Design Space Exploration of Heterogeneous Systems Applied to the Cloud Resource Allocation Problem

Danillo C. A. Arigoni[1], **Ricardo Ribeiro dos Santos**[1], **Liana D. Duenha Garanhani**[1]

[1] College of Computing – Federal University of Mato Grosso do Sul (UFMS)
Campo Grande – Brazil

{danillo.arigoni, ricardo.santos, liana.duenha}@ufms.br

Abstract. *Cloud computing services providers offer on-demand computing resources to applications. Finding the best cloud resource allocation that fits the users' budget, meets application performance and constraints are still a research challenge. The cloud resource allocation problem is quite akin to the Design Space Exploration (DSE) problem once they both have to find suitable hardware configurations in an ample design space, having incompatible objectives subject to several constraints. This work presents a solution to the cloud resource allocation problem by applying a design space exploration technique. We have designed and developed a software extension, MultiExplorer-VM, from a DSE tool, MultiExplorer, that has a workflow to provide virtual machine configurations according to the users' requirements and application constraints. A comprehensive set of experiments has been performed to evaluate and validate the proposed tool. We have also compared solutions from our proposal to other existing research work focused on the cloud resource allocation problem based on the Paramount Interaction (PI) technique. The results show that the MultiExplorer-VM achieves significant (better) results than the PI technique. The cost results brought by the MultiExplorer-VM were up to 8.8 times lower compared to the PI technique. The experiments also reveal that for most of the applications, MultiExplorer-VM achieved the optimal cloud configuration.*

1. Introduction

The increase in application workload density running on a cloud environment makes it paramount to optimize the resources and services that will be made available in this environment. Additionally, the applications complexity generates demands for heterogeneous computational resources; for example, workloads may require intensive CPU usage (CPU-intensive), while others may require intensive input and output usage (IO-intensive), or even GPU or other accelerators [Lee and Katz 2011]. Once the application demands are known, it becomes necessary to allocate a set of hardware and software resources that best meet those demands. For many applications, allocating computing resources should simultaneously achieve multiple objectives.

In the context of cloud computing applications, the virtual machine - VM - is a resource available as a combination of real hardware and virtualization software [Smith and Nair 2005]. From the user's perspective, the virtualization software

Acknowledgment: The authors wish to thank the FUNDECT for the financial support to this work, process number 71/700.151/2020.

enables the VM to present necessary hardware (processor, memory, I/O devices, storage) and software (operating system) resources in a way that is individualized to the application. However, the original hardware (host) can be shared among different applications or users. The suitable VM allocation for cloud applications should meet different objectives such as: maximizing application performance, minimizing cost, minimizing execution time, and minimizing communication time, among others [Buyya et al. 2009, Calheiros et al. 2011].

The cloud computing services industry has adopted tools to recommend appropriate configurations for user applications. For example, AWS cloud infrastructure and service provider offers the AWS Compute Optimizer tool that applies machine learning techniques to analyze historical utilization metrics from applications including IO, storage, CPU, and network utilization. Compute Optimizer creates VMs recommendations that reduce costs and optimize computing power and application performance [Amazon 2020]. This tool only generates recommendations for some AWS instances, and it can take up to 12 hours for the service to finish the analysis and resource recommendations.

In the field of computer systems, the Design Space Exploration (DSE) consists of algorithms and optimization techniques to choose, within a set of architectural parameters (the design space), computational solutions (or configurations) to meet applications demands. In general, DSE techniques aim at meeting multiple objectives (such as maximizing performance or minimizing energy consumption) subject to several design constraints (cost, area, power dissipation, among others). Cloud resource allocation and design space exploration problems share essential characteristics. While one demands suitable resource configurations, the other offers techniques to explore alternative configurations. In this context, this work aims to develop a solution to the cloud resource allocation problem by adopting design space exploration techniques. The motivation to propose design space exploration techniques as a potential solver for cloud resource allocation is to take advantage of the ability of DSE techniques to scale up resources solutions by addressing issues such as heterogeneity, multiple objectives, sets of constraints.

In this work, we propose an approach that adopts the MultiExplorer framework for the design space exploration in multicore systems as the building block for designing and developing an extension (MultiExplorer-VM). The MultiExplorer-VM goal is to offer alternative virtual machine configurations to meet cloud applications' resource demands. We evaluated our approach by searching lesser applications' runtime and lesser VM cost for 27 workloads and showed the results from MultiExplorer-VM were better than those from the PI technique achieving VM cost results up to 8.8 times lower. The main contributions of this work are:

- A new approach to provide alternative virtual machine solutions for resource demanding applications in cloud computing using DSE techniques.
- Predictor systems to evaluate cloud application runtime and cost running on VMs configurations. The design and developing of accurate runtime and cost predictors were also carried out in this work.
- A comprehensive validation and evaluation on a set of applications from the NPB benchmark and on AWS virtual machine configurations. Additionally, we compare the results of the MultiExplorer-VM to the Paramount Interaction Technique [Rosario et al. 2020].

