# Towards an Enhanced VM Placement Solution for Power-Aware Cloud Environments

Katja Gilly
Miguel Hernández University.
Elche (Alicante), Spain
Email: katya@umh.es

Sonja Filiposka
Ss. Cyril and Methodius University
Skopje, Macedonia
Email: sonja.filiposka@finki.ukim.mk

Carlos Juiz
University of the Balearic Islands
Palma de Mallorca, Spain
Email: cjuiz@uib.es

***Abstract*—This paper describes the already completed and verified first stage of a work in progress that aims to port a proposed optimised framework for virtual machine placement on hosts in a given cloud computing datacentre. The previously implemented and simulated framework using Matlab is being transferred to a richer cloud computing simulation environment that is CloudSim. The use of CloudSim permits, apart from validating the previous research, the introduction of many additional parameters and criteria in the simulated scenario thus enabling a far richer and in-depth analysis of the benefits and drawbacks of the proposed solution, such as power-awareness, that is an additional feature included in CloudSim libraries.**

## I. Introduction

With the steady rise of cloud computing as a preferred utility computing environment in an increasing number of different business, research and education cases, the need to optimise the main cloud computing characteristics is becoming the focal point of research efforts. This includes features like on-demand response, rapid elasticity, resource pooling and pay per use, that need to be considered in order to research and progress in the better understanding of cloud environments [1]. Many challenges need to be overcome towards the end user [2], starting from an intuitive and easy to use interface, added to the users demand of having access to the cloud service anytime, anywhere, from any device, and ending with the problems of accounting and billing, scaling resources up and down, and deciding on the optimal usage of the available resources. It is our task, as researchers, to work all the way out to optimising the full spectrum of characteristics that are the constituent part of one cloud datacentre.

Having in mind that the main technology behind cloud computing is virtualisation [3], in order to ensure more efficient investments, the design and the definition of the working sets within the datacentre are placed among the most important aspects that demand careful planning of the datacentre layout [4] in terms of physical hosts (PH) and their interconnection network, but also in terms of cooling and power consumption.

With the apparent complexity of this whole process it becomes obvious that researchers can not focus on one part of the whole problem disregarding how this component interacts with and affects the rest of the system. However, when starting to design a possible solution for one concrete component (i.e. VM placement algorithm) it is necessary to be able to set up the rules and decisions while isolating the component from the rest of the system, as a first stage, in order to thoroughly analyse and optimise it in a controlled environment. This permits to observe all possible reactions and outcomes of the designed component's behaviour. For these purposes, a fairly complex simulation environment aiming to capture the multifaceted nature of cloud computing is needed.

The purpose of this paper is to give an overview of this process and emphasise its importance by describing the initial results of a work in progress that aims to complete the second stage of the design process. Namely, starting from a fully developed (and analysed in isolation) optimised VM placement framework [5] designed in the authors' previous work, our goal is to port this component into a complex cloud datacentre simulation environment that will enable an extensive analysis of the way the framework can be tuned in order to provide the maximum optimisation for the complete set of parameters that describe the cloud datacentre. The process of porting in the case of composite components such as the one in question is far from trivial and demands a thorough knowledge of both the initial component implementation and the simulation environment that has been chosen for porting. Moreover, care must be taken to ensure that all the underlying component modules perform identically to their original implementation, that would mean that they are properly validated. Within the scope of this paper, we discuss the goals and steps of the porting procedure, the validation procedure and the extent of benefits that will be obtained upon completing the procedure.

The rest of the paper is structured as follows: In the second section, we give an overview of the original design of the VM placement framework, its inner structure, input parameters and the flexibility to simulate different scenarios. We also discuss its drawbacks that are the main motivation for the present work. In the third section, we give an overview of the complete process of porting the VM placement framework to a cloud simulator, starting from choosing a cloud simulator with the required features, and then moving to the decomposition of the porting process and its specific issues. The fourth section describes the validation of the work achieved so far, where a comparison of the original and ported version of the framework is presented. The fifth section discusses the benefits of the porting process and the possibilities that will be available upon the completion of our work. Special emphasis is given to the seamless integration of the available power consumption modelling in the chosen simulator. The last section concludes the work presented.

