Cluster Comput (2018) 21:1–13
https://doi.org/10.1007/s10586-017-0882-8

Support for spot virtual machine purchasing simulation

Ao Zhou[1] · Shangguang Wang[1] · Qibo Sun[1] · Jinglin Li[1] · Qinglin Zhao[2] · Fangchun Yang[1]

Received: 20 January 2017 / Revised: 18 April 2017 / Accepted: 20 April 2017 / Published online: 18 May 2017

Abstract With the rapid progress of cloud computing technology, a growing number of big data application providers begin to deploy applications on virtual machines rented from infrastructure as a service providers. Current infrastructure as a service provider offers diverse purchasing options for the application providers. There are mainly three types of purchasing options: reserved virtual machine, on-demand virtual machine and spot virtual machine. The spot virtual machine is a specific type of virtual machine that employs a dynamic pricing model. Because can be stopped by the infrastructure as a service providers without notice, the spot virtual machine is suitable for large-scale divisible applications, such as big data analysis. Therefore, spot virtual machine is chosen by many big data application providers for its low rental cost per hour. When spot virtual machine is chosen, a major issue faced by the big data application providers is how to minimize the virtual machine rental cost while meet service requirements. Many optimal spot virtual machine purchasing approaches have been presented by the researchers. However, there is a shortage of simulators that enable researchers to evaluate their newly proposed spot virtual machine purchasing approach. To fill this gap, in this paper, we propose SpotCloudSim to support for dynamic virtual machine pricing model simulation. SpotCloudSim provides an extensible interface to help researchers implement new spot virtual machine purchasing approach. In addition, SpotCloudSim can also study the behavior of the newly proposed spot virtual machine purchasing approaches. We demonstrate the capabilities of SpotCloudSim by using three spot virtual machine purchasing approaches. The results indicate the benefits of our proposed simulation system.

Keywords Cloud computing · Virtual machine · Big data analysis · Simulator · Dynamic pricing model

✉ Ao Zhou
aozhou@bupt.edu.cn

Shangguang Wang
sgwang@bupt.edu.cn

Qibo Sun
qbsun@bupt.edu.cn

Jinglin Li
jlli@bupt.edu.cn

Qinglin Zhao
zqlict@hotmail.com

Fangchun Yang
fcyang@bupt.edu.cn

[1] State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing, China

[2] Faculty of Information Technology, Macau University of Science and Technology, Avenida Wei Long, Taipa, Macau, China

1 Introduction

Cloud computing has emerged as a new paradigm for delivery of virtual resources and applications to end users in a pay-as-you-go model [1,2]. Migrating from a traditional model to the cloud model can help to save the infrastructure investments and reduce the maintenance complexity [3,4]. A growing number of big data application providers begin to deploy applications on virtual machines rented from the infrastructure as a service providers. The big data application providers pay for the virtual machines according to their use, likewise utilities such as electricity, gas, and water [5,6]. When cloud computing is adopted, a major issue faced by the big data application providers is how to minimize the virtual machine rental cost while meet service requirements [7,8].

Current infrastructure as a service providers offers diverse purchasing options for the application providers [9,10]. There are mainly three types of purchasing options: reserved virtual machine, on-demand virtual machine and spot virtual machine. The price of spot virtual machine changes dynamically and periodically based on current demand and supply [11,12]. The price of virtual machine is determined through an auction. If an application provider wins the bid, it obtains the right to use the virtual machines for which it bid [13]. The infrastructure as a service provider charges each hour by the last price. The virtual machine is stopped by the infrastructure as a service provider immediately when the bid is less than the current price. Therefore, spot virtual machine is suitable for large-scale divisible applications, e.g. big data analysis [14,15]. When the virtual machine is stopped in out-of-bid condition, the infrastructure as a service provider does not charge the latest partial hour. Spot virtual machine is chosen by many big data application providers for its low rental cost per hour. Many optimal spot virtual machine purchasing approaches have been presented by the researchers [16,17].

However, current cloud simulator can only support for reserved virtual machine purchasing and on-demand virtual machine purchasing. There is a shortage of simulators that enable researchers to evaluate the newly proposed spot virtual machine purchasing approach. To overcome these limitations of current cloud simulators, we propose SpotCloudSim, a CloudSim-based system which can support for spot virtual machine purchasing simulation. SpotCloudSim provides an extensible interface to help researchers implement the newly proposed spot virtual machine purchasing approach. In addition, SpotCloudSim can also study the behavior of the newly proposed spot virtual machine purchasing approaches. We demonstrate the capabilities of SpotCloudSim by using three spot virtual machine purchasing approaches. The results indicate the benefits of our proposed simulation system.

The rest of the paper is organized as follows. We discuss related work in Sect. 2. The background is introduced in Sect. 3. The design detail of our proposed SpotCloudSim is introduced in Sect. 4. The implementation detail of our proposed SpotCloudSim is introduced in Sect. 5. Section 6 gives an overview of our experiments. Finally, we conclude in Sect. 7.