The remainder of this paper is organized as follows: Section 3 presents a literature review and discussion of work related to the topic of this project. Section 4 details the development of the MultiExplorer-VM extension and the development and evaluation of the time and cost predictors for cloud virtual machines used in the extension. Section 5 presents the entire experimental procedure, validation, discussion of the results obtained, and comparisons with other techniques available in the literature in the area. Section 6 presents the conclusions and propositions for future work.

2. Related Work

Cloud providers like AWS [Amazon 2020], Google Cloud[1], and Microsoft Azure [2] provide to their users a diversity of virtual machine types and cloud configurations to be instantiated. There are different costs and performance depending on the chosen VM instance so that it is hard to a user to choose the best VM configuration to meet its application demands. The task of choosing a suitable VM configuration to the applications is subject to the Cloud Resource Allocation Problem and it is also a current topic of research in the literature.

There are proposals that seek to solve the cloud resource allocation problem by extracting information from multiple application runs to model the resources and performance demands of applications [Yadwadkar et al. 2017][Venkataraman et al. 2016]. The collected information constitutes a training set for machine learning algorithms aimed at recommending cloud configurations. Ernest and PARIS are examples of systems that have gone into this direction.

Ernest [Venkataraman et al. 2016] exploits the internal structure of a workload to generate a linear model that predicts execution time. The prediction needs only a small amount of data as input, significantly reducing the cost of measurement. PARIS [Yadwadkar et al. 2017], on the other hand, builds a measurement-based Random Forest model to make predictions of execution time or operation cost of workloads.

Some authors focused on exploring the search space of instance configurations. Techniques such as random search and network search achieved lower performance when compared to statistical methods such as Bayesian Optimization with Random Forest [Hsu et al. 2018][Rosario et al. 2020] or Gausian Process [Alipourfard et al. 2017]. Those approaches have the benefit of low-cost search by looking at a few configurations. In [Rosario et al. 2020] the search cost is even lower with the adoption of Paramount Interactions.

The proposal presented in this paper differs from the research work found in the literature by not resorting to the cloud provider at any stage of the working flow. MultiExplorer-VM, on the other hand, adopts the a simulation tool [Calheiros et al. 2011] to obtain the performance and cost data that will act as inputs (constraints) to the DSE module. The DSE module uses time and cost accurate predictors to estimate runtime and cost of the applications running on the proposed cloud resource. Another highlight of this work is the ability to search for VM instances that meet elasticity and heterogeneous VM configurations. Such features can lead to better cost $\times$ runtime relationships for determining the configuration of VM instances suitable to the workload.

[1]https://cloud.google.com

[2]https://azure.microsoft.com

3. MultiExplorer-VM

MultiExplorer-VM[3] carries out a new execution flow on the top of the MultiExplorer tool. In this flow, a new user interface is made available so that it is possible to simulate applications on virtual machines, besides offering architectural alternatives, via design space exploration, according to the users' demands. Figure 1 illustrates the MultiExplorer-VM main modules execution flow (the dashed modules were designed in this work):

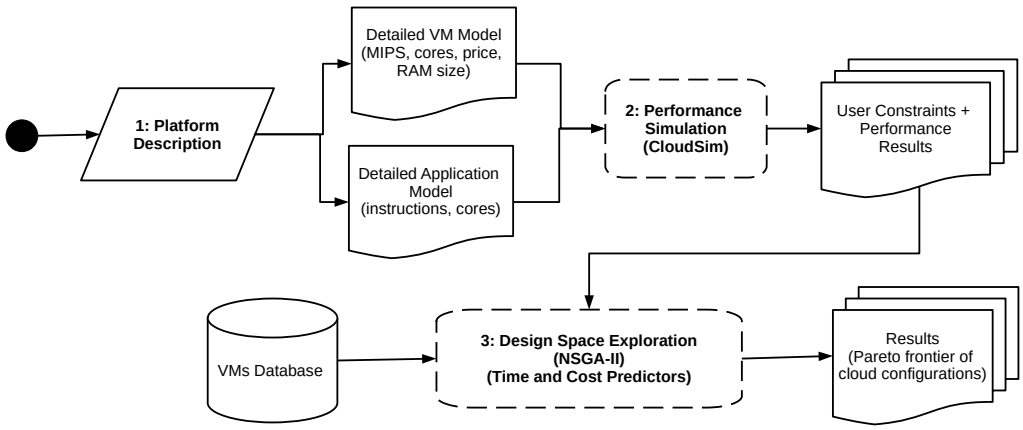


Figure 1. MultiExplorer-VM modules and execution flow.