## II. Initial design of the VM placement framework

In parallel to the development of the virtualisation technology, there have been many proposals of different heuristics and algorithms for VM placement, all of them aiming to optimise

978-1-5090-3843-5/17/$31.00 

one parameter or a set of them that describes the datacentre. The efforts started as a simple first fit greedy approach [6], working their way up to far more sophisticated approaches [7]. The problem of VM placement is, in fact, a subset of the problem of optimal resource usage (packing) [8]. However, despite the very high computational complexity, over time many heuristics have been developed focusing mostly on trying to optimise a given cost function. This function represents the optimisation goal of the heuristics under a set of constraints that define the compatibility of the VM needed resources and the host available resources. While in the past these heuristics were confined to a *one parameter, one optimisation goal space*, recent attempts deal with complex scenarios that describe the set of different resource characteristics (i.e., CPU, RAM, OS, storage, etc.) and are focused on achieving optimisations in a set of multi objective functions. In this way, new complex VM placement algorithms that take into account network throughput, overall power consumptions, or other parameters that additionally describe the datacentre, have emerged.

While the number of proposed solutions is overwhelming, the lack of thorough investigation of the impact of the VM placement implementation on the overall workings of the cloud datacentre is consistent. For these purposes, we decided to port a previously designed community-based VM placement framework [5], that incorporates several types of greedy heuristic implementations and combines them into a hierarchical decision making component based on a complex multi-objective function.

In more detail, the community-based VM placement framework offers the possibility to analyse the outcome of a VM placement or VM migration process implemented as a two level hierarchy where a matchmaking procedure decides on the optimal placement that takes into account the distribution of available resources across hosts and the nature (i.e. interdependency or interconnectivity) of the VMs that are to be placed. On both levels, there are different heuristic algorithm implementations that allow the user to independently choose what algorithm will be used on which level. The four currently available implemented algorithms are *bin centric*, *dot product*, *vector based load balancing*, and *vector based consolidation* [5]. All algorithms are based on greedy heuristics with a best fit approach, where the placement decision is made by comparing the available versus the demanded community-based resources.

Besides the choice of algorithms, when setting up a simulation scenario, the user can choose the number of hosts in the datacentre, the type of network connectivity in the datacentre (Fat-Tree or VL2 [9]), the characteristics of the physical servers (number of cores and RAM size), the number of cloud services, maximum number of VMs per cloud service, and the set of characteristics (cores and RAM) for the VMs. Within the simulation scenario a cloud service is defined as a group of tightly interconnected VMs (usually requested by the same tenant) that are working together in order to provide a complex service for the end users.

Taking into account all these considerations, the complete framework is implemented as a package in Matlab and enables different visualisations of the obtained results for the complete placement of all cloud services within the defined datacentre. This implementation enables straightforward creation of a number of different scenarios by varying the input parameters of the system. All scenarios are simulated in a very optimised environment using specialised matrix oriented computing. Also, each scenario is automatically run a given number of times (min 5) in order to ensure that the obtained averaged results are statistically significant and can be used as very good representatives of the expected outcome for the given scenario parameters.

However, while the Matlab implementation of the VM placement framework is a very convenient and incredibly fast tool that provides detailed insight into the process of VM placement, it also has a number of drawbacks. The implementation permits easily extending the number of parameters that describe physical hosts and VMs as long as the new parameters are of a similar nature as the ones implemented (i.e storage capacity). However, adding some other parameters (i.e OS, host architecture) would require redefining a large portion of the framework, especially the comparison functions used in the different heuristic implementations. Furthermore, the only available implementation takes into account the non-sharing model, where each VM requires exclusive resources from the host (i.e. VMs can not share the cores or RAM of the host where they are placed in).