2 Related work

In this section, related simulators are discussed. From past many years, distributed computing and Grid computing have gained widespread concern from the researches. Several simulators have been proposed to support the research and development of Grid computing. For example, SimGrid [18] and GridSim [19] are renowned grid simulators. However, none of them support for virtual and physical machine simulation, which is required by cloud computing environment.

To address the problem, a famous cloud simulator named CloudSim [20,21] has been proposed. CloudSim is a cloud simulator developed by the CLOUDS Laboratory of University of Melbourne. CloudSim provides an extensible simulation framework that enables the simulation of cloud computing infrastructures and cloud application services. Researchers can evaluate their virtual machine purchasing approaches or resource allocation approaches by using CloudSim.

However, CloudSim still has some limitations. For example, CloudSim can only support for reserved virtual machine purchasing and on-demand virtual machine purchasing. Dynamic virtual machine pricing model cannot be supported by CloudSim. Fortunately, the extendibility of Cloudsim allows the researchers to add new features to it. Many simulators have been proposed by extending CloudSim. For example, Cloud2Sim, WorkflowSim, and CloudAuction.

In Cloud2Sim [22], a distributed concurrent architecture is added to simulations. By extending CloudSim, Cloud2Sim can have several virtual machines execute the same task. In addition, Cloud2Sim can elastically scale the virtual resources in the simulation.

WorkflowSim [23] can support of workflow execution in cloud datacenter. Workflow engine, workflow parser and job scheduler are implemented in WorkflowSim. The researcher can set delays and other relative attributes in WorkflowSim.

CloudAuction [24] is a little similar to our simulator. CloudAuction can handle auction based service and help to allocate virtual machines to participants based on the combinatorial double auction principle. The researchers can design an optimized auction mechanism by implementing the interface provided by CloudAuction. In addition, researchers can study whether the newly proposed mechanism can satisfy other users and providers. However, CloudAuction is not able to evaluate the spot virtual machine purchasing approach.

There are many other cloud simulators [25–27]. However, none of the current cloud simulator can simulate the spot virtual machine purchasing perfectly. SpotCloudSim is presented by this paper to address the problem. We will illustrate the design of SpotCloudSim in the next section.

3 Preliminaries

3.1 Framework of CloudSim

Figure 1 illustrates the framework of CloudSim. As shown in Fig. 1, CloudSim consists of four layers, which are the cloud resource layer, the cloud service layer, the virtual machine service layer, and the user interface structures layer, respectively. In CloudSim simulation, a data center consists of host servers and switches. All the host servers and switches are connected by datacenter network. The virtual machines that

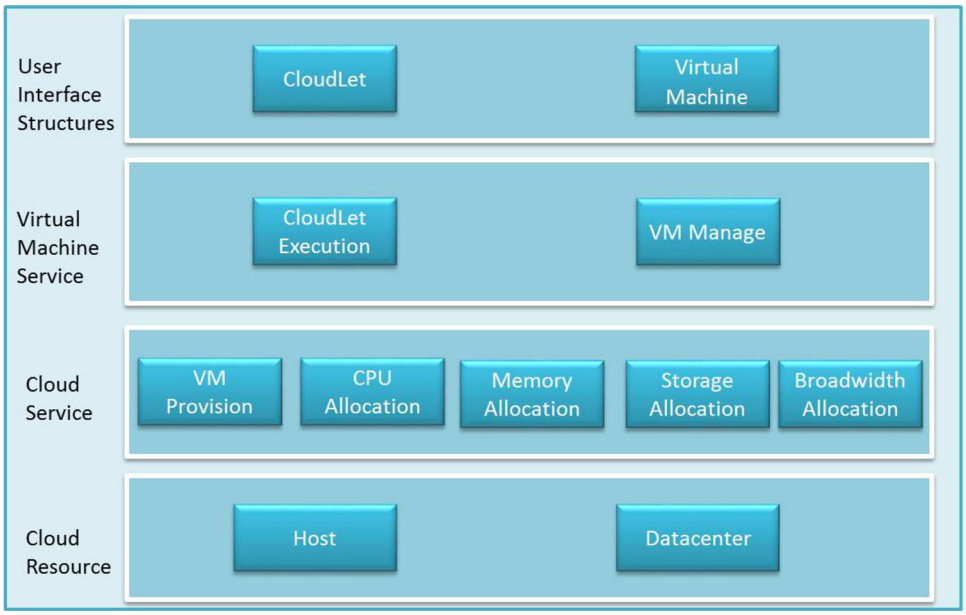


Fig. 1 Framework of CloudSim

are placed on the same host server share the physical resource with each other. The physical resources are allocated to the virtual machines based on the allocation policies in the cloud service layer. All the allocation policies can be re-defined by the users.