- The first step (Platform Description) allows a user to set the applications to be evaluated, an initial VM configuration to work as a cloud resource reference, the maximum budget (cost) that can be applied to run the application on a cloud resource, and the number of VMs to be used as parameters for the DSE step.
- In the performance simulation step, the CloudSim cloud simulator [Goyal et al. 2012] is run aiming to determine the performance of the application(s) on the virtual machine(s) chosen as an initial solution. The initial VM solution is named the original solution. The application runtime and cost together with the parameters from the first step are sent to the DSE module.
- In the design space exploration stage, a DSE algorithm (MultiExplorer uses a Non-Sorted Genetic Algorithm - NSGA-2 - heuristic) is run from the initial (original) solution in order to find new alternative VMs configurations to meet the applications demands for cost and performance subject to the users' constraints. The DSE algorithm constitutes the VMs configurations from a VMs database comprised of AWS instances dataset. The instances work as design space exploration building blocks forming up a new VM configuration that meet the constraints and objectives set by the user. Time and cost predictors are coupled to the DSE module in order to estimate application's performance and cost data for each VM configuration.

From Figure 1, one may observe that the first step is the platform description where a user should inform an initial VM model and some applications parameters to be

[3]https://github.com/lscad-facom-ufms/multiexplorer/tree/MultiExplorer-VM

used by the second step. The Performance Simulation module is responsible for simulating (on CloudSim) the application on the initial VM model. The Design Space Exploration module carries out a design space exploration from the applications' data coming from the second stage and from the user's constraints from the first stage. The module picks up VM configurations from the VMs Database and runs the runtime and cost predictors to estimate the time and cost of the user's application on the chose VM configuration. Once the configuration meets all the constraints and improve the time and cost objectives, it constitutes the Pareto Frontier of VM configurations solutions.

4. Dataset and VMs Configurations

The VMs configurations adopted in the design of the MultiExplorer-VM are comprised of nine families (available at AWS [Amazon 2020]) {c3, c4, c5, m3, m4, m5, r3, r4 and r5} and three sizes {large, xlarge and 2xlarge}. Table 1 presents a summary of the VM configurations and the price per hour.

Table 1. VMs configurations and pricing.

VM Type	vCPU	Memory (GB)	Price (USD/h)	VM Type	vCPU	Memory (GB)	Price (USD/h)
c3.large	2	4	0.163	r4.large	2	16	0.133
c3.xlarge	4	8	0.325	r4.xlarge	4	32	0.266
c3.2xlarge	8	16	0.650	r4.2xlarge	8	64	0.532
m3.large	2	8	0.190	c5.large	2	4	0.131
m3.xlarge	4	16	0.381	c5.xlarge	4	8	0.262
m3.2xlarge	8	32	0.761	c5.2xlarge	8	16	0.524
r3.large	2	16	0.350	m5.large	2	8	0.153
r3.xlarge	4	32	0.700	m5.xlarge	4	16	0.306
r3.2xlarge	8	64	1.399	m5.2xlarge	8	32	0.612
c4.large	2	4	0.100	r5.large	2	16	0.201
c4.xlarge	4	8	0.199	r5.xlarge	4	32	0.402
c4.2xlarge	8	16	0.398	r5.2xlarge	8	64	0.804
m4.large	2	8	0.100				
m4.xlarge	4	16	0.200				
m4.2xlarge	8	32	0.400				

The applications to validate and evaluate the MultiExplorer-VM are from the Numerical Aerodynamic Simulation Parallel Benchmarks (NPB) suite version 3.4.2 with MPI support [Bailey et al. 1995] [4]. The NPB is a set of parallel computing performance benchmarks with five kernels and three fluid dynamics applications. Each benchmark has a set of eight inputs divided into classes: S, W, A, B, C, D, E, and F. The S class consists of small workloads for quick testing; class W is designed to run on workstations; classes A, B, and C contain standard workloads; and classes D, E, and F are large workloads. The five available kernels were used with the input classes S, W, A, B, C, D, and E. Each of the kernels are detailed in Table 2.

From the VMs configurations and the applications of the NPB suite, we had to build the database of VM configurations to be used by the DSE module of the MultiExplorer-VM workflow. The VM database has 27 VM configurations (VM Types in Table 1) and 27 application configurations (5 applications $\times$ 5 entries + 1 application $\times$

[4]https://www.nas.nasa.gov/publications/npb.html

Table 2. NPB kernels.