Moreover, the main difficulty in using the framework for more detailed analysis is the lack of notion of time. While the cloud services are being placed one by one as they arrive in the placement queue, there is no implementation of the lifetime of a given VM. That is, once a VM is placed on a host, it will run on the host and consume the host's resources during the full run of the simulation. Because of this implementation choice, the simulations are very fast on one hand, but are not simulation time-aware on the other, meaning the user can not create cloud services that are to be deployed in a specific time moment or stop cloud services after a given time period.

## III. Porting to a complex cloud computing simulation environment

Before starting the porting process, a decision needs to be made about the target simulation platform to which the Matlab implemented VM placement framework will be ported to. Of course, the first of the requirements for the simulator is that is has to be a free open-source solution that can be further extended. The most popular open source choices [10] according to different literature and the amount of interest in the research and education community, are CloudSim [11], GreenCloud [12], and iCanCloud [13].

The primary objective of the CloudSim [11] project is to provide a generalised, and extensible simulation framework that enables seamless modelling, simulation, and experimentation of emerging cloud computing infrastructures and application services. By using CloudSim, researchers and industry-based developers can focus on specific system design issues that they want to investigate, without getting concerned about the low level details related to Cloud-based infrastructures and services. The simulator is written in Java and has become a very popular choice for many different design decisions analysis ranging from pricing, power consumption, VM placement, multi-datacentre environments, etc.

GreenCloud [12] is a sophisticated packet-level simulator for energy-aware cloud computing datacentres with a focus on

cloud communications. It offers a detailed fine-grained modelling of the energy consumed by the datacentre IT equipment, such as computing servers, network switches, and communication links. However, while the power aspects of the datacentre are implemented meticulously, GreenCloud improves simulation accuracy by maintaining a packet structure with all its protocol headers and the consequent protocol processing rather than simulation scalability.

iCanCloud [13] is a simulation platform aimed to model and simulate cloud computing systems, which is targeted to those users who deal professionally close to those kinds of systems. The main objective of iCanCloud is to predict the trade-offs between cost and performance of a given set of applications executed in a specific hardware, and then provide to users useful information about such costs. This simulator encompasses a wide range of different characteristics of cloud datacentres and has a GUI that simplifies the set-up of simulation scenarios. Lots of the implemented features are focused on the hardware details and network storage. It is Java based but uses OMNet++ and INET in the background.

After a detailed analysis of all different implemented features in these three options, as well as an evaluation of the flexibility to extend them, available support, etc., the authors opted for using the CloudSim platform, that encompasses a very broad range of cloud datacentre descriptors and has a vibrant community and a wide range of available documentation. Moreover, CloudSim permits the simulation of large scale cloud datacentres without excessively penalising the simulation time, that is an important issue to consider when replicating simulations is a need for statistical accuracy.

Because of the complexity of the framework, the porting process needs to be implemented carefully by dividing it into several steps. Essentially, each placement algorithm needs to be independently re-implemented in CloudSim as a new type of VM allocation policy. In this way, the lower level of the placement framework will be fully implemented. Then, the community-aware first level needs to be implemented by adding community-awareness into the CloudSim simulator. This second step is very complex since it includes adding a lot of network and topology information about the datacentre which is currently missing or is relatively rudimentarily implemented in CloudSim. However, before moving to step two, the implementation of the existing algorithms needs to be completed and validated against the original Matlab implementation.

CloudSim defines a layered architecture that can be configured by the user in order to generate the desired scenario for the cloud datacentre to be tested. It permits the specification of many options depending on the user requirements, details like host resources (number of cores, RAM, host storage, bandwidth), hosts architecture, OS or Virtual Machine Manager (VMM). The user interface structure provides access to the Java classes that manage VM services, cloud application services and resources. The jobs that are submitted to the CloudSim simulated cloud are defined as *cloudlets* (cloud-based application services) and are executed on VMs. Details as the VM image size, required RAM, number of processing elements, cloudlet scheduling model or bandwidth, can be specified when defining VMs, while characteristics as the length of the cloudlet code, number of needed processing elements or the utilisation model for the CPU, RAM or bandwidth, can be customised when describing cloudlets.

