Published as:

Paulo Ditarso Maciel Jr., Francisco Brasileiro, Raquel Lopes, Marcus Carvalho and Miranda Mowbray, "Evaluating the Impact of Planning Long-term Contracts on the Management of a Hybrid IT Infrastructure," *in Proceedings of the 12th IFIP/IEEE International Symposium on Integrated Network Management*, pp. 89 – 96, May 2011. DOI: 10.1109/INM.2011.5990678. Abstract online at http://ieeexplore.ieee.org/freeabs_all.jsparnumber=5990678

Evaluating the Impact of Planning Long-term Contracts on the Management of a Hybrid IT Infrastructure

Paulo Ditarso Maciel Jr.*, Francisco Brasileiro*, Raquel Lopes*, Marcus Carvalho* and Miranda Mowbray†

*Universidade Federal de Campina Grande
Departamento de Sistemas e Computação
Laboratório de Sistemas Distribuídos
Av. Aprígio Veloso, s/n, Bloco CO, 58.109-970 Campina Grande - PB, Brazil
E-mail: {pmaciel, fubica, raquel, marcus}@dsc.ufcg.edu.br

†HP Laboratories Bristol
Long Down Avenue, Stoke Gifford, Bristol BS34 8QZ, UK
E-mail: miranda.mowbray@hp.com 0

***Abstract*—The cloud computing market has emerged as an alternative for the provisioning of resources on a pay-as-you-go basis. This flexibility potentially allows clients of cloud computing solutions to reduce the total cost of ownership of their Information Technology infrastructures. On the other hand, this market-based model is not the only way to reduce costs. Among other solutions proposed, peer-to-peer (P2P) grid computing has been suggested as a way to enable a simpler economy for the trading of idle resources. In this paper, we consider an IT infrastructure which benefits from both of these strategies. In such a hybrid infrastructure, computing power can be obtained from in-house dedicated resources, from resources acquired from cloud computing providers, and from resources received as donations from a P2P grid. We take a business-driven approach to the problem and try to maximise the profit that can be achieved by running applications in this hybrid infrastructure. The execution of applications yields utility, while costs may be incurred when resources are used to run the applications, or even when they sit idle. We assume that resources made available from cloud computing providers can be either reserved in advance, or bought on-demand. We study the impact that long-term contracts established with the cloud computing providers have on the profit achieved. Anticipating the optimal contracts is not possible due to the many uncertainties in the system, which stem from the prediction error on the workload demand, the lack of guarantees on the quality of service of the P2P grid, and fluctuations in the future prices of on-demand resources. However, we show that the judicious planning of long term contracts can lead to profits close to those given by an optimal contract set. In particular, we model the planning problem as an optimisation problem and show that the planning performed by solving this optimization problem is robust to the inherent uncertainties of the system, producing profits that for some scenarios can be more than double those achieved by following some common rule-of-thumb approaches to choosing reservation contracts.**

(C) IEEE. Published as: Paulo Ditarso Maciel Jr., Francisco Brasileiro, Raquel Lopes, Marcus Carvalho and Miranda Mowbray, “Evaluating the Impact of Planning Long-term Contracts on the Management of a Hybrid IT Infrastructure,” *in Proceedings of the 12th IFIP/IEEE International Symposium on Integrated Network Management*, pp. 89 – 96, May 2011. DOI: 10.1109/INM.2011.5990678. Abstract online at http://ieeexplore.ieee.org/freeabs_all.jsparnumber=5990678

I. Introduction

Cloud computing has been experiencing a rapid growth, with many commercial companies currently providing on-demand virtual resources such as infrastructure, data storage and software services. In this paper the term “resource” may represent any type of service.

The flexibility to increase and reduce capacities at will, and pay only for resources that are actually used, imply that the use of cloud computing can result in substantial reductions in the costs of running IT infrastructures. However, this is not the only way to reduce costs. Another possible method for gaining access to extra computational resources at low cost is through *peer-to-peer grid computing*, which has been suggested by Cirne et al. [1] as a way to enable a simpler economy for the trading of idle resources. Peers in peer-to-peer (P2P) grids donate their in-house resources to other peers when these resources would otherwise be idle; in return, they receive resources from these peers for free, but with no quality of service guarantees, when these peers are themselves idle.

More generally, one can consider a hybrid IT infrastructure which benefits from both of these strategies for accessing IT resources [2], [3]. In such a hybrid system, computing power can be obtained from in-house dedicated resources, from resources acquired from cloud computing providers, and from resources donated by peers in a P2P grid.

One increasingly popular class of applications which are ideally suited to execute on this hybrid infrastructure is the class of *bag-of-tasks* applications. These are applications that are composed of a large number of independent tasks which can be scheduled in parallel using as many resources as are available. Many applications in science, e-commerce, and several industry sectors, such as pharmaceutical, energy, and engineering, are of this form. We will assume for the rest of the paper that the applications under consideration are bag-of-tasks applications. We will also assume that the applications are *time-constrained*: the utility obtained by the execution of the application decreases with the length of time taken to complete its execution, and is zero if this length is greater than or equal to some fixed value.

In previous work [2], [3], we and other colleagues proposed a business-driven management approach for maximizing the profitability obtained from the execution of time-constrained bag-of-tasks applications in this hybrid setting. Although at the time most cloud computing providers offered only resources on-demand, the authors projected that, as the market increased, providers would need to use a business model that included the possibility of reserving resources for later use. In the cloud computing market, both providers and consumers face risks. While providers face the risk of having too much of their capacity under utilised, consumers face the risk of not having resources available at reasonable prices when they need them. In this respect, reservation contracts are advantageous from the providers' perspective, since they increase the predictability of demand, as well as from the customers' perspective, since they protect against price fluctuations and resource unavailability. We are currently seeing some of the largest players in the market moving in this direction [4], [5]. However, previous work envisaged the use of short-term contracts, negotiated without human intervention. Instead, the major cloud computing providers are offering long-term contracts, typically lasting for a month or a year.