Benchmark	Description	Language
EP	Embarrassingly Parallel	Fortran
FT	Discrete 3D fast Fourier Transform	Fortran
MG	Block Tri-diagonal solver	Fortran
IS	Integer Sort, random memory access	C
CG	Conjugate Gradient	Fortran

2 entries). Along the experiments, the input classes D and E succeeded only for the EP kernel when run on a local machine (Intel Core i7-5500U CPU 2.40GHz $\times$ 4 processors, 12GB RAM, Ubuntu 20.04.2 LTS 64-bit operating system).

The time and cost of each VM are obtained from the CloudSim simulator which requires that each application (each application configuration is named cloudlet) should provide the number of application instructions and the applications' core demands. The Pin [Luk et al. 2005] profiling tool was adopted to get the total number of instructions from each cloudlet. For each benchmark application, Pin inserts a call to counter function before every instruction. When the program exits, it saves the count of the total number of instructions executed in a file. The cores demands for each cloudlet were evaluated in 5 configurations (2, 4, 8, 16, and 32 cores). The CloudSim simulator estimates the runtime of each cloudlet on a VM configuration. The application costs were calculated by multiplying the runtime results and pricing (from Table 1). The VMs dataset has also heterogeneous VM configurations. In this case, the virtual machine MIPS (Millions of Instructions Per Second) metric is calculated from the weighted average of the applications' instructions by the number of cores of each VM. Other VM metrics such as: the number of cores, RAM, and price, come from the sum of the values of the respective characteristics present in each VM of the heterogeneous system. The VMs dataset has a total of 82862 configurations.

5. Development of Time and Cost Predictors

Once that the DSE module will suggest VMs configurations to meet the user's objectives and constraints, the configuration may have a large number of VMs so that it is necessary to estimate the performance and cost of this new configuration. Those estimates will be given by predictors which are specifically designed to provide accurate time and cost estimates for each VM configuration. The development of the predictors were based on machine learning algorithms available in the scikit-learn library (version 1.0.2)[5]. Considering that the predictors addressed in this work may consider the same set of independent variables, a set of machine learning algorithms with a prominent focus on regression problems were selected for evaluation: Polynomial Regression (degree = 1,...,5), k-Nearest Neighbours, Decision Tree, Random Forest and Support Vector Machines (SVM) (kernel= RBF, linear, poly3, poly5, sigmoid).

The dataset of applications and VM configurations to build the predictors comes from the NPB kernels (simulated on CloudSim) and AWS instances. The dataset was divided into 20% for the training and 80% for the test set. The fitting procedure of each model separated the training set into ten rounds. In each round, 90% of the set was dedi-

[5]https://scikit-learn.org/stable/

cated to training, and 10% was dedicated to model validation. The following metrics were used to evaluate the prediction models: Coefficient of determination[6] R^2, training score, the response time of the test score, and the Mean Absolute Percentage Error (MAPE). Figure 2 illustrates the cross-validation strategy applied to the design and validation of the time and cost predictors.

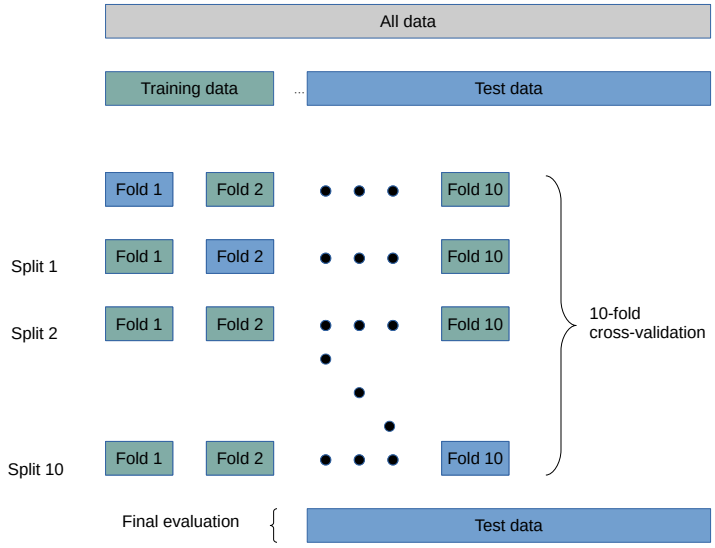


Figure 2. Cross-validation strategy.