For our porting process, there are several deployments in CloudSim (i.e. datacentres, VMs or hosts) that are already implemented in their corresponding Java classes and that can be easily customised or modified. This, obviously, simplifies the migration from the original framework scenario. However, there are other concepts that are not included in CloudSim as the notion of a closely collaborating VM group, that conforms a VM virtual community and that is defined as a *cloud service* in the Matlab implementation. This concept needed to be adapted to CloudSim, and we decided to use instead the entity that CloudSim provides named as *datacentre broker*, that manages the creation and execution of cloudlets on VMs. Apart from that, the role of brokers in our implementation also includes the function of managing the tenant VMs demanded from the same client.

## IV. Validation of the corresponding CloudSim extension

We present the validation of the re-implementation of one of the four heuristics available in the VM placement framework, namely BinCentric [14]. The multidimensional BinCentric objective is to place the largest not yet allocated VM to the smallest available host, where the size of both VMs and hosts is calculated as a normalised weighted sum of their resources.

Upon implementing the BinCentric heuristics, the validation of the implementation required an extension of CloudSim so that the Matlab placement scenarios could be recreated in the CloudSim environment.

### A. Simulation scenarios

The validation was done by comparing and analysing the output of a number of different simulation scenarios, defined to check that the CloudSim extension reacts in the same manner to the change of any of the defined input parameters of the VM placement framework. Table I summarises all different types of scenarios that were investigated and defines their input parameters. As it can be noticed there were simulations with two different datacentre sizes, and two different set-ups of host characteristics (8 cores, 16 GB RAM) and (16 cores, 32 GB RAM). As, for the generation of VMs, there are options of generating 1000 or 2000 cloud services (CS) and within each cloud service, a group of VMs was spawned where the number of VMs is chosen uniformly from the [1, maxVMs] interval.

The choices for the simulation scenarios were made based on the community-based VM placement framework results in [5] with the aim to be able to recapture the available published data [15]. The results of each scenario have been obtained by averaging the output of, at least, 3 simulation executions in order to avoid the inexactitudes that come along with the randomness of the distributions introduced.

### B. Simulation results

Due to the random nature of the generated scenario it is not possible to achieve full 100% match of the results since each run in each different implementation independently generates the queue of cloud services (and VMs) that need to be placed. However, by averaging across a number of run scenarios, if both implementations have the identical decision logic implemented,

TABLE I
SIMULATION SCENARIOS INPUT PARAMETERS

| scen no. | hosts | cores | RAM [GB] | no. of CS | max VMs |
|---|---|---|---|---|---|
| 1 | 5400 | 8 | 16 | 1000 | 10 |
| 2 | 5400 | 8 | 16 | 1000 | 20 |
| 3 | 5400 | 8 | 16 | 1000 | 30 |
| 4 | 5400 | 16 | 32 | 1000 | 10 |
| 5 | 5400 | 16 | 32 | 1000 | 20 |
| 6 | 5400 | 16 | 32 | 1000 | 30 |
| 7 | 5400 | 8 | 16 | 2000 | 10 |
| 8 | 5400 | 8 | 16 | 2000 | 20 |
| 9 | 5400 | 8 | 16 | 2000 | 30 |
| 10 | 5400 | 16 | 32 | 2000 | 10 |
| 11 | 5400 | 16 | 32 | 2000 | 20 |
| 12 | 5400 | 16 | 32 | 2000 | 30 |
| 13 | 10800 | 8 | 16 | 2000 | 10 |
| 14 | 10800 | 8 | 16 | 2000 | 20 |
| 15 | 10800 | 8 | 16 | 2000 | 30 |
| 16 | 10800 | 16 | 32 | 2000 | 10 |
| 17 | 10800 | 16 | 32 | 2000 | 20 |
| 18 | 10800 | 16 | 32 | 2000 | 30 |