3.2 Spot virtual machine purchasing

Because spot virtual machine is suitable for large-scale divisible applications, spot virtual machine is chosen by many big data application providers for its low rental cost per hour. As illustrated in Fig. 2, the price of spot virtual machine changes dynamically and spot virtual machine is purchased through an auction [28,29]. After receives a job from the end user, the application provider decides how to bid for the spot virtual machine at the beginning of each bidding interval. When the bid is higher than the current price ($t < 240$), the application provider obtains the right to use the virtual machine. The job is processed by the virtual machine. When the bid is lower than the current price ($t > 240$), the virtual machine is immediately stopped by the infrastructure as a service provider. The job execution process is interrupted. Therefore, the computation is lost, and the job should be restarted from the beginning. In addition, the job execution time increases and the reliability is low. To overcome the problem, the checkpoint interface is provided by the infrastructure as a service provider. The application provider can employ the checkpoint mechanism to enhance the service reliability [30,31]. When the checkpoint mechanism is adopted, the interrupted job can be restarted from the latest save checkpoint image in out-of-bid condition. Notice that the application provider can change the bid. We just take the fixed-bid strategy for example.

4 Design of SpotCloudSim

4.1 Framework of SpotCloudSim

As shown in Fig. 3, SpotCloudSim has added seven modules to CloudSim: rental fee charging module, spot virtual machine pricing module, spot virtual machine allocation module, out-of-bid event trigging module, purchasing decision-making module, coping strategy calculation module and experimental results displaying module. We will described the function of each module in this section.

4.2 Rental fee charging module

The rental fee charging module charges the application provider based on the following rules: (1) The virtual machine pricing module charges each hour by the last price. (2) When the virtual machine is stopped by the infrastructure as a service provider, the virtual machine pricing module does not charge the latest partial hour. (3) The virtual machine pricing module charges the last partial hour when an application provider terminates a virtual machine. (4) The rental fee of a partial-hour is the same as a full-hour. This module is extensible. The researcher can implement his own rental fee charging rule by extending the default one.

4.3 Spot virtual machine pricing module

Because the price of the spot virtual machine changes with time, the virtual machine pricing module decides the current price for each type of virtual machine. The module can import the historical prices of Amazon spot virtual machine from

