtained by solving linear programming (LP) models assume a static allocation instead, such as in the work from Lucas-Simarro et al. (2013); Tordsson et al. (2012). These provide a theoretical bound to the efficiency of the provisioning and can prove efficient when information about it and the number/size of client requests is known *a priori*.

Most of the investigated papers have as objectives the execution time and cost. Since much of the work on cloud scheduling started by extending grid solutions, the interest in them is understandable. However when considering complex web applications or HPC, new metrics need to be investigated: Dougherty et al. (2012) consider energy, Bonvin et al.

(2010, 2011) target HA, while Bonvin et al. (2011); Maurer et al. (2013) have SLAs as objective.

Recently, Galante and Bona (2012) have given a comprehensive survey on the existing solutions, both academia and industry related, for achieving scalability from both client and provider side. They also identified four main issues that can hinder its implementation: interoperability, availability, granularity (i.e. types of VMs), and start-up time.

3.1.1 Scaling / BoTs

Most of the investigated papers in this study deal with BoTs, although it is not explicitly specified in all cases.

Offline Only two studied papers addressed the problem from an offline point of view. Both Tordsson et al. (2012) and Lucas-Simarro et al. (2013) propose LP solutions to optimally allocate VMs. These two papers target costs and performance, respectively execution time and cost. Both works deal with the intercloud in offline scenarios. It comes as no surprise to find a small number of works in this category since the offline assumption is contrary to the dynamic nature of clouds.

Online Most papers in this category assume that decision making about scaling should be done online. Some papers assume an intercloud environment Calcavecchia et al. (2012); Calheiros et al. (2012); Bonvin et al. (2010, 2011). For example, Calheiros et al. (2012) describe an intercloud environment in which a centralized platform is designed for acquiring VMs through negotiation with other clouds. Another representative example is DEPAS, proposed by Calcavecchia et al. (2012), a framework aimed at provisioning VMs across multiple clouds, and to scale applications according to the probability of the neighborhood load. Among works considering a single cloud, Quiroz et al. (2009) propose a decentralized VM provisioning mechanism based on an analysis of the demand through a clustering method. Tasks are clustered depending on their requirements in VM classes which leads to a provisioning based on the number of requests in each class. Other examples include a multi-tenant VM allocation combined with a load balancing technique. Espadas et al. (2013) propose such a method for web applications following the OnDemand-1-blind (see section 3.2.1), i.e. one VM for each task (also found as part of the provisioning used in Genaud and Gossa (2011); Marshall et al. (2010); Vecchiola et al. (2012)) provisioning method. Energy efficiency, a hot topic nowadays especially for HPC, has been addressed in Dougherty et al. (2012). They give a solution relying on a generalized processor sharing queuing model.

Clouds are suited for adaptive scaling. A reinforced learning method has been proposed in Xu et al. (2012). The authors present an online throughput optimization based on VM reconfiguration. On the other hand, Maurer et al. (2013) propose a sophisticated runtime case based reasoning mechanism for guaranteeing the SLA.

Game theory has also been used to provision VMs based on dynamic pricing schemes. In Zaman and Grosu (2013) two auction-based policies are presented: CA-LP and CA-Greedy. The idea is to bid on bundles of VMs such that the client's cost is maximized. Compared to a fixed pricing scheme results show that CA-LP obtains higher revenue and resource utilization on systems with lower number of users. The reason is the limitation of the LP algorithms. In contrast CA-Greedy is applicable to systems with large number of users (100,000) producing similar results.

3.1.2 Scaling / Workflows

The few papers dealing with workflow scaling alone might indicate that the problem does not raise much questions separated from scheduling since workflows require ordering of tasks and proper scheduling besides mere provisioning VMs. Both Deelman et al. (2008) and Iqbal et al. (2011) rely on centralized online heuristics based provisioning targeting execution time, cost and SLA respectively. Deelman et al. (2008) was among the first authors to study the cost, in terms of budget and execution time, of porting scientific applications to clouds in three scenarios: only when the number of local resources is insufficient, when data is kept locally and the application is deployed on the cloud, and finally when both data and application are deployed on the cloud. Iqbal et al. (2011) target web applications and consider the satisfaction of the maximum average response times as SLA metric. The proposed method is able to reactively upscale and predictively downscale VMs. They do not consider rent costs in their experiments.

3.2 Scaling and Scheduling

Papers dealing with both scaling and scheduling are evenly divided between BoTs and worfklows. This provides a clue that the researchers' interests lie in studying both cases under realistic assumptions involving both VM provisioning and task scheduling. It also shows that when considering clouds, scalability and scheduling often must not be separated. Only but a few papers address the problem from an offline point of view. This is noticed especially for workflows where we notice that most work considers that information about execution times and deadlines is known *a priori* and that clouds are used only when local resources are exhausted.

3.2.1 Scaling and Scheduling / BoTs

Offline Only a couple of papers deal with offline scaling and scheduling of BoTs. These include the work from den Bossche et al. (2010) where LP is used to schedule tasks using deadline and cost constraints, and Gutiérrez-García and Sim (2013) who present in their work a novel agent based scheduling platform for BoTs and no less than 14 scheduling heuristics. These range from a completely random mapping to ones using several combinations of minimum and maximum remaining VM allocation times. Some of the policies are similar to the ones proposed by Genaud and Gossa (2011). They aim to optimize throughput and cost. It must be noted that not all of their heuristics are offline.

Online Online approaches address the additional difficulty to provision an appropriate number of VMs after a load change. A number of heuristics have been proposed in the literature. In the following, we use generic strategy names to refer to strategies that we found to be almost identical in different papers, although named differently.