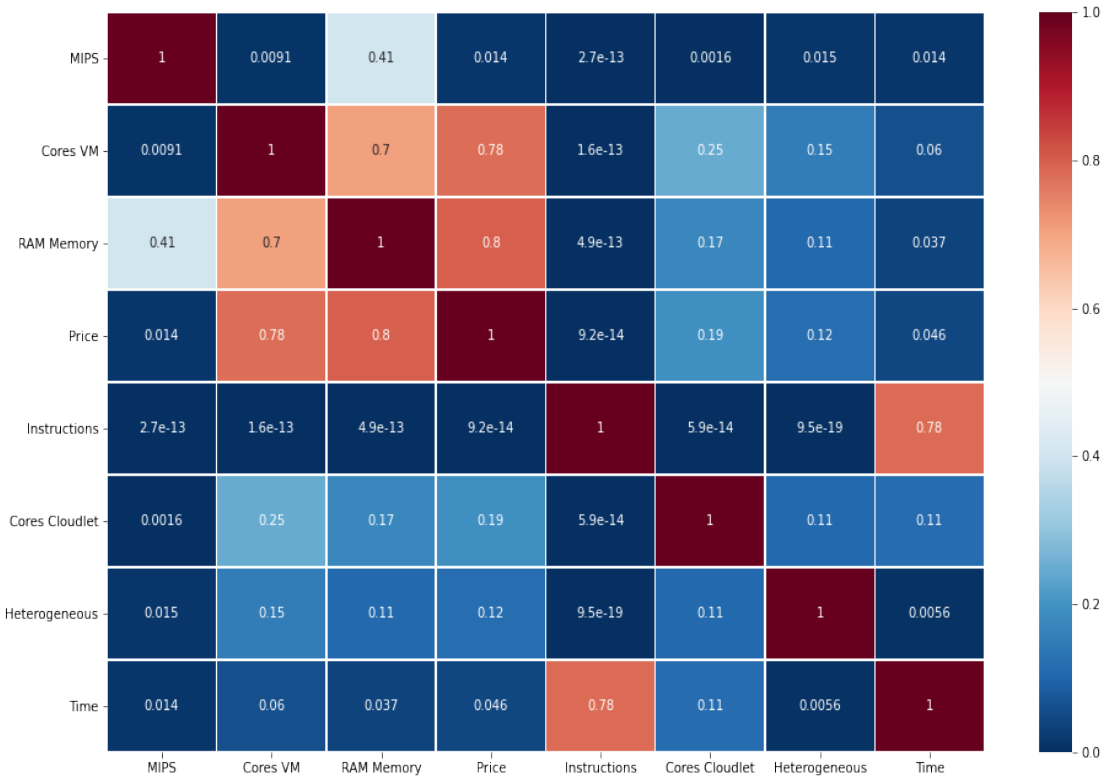
The MAPE[7](equation 1) represents the average percent difference between the observed (real) output Y^i and the predicted value $\hat{Y}^i$ for n samples ($i = 1, \ldots, n$).

$$MAPE = \frac{1}{n} \sum_{i=1}^{n} \frac{|Y^i - \hat{Y}^i|}{Y^i} \tag{1}$$

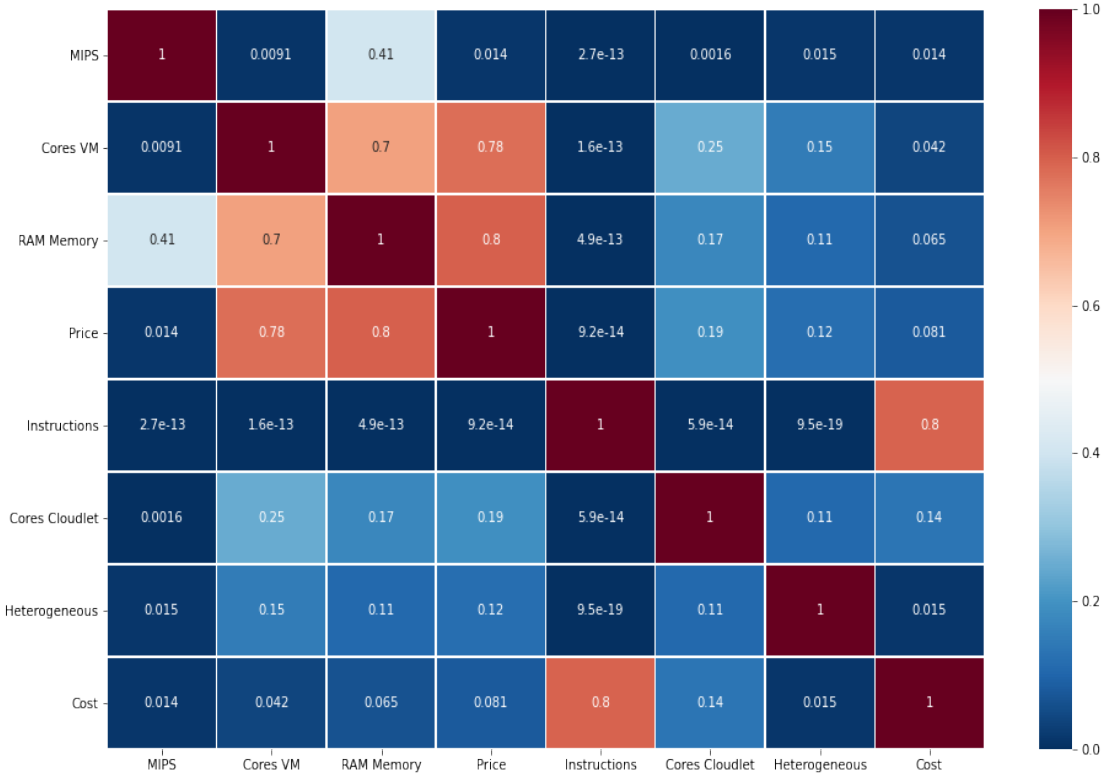
Figure 3 presents the heat maps generated from Pearson's correlation[8] considering only the response variables Time (a) and Cost (b). The variables chosen to the Time predictor were based on the highest Pearson's correlation and MAPE score: MIPS, Instructions, Cores Cloudlet, Cores VM and Heterogeneous. The variables chosen to the Cost predictor were: MIPS, Instructions, Cores Cloudlet, Cores VM, Price and Heterogeneous.

