Cost-optimized Parallel Computations using Volatile Cloud Resources

Jens Haussmann[1,2], Wolfgang Blochinger[1], and Wolfgang Kuechlin[2]

[1] Parallel and Distributed Computing Group, Reutlingen University, Germany
{jens.haussmann,wolfgang.blochinger}@reutlingen-university.de
[2] Symbolic Computation Group, University of Tuebingen, Germany
wolfgang.kuechlin@uni-tuebingen.de

Abstract. In recent years, the parallel computing community has shown increasing interest in leveraging cloud resources for executing parallel applications. Clouds exhibit several fundamental features of economic value, like on-demand resource provisioning and a pay-per-use model. Additionally, several cloud providers offer their resources with significant discounts; however, possessing limited availability. Such volatile resources are an auspicious opportunity to reduce the costs arising from computations, thus achieving higher cost efficiency. In this paper, we propose a cost model for quantifying the monetary costs of executing parallel applications in cloud environments, leveraging volatile resources. Using this cost model, one is able to determine a configuration of a cloud-based parallel system that minimizes the total costs of executing an application.

Keywords: Cloud computing · Parallel computing · Cost model.

1 Introduction

On-site compute clusters built of commodity hardware are a very popular platform for executing a broad range of HPC applications. However, this type of parallel platform requires considerable upfront investments and furthermore, scalability is limited to static scaling (by manually adding cluster nodes). In the last years, the cloud has emerged to a powerful and versatile platform for building parallel execution environments, establishing a promising alternative to conventional HPC clusters. In particular, cloud computing opens up new opportunities to explicitly control and optimize monetary costs on the level of individual parallel application runs. One can employ typical IaaS (Infrastructure-as-a-Service) cloud offerings to construct (virtual) parallel environments that share many characteristics with traditional on-site compute clusters. This holds especially when using the performance-optimized cloud resources designed for HPC workloads, recently introduced by many cloud providers. By employing a simple "copy & paste" approach, users can substitute their HPC cluster infrastructure with cloud-based virtual clusters, harnessing the on-demand self-service and pay-per-use characteristics of cloud offerings.

The pay-per-use model turns out to be especially beneficial for institutions which otherwise would have to deal with underutilized resources or have a restricted budget that prevents an investment for on-site clusters. Moreover, the *on-demand self-service* characteristic of cloud offerings allows novel execution scenarios. For example, jobs submitted to HPC clusters are typically handled by a scheduling system, stored in a queue and executed later when resources become available. In contrast, virtually unlimited and immediately available resources of cloud environments allow the execution of all jobs in parallel at the same cost but without delays. There already exists a number of activities on utilization of cloud environments for HPC workloads from both industry and academia. Large scale experiments in public cloud environments where over 150k processing units have been utilized give evidence of the feasibility of cloud based HPC [3].

However, there is still a tremendous untapped potential for savings, which can be harnessed for higher cost efficiency. Particularly promising in this context are low priced *volatile* resources, which, however, possess limited availability. Cloud providers offer such resources with significant discounts, in order to prevent idle cloud infrastructures. Users can seize these resources for a fraction of the usual price, under the limitation of having no guarantee of availability. Volatile resources are well suited for a variety of parallel applications. Decisive factors are the degree of coupling and scalability. The better an application meets these properties, the higher is the potential to decrease the costs of executions. Among others, this includes discrete optimization, graph search, constraint satisfaction solving, and MapReduce. Several published studies have considered the utilization of volatile resources for HPC in cloud environments [17], [16], [5]. However, relationship and impact on the cost efficiency is not yet fully understood.

The work presented in this paper is motivated by the following hypothesis: Due to their high pricing, it is not adequate to solely rely on traditional *reserved* cloud resources. We argue that cost-efficient computations require a fine-tuned and balanced execution environment configuration, consisting of both volatile and reserved resources. Which combination of volatile and reserved resources offers the best cost efficiency depends on the characteristics of both application as well as resources. In particular, we make the following contributions: 1) The key research question of our study is to find the number of processors and, more importantly, the concrete type (reserved or volatile) for which the total monetary costs of a parallel computation are minimal. 2) To address this question, we propose a novel cost model for quantifying the monetary costs of parallel computations employing a mix of reserved and volatile processors.