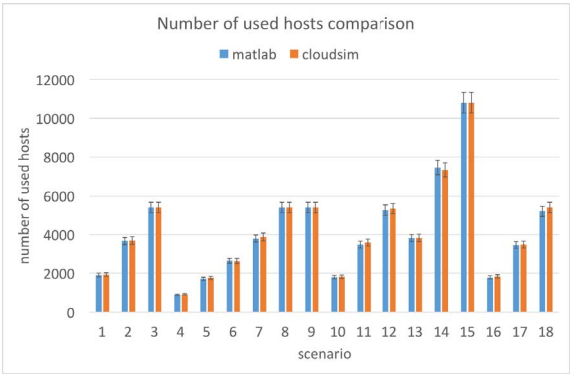



Fig. 1. Comparison of the number of used hosts for both implementations presented with 5% error bars.

the discrepancies of the averaged output results should fall well within the 95% confidence interval.

In Figure 1, the side by side comparison of the averaged number of used hosts for both implementations is presented together with 5% error bars. The exact 95% confidence interval for the runs is, in fact, smaller than the presented 5% error bars and falls very closely around the given average value. It is straightforward to conclude that both implementations yield to the same placement decisions and thus, the number of used hosts for the overall placement process of all defined cloud services is very similar.

In order to clearly demonstrate the negligible difference in the obtained results, in Figure 2, the relative difference in the number of used hosts for both implementations is shown. As it can be seen, the maximum relative difference is less than 4%, which is the result of the random nature of the generated scenarios.

Apart from the macro results validation, we went one step further and analysed the difference in the distribution of the number of placed VMs across the hosts in the datacentre. In Figure 3, the distribution of the VMs per host for the case study of scenario no. 4 is presented using 5% error bars. Again, the results show that there is a very high degree of similarity in the placement decision process with a normal like distribution around the value of 6 VMs per host.

Another detailed analysis of the same scenario is given in Figure 4, where we analyse the absolute difference in the

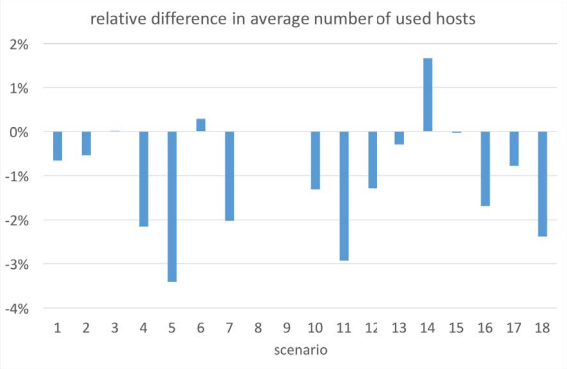



Fig. 2. Relative difference in the number of used hosts for both implementations.

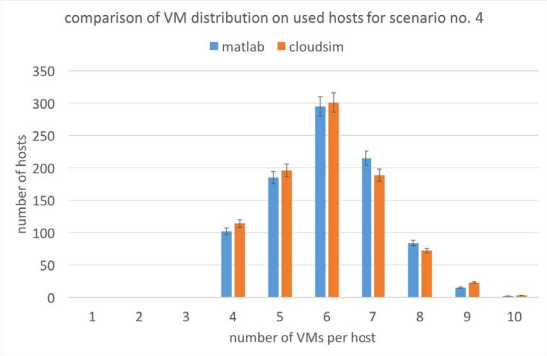



Fig. 3. Comparison of the VM distribution on the used hosts for scenario no. 4.