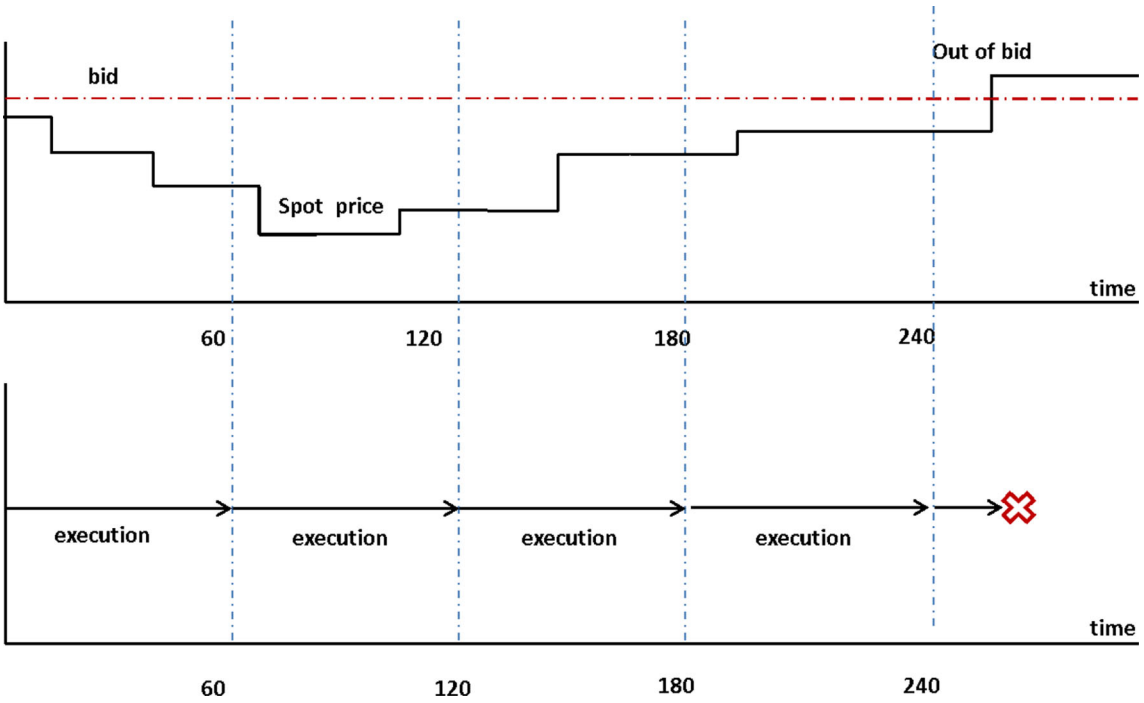


Fig. 2 Execution process under dynamic pricing model

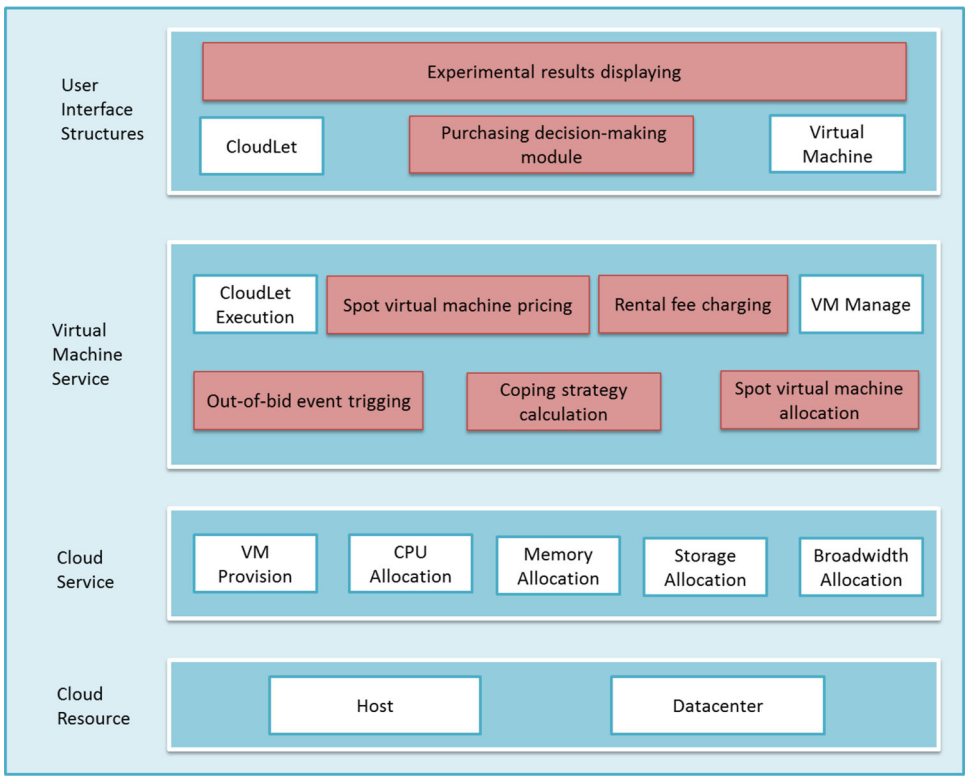


Fig. 3 Framework of SpotCloudSim

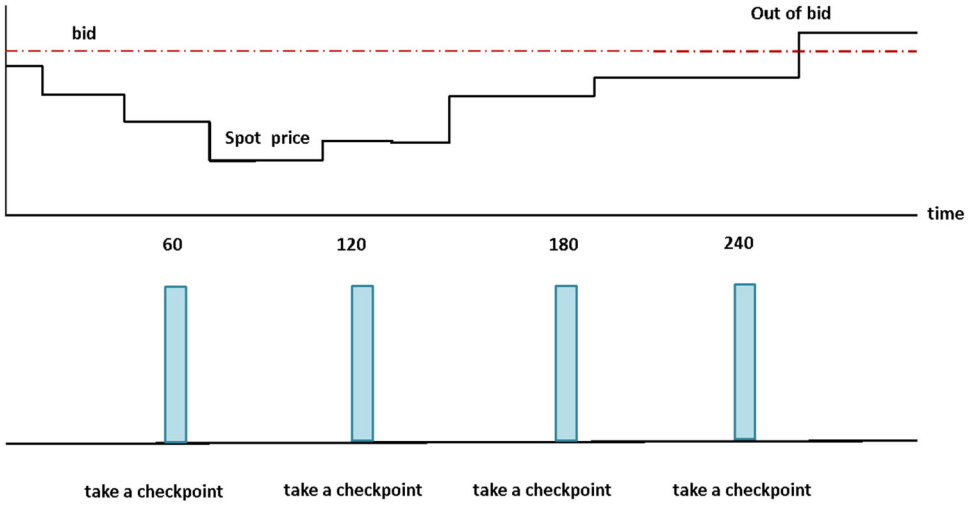


Fig. 4 Taking checkpoint periodically

.xml file, and set the current price of virtual machine based on the file.

4.4 Spot virtual machine allocation module

The spot virtual machine allocation module collects the bids from the application providers at the beginning of each bidding interval. Then, it allocates the virtual machines to the application providers based on the pre-defined rules. When the bid of an application provider is higher than the current price, the spot virtual machine allocation module allocate the virtual machines for the application provider as required. The spot virtual machine allocation module is extensible. The researcher can re-define the allocation rules as needed.