The simplest one (OnDemand-1-blind) starts a new VM for each job regardless of the platform state. This approach is used by Genaud and Gossa (2011); Marshall et al. (2010); Vecchiola et al. (2012). A straightforward optimization consists in reusing an idle VM still running (OnDemand-1-P), used by Genaud and Gossa (2011); Duong et al. (2011); Villegas et al. (2012). Another heuristic (OnDemand-Steady), designed for regular and moderate loads, consists in keeping a spare VM can be found in Marshall et al. (2010) and Duong et al. (2011). It avoids the trashing that would happen with previous strategies when a job arrives shortly after the last VM has been terminated. Observing the number of jobs waiting in the

queue or their waiting times enables alternative strategies (OnDemand-ExecTime) to refine the number of VMs to start. It has been studied by Marshall et al. (2010); Duong et al. (2011) and Villegas et al. (2012), where the number of VMs to boot has been empirically set to $W/2b$, W and b being respectively the total wait time of jobs in queue and the average boot time of the VM. A variant based on a mix of the number of jobs, waiting time, and remaining run times has been proposed by Duong et al. (2011). An interesting aspect concerning online scheduling and scaling is that of comparing the efficiency of those various VM provisioning techniques, as done by Michon et al. (2012) and Villegas et al. (2012) for BoTs.

An exception to heuristic-based solutions is the exact approach presented in Oprescu and Kielmann (2010), in which a BoT is scheduled onto a set of (known) available VMs so as to maximize the speed within a budget. The problem is expressed as a bounded knapsack problem: pick a number of items of different type (machines in different clusters), each characterized by a profit (machine speed) and a weight (cost) so that the total cost of selected objects does not exceed the capacity (budget). The problem is solved using dynamic programming. This work falls into the semi-online category since it uses a moving average of the task execution time and updates the allocation iteratively to check if it still fits into the budget.

The system performance is a key metric when considering scaling as it provides a clue on the efficiency of the deployed VMs. Nathani et al. (2012) propose an online centralized policy for optimizing it. Energy saving is addressed by Li et al. (2012). The authors rely on a technique for minimizing the number of active servers since they consume more energy than the idle ones. At the same time they aim to minimize execution time. Scaling has also been proposed to extend the capacity of clusters. According to de Assunção et al. (2010), backfilling techniques used in batch-scheduling systems can be adapted to deal with environments mixing dedicated cluster and cloud based resources. Their work evidence situations in which scaling and scheduling in this type of environment improves the application average response time and cost.

3.2.2 Scaling and Scheduling / Workflows

Regarding workflows it seems that offline vs. online approaches gather equal interest. Most of the offline cases are extensions of known grid oriented algorithms. We also noticed that none of the investigated articles focused on decentralized solutions.

Offline Caron et al. (2012) propose an algorithm for executing budget constraint non-deterministic workflows, i.e., workflows that contain loops, joins or splits. They propose a method for splitting these into a series of deterministic DAGs each allocated a fraction of the overall budget. SHEFT, a version of the known HEFT (Zhao and Sakellariou, 2003) heuristics for scheduling workflows on grids, is presented in Lin and Lu (2011). The novelty of the proposed algorithm is its ability to scale resources depending on whether or not a task can execute by its estimated finish time. The makespan of SHEFT is then compared with HEFT for over 50,000 randomly generated graphs for large scale, compute and data intensive workflows. Results show that, as the number of tasks increases, SHEFT outperform HEFT in almost all tests. An algorithm called Scaling-Consolidation-Scheduling (SCS), for auto-scaling workflow applications is proposed in Mao and Humphrey (2011). The paper uses a five steps approach in the scheduling process: process bundling, deadline assignment, VM scaling, instance consolidation, and scheduling. The algorithm is then compared with a greedy approach and the Gain algorithm (Sakellariou et al., 2007). A comparison of different techniques is proposed by Wu et al. (2013), who compare Ant

Colony Optimization (ACO), Genetic Algorithms (GA), and Particle Swarm Optimization (PSO) for workflow scheduling. Although there is no definite "best" algorithm, ACO seems to be the best candidate in terms of makespan, cost, and CPU time. Their approach is to divide the scheduling in two layers: service and task scheduling. At service level, a package based random scheduling is performed, while at task level methods involving GA, ACO and PSO are used. Byun et al. (2011) present the Balanced Time Scheduling (BTS) algorithm for scheduling workflows on elastic resources. Its aim is to minimize the number of resources needed to finish the workflow within a given deadline. Their objective is to optimize time and costs. Tests are performed on five real workflow applications as well as on randomly generated ones. Results show that BTS outperforms IterHEFT (a version of HEFT) in overestimating the needed capacity. In Michelle Zhu et al. (2012) a provider side, with applicability on the client side, offline algorithm targeting resource utilization is proposed. The algorithm comprises two phases: first makespan minimization and then the costs are reduced by relaxing task mappings in case the resulted makespan is smaller than the user given deadline.

Online In case of online workflow scheduling, besides time and cost optimization, there are papers dealing with energy saving, preemptable tasks or web applications. The reason why these topics are considered in online scenarios is that these cases usually require runtime adaptations since processor usage and user hit rates change over time.

An online scheduling algorithm for long running component based applications is presented in Frincu (2014). Several objectives are targeted including cost, load and high availability. Results show that the algorithm achieves HA while keeping the resources balanced and at a cost smaller than a round robin approach. In Hwang and Kim (2012) two scheduling policies minimizing VM rent costs for Map-Reduce tasks are proposed: List and First-Fit (LFF) – sorts prices and the corresponding VMs are allocated to map and reduce tasks – and Deadline-aware Tasks Packing – uses the estimated deadline to schedule map tasks.

Li et al. (2012) propose a scheduling method for preemptable workflow tasks on the intercloud. The primary goal is the makespan minimization. The authors also address the problem of energy saving by maximizing resource utilization on active servers. Bossche et al. (2013) recently proposed two online provisioning algorithms for BoTs using modified first-come-first-serve and earliest-deadline-first (EDF) methods. They consider a hybrid cloud, and use public clouds whenever a deadline on the private one is not met for one task (i.e., unfeasible). Two policies are used Unfeasible-to-public (sends the unfeasible task to the public cloud) and Cheapest-to-public (sends the cheapest predecessor of the unfeasible task to the public cloud). The EDF version together with both policies shows to be the most robust and efficient in terms of met deadlines and cost.