The remainder of the paper is organized as follows: In Section 2, we briefly address the background topics of our work. Furthermore, in Section 3, we discuss in more detail the specific problem we are addressing. Next, Section 4 describes our cost model for parallel computations in cloud environments. Later, in Section 5, we evaluate the cost model and investigate the effects of individual parameters on the total costs. Section 6 gives an overview of related work. Finally, Section 7 concludes the paper and outlines directions for future research.

2 Background

Analytical modeling of parallel systems - Performance metrics of parallel systems (a particular combination of application and architecture) are an essential instrument for evaluation purposes. The most fundamental metrics are the *sequential execution time* T_{seq} and the *parallel execution time* $T_{par}(p)$. While the former is the time required to solve a given problem by the fastest known sequential algorithm, the latter is the time required to solve the same problem in parallel, using p processors. Based on these two fundamental metrics, additional metrics can be derived. *Speedup* $S(p) = \frac{T_{seq}}{T_{par}(p)}$ indicates the performance improvement of solving a problem in parallel over a sequential execution. *Parallel efficiency* $E(p) = \frac{S(p)}{p}$ represents the fraction of processing time spent on essential work. For economic considerations, parallel efficiency is of particular interest, as it indicates the capitalized fraction of the invested processing capacity. Besides the ideal case, parallel systems exhibit *overhead* which manifests itself as processor idling, inter-processor communication, and excess computation. The parallel overhead negatively impacts the parallel efficiency and thus also the *scalability* of a parallel system, which is characterized by $E(p)$ within a range of different numbers of processors p [10].

Volatile cloud resources - More and more cloud providers offer volatile resources, including *spot-instances* of *Amazon EC2* [6], *preemptible VMs* of *Google Compute Cloud* [9], and *low-priority VMs* of *Microsoft Azure* [4]. Although based on the same principle, each provider has a slightly different manifestation of this offering. For example, Amazon employs a sophisticated auction system where resources are claimed through a bidding process. In contrast, Google and Microsoft offer such resources for a fixed price that is significantly lower than the price of traditional resources. In this paper, we assume a prototypical fixed price model for volatile resources, derived from respective cloud offerings of Google and Microsoft. According to the conditions of many providers, we also assume that resources are not withdrawn spontaneously. Prior to resource withdrawal, a signal is emitted that notifies about the imminent retraction. In this way, cleanup operations and state storing are performed on demand, which significantly reduces the overhead. Moreover, there exists a period that guarantees the least time of resource availability, otherwise it is not charged. This also means that a minimum granularity is given for our cost model.

3 Problem Statement

Unlike executing a job on a physical on-site cluster, cloud-based virtual clusters expose direct cost visibility to the users executing jobs, i.e., the charged costs are in direct relation to resources requested. This allows flexible, fine-grained approaches for cost optimization. For example, the user can construct for every run of an application an individually configured cluster that minimizes the cost with respect to the concrete situation in which the results of the computation are needed. However, this flexibility imposes a significant burden on the user. She/he has to determine among a multitude of options the concrete configuration of the cloud-based virtual cluster.

In the first place, the employed number of processors has a significant impact on the resulting costs. An important factor that determines the optimal number of processors with respect to the monetary costs is the scalability of the parallel application at hand: At a certain scale, increasing the number of processors is not *profitable*, i.e., it is not possible to retain adequate benefits from purchasing additional processors, like an increased speedup of computation or a higher processing rate. Basically, the scalability characteristics of the application at hand must be determined by the user. This can, for example, be accomplished by measuring $T_{par}(p)$ for different numbers of processors p, using prototypical input data. Other cost-related parameters that must be considered are the ratio of reserved and volatile processors, the availability of volatile processors, and the prices for both types of processors.

In this paper, we aim to systematically simplify the process of constructing cost-optimal cloud-based clusters by formalizing their costs with a cost-model, resulting in a novel approach for cost optimization of parallel cloud computations. We discuss our cost model in detail in the next section.