4.5 Out-of-bid event trigging module

When the bid of an application provider is less than the current price, the out-of-bid event trigging module stops the virtual machine immediately without any notice. In addition, the physical resources that have been allocated to the virtual machine are released.

4.6 Purchasing decision-making module

The prices of spot virtual machines fluctuate dynamically based on current demand and supply. The price of spot virtual machine is determined through an auction. An application provider can partially control the rental cost by setting the bid price. However, a lower bid price may lead to out-of-bid. Part of the computation is lost, and the total task execution time increases.

The purchasing decision-making module decides how to purchase the spot virtual machine based on the service requirements. For example, the application provider can set current bid price as the highest historical spot price if current task requires a quick response time. Otherwise, application provider can consider how to avoid high price, and purchase the spot virtual machine when the price is low. We have implement several basic purchasing decision-making strategies, such as random strategy, average price strategy and highest price strategy, in SpotCloudSim. The purchasing decision-making module is extensible. The researcher can implement his own spot virtual machine purchasing approach by extending the default ones.

4.7 Coping strategy calculation module

When an out-of-bid event happens, the virtual machine is stopped by the infrastructure as a service provider immediately without notice. The application provider should deal with the cost-reliability trade-offs. The infrastructure as a service provider, such as Amazon, provide a checkpoint interface to reduce the negative effects. Figures 4 and 5 illustrate two types of checkpoint-based strategy. In Fig. 4, a check-point is taken periodically. In Fig. 5, a check-point is taken when the spot price rises compared to the latest historical spot price. We add the coping strategy calculation module to support the checkpoint mechanism. We have implemented two basic checkpointing strategies in SpotCloudsim: taking checkpoint periodically and taking a checkpoint in rise price. By implementing the provided interface, the researchers can decide when to save the execution state of a virtual machine. The checkpoint interval has impact on the total rental cost. The task execution is suspended when a checkpoint is being taken. Spotcloudsim includes checkpoint as cost. Therefore, the application provider should make trade-off between the lost time for checkpoint taking and the lost time for out-of-bid.

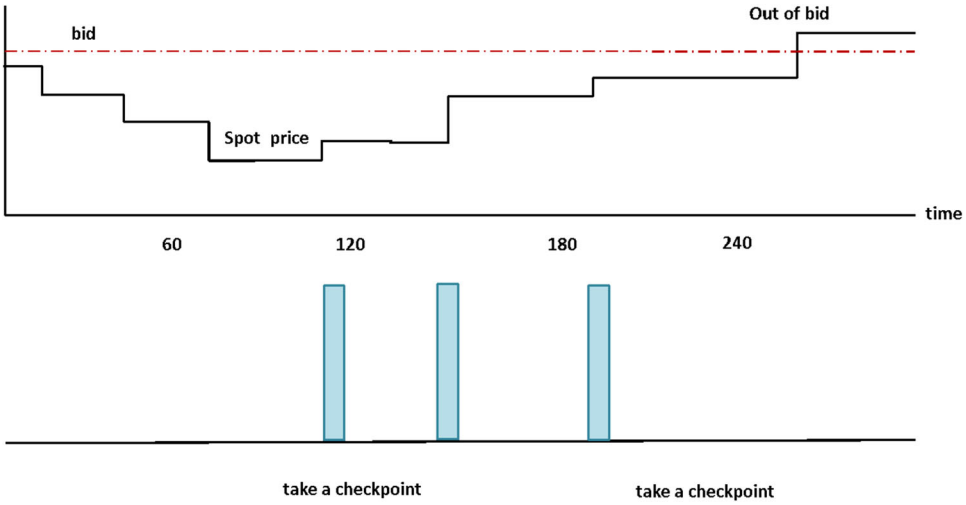


Fig. 5 Taking a checkpoint in rise price

4.8 Experimental results displaying module

Experimental results displaying module illustrates the simulation results to the user. In cloud computing environments, application providers should minimize the virtual machine rental cost while meet service requirements.

In other words, the task execution time should satisfy the service level agreement. Therefore, SpotCloudSim provides three types of performance metric to highlight the advantages and shortcomings of each spot virtual machine purchasing approach. The first performance metric type is the ability to save rental cost. This type of metric includes the rental cost (the rental fee the application provider has paid for completing each task), the total rental cost (the total rental fee the spot virtual machine purchasing approach has paid for completing all tasks). The second performance metric type is related to service level agreement satisfying. This type of metric includes the execution time (the time the spot virtual machine purchasing approach takes to complete each task), the total execution time (the total time the spot virtual machine purchasing approach takes to complete all tasks), the average lost time (the total lost time because of out-of-bid), and the out-of-bid frequency.