This reality has motivated us to extend our previous work to deal with the problem of planning the capacity of a hybrid IT infrastructure whose contracts established with cloud computing providers are long-term commitments. Similarly to our previous work, we take a business-driven approach and propose a strategy to plan the long-term contracts and to use the reserved resources, in order to maximize the profit obtained from running a given workload (comprising multiple applications) on the infrastructure over a long time period.

The rest of the paper is organized as follows. Related work is discussed in Section II. Section III presents the system model and gives a formal statement of the problem of long-term management of a hybrid IT infrastructure. Capacity planning and run-time scheduling are discussed in Section IV. A performance evaluation is presented in Section V. Finally, Section VI closes the paper with our concluding remarks and directions for future work.

II. RELATED WORK

Several papers propose heuristics for scheduling applications using cloud resources, for instance Pandey et. al. [9] and Silva et. al. [10]. More recently there has also been work on scheduling strategies for the use of *CloudBursts*, in which in-house resources are complemented by resources from cloud providers to address a spike in demand (Marshall et. al. [11], Assunção et. al. [12]). These papers, however, only consider in-house and cloud resources.

Kim et al. [13] and Maciel Jr. et al. [2], [3] propose scheduling strategies for a hybrid infrastructure similar to the one considered in this paper, which in addition includes resources obtained from a grid. Kim et al. propose two strategies. The first aims to finish the application as soon as possible, while the second aims to improve the speedup without exceeding a budget. However, Kim et al. only consider the scheduling process at run-time, and do not consider resource reservation. As mentioned in Section I, our approach is similar to that taken by Maciel Jr. et al., however, while they consider only short-term planning, and one application at a time, we focus on long-term contracts and workloads that comprise many applications to be executed over a long period of time. In other words, the underlying ideas for the model, capacity planning method, and run-time scheduling method are similar to those presented in [2], [3], however unlike those papers we consider long-term planning, with the execution of several applications being planned at once.

III. SYSTEM MODEL AND PROBLEM STATEMENT

We consider the problem from the perspective of a Chief Operations Officer (COO) of a corporation who needs to run a workload $\mathcal{W}$ over a long time period—typically a year. The workload consists of multiple short-lived applications. A set of resource providers ($\mathcal{P}$) are at the disposal of the COO. The COO's job is to decide on the long-term contracts ($\mathcal{K}$) that need to be established with the providers, and the appropriate scheduling ($\mathcal{S}$) to allocate the workload on the available infrastructure, so that the profit achieved is maximized. The best choice for $\mathcal{S}$ will depend on the values of $\mathcal{W}$ and $\mathcal{K}$.

Of course, it is not realistic to expect that the COO will know precisely what the workload will be in the year ahead. We assume that the COO will plan which contracts to establish based on an estimate of this workload, and will adjust the run-time scheduling of applications in response to variations from this estimate. We will show that in the implementation of our model, good performance can be achieved without a very accurate workload estimate.

A. System Model

Let $\{\mathcal{A}_1, ..., \mathcal{A}_m\}$ be the set of applications making up the workload $\mathcal{W}$. We characterize each application $\mathcal{A}_i$ by the tuple $\langle d_i, tr_i, u_i(\cdot)\rangle, 1 \leq i \leq m$, where d_i is the demand of application $\mathcal{A}_i$ (one unit of demand corresponds to one resource unit used for a unit of time); tr_i is the instant of time when the application $\mathcal{A}_i$ is ready for execution; and $u_i(\cdot)$ is the application $\mathcal{A}_i$'s utility function, which specifies the total utility obtained as a function of the amount of time taken to complete the execution of the application. We write tc_i for the time that $\mathcal{A}_i$ completes its execution; the value of tc_i will depend on the contracts and scheduling chosen. The utility gained from the execution of this application will thus be $u_i(tc_i - tr_i)$.

The cost of using a resource is normally calculated over a short time interval, with the use of the resource for a fraction of the time interval being charged the same as if it had been used for the whole interval. We therefore assume that the time period under consideration consists of discrete time intervals of this short length, and that the times tr_i and tc_i are always at the beginning of one of these discrete time intervals. We assume that the time period for which the capacity is first planned, and then used, runs from $t = 1$ to $t = T$, and that the discrete time intervals comprising it are of unit length.

Let $\mathcal{P} = \langle \mathcal{P}_1, ..., \mathcal{P}_n \rangle$ represent the available providers. A single provider $\mathcal{P}_j$ is characterized by a tuple $\langle pr_j, pu_j(t) \rangle, 1 \leq j \leq n$, where pr_j is the fixed reservation fee that provider $\mathcal{P}_j$ charges for the long-term reservation of one resource unit for one unit of time, and for each time t within the reservation period, $pu_j(t)$ is the price that this provider charges at time t for the use of one resource unit for a unit of time. The time-dependence of the usage price may reflect, for example, seasonal pricing.

We represent a capacity plan

$$\mathcal{K} = \langle \langle \mathcal{K}_1(t), b_1, e_1 \rangle, \ldots, \langle \mathcal{K}_n(t), b_n, e_n \rangle \rangle$$

as the contracts established with each of the available service providers, where the function $\mathcal{K}_j(t)$ specifies the number of resource units reserved for use from provider $\mathcal{P}_j$ at time t, for all t such that $b_j \leq t \leq e_j$. (Typically $\mathcal{K}_j(t)$ is constant over time, but we prefer to model it in a more general way.) Thus, at time t, no more than $\mathcal{K}_j(t)$ units of resources may be used from provider $\mathcal{P}_j$ by the contractor; and no resources may be used from this provider at times $t < b_j$ or $t > e_j$.