4 Cost Model

Basically, a cost model for parallel cloud computing should consider pay-per-use billing and also reflect the two conflicting objectives *fast processing* versus *low monetary costs*. In our previous work, we presented a cost model that applies the concept of *opportunity costs* to model the corresponding trade-off [12]. In our case, the opportunity costs express the lost monetary profits of delayed results from computations. The cost function is given in Equation 1.

$$C(p) = \underbrace{T_{par}(p) * p * c_{\pi}}_{\text{Proc. costs}} + \underbrace{T_{par}(p) * c_{\omega}}_{\text{Opport. costs}} \tag{1}$$

The cloud provider's price for a processor per time unit is denoted by c_{π}, whereas c_{ω} represents the lost monetary profits of delayed results per time unit.

In this work, we extend the model to capture an execution environment consisting of reserved and volatile processors. These processor types do not only differ in terms of their price but also exhibit a different degree of availability. While a processor of type reserved is available at any time, for a processor of type volatile only a limited availability is guaranteed. Figure 1 illustrates an example of a parallel computation, employing both reserved and volatile processors. Given that a volatile processor is not available anymore, it is compensated through a (more expensive) reserved processor. Let v denote the current number of volatile processors employed for computation, and let r denote the current number of reserved processors. The total number of required processors is denoted by p, whereby $p = r + v$ holds true at any time. Referring to a single bar of this chart, we will use the term *configuration*. A configuration is a tuple *(reserved, volatile)* that defines the number of utilized processors of each type.

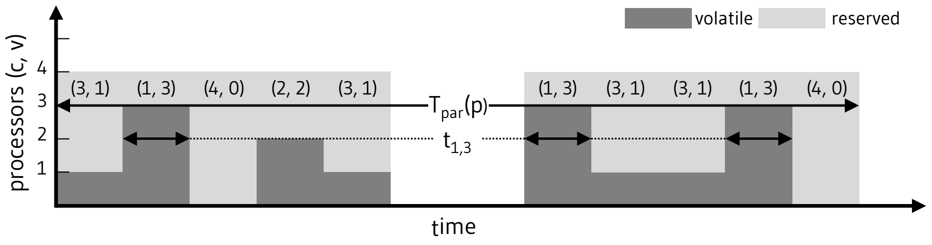

Fig. 1. Exemplary parallel computation employing volatile and reserved processors.

An essential characteristic of a volatile processor is the availability α, defining the probability of being available at any given point in time. For example, volatile processors of the *Google Cloud Platform* are advertised with an availability of 85%-95%. Equation 2 defines the probability of having a configuration with v volatile processors available, out of the total number of required processors p.

$$\underbrace{R(v,p,\alpha)}_{\text{probability of ...}} = \underbrace{(\alpha^v)}_{\substack{\text{...}v\text{ volatile procs.}\\ \text{being available}}} * \underbrace{(1-\alpha)^{p-v}}_{\substack{\text{...}(p-v)\text{ volatile procs.}\\ \text{being unavailable}}} * \underbrace{\binom{p}{v}}_{\text{no. of comb.}} \tag{2}$$

Let $t_{r,v}$ denote the sum of periods spent in configuration (r, v). The total of all periods $t_{r,v}$, where execution takes place with configuration (r, v), constitute the total execution time of a parallel computation $T_{par}(p)$ (cf. Equation 3).

$$T_{par}(p) = \sum_{i=0}^{p} t_{p-i,i} \tag{3}$$

$$t_{r,v} = T_{par}(p) * R(v,p,\alpha) \quad | \quad p = r + v \tag{4}$$

The probability $R(v, p, \alpha)$ for the configuration (r, v) further allows quantifying the fraction of $T_{par}(p)$ that is spent in this configuration, i.e, $t_{r,v}$, which is shown in Equation 4.

With the findings from this discussion, we propose the cost function given in Equation 5 to formalize our cost model.

$$C(p) = \sum_{i=0}^{p} \left(\underbrace{\Big(T_{par}(p) * R(i,p,\alpha)\Big)}_{\substack{t_{p-i,i}\\ \textbf{[A]}}} * \Big(\underbrace{(p-i) * c_{\pi_r}}_{\substack{\text{reserved}\\ \text{processors}\\ \textbf{[B]}}} + \underbrace{i * c_{\pi_v}}_{\substack{\text{volatile}\\ \text{processors}\\ \textbf{[C]}}} \Big) \right) + \underbrace{T_{par}(p) * c_\omega}_{\substack{\text{opport.}\\ \text{costs}\\ \textbf{[D]}}} \tag{5}$$

The total costs are comprised of the costs resulting from each configuration, determined by the period $t_{r,v}$ spent in it during execution as well as the reserved and volatile processors' prices c_{π_r} and c_{π_v}, respectively.