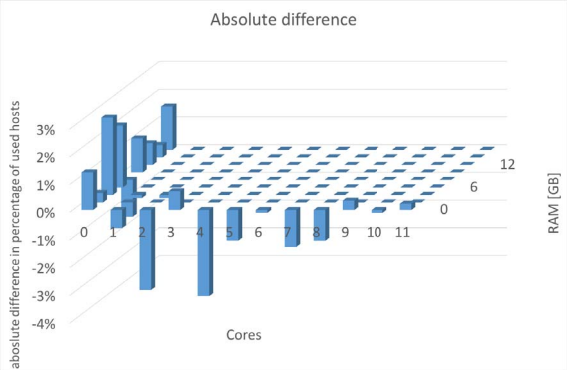
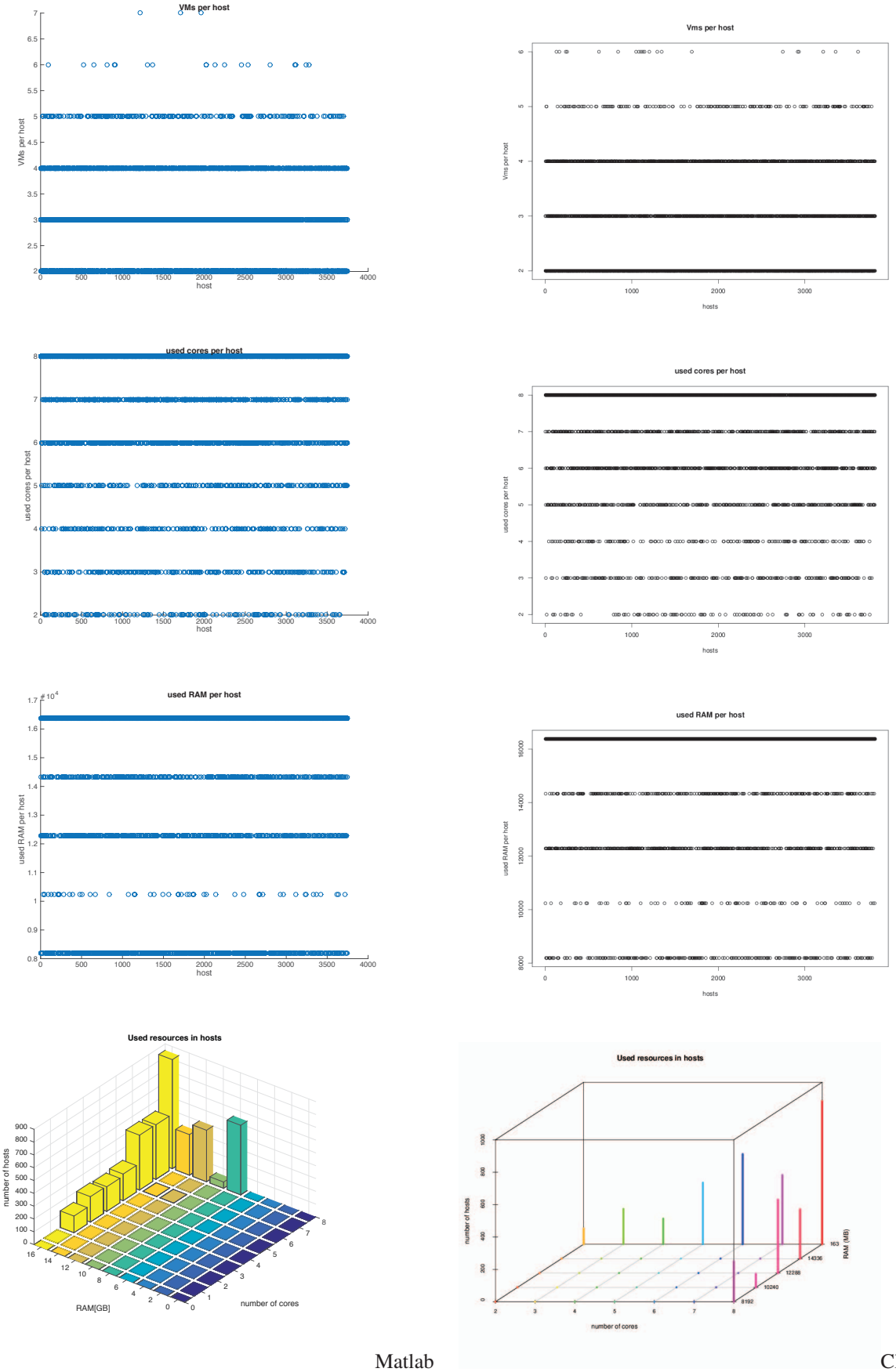