Notice that the fact that a capacity plan comprises contracts with all providers does not imply that a contract is really established with all providers. A contract $\mathcal{K}_j(t)$ such that $\mathcal{K}_j(t) = 0$ for all t will have a reservation cost equal to zero and will make no resource units available for future use by the contractor, being essentially void. This modelling, however, simplifies the formalization of the problem.

With the model defined above, we are able to define different types of providers, according to the reservation and usage prices specified in their associated contracts. For instance, if $pr_j \neq 0$ and $pu_j(t) \neq 0$, then $\mathcal{P}_j$ provides reserved resources, in which a reservation precedes the usage of resources. On the other hand, if $pr_j = 0$ and $pu_j(t) \neq 0$, then $\mathcal{P}_j$ provides only on-demand resources without requiring previous reservation. In this case, $\mathcal{K}_j(t)$ is the maximum amount of resources that can be acquired from the on-demand provider $\mathcal{P}_j$ at time t. A cloud provider offering both reserved and on-demand resources can be modelled as two providers, one offering only reserved resources and one only online resources. If we assume that the cost of running the in-house IT infrastructure (acquiring hardware and software, housing, power, people etc.) can be evenly amortized over the long-term reservation period, then by setting $pr_j \neq 0$ and $pu_j(t) = 0$ for all t, we can also model the in-house infrastructure as a special type of provider; $\mathcal{K}_j(t)$ in this case is the maximum amount of resources available in-house at time t. Finally, a P2P grid can be modelled as a provider for which $pr_j = pu_j(t) = 0$ for all t, since there is no reservation of resources in the grid and its cost of use is negligible in comparison to the costs for the other providers.

A schedule

$$\mathcal{S} = \{uc_j^i(t), \forall i, j, t | 1 \leq i \leq m, 1 \leq j \leq n, 1 \leq t \leq T\}$$

specifies how the resources available under contract $\mathcal{K}$ are used to compute the workload $\mathcal{W}$; $uc_j^i(t)$ is the number of resources from provider $\mathcal{P}_j$ that will be used to compute application $\mathcal{A}_i$ at time t.

We write $\mathcal{R}(\mathcal{K})$ for the reservation cost of a long-term contract $\mathcal{K}$, $\mathcal{U}(\mathcal{W}, \mathcal{S})$ for the utility achieved by the execution of the workload $\mathcal{W}$ under schedule $\mathcal{S}$, and $\mathcal{C}(\mathcal{W}, \mathcal{S})$ for the cost incurred to compute the workload (that is, the usage cost rather than the reservation cost). We define the profit of the hybrid infrastructure to be the difference between the utility obtained from the execution of the workload and the sum of the cost of provisioning the infrastructure for the long-term period considered and the cost of processing the workload. Thus, the profit ($\wp$) achieved by a given plan $\mathcal{K}$ to process workload $\mathcal{W}$ under schedule $\mathcal{S}$ is given by:

$$\wp(\mathcal{W}, \mathcal{K}, \mathcal{S}) = \mathcal{U}(\mathcal{W}, \mathcal{S}) - [\mathcal{R}(\mathcal{K}) + \mathcal{C}(\mathcal{W}, \mathcal{S})].$$

B. Problem Statement

The problem to be solved is how to optimize the profit gained by running the hybrid IT infrastructure. Generalizing the formulation of the problem given in [2] to the case of multiple applications, this problem can be formalized as follows:

Given $\mathcal{W}$, choose $\mathcal{K}$ and $\mathcal{S}$ so as to maximize

$$\wp(\mathcal{W}, \mathcal{K}, \mathcal{S}) = \mathcal{U}(\mathcal{W}, \mathcal{S}) - [\mathcal{R}(\mathcal{K}) + \mathcal{C}(\mathcal{W}, \mathcal{S})],$$

where

$$\mathcal{U}(\mathcal{W}, \mathcal{S}) = \sum_{i=1}^{m} u_i(tc_i - t_r),$$

$$\mathcal{R}(\mathcal{K}) = \sum_{j=1}^{n} (pr_j \cdot \sum_{t=1}^{T} \mathcal{K}_j(t)),$$

and

$$\mathcal{C}(\mathcal{W}, \mathcal{S}) = \sum_{i=1}^{m} \sum_{j=1}^{n} \sum_{t=tr_i}^{tc_i} (uc_j^i(t) \cdot pu_j(t)),$$

subject to

$$\forall j, t, \sum_{i=1}^{m} uc_j^i(t) \leq \mathcal{K}_j(t)$$

and

$$\forall i, \sum_{j=1}^{n} \sum_{t=tr_i}^{tc_i} uc_j^i(t) = d_i.$$

Note that for a given workload $\mathcal{W}$ and capacity plan $\mathcal{K}$ there may be several schedules $\mathcal{S}$ that are feasible, leading to different utilities and associated costs. Thus, calculating the maximum profit that a given plan $\mathcal{K}$ yields, requires solving another optimization problem to find out the optimal schedule $\mathcal{S}_{opt}(\mathcal{W}, \mathcal{K})$ for plan $\mathcal{K}$ and workload $\mathcal{W}$.