4 Objective based Taxonomy

While much work has been done in developing various scheduling and scaling algorithms for client-side resource management each of the papers introduces its own notations and formalism. As a result it is difficult for readers to link together the objectives and have a unitary view. It is the intent of this section to provide a single formalized view of the objectives targeted by each paper which can be used by readers as a starting point when studying the details of each scientific result. We provide unified notations of all the metrics

Table 2 Objectives and metrics used in scheduling/provisioning

		Objective	Formula	Unit	References
scheduling	time	makespan	$\sum_{t_i \in T} et_i$	s	Frincu (2014); Gutiérrez-García and Sim (2013); Liu (2009); Fard et al. (2013); Villegas et al. (2012); Wu et al. (2013)
		wait time	$st_i - t_i^{\beta_3}$	s	Michon et al. (2012); de Assunção et al. (2010)
		deadline	$t_i^{\beta_5}$	s, h	Bittencourt and Madeira (2011); Celaya and Arronategui (2011); Hwang and Kim (2012)
		response time	$R_{total} = \frac{packetsize}{I/O_{in}-inbytes} + \frac{packetsize}{I/O_{out}-outbytes}$	s	Emeakaroha et al. (2010)
	cost	charged cost	$\sum_{t_i \in T} \sum_{vm_j \in allocated(t_i)} \times r_{VMC_j}$	$	Frincu (2014); Gutiérrez-García and Sim (2013); Fard et al. (2013); Villegas et al. (2012); Liu (2009); Wu et al. (2013)
		charged cost	$pay(i,j) = \begin{cases} \frac{eft_{i,s} \times r_s}{eft_{i,j}} & j = b \& ft_{i,j} \leq eft_{i,j} \\ f(s_{i,.}) & j = b \& ft_{i,j} > eft_{i,j} \\ 0 & j \neq b \end{cases}$	$	Fard et al. (2013)
		communication overhead	$\sum_{(i,j) \in \mathcal{D}} (allocated(t_i)_{\alpha_4} + allocated(t_j)_{\alpha_4})$	$	Bittencourt and Madeira (2011)
		economic fitness (balance)	$balance = utility(j) - rent(j)$	n/a	Bonvin et al. (2011); Liu et al. (2010, 2011); Fard et al. (2013)
		availability	$A = 1 - \frac{downtime(j)}{uptime(j)}$	n/a	Emeakaroha et al. (2010); Frincu (2014)
		availability	$A = \sum_i \sum_j conf_i \times conf(j) \times diversity(i,j)$	n/a	Bonvin et al. (2011)
		SLA measurement cost	$\nu C_m + \sum_\alpha \alpha C_u$	$	Emeakaroha et al. (2010)
		number of requests per second	–	msg/s	Ferrer et al. (2012); Gutiérrez-García and Sim (2013); Liu (2009)
		priority	–	n/a	Ghanbari and Othman (2012)
scaling		random	–	n/a	Michon et al. (2012); Villegas et al. (2012)
	cost	charged cost	$\sum_{t_i \in T} g_i(et_i)$	$	Frincu (2014); Gutiérrez-García and Sim (2013); Villegas et al. (2012); Wu et al. (2013)
		actual cost	total $CPUtime$	h	Villegas et al. (2012); Wu et al. (2013)
	performance	VM startup time	–	s	Pawluk et al. (2012)
		VM migration time	–	s	Shen et al. (2011)
		performance	–	MFLOPS	Lucas-Simarro et al. (2013)
		CPU usage	–	%	Frincu (2014); Shen et al. (2011)
		system utilization	$U = \frac{used\_resources}{total\_resources}$	n/a	Nathani et al. (2012)
	energy	energy consumption	$E(Q_j) = \sum_{j=0}^{n} q_{ij} E_j$	J	Dougherty et al. (2012)
		power consumption	$N(f(1)\rho + f(0)(\delta - \rho))$	W	Dougherty et al. (2012)

and formulae used in the various papers in order to compare them. These objectives and metrics are presented in Table 2 together with the most important notations, summarized in Table 3. In addition, we provide some introductory notations in the text below.

A cloud (named "region" by some providers, e.g. Amazon's EC2 regions) is specified by $C = \{\alpha_1, \alpha_2, \alpha_3, \alpha_4\}$, where α_1 represents the geographical location; α_2 contains the collection of heterogeneous allocated VMs; α_3 holds the resource rent model; and α_4 represents the inbound/outbound pricing model.

Each $VM_i \in \alpha_2$ has a set of characteristics $VMC_i = \{p, c, m, s, n, e_i, e_b, r\}$, where p represents the CPU speed, c the number of cores; m the memory size; s the disk space; n the network bandwidth; e_i the energy consumed when idle; e_b energy consumed when busy; and r the rent cost per BTU (Billing Time Unit). For simplicity VMs are assumed to behave as uniform parallel machines. Since we deal with client side scheduling, we do not consider in our model the PM, i.e., we assume that for every VMC_i there exists a suitable PM to host it.

The resource types together with their billing model are contained in α_3. Each resource depending on the number of rented instances and on their properties has a certain rent cost. For example Amazon uses fixed prices for on-demand instances but these vary depending on OS type and employed hardware. However for the reserved instances – used for long term renting, up to one or three years – it uses a large initial billing cost and smaller hourly fees from then on. In addition, as costs increase and start exceeding given thresholds, larger discounts are applicable. A third type of resources used by Amazon is called spot instances.