Tables 3 and 4 present the model predictors, the configuration parameters that achieved the best results for each model, and the results according to the metrics R^2, MAPE, and Runtime. The Decision Tree-based was the model chosen as the time predictor, achieving the lowest response time, the lowest MAPE, and the highest R^2 for training and testing. The models based on polynomial regression (degree = 5), Decision Tree, and Random Forest achieved the highest R^2 scores on the cost predictor. Random Forest

[6]https://en.wikipedia.org/wiki/Coefficient_of_determination
[7]https://en.wikipedia.org/wiki/Mean_absolute_percentage_error
[8]https://en.wikipedia.org/wiki/Pearson_correlation_coefficient



(a) Time



(b) Cost

Figure 3. Pearson's correlation coefficients between the independent variables and the dependent (Time and Cost).

and Decision Tree had the lowest MAPEs, but the decision tree model's response time is about 30 times lowest than the second-best result so that the Decision Tree-based models were chosen for time and cost predictions in the DSE module.

Table 3. Characteristics of time predictor models with best scores.

Model	Parameters	R^2 Training (MAPE) (%)	R^2 Testing (MAPE) (%)	Runtime (s)
Decision Tree	max_depth: None, min_samples_split: 2	99.86 (0.009)	99.89 (0.008)	0.020
Random Forest	max_depth: 20, min_samples_split: 2, n_estimators: 50	99.85 (0.013)	99.90 (0.012)	0.613
Polynomial	degree: 5	99.50 (50.77)	99.89 (46.72)	0.599
KNN	n_neighbors: 3	97.27 (1.46)	97.76 (1.74)	1.127
SVR	kernel: poly, degree: 5	93.85 (113.41)	94.65 (84.94)	19.973

Table 4. Characteristics of cost predictor models with best scores.

Model	Parameters	R^2 Training (MAPE) (%)	R^2 Testing (MAPE) (%)	Runtime (s)
Decision Tree	max_depth: 25, min_samples_split: 2	96.85 (0.025)	97.64 (0.025)	0.017
Random Forest	max_depth: 20, min_samples_split: 2, n_estimators: 50	98.27 (0.024)	98.27 (0.023)	0.525
Polynomial	degree: 5	99.89 (0.134)	99.88 (0.135)	0.751
KNN	n_neighbors: 2	95.88 (0.107)	96.55 (0.105)	0.777
SVR	kernel: poly, degree: 5	93.02 (0.413)	93.12 (0.536)	20.851

6. Experiments and Results

We have performed validation experiments on the MultiExplorer-VM and pairwise comparisons to the PI technique [Rosario et al. 2020] and a brute force algorithm that finds optimal VM configurations but takes a long runtime. The validation experiments were based on statistical tests: estimating the power of the pairwise test, testing for normality of samples (Shapiro-Wilk test[9]), testing for homogeneity of variances (Bartlett[10] and Fligner-Killeen[11] tests). The pairwise comparison experiments targeted to explore VMs configurations for each application of the NPB benchmark.