In addition to the complexity brought by the formulation of the problem in terms of a double optimization, there are other difficulties in solving it. The extra difficulties are all related to uncertainties on the information available at the time that the optimizations need to be performed, i.e. at the capacity planning time. First of all, except in very specific cases, it is not possible to anticipate with complete accuracy the workload that will need to be processed over the whole period of interest. Second, the future capacity available from a provider $\mathcal{P}_j$ might not be easy to estimate when resources are not reserved in advance. In more concrete terms, it is not easy to foresee in advance exactly how many resources will be available, either from an on-demand service provider, or from a best-effort P2P grid. Finally, it may not be possible to predict with certainty the future usage prices of resources acquired from the on-demand service providers (i.e. those with $pr_j = 0$ and $pu_j(t) > 0$).

In order to overcome these difficulties we take a simplifying divide-and-conquer approach. We break the solution of the problem in two distinct mechanisms that are executed at different points in time. At capacity planning time, we solve a simplified optimization problem that uses estimates based on the information that is available at that time. Then, at run-time, we optimize the actual execution of the workload, using whatever resources are available at each point in time. Notice that the availability of some of these resources depends on the suboptimal planning that has been performed earlier. The planning mechanism tries to make the best reservations that it can, considering the information that it has, while the scheduling tries to "fix", at run-time, the limitations of the suboptimal planning. The next section describes the suboptimal capacity planning and the run-time scheduling mechanisms that we use.

IV. Capacity Planning and Run-time Scheduling

For the sake of simplicity, we consider a system with a single P2P grid $\mathcal{P}_1$, a single in-house provider $\mathcal{P}_2$, and cloud providers $\mathcal{P}_3$ to $\mathcal{P}_n$ which offer either only reserved resources or only on-demand resources. Due to the uncertainty on the usage prices from the providers that offer on-demand services one year ahead, we use an estimate of these prices based on the knowledge that is available at the time the planning is performed. Solving the planning problem boils down to determining the number of resource units that need to be reserved from each of the providers that offer a reservation service, so as to maximize the expected profit.

We assume that in-house capacity is constant over time and known; we will denote this value $\mathcal{K}_2$. We also assume that the workload $\mathcal{W}$ used to do the capacity planning is a good estimation of the real workload that will be submitted to the infrastructure. If there is a good characterization of the workload distribution for the duration of the long-term period assumed, then $\mathcal{W}$ can be generated from this distribution.

We refer to the peer that is in charge of trading the idle resources from the in-house infrastructure in the P2P grid as the *local* peer. To do the planning we estimate the number of resources available in the P2P grid at the beginning of the interval of time t by the value $b(t)$, which records the aggregate balance at time t of past interactions of the local peer with all other peers in the P2P grid. The balance can be regarded as the amount of resources that the grid owes to the local peer. The value of the balance at the beginning of the interval of time t is the value at the beginning of the previous interval of time, minus the amount of resources consumed from the grid in the previous interval of time, plus the amount of resources donated to collaborators[1]. The balance function $b(t)$ is given by:

$$b(t) = b(t-1) - \sum_{i=1}^{m} uc_1^i(t-1) + (\mathcal{K}_2 - \sum_{i=1}^{m} uc_2^i(t-1)) \cdot p_d$$

where p_d is the grid quality of service, i.e. the probability of there being a demand for resources from the grid at a given time, times the probability that, if a resource is donated, it is donated to a collaborator.

Our planning method proceeds as follows. It considers an estimation of the real workload $\mathcal{W}$ that will be submitted to the infrastructure one year ahead and, each time an application i is ready for execution, the solver finds its best completion time (tc_i), given an input reservation scenario. In other words, the solver seeks for the best completion time for each application, taking into account one-year workload activity and a reservation scenario. Formally, let $\mathcal{W}(t) \subseteq \mathcal{W}$ be the subset of applications that become ready to start at time t, i.e. $\mathcal{W}(t) = \{\mathcal{A}_i | tr_i = t\}$. For each time interval t, the planning runs an optimization procedure that calculates the best completion times (tc_i) for all applications $\mathcal{A}_i \in \mathcal{W}(t)$, and the values of $uc_j^i(t)$ that maximize the profit subject to achieving these completion times. This optimization takes into account only the applications that are ready to execute at the beginning of time interval t. Moreover, the optimization assumes that up to $b(t)$ resources may be reclaimed from the P2P grid at time t and that no resources are used from the grid by these applications in later time intervals. Resources are reserved for time interval t according to the values of $uc_j^i(t)$ found by the optimizer. These values imply which contracts to make with which providers.

The projected scheduling plan is then reassessed at run-time, using the information which is then available; this includes the

[1] A collaborator is a peer that fully participates in the system, consuming and also donating resources. A *free-rider*, on the other hand, is a peer who just consumes resources from the system [6].

actual workload to be processed (at least for the applications that are ready to be executed), the amount of resources that can be attained from both the P2P grid and the on-demand service providers, and the current price of the on-demand resources. The run-time scheduling assumes that the on-demand price of resources does not vary during the time interval over which the application runs. Since we assume that the workload is comprised of short-lived applications, this is a reasonable approximation.

V. Performance Evaluation

In this section we evaluate the performance of the planning and run-time scheduling mechanisms outlined in the previous sections. Firstly, we describe how the model was implemented and how the experiments were conducted. Then, we describe the evaluated scenarios and how the model's parameters were instantiated. At the end of this section we analyse the results.

A. Model Implementation and Experiment Description

We have implemented the model proposed in Section III-A using a solver to evaluate the optimization problem stated in Section III-B. It works as follows. Given a workload $\mathcal{W}$, the set of providers $\mathcal{P}$, and a plan $\mathcal{K}$, the algorithm chooses the best completion time for each application A_i between tr_i and its deadline, considering all resources available. The amount of resources available depends on the amount of resources reserved with the cloud computing providers, as well as the amount reclaimable from the P2P grid. The solver executes the planning algorithm, trying out different options for the amount of resources reserved with each provider in order to find the option that achieves the greatest expected profit.