Table 3 List of the notations used in Table 2

Notation	Explanation
et_i	execution time of task t_i. It depends on the properties of the allocated VMs
st_i	actual start time of task t_i
$allocated(i)$	set of VMs allocated to t_i
$eft_{i,j} = t_i^{\beta 3} + t_i^{\beta 5} - t_i^{\beta 4}$ $ft_{i,j} = st_i + eet_i$ $b \leftarrow VM_{C_j}^b$	estimated completion time of task i on resource j real completion time of task i on resource j best bidder $\in \mathcal{M}$
$s \leftarrow VM_{C_j}^s$ $f(s_i, .)$	second smallest bidder $\in \mathcal{M}$ penalty function $\leq min_j\{c_{i,j}^{real}\}$
$\mathcal{D} = (v, \epsilon)$ $v_{i,j}$	workflow (DAG) of the application edge from t_i to task t_j
$utility(i)$ $rent(i, j)$	the value a task gives to an application the virtual rent paid by a t_i to a vm_j. This should not be confused with the rent cost r_{VMC_i}
$downtime(j)$ $uptime(j)$	time in which vm_j is unavailable (mean time to repair) time in which vm_j is online (mean time between failures)
$conf(i) \in [0, 1]$ $diversity(i, j)$	confidence level of vm_i n-bit number correlated with the geometrical distance between VMs i and j
ν C_m α C_u	total number of measurements cost of one measurement number of undetected SLA violations cost of one undetected SLA violation
$\mathcal{M}$ $n = sum_{C_i \in \mathcal{M}} \mid C_i^{\alpha 2} \mid$ $\overline{m}$ e_i e_b	A set of federated clouds $\{C_1, ..., C_n\}$ total number of VMs average # VMs running tasks energy consumed per unit by idle VMs energy consumed per unit by busy VMs
q_{ij} E_i	$q_{ij} = 1$ if t_j runs on vm_i, otherwise it is 0 energy consumption of t_j
$f(s)$ ρ δ N	power consumed by processor at speed s $\sum_r \beta_r \rho_r$ where r represents a VM type, $\beta = \frac{\#types}{\#VMs}$ and $\rho_r = \frac{meanArrivalRateOfJob}{meanSizeOfJob}$ positive number # VMs

These allow to bid for unused capacity and use it as long as the spot price is below ones bid and the job is unfinished.

The network pricing model α_4 specifies the template for billing incoming and outgoing traffic. For instance Amazon EC2 does not charge incoming traffic but uses different pricing for outgoing traffic depending on the monthly size – no charge if traffic is below 1GB, \$0.120/GB if traffic $\in (1GB, 10TB]$ and so forth. Hence this template should contain enough information to be able to determine the corresponding costs based on the traffic size and type.

An application is seen as a collection T of tasks t_i, each having a set of characteristics $\{\beta_1, \ldots, \beta_5\}$ where β_1 indicates whether preemption is allowed or not, β_2 specifies any precedence relations between tasks, β_3 specifies the desired start time, β_4 indicates any restrictions on execution time, β_5 indicates the deadline. When deployed, each t_i is allocated to one or more VMs: $allocated(i)$.

A cloud scheduling problem usually tries to map tasks on VMs such that a total cost function $g_{max} = max\{g_i(et_i),\ \forall t_i \in T\}$ is minimized. Here et_i represents the execution time of t_i, and $g_i(et_i)$ its associated cost. In the case of cloud computing, it usually represents a multi-objective function made up of the rent cost $\sum_{vm_j} \in allocated(i) \times r_{VMC_i}$ plus other objectives such as makespan, lateness, energy consumption, throughput, etc.

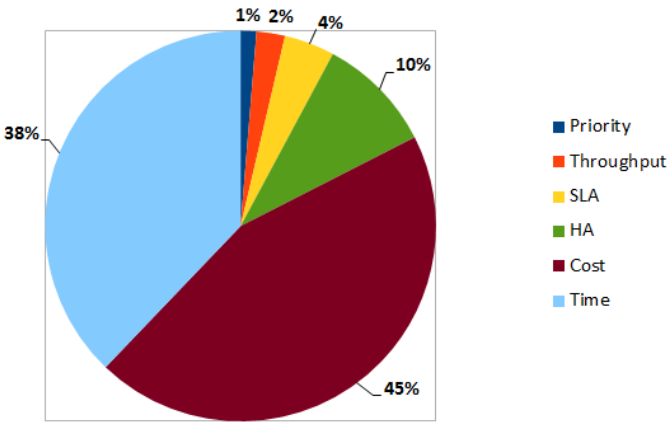
5 Discussion

This section discusses the general trends observed from our analysis of the literature. While the taxonomy itself provides one of the direction research in client-side resource management is heading to, the impact of the published work offers a finer detail. To have a clear picture of the impact the current work has on future research, we compare the number of papers related to a given objective we cite in this survey (indicator termed *papers*) to the number of citations these papers receive globally in the literature, as reported by Google *Scholar* and omitting self-citations (indicator termed *citations*).

Figures 2 and 3 depict the distribution of *citations* depending on the objective. It can be easily noticed that cost and time related objectives dominate the cloud scheduling aspect while cost and VM performance make the bulk of citations for papers dealing with cloud scaling. While other cloud specific objectives such as SLA or HA do not seem to have such a great impact it is only when we look at the correlation between *papers* and their *citations* that the situation becomes clear. Figure 4 presents the median value of citations generated by each objective against the total number of cited papers listing that particular objective. The median value was preferred in order to reduce the impact from the outliers.

From the figure, we notice that while the number of *papers* related to energy efficiency, VM performance, HA, SLA, or scaling cost is relatively small, the *citations* they generate are far greater, indicating an increased interest in the domain for these topics. Contrary, time objectives that have long interested grid researchers, seem to have lost interest as far as the citation impact is concerned.