After the sample data validation, t-tests[12] and Wilcoxon tests[13] were applied to evaluate hypotheses of statistical significant difference among the best (following time and cost criteria) VM configurations from the MultiExplorer-VM, Brute Force and the PI technique. The null (H_0) and alternative (H_1) hypotheses for the pairwise tests were:

$$H_0 : \quad Cost(VM_{ME}) = Cost(VM_{BF,PI}) \tag{2}$$

$$H_0 : \quad Time(VM_{ME}) = Time(VM_{BF,PI}) \tag{3}$$

[9]https://en.wikipedia.org/wiki/Shapiro–Wilk_test

[10]https://en.wikipedia.org/wiki/Bartlett%27s_test

[11]https://www.rdocumentation.org/packages/stats/versions/3.6.2/topics/fligner.test

[12]https://en.wikipedia.org/wiki/Student%27s_t-test

[13]https://en.wikipedia.org/wiki/Wilcoxon_signed-rank_test

$$H_1: \quad Cost(VM_{ME}) \neq Cost(VM_{BF,PI}) \tag{4}$$
$$H_1: \quad Time(VM_{ME}) \neq Time(VM_{BF,PI}) \tag{5}$$

H_0 hypotheses means that there is no difference in cost (2) or time (3) between the best solutions from the MultiExplorer-VM (ME) and the solutions from a Brute Force (BF) algorithm and the Paramount Interaction (PI) technique. The alternative hypotheses (H_1) represent a statistical significant difference between the solutions following the same cost (4) and time (5) metrics. The significance level chosen for the tests is $\alpha = 0.05$. For a power of the test of 80% (reduction in the occurrence of false-negative or type II error), 21 runs of MultiExplorer-VM were generated for each application. The constraints applied to the design space exploration by the MultiExplorer-VM technique were: a maximum runtime of one hour and a cost of 2 USD/h for each application.

Table 5 presents the runtime and cost of the best VM configurations, considering the lowest runtime and cost, suggested by MultiExplorer-VM (ME), Brute Force (BF), and PI technique for each application. One may notice that ME and BF have quite akin best results, thus indicating that ME finds the optimal VM configuration. Another interesting finding from this experiment is that the ME results (time and cost) are better than the PI results for all applications. Since PI searches for VM configurations that minimize the cost, we compare the cost improvement of ME compared to PI. The ME cost improvement is up to 8.8 times (EP D application). Table 6 presents the VM configurations details (amount of each VM) suggested by ME, BF, and PI for the EP D application. The first three rows present the VM configurations of each technique that achieved the lowest runtime. The last three rows present the VM configurations with the lowest cost of each technique. The results of ME and BF, in Table 6, reveal a tradeoff between time and cost objectives of the VM configurations.

Despite not presented, we performed statistical pairwise comparison tests (t, and Wilcoxontests) among all three techniques. The statistical tests aimed to evaluate if the results have statistical differences between ME and PI. The results showed statistical evidence to accept the alternative hypothesis (H_1) that ME and PI results are not the same ($p - value < \alpha = 0.05$). Seven (EP A, EP B, EP C, EP D, FT C, IS C, CG C) out of eight applications achieved better (lowest) cost on ME than PI. The tests also reveal a significant difference between ME and BF results.

One of the reasons for ME achieving better results is on exploiting VM heterogeneous configurations. PI and most cloud resource allocation techniques in the literature approach the problem by searching for homogeneous VM configurations. The PI results may also have been affected by the constraints (4 paramount interactions for each application) adopted in this technique.

7. Conclusion

This work designed and developed a design space exploration tool (Multiexplorer-VM) to provide alternative VM configurations for applications that demand cloud computing resources. In addition to MultiExplorer-VM, we also designed and developed accurate prediction systems that estimate runtime and cost of applications according to VM configurations. The predictors were built on the top of the Decision Tree technique due to the low prediction error (MAPE), high test score, and low response time compared to other

Table 5. Time and Cost of VMs configurations suggested by MultiExplorer-VM (ME), Brute Force (BF), and Paramount Interaction (PI) techniques.

Application	Time (sec)			Cost (USD)		
	ME	BF	PI	ME	BF	PI
EP S	0.385	0.322		0.17	0.17	
EP W	0.774	0.644		0.204	0.17	
EP A	5.112	5.112		0.193	0.17	
EP B	20.628	20.628		0.255	0.255	
EP C	77.76	77.76	2268	0.34	0.34	0.476
EP D	1249.2	1249.2	150048	1.02	1.02	9.07
FT S	0.117	0.117		0.17	0.17	
FT W	0.290	0.235		0.17	0.17	
FT A	3.96	3.96		0.17	0.17	
FT B	45.72	45.72		0.255	0.255	
FT C	221.4	221.4	4896	0.425	0.425	0.68
MG S	0.015	0.015		0.17	0.17	
MG W	0.306	0.255		0.17	0.17	
MG A	2.257	1.882		0.17	0.17	
MG B	5.904	5.904		0.17	0.17	
MG C	56.52	47.16	1188	0.278	0.255	0.34
IS S	0.011	0.009		0.193	0.17	
IS W	0.097	0.097		0.17	0.17	
IS A	0.727	0.727		0.17	0.17	
IS B	3.513	2.926		0.17	0.17	
IS C	11.016	11.016	5119	0.193	0.17	0.238
CG S	0.083	0.083		0.193	0.17	
CG W	0.450	0.450		0.193	0.17	
CG A	1.71	1.71		0.193	0.17	
CG B	43.92	43.92		0.255	0.255	
CG C	155.52	129.6	1623	0.34	0.34	1.36

Table 6. VM configurations details from MultiExplorer-VM (ME), Brute Force (FB), and Paramount Interaction (PI) suggested to the EP D application.