The experiments are conducted as follows. At planning time, we consider a *baseline* scenario, which represents particular choices for the values assumed by the system parameters (workload demand, grid quality of service, and usage price of on-demand resources). These choices are justified in the next subsection. Then, at run-time we introduce perturbations in the values used in the baseline scenario, to represent the effect of errors in the prediction of the parameters. Thus, the optimization at planning time is executed assuming a given scenario (the baseline), while the optimization at scheduling time is executed for another (a particular perturbation of the baseline). For each perturbed scenario we calculate an optimal plan ($\mathcal{K}_{optimal}$) and associated profit by feeding the planning phase with the perturbed scenario, instead of the baseline. We can then quantify the impact that wrong estimations of the system parameters have on the profit achieved, using the following efficiency metric:

$$\mathcal{E}(\mathcal{K}) = 100 \cdot \frac{\wp(\mathcal{K}))}{\wp(\mathcal{K}_{optimal})},$$

which expresses the profit obtained by using the computed reservation plan as a percentage of the profit obtained by using an optimal plan.

The execution of the experiments' scenarios was carried out in parallel using a 64-core cluster and, for each scenario, the solver took appropriately 5 hours to find the best completion times for each workload's application. Thus, the whole planning process can take less than a work day to decide which reservations to make, considering several reservations scenarios in parallel.

B. Evaluated Scenarios

We assumed a system with 5 providers, comprising a P2P grid, the in-house resources, and 3 cloud computing providers; two offering reserved services and one offering on-demand services.

In order to evaluate the uncertainty that comes from the provider which offers only on-demand services, we modelled errors in the prediction of future prices. We assume the current on-demand price practised by the Amazon EC2 service (USD$\$0.10$) [4], [5] as a baseline, and for the perturbed scenarios we perturb this by prediction error values in the set $\{-20\%, -10\%, 0\%, +10\%, +20\%\}$. One of the providers of reserved services, $\mathcal{P}_4$, charges the same prices as those currently charged by the Amazon EC2 service for the reservation and use of a small virtual machine instance for a period of one year [4], [5], while the other, $\mathcal{P}_3$, has half the usage cost and double the reservation cost. In both cases, the usage price is assumed to be constant over time. We set the capacity $\mathcal{K}_2$ of the in-house infrastructure to be 40% of the average capacity required per unit time interval to run the expected annual workload, and set the in-house reservation cost pr_2 to 5 times the cost per time unit of reserving a small instance from EC2 (pr_4). In other words, we assume that the cost of running the in-house infrastructure for the period of one year is the cost of reserving the same amount of resources from EC2, multiplied by an inefficiency factor of 5. We assume that up to 20 resource units are available from the on-demand provider at each interval of time, following the restrictions imposed by Amazon for the automated negotiation of EC2 instances. We assume that the on-demand usage price is constant over time.

For the P2P grid provider, we assume that at the beginning of each experiment's execution there are no credits left in the grid, i.e. $b(1) = 0$. We set the grid quality of service p_d to 0.72 in the baseline scenario, and use prediction error values $\{-20\%, -10\%, 0\%, +10\%, +20\%\}$ for the perturbed scenarios. The value 0.72 was estimated in Carvalho et al. [7] using data based on application traces.

The workload we have used in the baseline scenario comes from traces of the Grid Workloads Archive [8] (GWA-T3 — NorduGrid). Each scenario uses the trace from a particular group of users, running computing-intensive scientific applications. More specifically, we grouped applications in bag-of-tasks and collected information about tasks' submission times and run-times. We assume that the tasks' run-times are good indications of their computational demand. The workload used in the perturbed scenarios is obtained by introducing a fixed percentage error on the demand d_i of each application $\mathcal{A}_i$ that comprises the workload $\mathcal{W}$. The prediction error values $\Delta\mathcal{W}$ used are in the set $\{-40\%, -20\%, 0\%, +20\%, +40\%\}$.

We assumed that the utility gained by the execution of each application $\mathcal{A}_i$ has a maximum value M_i when $tc_i = tr_i$, and decreases linearly as tc_i increases until it reaches a value of zero at $tc_i = td_i$, where td_i is the deadline for application $\mathcal{A}_i$; in other words,

$$u_i(tc_i - tr_i) = max\{0, M_i \cdot (1 - (tc_i - tr_i)/(td_i - tr_i))\}.$$

In the absence of real data to determine the utility functions, we follow the proposal of C. Lee and A. Snavely [14] for generating synthetic utility functions. More specifically: M_i is set to $(10 \cdot pr_2 \cdot d_i)$, i.e. a maximum utility equivalent to 10 in-house instances times the application's demand; and $td_i - tr_i = d_i$, i.e. if there is just one unit of resource available at tr_i, the application $\mathcal{A}_i$ will finish exactly on the deadline.

C. Results and Analysis

The planning phase considers all possible values for the amount of resource units reserved from the two cloud providers, varying from 0 to 40 from each cloud provider. The output of this phase specifies the amount of resources that should be reserved from providers $\mathcal{P}_3$ and $\mathcal{P}_4$ in order to maximize the expected profit.

Figure 1 is a plot of the profit achieved by all the combinations of values for $\mathcal{P}_3$ and $\mathcal{P}_4$, in experiments in which the baseline scenario is considered. The x and y axes represent the number of resources reserved from providers $\mathcal{P}_3$ and $\mathcal{P}_4$, respectively; while the z axis shows the profit obtained. From Figure 1 we can see the importance of the planning phase, since the profit obtained can vary significantly depending on the amount of resources reserved.