5 Implementation of SpotCloudSim

The class diagram of SpotCloudSim is illustrated in Fig. 6. The Bidder class represents an application provider. The Cashier class charges the application provider at the end of each charge interval. The Pricer class decides the current price for the spot virtual machines. The PriceProcesser class reads the historical spot prices from the excel file. The OBTrigger class triggers an out-of-bid event when the bid is lower than the current price. The Logger class is extended to the Recorder class. The Recorder class stores the results to the .log file and the database. The VmAllocationPolicy class is extended to the SpotVmAllocationPolicy class. The SpotVmAllocationPolicy class allocates the spot virtual machines to the application providers based on the bid and the current price. The CheckpointingScheduler class saves the state of the virtual machine based on the pre-defined strategy. We extend the class CloudSimTag and CloudSimConfig to add some configuration parameters.

We now describe the basic execution sequence of SpotCloudSim. The sequence diagrams are illustrated in Figs. 7 and 8. Figure 7 illustrates Spot virtual machine purchasing sequence. Firstly, the Bidder sets the highest acceptable per hour rental fee. Secondly, the Pricer calculates the current price of each type of spot virtual machine. For each bidder, if his bid is higher than the current price, the VmAllocationPolicy allocates the virtual machines for the application provider as required. Then, the Cashier charges the bidder at the end of each charge interval. If the bid is lower than current price and the application provider has already occupied the virtual machine, the OBTrigger stops virtual machine and releases the physical resource without notice. The Recorder stores required information to the database. Figure 8 illustrates the sequence of taking a checkpoint in rise price. The Pricer calculates the current price of each type of spot virtual machine. If the spot has changed, a price update event is sent to CheckpointScheduler. The CheckpointScheduler checks whether the price has risen. If the current price is higher than the latest historical spot price, a checkpoint-taking event is sent to relative virtual machine. The virtual machine starts to take a checkpoint as soon as the event has been received.

To validate and understand the effectiveness of SpotCloudSim, we conduct a set of experiments on real world spot dataset. We will discuss the experiment results in the next section.