A drawback of utilizing volatile processors is the higher overhead, which is caused by mechanisms for fault tolerance. However, as stated in Section 2, during regular operation, this overhead is small since cleanup and state storing are performed on demand when receiving a withdrawal notification. Referring to the guaranteed period of resource availability, this also holds true for the overhead of re-establishing a computation on another resource. In a first step, we consider this overhead with a constant cost factor for each volatile processor, which is part of c_{π_v}. For an examination of the individual cost constituents, we divided Equation 5 into four parts. [A], [B], and [C] are configuration specific, defining the costs of the cloud resources. Based on Equation 4, [A] quantifies $t_{r,v}$ for each configuration and is applied in [B] (reserved) and [C] (volatile) to determine the total costs of each. Finally, [D] expresses the opportunity costs, which are independent of a computation's configurations.

5 Evaluation

For evaluating the cost models' validity, we examine different aspects of Equation 5. Specifically, we investigate the characteristics that influence the total costs of computations for exemplary scenarios with different scalability. In each scenario, we assume a sequential execution time of $T_{seq} = 12h$ and model the parallel execution time $T_{par}(p)$ for a constant number of processors p by Amdahl's law: $T_{par}(p) = \beta * T_{seq} + \frac{(1-\beta)*T_{seq}}{p}$. The scalability of a parallel application is characterized by its sequential fraction $0 \leq \beta \leq 1$.

First, we consider five different parallel applications, which are characterized by different sequential fractions $0.01 \leq \beta \leq 0.3$. The left graph in Figure 2 illustrates the total monetary costs $C(p)$ for parallel computations of these applications. We set the prices for reserved and volatile processors c_{π_r} and c_{π_v} as well as the availability α in accordance with the advertised ideal situation of the *Google Cloud Platform* [9]. The values of all parameters are shown in the figure. Since opportunity costs are highly application-specific, we assume the shown value for demonstration purposes. Concerning the total costs, it is apparent from this graph that all computations possess a similar behavior. Particularly, all curves of the cost function $C(p)$ have a unimodal shape, exhibiting only a single minimum. At all scales beyond this cost minimum, one would pay for inefficiently used computing resources, i.e., the return (in the form of decreasing opportunity costs) is lower than the investment for additional processors. We also see that the scalability (i.e., the sequential fraction β) has a significant influence on how strong the costs increase after the minimum is reached.

Next, we consider the influence of the opportunity costs on the total costs. As before, we examine different scenarios, which are characterized by different values of c_ω, where $0\$/h \leq c_\omega \leq 10\$/h$. The values of c_{π_r}, c_{π_v}, and α are identical to the previous scenario, whereas the sequential fraction is set to $\beta = 0.1$. The right graph in Figure 2 illustrates the corresponding total monetary costs $C(p)$. Apparent is the strong correlation between the shape of the cost curves in both graphs, however, in a mirrored arrangement. This confirms that the model accurately specifies the two conflicting objectives *fast processing* and *low monetary costs*. The influence of the opportunity costs is particularly evident in the case of $c_\omega = 0\$/h$. Since there are no cost benefits at all for speeding up the computation, $p = 1$ is the cost minimal computing infrastructure. In contrast,

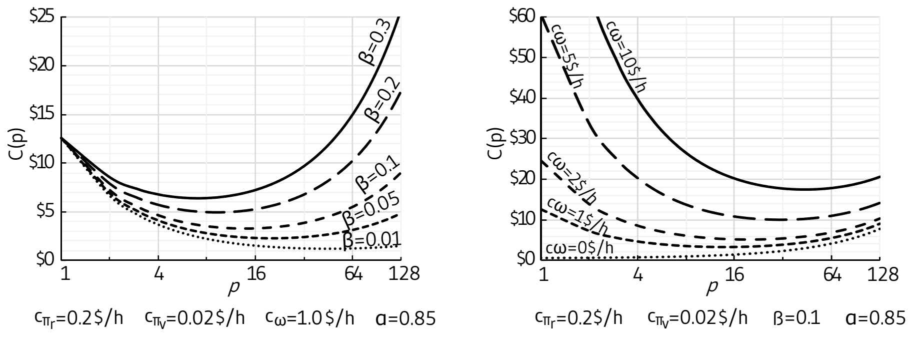

Fig. 2. Costs for parallel computations with different scalability and opportunity costs.