Figure 2 Impact of research w.r.t the objectives identified in Table 2 and related to cloud scheduling



From these we get a fairly clear image of the direction client-side cloud resource management is heading. Energy seems to gain an increased attention even on the client side. Most of the rented resources use only a fraction of the rent time – usually 20-30% according to Dougherty et al. (2012), and while there are not currently many ways for a provider to make the client energy aware, future approaches could include fluctuating costs based on the data center total gross (number of free data center resources) and net (total actual VM usage) load or pricing schemes that would favor better VM scaling strategies in order to maximize VM usage load. Furthermore energy based models driving the pricing

Figure 3 Impact of research w.r.t the objectives identified in Table 2 and related to cloud scaling

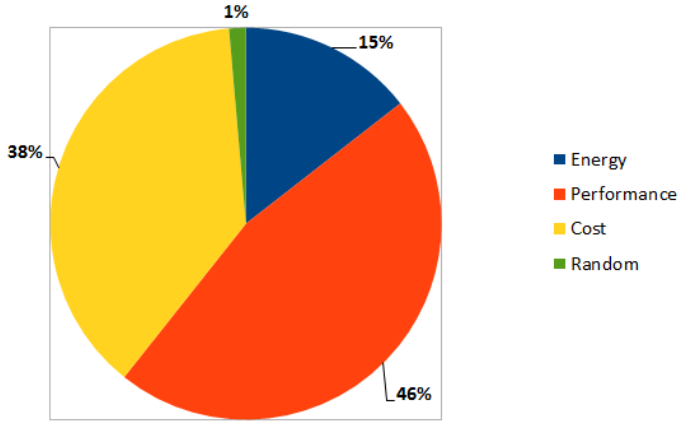
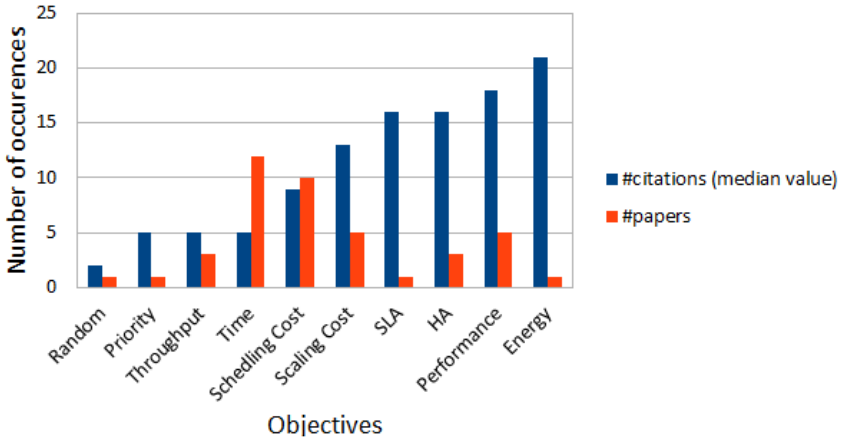


Figure 4 Comparison between the numbers of published papers and the citations they receive regarding objectives (up to oct-2013)



of Amazon's spot instances for example could lead to more energy efficient data centers. While initiated by the provider all of these would have impact on the client's resource management policies.

Moving to the research being done related to specific application types, i.e. BoTs or workflows, we see another pattern emerging, one in which cloud oriented workflows scheduling has been largely restricted to extending existing grid solutions. The complex nature of workflows – long/short running, e.g., web applications vs. scientific experiments, deterministic vs. non-deterministic, data vs. CPU intensive – opens a wide range of questions in terms of providing adequate HA, response times, or throughput depending on their objectives and purpose.

Another future direction for the client side-energy aware cloud scheduling would be to standardize the energy consumption models. This could eventually lead to something similar to what LoM2HiS (Emeakaroha et al., 2010) does for SLA but in the context of

energy consumption. It could eventually lead to the integration of energy in SLAs in a manner which would also bound the client to provider not only the provider to the client. This would ensure in some manner the provider the datacenter energy consumption pattern will not lead to cost increases which in turn will have benefits for the clients too. How this would affect the efficiency of the client's application depends on the usecase and definitely raises many challenging questions. As SLAs begin to play an important role there is still no way of enforcing them and better monitoring methods to ensure their fulfillment are needed. These include more accurate measurements on VM boot times, SLA violations, HA, transfer times across and inside clouds, gathering of traces for user behavior, types of applications. All of these are essential for testing and running an intercloud RMS.

In the future, SLAs and provisioning strategies could become as interdependent as scaling and scheduling are nowadays. One particular interesting aspect would be to have a two way SLA between clients and providers. This would also bind clients to some rules of conduct in terms of energy consumption for instance. Monitoring tools in this respect will need to undergo change in the sense that providers will have to offer clients access to more infrastructure data as currently available.

While the economical aspect of clouds is unquestioned, there is still no way for clients to be reimbursed if SLAs are violated. More solutions from fields such as game theory are still waiting. Reimbursements based on unused time, dynamic pricing based on user loyalty, time of access or sensitivity of data and application, could all be valid in a highly competitive cloud-enabled Internet.

The increasing numbers of cloud providers, the advent of hybrid clouds, the diversification of the billing unit – from hour (e.g., Amazon) to minute (e.g., Google) –, the various privacy and legal issues arising from the fact that clouds are geographically not politically bound, lead undoubtedly to dynamic SLAs in which negotiation will play an important role. Scheduling strategies either at cloud or broker level will have to incorporate them. How these new SLAs will be standardized, what will they contain and how will they impact the efficiency of applications geographically distributed in nature, represent future research directions. In this sense client side resource management will need to go beyond the simple objectives considered nowadays and become semantic and context aware as more dynamic aspects part of our day to day activities begin to be incorporated in cloud systems. Complex ontologies and reasoners on top of them will have to be built and used in order to make the right choices for each cloud user.

We already mentioned above the change in billing policies. While initially we had one per hour billing with either spot or on-demand instances we now have the per minute billing introduced by Google. This changes the way scheduling policies regard pricing. The transition from homogeneous to heterogeneous billing units will add a new dimension to the already heterogeneous pricing scheme based on the type of rented resources.

Another interesting aspect concerning the intercloud is the migration of VMs. Amazon's recent introduction of the import VM feature lets users upload their own custom VM. This degree of freedom together with the adoption of API standards by private cloud providers such as Eucalyptus or OpenStack leads to a whole new approach in terms of doing intercloud. While we current focus on application migration soon we will deal with VM migration and interoperability at VM support infrastructure level. This step will become essential if we think of the emerging big data where gigabytes if not terabytes of data stored across the world will need to be processed in a timely manner. This will lead to a different kind of workflows one in which it is not the data that is passed between tasks but tasks between workflows.