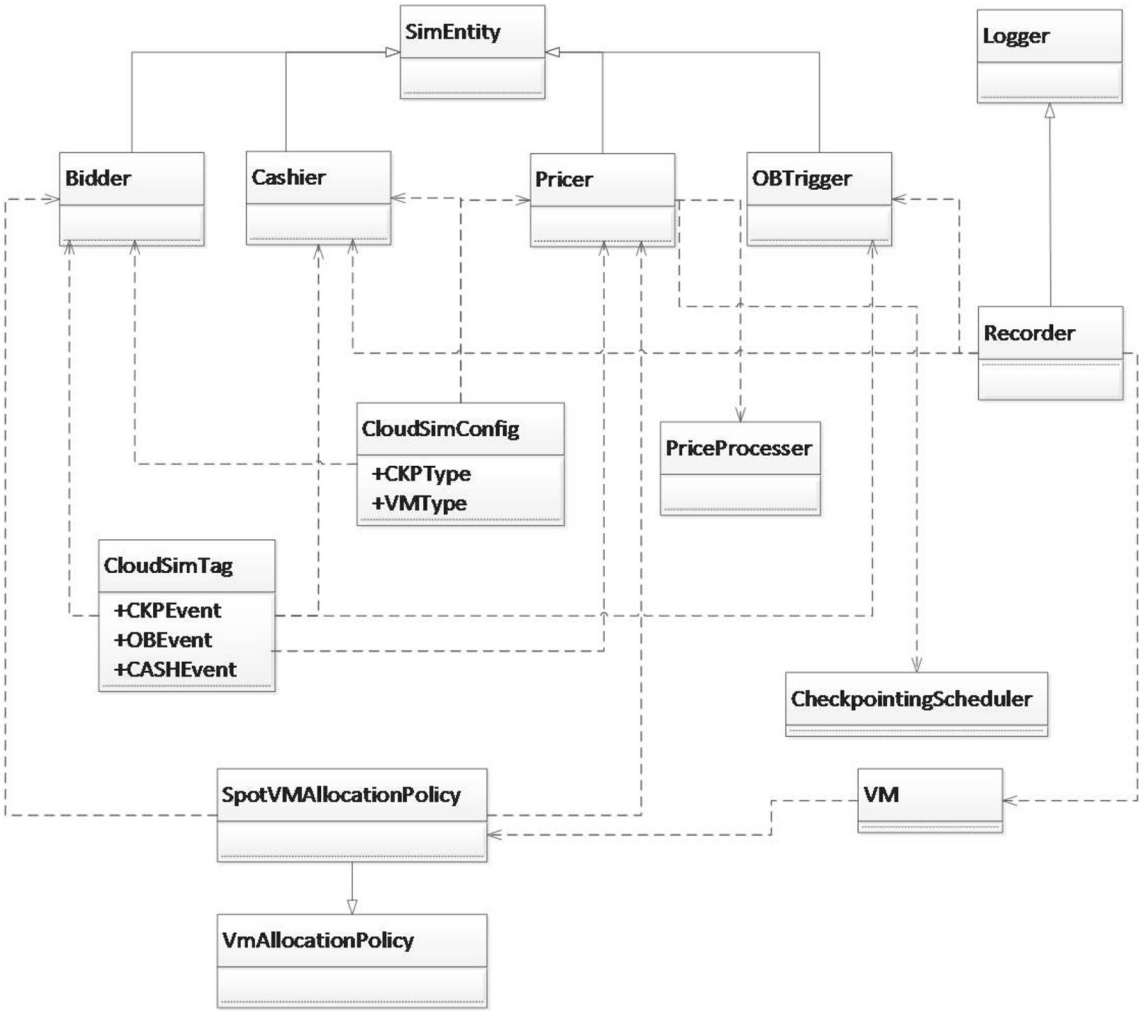


Fig. 6 Class diagram of SpotCloudSim

6 Experiments

In this section, we illustrate the new features of SpotCloudSim by using three spot virtual machine purchasing approaches. Firstly, we describe the experimental setup. Then, experimental results are analyzed.

6.1 Experimental setup

All comparisons are carried out under the same experimental environment. The CPU is the Intel Core i7-5500U 2.4GHz, the RAM size is 8 GB, and the operating system is Windows 8. We experiment with three datasets: (1) dataset 1: the spot prices of eu linux m2.2xlarge, Amazon, from 2011 January 1 to 2011 January 30; (2) dataset 2: the spot prices of us suse m2.4xlarge, Amazon, from 2011 January 1 to 2011 January 30; (3) dataset 3: the spot prices of eu linux m2.4xlarge, Amazon, from 2011 January 1 to 2011 January 30. In our experiment, we generate 40 tasks. For each type of virtual machine, the task size is uniformly distributed between 200 min to 400 min. The checkpoint taking time is 2 min. We will demonstrate the capabilities of SpotCloudSim by using the following three spot virtual machine purchasing approaches:

- *HP (Highest Price)*: HP always selects the highest value of the historical spot prices as the bid price.
- *AP (Average Price)*: AP always selects the average value of the historical spot prices as the bid price.
- *RP (Random Price)*: RP always randomly generates a price between the highest price and the lowest price as the bid price.

Two types of reliability enhancement strategies are employed by these approaches:

- *None:* When an out-of-bid event occurs, the interrupted task is restarted from the beginning.
- *Period:* A checkpoint is taken every 30 min.