The profit is in fact calculated as the utility received by executing the workload, minus the reservation and usage costs incurred. Explaining the simplest graph first, the reservation cost (Figure 1(d)) increases linearly in both directions as the amount of reserved resources increases. We can also note that the reservation price of provider $\mathcal{P}_3$ impacts more on the final cost since its reservation price is higher than the one for $\mathcal{P}_4$. On the other hand, the usage cost (Figure 1(c)) is impacted more by the use of resources from $\mathcal{P}_4$. Moreover, we can note a discontinuity in Figure 1(c). This happens from the moment it is more profitable to use more resources from providers $\mathcal{P}_3$ and/or $\mathcal{P}_4$ instead of the on-demand provider $\mathcal{P}_5$, which presents higher usage cost. The utility obtained by executing the workload (Figure 1(b)) is flat after a certain amount of reserved resources is reached, because once the whole workload is executed with $tc_i = tr_i$ for all applications A_i, the utility can no longer increase.

For this particular scenario the reservations that the capacity planning algorithm proposed were $(0, 8)$ - in other words, no reservation from $\mathcal{P}_3$, and a reservation of 8 resource units from $\mathcal{P}_4$.

In order to measure the impact of a bad estimation of the workload demand, of the on-demand market price, and of p_d on the planning results, we carried out experiments varying all parameters, as explained in the previous subsection. The values of the efficiency measure for the reservations $(0, 8)$ are shown in Table I. We only show results for scenarios in which the prediction errors of the on-demand price and grid quality of service were in the set $\{0\%, \pm 20\%\}$, since the other results were similar.

Table I

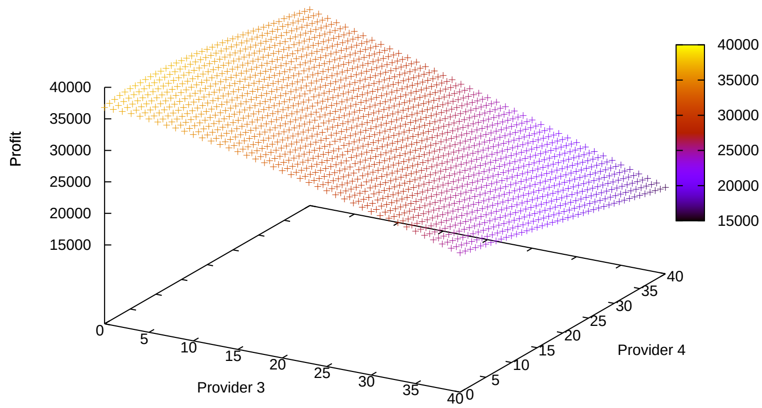
$\mathcal{E}(0,8)$ FOR DIFFERENT DEMAND SCENARIOS, VARYING THE PREDICTION ERRORS IN THE WORKLOAD, ON-DEMAND PRICE AND GRID QUALITY OF SERVICE.

$\Delta\mathcal{W}$	Price Variation of -20%	Price Variation of 0%	Price Variation of $+20\%$
Grid Prediction Error of -20%			
-40%	96.82	96.97	97.12
-20%	99.55	99.62	99.69
0%	100.00	99.98	99.97
$+20\%$	99.75	99.69	99.62
$+40\%$	99.14	99.05	98.94
Grid Prediction Error of 0%			
-40%	96.67	96.81	96.95
-20%	99.46	99.54	99.61
0%	100.00	100.00	100.00
$+20\%$	99.80	99.74	99.67
$+40\%$	99.25	99.16	99.07
Grid Prediction Error of $+20\%$			
-40%	96.02	96.15	96.27
-20%	99.38	99.47	99.56
0%	100.00	100.00	100.00
$+20\%$	99.86	99.80	99.74
$+40\%$	99.33	99.26	99.17

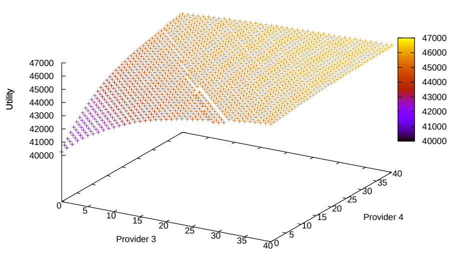
It can be seen from the table that errors in predicting the on-demand price and the grid quality of service have little effect; and that when there is a good estimate of the workload demand, i.e. $\Delta\mathcal{W}$ is small, the planning algorithm gets good results. Negative values of $\Delta\mathcal{W}$ have a stronger impact on efficiency. However, the performance of the proposed algorithm is close to that for the optimal plan in all scenarios evaluated.

We also compared our planning and scheduling strategy against some strategies that use a rule-of-thumb approach instead of a planning process. To do this, we carried out experiments comparing the results of using the reservations $(0, 8)$ with using seven different reservations, $(0, 0)$, $(34, 0)$, $(0, 34)$, $(17, 17)$, $(14, 0)$, $(0, 14)$, and $(7, 7)$. The first of these reserves nothing. The next three each reserve a total of 34 resource units, which is the average capacity per unit time period required to run the expected annual workload, minus the capacity available in-house. The last three each reserve a total of 14 resource units, and assume that 20 will be acquired from the on-demand market. These reservations represent naive but common strategies for reserving resources. Results are shown in Tables II, III and IV. For brevity, these tables only show results for the baseline on-demand price, as in the middle column of Table I.