6 Conclusions

In this paper, we provided a comprehensive vie ew on the current state of the art regarding client-side cloud scheduling. We have carefully analyzed the environments, the methods, the objectives, and the metrics used in each paper to come up with the taxonomy proposed in Section 3, depicted in Figure 1. Our first classification criterion reflects the importance given to the resource management in clouds: a vast majority of papers address either scaling alone or simultaneous scheduling and scaling. At the second level of the classification, we place the application type, while the time of decision and decision level are our extra classification criteria. While some would argue that cloud specific user goals like cost, SLA or high availability could act as criteria we see them as being part of the more general category of scheduling objectives. We have further analyzed these objectives and devoted a specific section to these, in which we have unified the metrics and formulae of the original papers. Results are gathered in Table 2.

As a conclusion, we see that client side cloud resource management has evolved much in less than half a decade and given the increasingly number of papers published every year it is likely going to advance even faster. Its specific placement at the border between grids, economy, and Internet, and its numerous open research challenges make it an appealing subject for scientists across different fields of study, opening ways for bold future developments and ideas, some of which have been briefly presented here. They should however not be treated exhaustively but seen as starting points for future research directions.

Finally, the impact of each direction will be translated into degree of adoption by various providers: Some, like negotiation or more dynamic client oriented SLAs, could meet resistance; Others, like energy oriented pricing or standardization, could be more easily embraced especially by smaller providers.

Acknowledgement

This work is partially supported by the ANR project SONGS (11-INFRA-13).

References

David Bernstein, Erik Ludvigson, Krishna Sankar, Steve Diamond, and Monique Morrow. Blueprint for the intercloud - protocols and formats for cloud computing interoperability. In *Fourth International Conference on Internet and Web Applications and Services, 2009. ICIW '09*, pages 328–336, may 2009. doi: 10.1109/ICIW.2009.55.

Luiz Bittencourt and Edmundo Madeira. Hcoc: a cost optimization algorithm for workflow scheduling in hybrid clouds. *Journal of Internet Services and Applications*, 2:207–227, 2011. ISSN 1867-4828. doi: 10.1007/s13174-011-0032-0.

Nicolas Bonvin, Thanasis G. Papaioannou, and Karl Aberer. An economic approach for scalable and highly-available distributed applications. In *Proceedings of the 2010 IEEE 3rd International Conference on Cloud Computing*, CLOUD '10, pages 498–505, Washington, DC, USA, 2010. IEEE Computer Society. ISBN 978-0-7695-4130-3. doi: 10.1109/CLOUD.2010.45.

Nicolas Bonvin, Thanasis G. Papaioannou, and Karl Aberer. Autonomic sla-driven provisioning for cloud applications. In *CCGRID'11*, pages 434–443, 2011.

Ruben Van Den Bossche, Kurt Vanmechelen, and Jan Broeckhove. Online cost-efficient scheduling of deadline-constrained workloads on hybrid clouds. *Future Generation Computer Systems*, 29 (4):973 – 985, 2013. ISSN 0167-739X. doi: 10.1016/j.future.2012.12.012.

Eun-Kyu Byun, Yang-Suk Kee, Jin-Soo Kim, and Seungryoul Maeng. Cost optimized provisioning of elastic resources for application workflows. *Future Generation Computer Systems*, 27(8):1011 – 1026, 2011. ISSN 0167-739X. doi: 10.1016/j.future.2011.05.001.

Nicolò Maria Calcavecchia, Bogdan Alexandru Caprarescu, Elisabetta Di Nitto, Daniel J. Dubois, and Dana Petcu. Depas: a decentralized probabilistic algorithm for auto-scaling. *Computing*, 94 (8-10):701–730, 2012.

Rodrigo N. Calheiros, Adel Nadjaran Toosi, Christian Vecchiola, and Rajkumar Buyya. A coordinator for scaling elastic applications across multiple clouds. *Future Generation Computer Systems*, 28(8):1350 – 1362, 2012. ISSN 0167-739X. doi: 10.1016/j.future.2012.03.010.

Eddy Caron, Frederic Desprez, and Adrian Muresan. Forecasting for grid and cloud computing on-demand resources based on pattern matching. In *Proceedings of the 2010 IEEE Second International Conference on Cloud Computing Technology and Science*, CLOUDCOM '10, pages 456–463. IEEE Computer Society, 2010. ISBN 978-0-7695-4302-4. doi: 10.1109/CloudCom.2010.65.

Eddy Caron, Frédéric Desprez, Adrian Muresan, and Frédéric Suter. Budget constrained resource allocation for non-deterministic workflows on an iaas cloud. In *Algorithms and Architectures for Parallel Processing*, volume 7439 of *Lecture Notes in Computer Science*, pages 186–201. Springer, 2012. ISBN 978-3-642-33077-3. doi: 10.1007/978-3-642-33078-0_14.

Javier Celaya and Unai Arronategui. A highly scalable decentralized scheduler of tasks with deadlines. In *GRID'11*, pages 58–65, 2011.

Marcos de Assunção, Alexandre di Costanzo, and Rajkumar Buyya. A cost-benefit analysis of using cloud computing to extend the capacity of clusters. *Cluster Computing*, 13:335–347, 2010. ISSN 1386-7857. doi: 10.1007/s10586-010-0131-x.

Ewa Deelman, Gurmeet Singh, Miron Livny, G. Bruce Berriman, and John Good. The cost of doing science on the cloud: the montage example. In *SuperComputing'08*, page 50, 2008.

Ruben Van den Bossche, Kurt Vanmechelen, and Jan Broeckhove. Cost-optimal scheduling in hybrid IaaS clouds for deadline constrained workloads. In *IEEE CLOUD*, pages 228–235, 2010.

Brian Dougherty, Jules White, and Douglas C. Schmidt. Model-driven auto-scaling of green cloud computing infrastructure. *Future Generation Computer Systems*, 28(2):371 – 378, 2012. ISSN 0167-739X. doi: 10.1016/j.future.2011.05.009.

Ta Nguyen Binh Duong, Xiaorong Li, and Rick Siow Mong Goh. A framework for dynamic resource provisioning and adaptation in IaaS clouds. In *CloudCom'11*, pages 312–319, 2011.