increasing c_ω also increases the number of processors p with the cost minimal computing infrastructure, since the reduction of opportunity costs outweighs the costs for additional processors.

Next, we focus on the availability α and price c_{π_v} of a volatile processor. Particularly, we want to assess to what extent they influence the total costs. Hence, we consider two scenarios with different c_{π_v} and $0 \leq \alpha \leq 1$, while keeping the other values constant. The left graph in Figure 3 shows the total costs for the first scenario, whereas the right graph shows the second scenario, which is characterized by an increased volatile processor price. The first one shows a clear trend of decreasing total costs as the availability of volatile processors increases. While this behavior is no surprise at all, one can furthermore observe that in the right graph it is much less pronounced. The potential benefit of employing volatile processors for parallel computations can only be capitalized if their price is significantly smaller, compared to reserved processors. If this is the case, they offer enormous economic potential for cost minimization. As illustrated in the left graph, the savings are considerable, even at lower degrees of availability.

6 Related Work

There exists a growing body of research on the utilization of cloud environments for HPC workloads. To benefit from the cloud, recent studies found that HPC applications have to be adapted to suit cloud characteristics like on-demand resource access, elasticity, and pay-per-use [7], [15], [13]. In [14], the authors deal with elastic scaling and investigate on a framework and runtime system for applications with dynamic task parallelism. The demand for high-performance cloud systems was also recognized by cloud providers, which launched performance-optimized VMs with InfiniBand like Microsoft *H-Series* [1]. Complementary, tools like *elasticHPC* [2] facilitate the execution of HPC workloads by automating the creation of clusters in the cloud, enabling monitoring, and providing a cost management to start and terminate jobs based on specific price constraints.

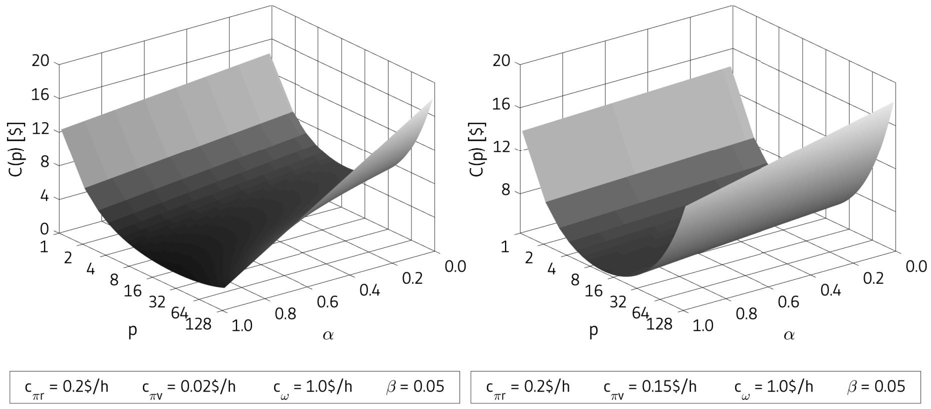

Fig. 3. Costs for parallel comp. with volatile procs. of different availability and price.

In recent years there has been considerable interest in utilizing volatile cloud resources for HPC [5], [2]. Several studies investigated the benefits for various classes of parallel applications, covering traditional applications that employ MPI [17], as well as decoupled MapReduce applications [16]. Contrary to our work, the focus of most of these studies has been mainly on bidding strategies and automated bidding processes.