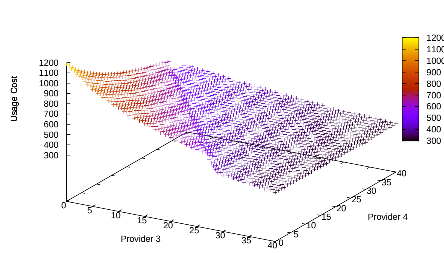
From Table II we can see that the efficiency obtained by the reservations $(0, 8)$ determined by the planning algorithm is higher than the efficiency obtained by reservations $(0, 0)$ in almost all the scenarios evaluated; the exceptions are scenarios where the actual workload is 40% smaller than predicted, and one scenario where it is 20% smaller. In these cases it is possible to achieve high efficiency using only on-demand resources,



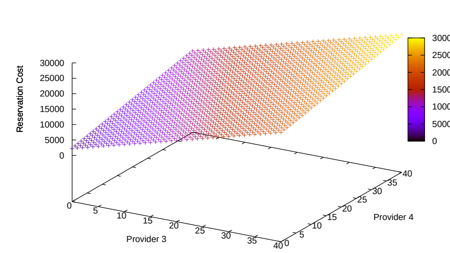
(a) Profit



(b) Utility



(c) Usage Cost



(d) Reservation Cost

Figure 1. Planning results for the baseline scenario.

Table II
$(0, 8)$ RESULTS IN COMPARISON WITH THE STRATEGY THAT DOES NOT RESERVE RESOURCES.

$\Delta\mathcal{W}$	$\mathcal{E}(0,0)$	$\mathcal{E}(0,8)$
Grid Prediction Error of -20%		
-40%	100.00	96.97
-20%	99.46	99.62
0%	98.12	99.98
$+20\%$	93.42	99.69
$+40\%$	91.02	99.05
Grid Prediction Error of 0%		
-40%	100.00	96.81
-20%	99.53	99.54
0%	98.21	100.00
$+20\%$	93.65	99.74
$+40\%$	91.29	99.16
Grid Prediction Error of $+20\%$		
-40%	100.00	96.15
-20%	99.68	99.47
0%	98.35	100.00
$+20\%$	93.88	99.80
$+40\%$	91.59	99.26

with no need for resource reservation. This can be explained by the fact that when the workload demand is smaller, fewer resources are needed to compute the applications. However, in the scenarios where $\mathcal{E}(0,0)$ is greater than $\mathcal{E}(0,8)$, the difference between the two is small.

From Table III, it can be seen that the results for the reservations $(0, 8)$ are all better than for the three strategies reserving 34 resource units. For the scenarios with $\Delta\mathcal{W} = -40\%$, the profits obtained using reservations $(0, 8)$ are more than double those obtained using $(34, 0)$. All the results for strategies that reserved some resources from provider $\mathcal{P}_3$ are worse than those for strategies that only reserved resources from provider $\mathcal{P}_4$. This is because $pr_3 + pu_3 > pr_4 + pu_4$ and $pr_3 > pr_4$, and as a result the total cost of reserving and using a unit of resource from $\mathcal{P}_3$ is always higher than the cost of reserving a unit of resource from $\mathcal{P}_4$ and using it during the same time intervals.

Finally, Table IV shows the results of the planning-based strategy in comparison with the strategies that reserve 14 resource units. As we can see, the results for the $(0, 8)$ reservations are all better than the ones reserving resources from the provider $\mathcal{P}_3$ — $(14, 0)$ and $(7, 7)$ — since this provider has a higher reservation cost. However, the results for the $(0, 14)$ scenario are better in the cases of an underestimate of workload demand. Where workload demand is overestimated, the reservations $(0, 14)$ and $(0, 8)$ both have close to optimal

Table III
$(0,8)$ RESULTS IN COMPARISON WITH THE STRATEGIES THAT RESERVE 34 RESOURCE UNITS.

$\Delta\mathcal{W}$	$\mathcal{E}(34,0)$	$\mathcal{E}(0,34)$	$\mathcal{E}(17,17)$	$\mathcal{E}(0,8)$
		Grid Prediction Error of -20%		
-40%	45.44	77.40	61.58	96.97
-20%	62.20	86.94	74.83	99.62
0%	72.36	92.21	82.53	99.98
$+20\%$	79.54	95.59	87.84	99.69
$+40\%$	83.28	97.67	90.49	99.05
		Grid Prediction Error of 0%		
-40%	45.11	76.82	61.03	96.81
-20%	62.07	86.73	74.64	99.54
0%	72.24	92.04	82.40	100.00
$+20\%$	79.47	95.51	87.76	99.74
$+40\%$	83.18	97.56	90.39	99.16
		Grid Prediction Error of $+20\%$		
-40%	44.49	75.64	60.09	96.15
-20%	61.78	86.38	74.27	99.47
0%	72.14	91.95	82.28	100.00
$+20\%$	79.41	95.42	87.70	99.80
$+40\%$	83.07	97.43	90.26	99.26

efficiency.

Table IV
$(0,8)$ RESULTS IN COMPARISON WITH THE STRATEGIES THAT RESERVE 14 RESOURCE UNITS.