Vincent C. Emeakaroha, Ivona Brandic, Michael Maurer, and Schahram Dustdar. Low level metrics to high level slas - lom2his framework: Bridging the gap between monitored metrics and sla parameters in cloud environments. In *International Conference on High Performance Computing and Simulation (HPCS'10)*, pages 48–54, July 2010. doi: 10.1109/HPCS.2010.5547150.

Javier Espadas, Arturo Molina, Guillermo Jiménez, Martín Molina, Raúl Ramírez, and David Concha. A tenant-based resource allocation model for scaling software-as-a-service applications over cloud computing infrastructures. *Future Generation Computer Systems*, 29(1):273 – 286, 2013. ISSN 0167-739X. doi: 10.1016/j.future.2011.10.013.

Hammid Mohammadi Fard, Radu Prodan, and Thomas Fahringer. A truthful dynamic workflow scheduling mechanism for commercial multi-cloud environments. *IEEE Transactions on Parallel and Distributed Systems*, 24(6):1203–1212, 2013. ISSN 1045-9219. doi: 10.1109/TPDS.2012.257. published sept 2012.

Ana Juan Ferrer, Francisco Hernández, Johan Tordsson, Erik Elmroth, Ahmed Ali-Eldin, Csilla Zsigri, Raül Sirvent, Jordi Guitart, Rosa M. Badia, Karim Djemame, Wolfgang Ziegler, Theo Dimitrakos, Srijith K. Nair, George Kousiouris, Kleopatra Konstanteli, Theodora Varvarigou, Benoit Hudzia, Alexander Kipp, Stefan Wesner, Marcelo Corrales, Nikolaus Forgó, Tabassum Sharif, and Craig Sheridan. Optimis: A holistic approach to cloud service provisioning. *Future Generation Computer Systems*, 28(1):66 – 77, 2012. ISSN 0167-739X. doi: 10.1016/j.future.2011.05.022.

Marcelo Finger, Germano C. Bezerra, and Danilo R. Conde. Resource use pattern analysis for predicting resource availability in opportunistic grids. *Concurr. Comput. : Pract. Exper.*, 22(3): 295–313, March 2010. ISSN 1532-0626. doi: 10.1002/cpe.v22:3.

Ian Foster and Carl Kesselman, editors. *The grid: blueprint for a new computing infrastructure*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1999. ISBN 1-55860-475-8.

Marc Frincu and Adrian Spataru. Minimizing resource rent loss while maximizing user availability in cloud applications through online switching of the scaling method. In *Proceedings of the 2012 Symposium on Symbolic and Numeric Algorithms for Scientific Computing*, SYNASC '12. IEEE Computer Society, 2012. in print.

Marc Eduard Frincu. Scheduling highly available applications on cloud environments. *Future Generation Computer Systems*, 32:128–153, March 2014. ISSN 0167-739X. doi: 10.1016/j.future.2012.05.017.

Borko Furht and Armando Escalante, editors. *Handbook of Cloud Computing*. Springer US, 2010. ISBN 978-1-4419-6523-3.

Guilherme Galante and Luis Carlos E. de Bona. A survey on cloud computing elasticity. In *Utility and Cloud Computing (UCC), 2012 IEEE Fifth International Conference on*, pages 263 –270, November 2012. doi: 10.1109/UCC.2012.30.

Stéphane Genaud and Julien Gossa. Cost-wait trade-offs in client-side resource provisioning with elastic clouds. In *IEEE CLOUD'11*, pages 1–8, 2011.

Shamsollah Ghanbari and Mohamed Othman. A priority based job scheduling algorithm in cloud computing. *Procedia Engineering*, 50:778 – 785, 2012. ISSN 1877-7058. doi: 10.1016/j.proeng.2012.10.086.

J. Octavio Gutiérrez-García and Kwang Mong Sim. A family of heuristics for agent-based elastic cloud bag-of-tasks concurrent scheduling. *Future Generation Comp. Syst.*, 29(7):1682–1699, 2013. doi: 10.1016/j.future.2012.01.005. published online sept 2012.

Eunji Hwang and Kyong Hoon Kim. Minimizing cost of virtual machines for deadline-constrained mapreduce applications in the cloud. In *GRID'12*, pages 130–138, 2012.

Waheed Iqbal, Matthew N. Dailey, David Carrera, and Paul Janecek. Adaptive resource provisioning for read intensive multi-tier applications in the cloud. *Future Gener. Comput. Syst.*, 27(6):871–879, June 2011. ISSN 0167-739X. doi: 10.1016/j.future.2010.10.016.

Sadeka Islam, Jacky Keung, Kevin Lee, and Anna Liu. Empirical prediction models for adaptive resource provisioning in the cloud. *Future Generation Comp. Syst.*, 28(1):155–162, 2012.

J. Kupferman, J. Silverman, P. Jara, and P. Browne. Scaling into the cloud, 2009. URL `http://www.techrepublic.com/whitepapers/scaling-into-the-cloud/3302611`. (accessed April 7th 2014).

Jiayin Li, Meikang Qiu, Zhong Ming, Gang Quan, Xiao Qin, and Zonghua Gu. Online optimization for scheduling preemptable tasks on IaaS cloud systems. *J. Parallel Distrib. Comput.*, 72(5): 666–677, 2012.

Cui Lin and Shiyong Lu. Scheduling scientific workflows elastically for cloud computing. In *IEEE CLOUD'11*, pages 746–747, 2011.

Ke Liu. *Scheduling Algorithms for Instance-Intensive Cloud Workflows*. PhD thesis, University of Swinburne Australia, 2009.

Shuo Liu, Gang Quan, and Shangping Ren. On-line scheduling of real-time services for cloud computing. In *6th World Congress on Services*, pages 459 –464, july 2010. doi: 10.1109/SERVICES.2010.109.

Shuo Liu, Gang Quan, and Shangping Ren. On-line preemptive scheduling of real-time services with profit and penalty. In *Proceedings of IEEE Southeastcon*, pages 287 –292, march 2011. doi: 10.1109/SECON.2011.5752951.

