Dynamic Windows Scheduling for Virtual Machine Placement*

Katia Leal[1][0000-0002-1405-6756], Jose Herrera[2][0000-0002-6547-6099], and Virginia Escuder[3]

[1] Universidad Rey Juan Carlos, ETSIT, Camino del Molino s/n, 28943 Fuenlabrada, Madrid, Spain - katia.leal@gmail.com

[2] IMDEA Networks, Avda. del Mar Mediterráneo 22, 28918 Leganés, Madrid, Spain - jose.herrera@imdea.org

[3] Universidad de Alcalá de Henares, EPS, Ctra. NII Km 31.600 28871, Alcalá de Henares, Madrid, Spain - virginia.escuder@uah.es

Abstract. We consider applying Windows Scheduling (WS) to Virtual Machine(VM) placement in cloud computing. In brief, the problem is to schedule the use of Physical Machines(PM) in terms of mips, processing elements, ram, storage and/or bandwidth to clients. Each client is characterized by an active cycle and a broker which is acting on behalf of that client. The broker hides virtual machine management, as VM creation, submission of cloudlets to VM and destruction of VMs. During the period of time that any given client is active, there must be transmissions from the broker to the cloud resource provider, for example, a Data Center, for VM creation and destruction. The goal of the Data Center Virtual Machine Allocation Policy is to minimize the number of PMs used for VM placement. With this online model, decisions are dynamic, we assume that VMs may be reallocated at some cost. We assume that such cost is a constant amount paid per reallocation. That is, we also aim to minimize the number of reallocations. We will present several online reallocation algorithms for VM placement. We will evaluate experimentally these policies showing that, in practice, all achieve constant amortized reallocations with close to optimal PM usage. Our simulations will also expose interesting trade-offs between reallocations and PM usage. We are developing the simulation model termed VirtualMachinePlacementSim (VMPSim) based on CloudSim to perform the experiments.

Keywords: VM placement · Online Windows Scheduling · Cloud Computing.

* This work has been supported by the European Commission under project ElasTest (H2020-ICT-10-2016, GA-731535); by the Regional Government of Madrid (CM) under project EDGEDATA-CM (P2018/TCS-4499) cofunded by FSE & FEDER; and by the Spanish Government under projects LERNIM (RTC-2016-4674-7) and BugBirth (RTI2018-101963-B-I00) cofunded by the Ministry of Economy and Competitiveness, FEDER & AEI. Also, this work has been partially supported by the Comunidad de Madrid grant EdgeData-CM (P2018/TCS-4499)

1 Introduction

In origin, the WS problem is to schedule the use of communication channels to clients. Each client $client_i$ is characterized by an active cycle (with arrival and departure times) and a window $window_i$ also called laxity[4] (in the context of job scheduling). During the period of time that any given client $client_i$ is active, there must be at least one transmission from $client_i$ scheduled in any $window_i$ consecutive time slots. The optimization criterion is to minimize the number of channels used. The WS problem appears in many areas such as Operations Research, Networks, Streaming, etc. More application details can be found in a previous publication [3] and the references therein.

In that paper, we presented three protocols for online WS with reallocation. Namely, *preemptive*, *lazy*, and *constant*. In brief, the first is a repeated (upon each client arrival or departure) application of the offline protocol in [1]. Aiming to minimize channel usage, clients are preemptively reallocated to guarantee the same offline packing. Instead, in *lazy*, clients are not reallocated as long as a maximum number of channels in use is not exceeded. The idea is to save reallocations taking advantage of all possible channels. Finally, *constant* is designed to guarantee an amortized constant number of reallocations. The main approach is to classify clients by laxity. Thus, we plan to apply these algorithms to VM placement in cloud computing. In this case, we will schedule the use of PMs, first in terms of mips. The goal is to minimize the number of PMs used and to minimize the number of reallocations.

We will evaluate experimentally all three protocols against default and well known VM placement schemes in cloud computing [8]. In the past, we have successfully developed the CommunicationChannelsSim [3] simulation models to study the dynamic WS with reallocations, the FederatedGridSim [4] for resource sharing in Federated Grids, and the GridWaySim [7] for decentralized scheduling in Federated Grids. At this time, we are working in the VMPSim, which is based on the CloudSim 4.0 [2] , to study the VM placement protocols proposed.

2 VMPlacementSim

At the time of writing this paper, we have already implemented a default VM Allocation Policy for VM placement in our VMPSim simulator. The implementation of WS algorithms on a real environment requires the involvement of a large number of users and resources, which is very hard to coordinate and build, and would hinder the repeatability of experiments. Thus, as we did before, we will evaluate the WS algorithms, and even further extensions, through simulations. The results of this evaluation will later encourage or discourage the implementation on a real production environment.

[4] The term *laxity* can be used instead of *window* since it conveys better the concept of maximum delay between transmissions. Nonetheless, we will use the term *window* when describing the related literature.

The simulations are performed from the point of view of one user, which submits VM creation or destruction to the broker. The VMs are loaded from an XML input file. For our experiments, we have created three different input files each containing 4000 clients with different laxities, arrival (VM creation time) and departure times (VM destruction time). For each client, we defined arrival/departure times and laxity aiming to simulate multiple real world situations, such as bursts of arrivals, bursts of departures, a stable heavily loaded system, and an almost empty system.

For each VM arriving or leaving the system, the VM Allocation Policy performs the required action according to the specific WS algorithm simulated. Also, we will implement default and well known VM placement schemes in cloud computing to validate our results: minimize the number of PMs used and minimize the number of reallocations.

Finally, we also plan to make an energy consumption analice of our proposed algorithms, as we have done previously [5, 6]. More than ever, energy aware scheduling and placement policies are required.

References

1. Bar-Noy, A., Ladner, R.E.: Windows scheduling problems for broadcast systems. SIAM Journal on Computing **32**(4), 1091–1113 (2003)
2. Calheiros, R.N., Ranjan, R., Beloglazov, A., Rose, C.A.F.D., Buyya, R.: "CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms". Software Practice and Experience **41**, 23–50 (2010)
3. Farach-Colton, M., Leal, K., Mosteiro, M.A., Thraves, C.: " Dynamic Windows Scheduling with Reallocations". In: Symposium on Experimental Algorithms (SEA 2014). vol. 8504, pp. 99–110. Lecture Notes in Computer Science (LNCS) (2014)
4. Leal, K.: "Anticipating resource saturation in Federated Grids". Future Generation Computer Systems **45**, 114–122 (April 2015)
5. Leal, K.: "Energy efficient scheduling strategies in Federated Grids". Sustainable Computing: Informatics and Systems **9**, 33–41 (March 2016)
6. Leal, K., Herrera, J.: " Calculating scheduler's contamination rate.". In: International Conference on Sustainable Energy and Environment Sensing (SEES 2018). ISBN: 978-1-912532-01-8 (2018)
7. Leal, K., Huedo, E., Llorente, I.M.: "A Decentralized Model for Scheduling Independent Tasks in Federated Grids". Future Generation Computer Systems **25**(8), 840–852 (September 2009)
8. Masdari, M., Nabavi, S.S., Ahmadi, V.: "An overview of virtual machine placement schemes in cloud computing". Journal of Network and Computer Applications **66**, 106–127 (2016)