Fig. 4. Absolute difference in the remaining free resources of used hosts for scenario no. 4.

percentage of hosts that are left with a given combination of remaining free resources against the total number of used hosts. We obtained that the absolute difference is at maximum 3%, with the discrepancies symmetrically compensated across the two resources. It is important to note that, in both cases, almost all of the used hosts were left in a state where the remaining free resources are either cores, or RAM. Rarely there could be one to two cases of hosts where there is something left from both of their resources. This is an expected behaviour of the chosen BinCentric heuristic.

For a complete head to head comparison of two runs of the same simulation scenario (no. 14 is randomly chosen as a use case example) please refer to Figure 5, where the output results from the two implementations are placed side by side for visual inspection. There are four different comparisons presented: number of VMs per host, used cores per host, used RAM per host, and distribution of used resources across hosts.




Matlab CloudSim

Fig. 5. Side by side simulation results for different distributed resources for an example scenario no. 14.

## V. Benefits of the approach

The validation of the successful porting of one of the algorithms to CloudSim and the ability to reproduce the results have paved the way for the rest of the framework. The work done so far is a cornerstone project that has provided insight in the CloudSim architecture in great details, which is of utter importance when aiming to make a seamless extension that will enable the newly added component to interact with the rest of the system in a dynamic and flexible way without any impedance whatsoever.

The considerable efforts put into porting the community-based VM placement framework to CloudSim will have many-fold benefits. Once the whole framework is incorporated into CloudSim, we can analyse the effects of the chosen placement algorithms on a number of different aspects of the system which are not available in the original implementation of the framework. We can also try out the differences in performance when using the space sharing against the time sharing principle of providing resources to the VMs, since both of these mechanisms are implemented in CloudSim already.

The current version of CloudSim also implements the notion of datacentre network topology and network based process (cloudlet) intercommunication. Although this feature is a bit rudimentary, it is still a good start for investigating the amount of traffic that passes between the VMs and using this information for further optimisation of the system. Combined with the possibility to create and destroy cloud services on the fly during simulation time, this is the perfect background needed to try out the performance of the VM placement framework for real-time optimisation via migration of VMs.

However, one of the main reasons for this process of porting was to be able to analyse the effects of the framework on the power consumption within the datacentre. We have tested the power-aware models that come with CloudSim by replicating some of the previously simulated scenarios while changing the *VMs*, *hosts datacentre*, *broker datacentre* Java classes with the equivalent power-aware classes [16]. We have used the already implemented power profiles for the servers in order to ensure that our model can be adapted to the existent power-aware deployments in CloudSim, an with the intention of planning an in-depth study in this area considering energy-aware policies [17] applied to the original VM placement framework. We can predict that once the energy profiles of the hosts' hardware of the datacentre are fully investigated and implemented in the simulator, the results from the simulation are going to give us a much better insight in the research of power-aware cloud computing deployments.

## VI. Conclusion and future work

With the increasing complexity of the cloud datacentre environments it is of utter importance to make an in-depth analysis of the way each design decision affects the overall performance. For these purposes, upon performing a focused investigation on some specific problem or process that is an inherent part of the datacentre, one must put the solution in perspective by analysing it in a realistic simulation environment. In this way, the implications and interdependencies between different processes will become visible, and an expanded view with many more macro and micro parameters will be available. For instance, one of the main drivers in this field today is to investigate the effects of different components on the power-consumption of the complete datacentre.

Starting with this exact goal in mind, in this paper we describe our work in progress of porting a highly flexible, yet very efficient hierarchical VM placement framework to the CloudSim platform. Using carefully made design decisions and a step by step process with continuous validation, we aim to make the VM placement a seamlessly integrated part into CloudSim. This will enable us to analyse the framework performance using a large variety of different scenarios.

Furthermore, the readily available additional features that CloudSim offers will make the creation of federated clouds scenarios, where complex cloud services can be created and destroyed during simulation runtime, a fairly easy task that can offer a very wide range of insights into different aspects and impacts of the process of VM placement.