Jose Luis Lucas-Simarro, Rafael Moreno-Vozmediano, Ruben S. Montero, and Ignacio M. Llorente. Scheduling strategies for optimal service deployment across multiple clouds. *Future Generation Computer Systems*, 29(6):1431—1441, August 2013. ISSN 0167-739X. doi: 10.1016/j.future.2012.01.007.

Sunilkumar S. Manvi and Gopal Krishna Shyam. Resource management for infrastructure as a service (iaas) in cloud computing: A survey. *Journal of Network and Computer Applications*, 41(0): 424–440, 2014. ISSN 1084-8045. doi: 10.1016/j.jnca.2013.10.004. (accepted oct 2013).

Ming Mao and Marty Humphrey. Auto-scaling to minimize cost and meet application deadlines in cloud workflows. In *Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis*, SC '11, pages 49:1–49:12, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0771-0. doi: 10.1145/2063384.2063449.

Paul Marshall, Kate Keahey, and Timothy Freeman. Elastic site: Using clouds to elastically extend site resources. In *CCGRID'10*, pages 43–52, 2010.

Michael Maurer, Ivona Brandic, and Rizos Sakellariou. Adaptive resource configuration for cloud infrastructure management. *Future Generation Computer Systems*, 29(2):472 – 487, 2013. ISSN 0167-739X. doi: 10.1016/j.future.2012.07.004.

Mengxia Michelle Zhu, Qishi Wu, and Yang Zhao. A cost-effective scheduling algorithm for scientific workflows in clouds. In *Performance Computing and Communications Conference (IPCCC), 2012 IEEE 31st International*, pages 256 –265, December 2012. doi: 10.1109/PCCC.2012.6407766.

Etienne Michon, Julien Gossa, and Stéphane Genaud. Free elasticity and free CPU power for scientific workloads on IaaS Clouds. In *18th IEEE International Conference on Parallel and Distributed Systems (ICPADS'12)*, pages 85–92. IEEE, December 2012.

Amit Nathani, Sanjay Chaudhary, and Gaurav Somani. Policy based resource allocation in IaaS cloud. *Future Generation Comp. Syst.*, 28(1):94–103, 2012.

OASIS. Ws-bpel 2.0, 2014. URL `http://ode.apache.org/ws-bpel-20.htmll`. http://ode.apache.org/ws-bpel-20.html (accessed July 16th 2014).

Ana-Maria Oprescu and Thilo Kielmann. Bag-of-tasks scheduling under budget constraints. In *CloudCom*, pages 351–359, 2010.

Przemyslaw Pawluk, Bradley Simmons, Michael Smit, Marin Litoiu, and Serge Mankovski. Introducing stratos: A cloud broker service. In *IEEE CLOUD'12*, pages 891–898, 2012.

Andres Quiroz, Hyunjoo Kim, Manish Parashar, Nathan Gnanasambandam, and Naveen Sharma. Towards autonomic workload provisioning for enterprise grids and clouds. In *Proceedings of the 10 th IEEE/ACM International Conference on Grid Computing (Grid 2009)*, pages 50 – 57, 2009.

Rightscale, 2014. URL `http://www.rightscale.com/`. (accessed April 4th 2014).

Rizos Sakellariou, Henan Zhao, Eleni Tsiakkouri, and Marios D. Dikaiakos. Scheduling workflows with budget constraints. In *in Integrated Research in Grid Computing, S. Gorlatch and M. Danelutto, Eds.: CoreGrid series*. Springer-Verlag, 2007.

Zhiming Shen, Sethuraman Subbiah, Xiaohui Gu, and John Wilkes. Cloudscale: elastic resource scaling for multi-tenant cloud systems. In *Proceedings of the 2nd ACM Symposium on Cloud Computing*, SOCC '11, pages 5:1–5:14, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0976-9. doi: 10.1145/2038916.2038921.

Johan Tordsson, Rubén S. Montero, Rafael Moreno-Vozmediano, and Ignacio M. Llorente. Cloud brokering mechanisms for optimized placement of virtual machines across multiple providers. *Future Gener. Comput. Syst.*, 28(2):358–367, February 2012. ISSN 0167-739X. doi: 10.1016/j.future.2011.07.003.

Christian Vecchiola, Rodrigo N. Calheiros, Dileban Karunamoorthy, and Rajkumar Buyya. Deadline-driven provisioning of resources for scientific applications in hybrid clouds with aneka. *Future Gener. Comput. Syst.*, 28(1):58–65, January 2012. ISSN 0167-739X. doi: 10.1016/j.future.2011.05.008.

David Villegas, Athanasios Antoniou, Seyed Masoud Sadjadi, and Alexandru Iosup. An analysis of provisioning and allocation policies for infrastructure-as-a-service clouds. In *CCGRID'12*, pages 612–619, 2012.

Zhangjun Wu, Xiao Liu, Zhiwei Ni, Dong Yuan, and Yun Yang. A market-oriented hierarchical scheduling strategy in cloud workflow systems. *The Journal of Supercomputing*, 63(1):256–293, January 2013. ISSN 0920-8542.

Cheng-Zhong Xu, Jia Rao, and Xiangping Bu. Url: A unified reinforcement learning approach for autonomic cloud management. *Journal of Parallel and Distributed Computing*, 72(2):95 – 105, 2012. ISSN 0743-7315. doi: 10.1016/j.jpdc.2011.10.003.

Sharrukh Zaman and Daniel Grosu. Combinatorial auction-based allocation of virtual machine instances in clouds. *Journal of Parallel and Distributed Computing*, 73(4):495 – 508, 2013. ISSN 0743-7315. doi: 10.1016/j.jpdc.2012.12.006.

Henan Zhao and Rizos Sakellariou. An experimental investigation into the rank function of the heterogeneous earliest finish time scheduling algorithm. In *Euro-Par 2003 Parallel Processing*, volume 2790 of *Lecture Notes in Computer Science*, pages 189–194. Springer, 2003. ISBN 978-3-540-40788-1. doi: 10.1007/978-3-540-45209-6_28.