$\Delta\mathcal{W}$	$\mathcal{E}(14,0)$	$\mathcal{E}(0,14)$	$\mathcal{E}(7,7)$	$\mathcal{E}(0,8)$
		Grid Prediction Error of -20%		
-40%	80.03	93.25	86.65	96.97
-20%	87.50	97.80	92.66	99.62
0%	91.26	99.55	95.41	99.98
$+20\%$	93.18	99.97	96.58	99.69
$+40\%$	94.12	99.99	97.06	99.05
		Grid Prediction Error of 0%		
-40%	79.94	92.99	86.47	96.81
-20%	87.36	97.63	92.50	99.54
0%	91.22	99.49	95.36	100.00
$+20\%$	93.19	99.97	96.58	99.74
$+40\%$	94.13	100.00	97.06	99.16
		Grid Prediction Error of $+20\%$		
-40%	79.22	92.01	85.62	96.15
-20%	87.21	97.41	92.31	99.47
0%	91.15	99.41	95.28	100.00
$+20\%$	93.18	99.95	96.57	99.80
$+40\%$	94.13	100.00	97.07	99.26

As we can see from the results, prediction errors of 20% in both the on-demand price and grid quality of service do not impact significantly on profits. Poor estimations of the workload demand can lead to situations where a naive strategy for reserving resources achieves higher profits. However, for all the evaluated scenarios in which where the workload is as expected ($\Delta\mathcal{W} = 0\%$), the proposed approach is more profitable. More importantly, in all scenarios evaluated the profit obtained by the proposed approach is close to optimal, with a maximum shortfall of about 4%.

VI. Conclusions

In this paper, we presented an approach for planning and scheduling the long-term capacity of a hybrid infrastructure. From the results obtained, we conclude that this approach can achieve higher profits than (some) strategies that use simple rules of thumb. Moreover, the proposed algorithm achieves close to optimum profits even when the estimates of workload demand, the usage price of on-demand resources, and the grid quality of service are all out by 20%.

As future work, we intend to better characterize the workload demand in order to instantiate the model's parameters with a larger set of suitable values. Moreover, we intend to evaluate how more accurate information on the available resources can be used at the run-time scheduling to improve efficiency. For instance, Amazon EC2 spot instances could be used to execute the applications and the strategy to bid the spot-price could take into account the current reserved and on-demand prices, as well as the grid quality of service. Finally, we intend to investigate if a mix of long-term and short-term contracts could improve profitability for the hybrid infrastructure.

Acknowledgement

This research was conducted in cooperation with Hewlett-Packard Brasil Ltda. using incentives of the Brazilian Informatics Law (Law nº 8.2.48 of 1991).

References

[1] W. Cirne, F. Brasileiro, N. Andrade, L. Costa, A. Andrade, R. Novaes, M. Mowbray, Labs of the world, unite!!!, Journal of Grid Computing 4 (3) (2006) 225–246. doi:10.1007/s10723-006-9040-x.

[2] P. D. Maciel Jr., F. Figueiredo, D. Candeia, F. Brasileiro, A. Coêlho, On the Planning of a Hybrid IT Infrastructure, in: IEEE/IFIP Network & Operations Management Symposium (NOMS'08), Salvador, Bahia, Brazil, 2008, pp. 496–503.

[3] P. D. Maciel Jr., M. Carvalho, N. Andrade, F. Brasileiro, R. Araujo, D. Maia, R. Miceli, R. Lopes, M. Mowbray, Business-Driven Short-Term Management of a Hybrid IT Infrastructure. Submitted to *JPDC*; available for download at http://lsd.ufcg.edu.br/˜ fubica/short-term.pdf.

[4] Amazon Web Service EC2 Pricing Model [On-line]: http://aws.amazon.com/ec2/#pricing.

[5] W. Vogels, Introducing Amazon EC2 Reserved Instances - A way to further reduce IT costs [On-line]: http://www.allthingsdistributed.com/2009/03/amazon_ec2_reserved_instances.html.

[6] N. Andrade, F. Brasileiro, W. Cirne, M. Mowbray, Automatic Grid Assembly by Promoting Collaboration in Peer-to-Peer Grids, Journal of Parallel and Distributed Computing 67 (8) (2007) 957–966. doi:http://dx.doi.org/10.1016/j.jpdc.2007.04.011.

[7] M. Carvalho, R. Miceli, P. D. Maciel Jr., F. Brasileiro, R. Lopes, Predicting the Quality of Service of a Peer-to-Peer Desktop Grid, in: Proceedings of the 4^{th} Workshop on Desktop Grids and Volunteer Computing Systems (PCGrid 2010), 2010.

[8] A. Iosup, H. Li, M. Jan, S. Anoep, C. Dumitrescu, L. Wolters, D. H. J. Epema, The Grid Workloads Archive, Future Generation Computer Systems 24 (7) (2008) 672–686. doi:http://dx.doi.org/10.1016/j.future.2008.02.003.

[9] S. Pandey, L. Wu, S. Guru, R. Buyya, A Particle Swarm Optimization-based Heuristic for Scheduling Workflow Applications in Cloud Computing Environments, International Conference on Advanced Information Networking and Applications (AINA 2010).

[10] J. N. Silva, L. Veiga, P. Ferreira, Heuristic for Resources Allocation on Utility Computing Infrastructures, 6th International Workshop on Middleware for Grid Computing (MGC 2008).

[11] P. Marshall, K. Keahey, T. Freeman, Elastic Site Using Clouds to Elastically Extend Site Resources, 10th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid 2010).

[12] M. de Assunção, A. di Costanzo, R. Buyya, Evaluating the Cost-Benefit of Using Cloud Computing to Extend the Capacity of Clusters, in: Proceedings of the 18th ACM International Symposium on High Performance Distributed Computing, ACM New York, NY, USA, 2009, pp. 141–150.

[13] H. Kim, M. Parashar, D. Foran, L. Yang, Investigating the Use of Autonomic Cloudbursts for High-Throughput Medical Image Registration, 10th IEEE / ACM International Conference on Grid Computing (Grid 2009).

[14] Cynthia B. Lee, Allan E. Snavely, Precise and Realistic Utility Functions for User-centric Performance Analysis of Schedulers, Proceedings of the 16th International Symposium on High Performance Distributed Computing (HPDC '07), pp. 107–116, Monterey, California, USA, 2007.