## References


[1] J. L. Lucas-Simarro, R. Moreno-Vozmediano, R. S. Montero, I. M. Llorente. Cost optimization of virtual infrastructures in dynamic multi-cloud scenarios. Concurrency and Computation: Practice and Experience, 27(9):2260–77, 2015.

[2] D. Puthal, B. P. Sahoo, S. Mishra, S. Swain. Cloud computing features, issues, and challenges: a big picture. In Computational Intelligence and Networks (CINE), 2015 International IEEE Conference on, 116–123, 2015.

[3] L. Ren, L. Zhang, F. Tao, C. Zhao, X. Chai, X. Zhao. Cloud manufacturing: from concept to practice. Enterprise Information Systems, 17;9(2):186–209, 2015.

[4] L. A. Barroso, J. Clidaras, U. Holzle U. The datacenter as a computer: An introduction to the design of warehouse-scale machines. Synthesis lectures on computer architecture, 31;8(3):1–54, 2013.

[5] S. Filiposka, A. Mishev and C. Juiz. Community-based VM Placement Framework. Journal of Supercomputing, 71:12, 4504–4528, 2015.

[6] B. Xia, Z. Tan Tighter bounds of the First Fit algorithm for the bin-packing problem. Discrete Applied Mathematics 158 (15): 1668–1675, 2010.

[7] M. Sindelar, R. K. Sitaraman, P. J. Shenoy. Sharing-aware algorithms for virtual machine colocation. In ACM Symposium on Parallelism in Algorithms and Architectures, 367–378, 2011.

[8] E. G. Coffman Jr, J. Csirik, G. Galambos, S. Martello, D. Vigo. Bin packing approximation algorithms: survey and classification. In Handbook of Combinatorial Optimization, 455–531. Springer New York. 2013.

[9] A. Greenberg, J. R. Hamilton, N. Jain, S. Kandula, C. Kim, P. Lahiri, D. A. Maltz, P. Patel, S. Sengupta. VL2: a scalable and flexible data center network. In ACM SIGCOMM computer communication review 39(4), 51–62), ACM. 2009.

[10] A. Ahmed, A. S. Sabyasachi. Cloud computing simulators: A detailed survey and future direction. In Advance Computing Conference (IACC), 2014 IEEE International, 866–872, IEEE. 2014.

[11] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. De Rose, and R. Buyya. CloudSim: A Toolkit for Modeling and Simulation of Cloud Computing Environments and Evaluation of Resource Provisioning Algorithms. Software: Practice and Experience (SPE) 41(1) 23–50, Wiley Press, New York, USA, January 2011.

[12] D. Kliazovich, P. Bouvry, and S. U. Khan A Packet-level Simulator of Energy-aware Cloud Computing Data Centers. Journal of Supercomputing, 62(3), 1263–1283. 2012.

[13] A. Nunez, J. L. Vazquez-Poletti, A. C. Caminero, G. G. Castane, J. Carretero and I. M. Llorente. iCanCloud: A Flexible and Scalable Cloud Infrastructure Simulator. Journal of Grid Computing, 10(1), 185–209. 2012.

[14] R. Panigrahy, K. Talwar, L. Uyeda, U. Wieder. Heuristics for vector bin packing. Microsoft research, 2011.

[15] S. Filiposka, A. Mishev, C. Juiz. Balancing Performances in Online VM Placement. In ICT Innovations 2015, 153–162. Springer International Publishing. 2016.

[16] A. Beloglazov, R. Buyya. Optimal Online Deterministic Algorithms and Adaptive Heuristics for Energy and Performance Efficient Dynamic Consolidation of Virtual Machines in Cloud Data Centers, Concurrency Computat.: Pract. Exper., 24, 1397–142. 2012

[17] J. M. Sola-Morena, K. Gilly, C. Juiz. Sustainability in Web server systems. Computers in Industry, 2014, 65, 401-407