Technique	Application	Time (sec)	Cost (USD)	VM configuration
ME	EP D	1249.2	1.7	20 ×c5.large
BF		1249.2	1.7	20 ×c5.large
PI		150048	9.07	2 ×c5n.large
ME	EP D	3560.8	1.02	12 × c5.large
BF		3560.8	1.02	12 × c5.large
PI		150048	9.07	2 ×c5n.large

machine learning models. A highlighted feature of MultiExplorer-VM is the capability to explore heterogeneous VM configurations that meet the applications' demands.

The experiments were carried out to validate and evaluate MultiExplorer-VM. The results were compared to an optimal algorithm (brute force search) and the PI technique. The results demonstrated the feasibility of DSE as an alternative for solving the cloud resource allocation problem. For most of the applications evaluated, the results from MultiExplorer-VM were better (lesser applications' runtime and lesser VM cost) than those from the PI technique achieving VM cost results up to 8.8 times lower. Some opportunities for future work are on evaluating the suggested heterogeneous configurations in cloud computing infrastructure providers and extending MultiExplorer-VM by covering the search for storage and memory-based cloud systems.

References

Alipourfard, O., Liu, H. H., Chen, J., Venkataraman, S., Yu, M., and Zhang, M. (2017). Cherrypick: Adaptively unearthing the best cloud configurations for big data analytics. In *14th Symposium on Networked Systems Design and Implementation*, pages 469–482.

Amazon (2020). Amazon Web Services. `https://aws.amazon.com/`. [Online; accessed: 08-February-2020].

Bailey, D., Harris, T., Saphir, W., Van Der Wijngaart, R., Woo, A., and Yarrow, M. (1995). The NAS parallel benchmarks 2.0. Technical report, Technical Report NAS-95-020, NASA Ames Research Center.

Buyya, R., Ranjan, R., and Calheiros, R. N. (2009). Modeling and simulation of scalable cloud computing environments and the cloudsim toolkit: Challenges and opportunities. In *2009 International Conference on High Performance Computing & Simulation*, pages 1–11. IEEE.

Calheiros, R. N., Ranjan, R., Beloglazov, A., De Rose, C. A., and Buyya, R. (2011). Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. *Software: Practice and experience*, 41(1):23–50.

Goyal, T., Singh, A., and Agrawal, A. (2012). Cloudsim: simulator for cloud computing infrastructure and modeling. *Procedia Engineering*, 38:3566–3572.

Hsu, C.-J., Nair, V., Freeh, V. W., and Menzies, T. (2018). Arrow: Low-level augmented bayesian optimization for finding the best cloud vm. In *2018 IEEE 38th International Conference on Distributed Computing Systems*, pages 660–670. IEEE.

Lee, G. and Katz, R. H. (2011). Heterogeneity-aware resource allocation and scheduling in the cloud. *HotCloud*, 11:4–8.

Luk, C.-K., Cohn, R., Muth, R., Patil, H., Klauser, A., Lowney, G., Wallace, S., Reddi, V. J., and Hazelwood, K. (2005). Pin: building customized program analysis tools with dynamic instrumentation. *ACM sigplan notices*, 40(6):190–200.

Rosario, V. M., Silva Camacho, T. A., Napoli, O. O., and Borin, E. (2020). Fast and low-cost search for efficient cloud configurations for HPC workloads. *arXiv e-prints*.

Smith, J. and Nair, R. (2005). *Virtual machines: versatile platforms for systems and processes*. Elsevier.

Venkataraman, S., Yang, Z., Franklin, M., Recht, B., and Stoica, I. (2016). Ernest: Efficient performance prediction for large-scale advanced analytics. In *13th Symposium on Networked Systems Design and Implementation*, pages 363–378.

Yadwadkar, N. J., Hariharan, B., Gonzalez, J. E., Smith, B., and Katz, R. H. (2017). Selecting the best vm across multiple public clouds: A data-driven performance modeling approach. In *Proceedings of the 2017 Symposium on Cloud Computing*, pages 452–465.