Cost modeling in the domain of parallel- as well as cloud-computing has been the topic of several studies during the last decade. Current cost models for cloud-based web-applications are typically based on pay-per-use and defined in terms of monetary costs [18], [8]. For parallel computations costs are more abstractly defined, using performance metrics like total processing time [10]. The authors of [11] put together both concepts, extending the model for parallel computations towards pay-per-use of the cloud model. Our previous study [12], presents a cost model for parallel cloud applications to determine the monetary costs based on both execution time and utilized cloud resources.

7 Conclusion

In this paper, we presented a cost model for quantifying the monetary costs of parallel computations, performed in cloud-based environments that consist of both reserved and volatile resources. To adequately model the costs, we addressed a wide range of economic aspects. Specifically, we considered the trade-off between availability and price of volatile processors in the context of costs caused by delayed results of a computation. Our evaluation revealed that volatile processors offer enormous economic potential that can be harnessed for cost minimization. Thus, our approach helps HPC users to exploit the potential of cost-saving further when employing cloud resources for executing their parallel applications. In future work, we plan to extend our cost model towards auction-based pricing models for volatile resources like the Amazon EC2 spot instances.

References

1. Availability of H-series VMs in Microsoft Azure, https://azure.microsoft.com/en-us/blog/availability-of-h-series-vms-in-microsoft-azure/
2. ElasticHPC, http://www.elastichpc.org/
3. HPC Cloud Hits Petaflop, https://www.schrodinger.com/news/schrodinger-partners-cycle-computing-accelerate-materials-simulation-using-cloud
4. Microsoft Azure Low-Priority VMs, https://docs.microsoft.com/en-us/azure/batch/batch-low-pri-vms
5. Spotinst HPC, https://spotinst.com/solutions/hpc/
6. Amazon: Amazon EC2 Spot Instances (2014), https://aws.amazon.com/ec2/spot/
7. Da Rosa Righi, R., Rodrigues, V.F., Da Costa, C.A., Galante, G., De Bona, L.C.E., Ferreto, T.: Autoelastic: Automatic Resource Elasticity for High Performance Applications in the Cloud. IEEE Transactions on Cloud Computing **4**(1) (2016)
8. Deelman, E., Singh, G., Livny, M., Berriman, B., Good, J.: The Cost of Doing Science on the Cloud: The Montage Example. In: Proc. of the ACM/IEEE Conference on Supercomputing (2008)
9. Google Cloud: Preemptible VMs, https://cloud.google.com/preemptible-vms/
10. Grama, A., Kumar, V., Karypis, G., Gupta, A.: Introduction to Parallel Computing. Addison-Wesley, 2 edn. (2003)
11. Gupta, A., Milojicic, D.: Evaluation of HPC Applications on Cloud. In: Proc. of the 6th Open Cirrus Summit. IEEE (2012)
12. Haussmann, J., Blochinger, W., Kuechlin, W.: Cost-efficient Parallel Processing of Irregularly Structured Problems in Cloud Computing Environments. Cluster Computing (2018). https://doi.org/10.1007/s10586-018-2879-3
13. Kehrer, S., Blochinger, W.: A Survey on Cloud Migration Strategies for High Performance Computing. In: Proc. of the 13th Advanced Summer School on Service-Oriented Computing. IBM Research Division (2019)
14. Kehrer, S., Blochinger, W.: TASKWORK : A Cloud-aware Runtime System for Elastic Task-parallel HPC Applications. In: Proc. of 9th International Conference on Cloud Computing and Services Science (2019)
15. Rajan, D., Canino, A., Izaguirre, J.A., Thain, D.: Converting a High Performance Application to an Elastic Cloud Application. In: Proc. of the 3rd IEEE International Conference on Cloud Computing Technology and Science (2011)
16. Taifi, M.: Banking on Decoupling: Budget-driven Sustainability for HPC Applications on Auction-based Clouds. ACM SIGOPS Operating Syst. Rev. **47**(2) (2013)
17. Taifi, M., Shi, J.Y., Khreishah, A.: SpotMPI: A Framework for Auction-Based HPC Computing Using Amazon Spot Instances. In: Proc. of the 11th International Conference on Algorithms and Architectures for Parallel Processing (2011)
18. Viana, V., De Oliveira, D., Mattoso, M.: Towards a Cost Model for Scheduling Scientific Workflows Activities in Cloud Environments. In: Proc. of the IEEE World Congress on Services (2011)