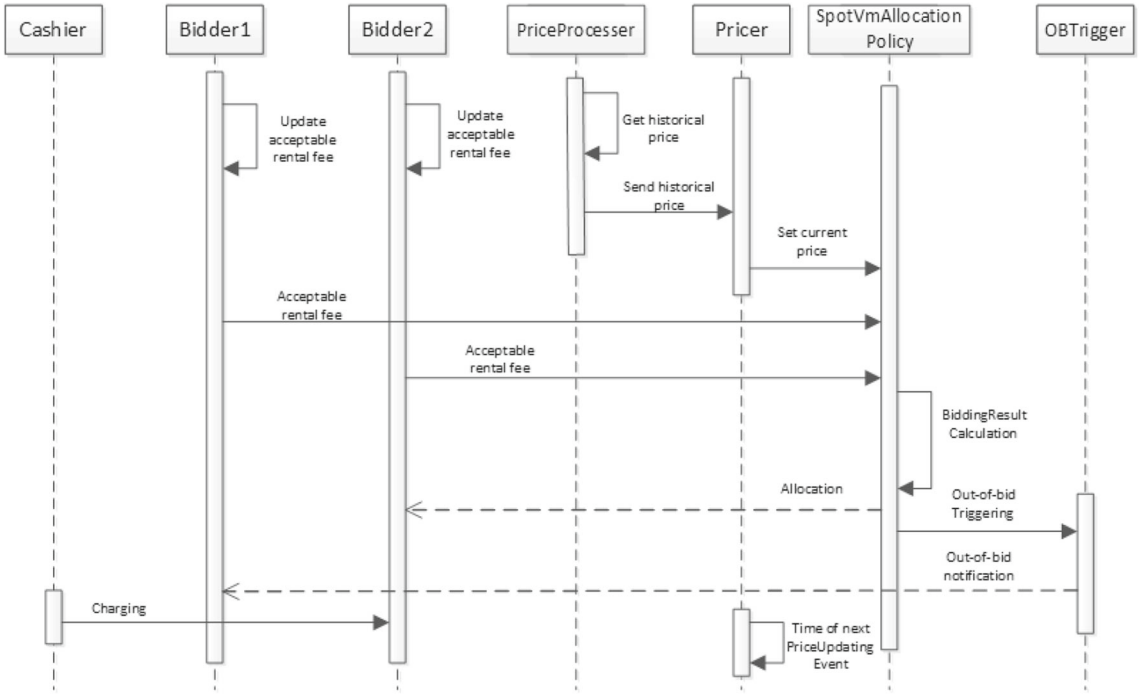


Fig. 7 Spot virtual machine purchasing sequence diagram

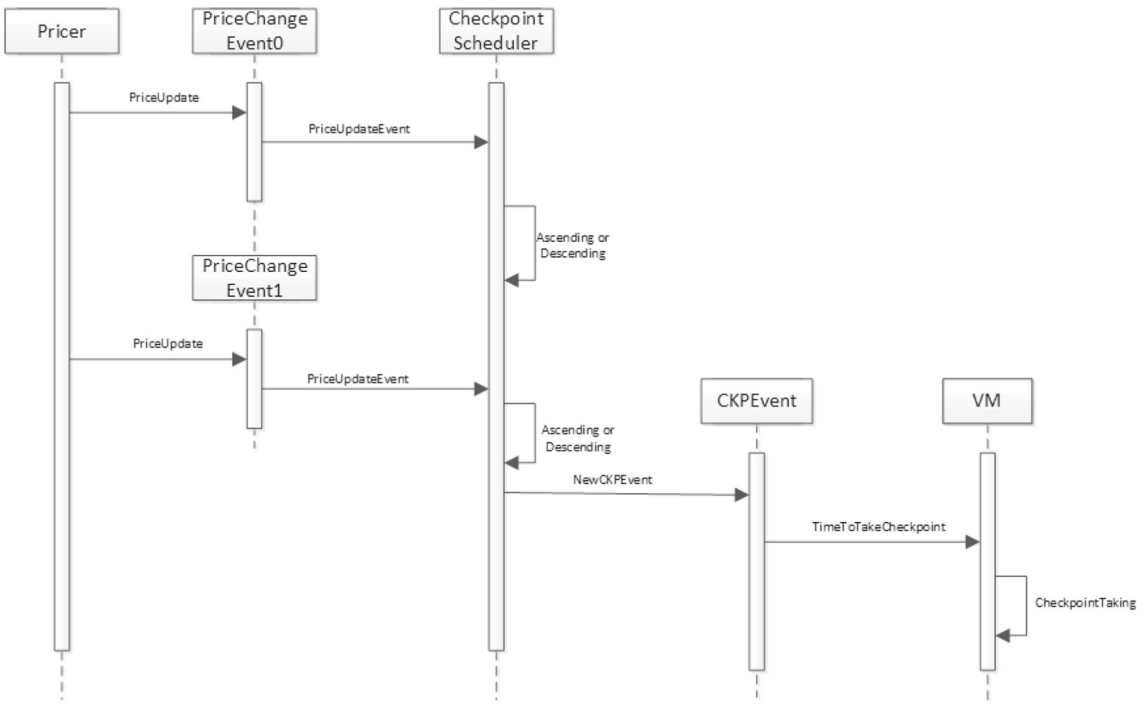


Fig. 8 Taking a checkpoint in rise price sequence diagram

We will evaluate the three spot virtual machine purchasing approaches by using the following two metric:

- *Total rental cost.* The total rental rate paid by the application provider for completing all the tasks submitted by the end users. Total rental cost can be calculated as follows:

$$Rental = \sum_{i=1}^{n}[rental(T_i)] \quad (1)$$

where T_i denotes a task, and n denotes the total number of tasks. $rental$ returns the rental rate paid by the application provider for completing T_i.

- *Total execution time.* The total time the approach takes to complete all the tasks submitted by the end users. Total execution time can be calculated as follows:

$$Time = \sum_{i=1}^{n} t_{end}(T_i) - \sum_{i=1}^{n} t_{start}(T_i) \quad (2)$$

where T_i denotes a task, and n denotes the total number of tasks. t_{start} is the time the task is received, t_{end} is the time the task is completed.

6.2 Impact of the bidding strategy

To study the impact of the bidding strategy on the total rental cost and the total execution time, we experiment on different spot datasets. No reliability enhancement strategy is employed. Figures 9 and 10 shows the experimental results. The experimental results in Figs. 9 and 10 show that: (1) the rental cost and the execution time are influenced by the bidding strategy; (2) HP outperforms other approaches consistently in the execution time and the rental cost; (3) AP performs poorly in the execution time and the rental cost. The average price is always lower or equal to the highest price. Therefore, the bid price of AP and RP are always lower or equal to the bid price of HP. When an application provider has a very low bid price, there is a higher chance that the bid price is lower than the final price of the spot virtual machine. The spot virtual machine frequently becomes unavailable. Therefore, the fraction of time that a virtual machine is available decreases with the decrease of the bid price. The continuous available periods of the spot virtual machine are relatively short, and the execution time becomes larger. Thats why HP outperforms AP and RP.

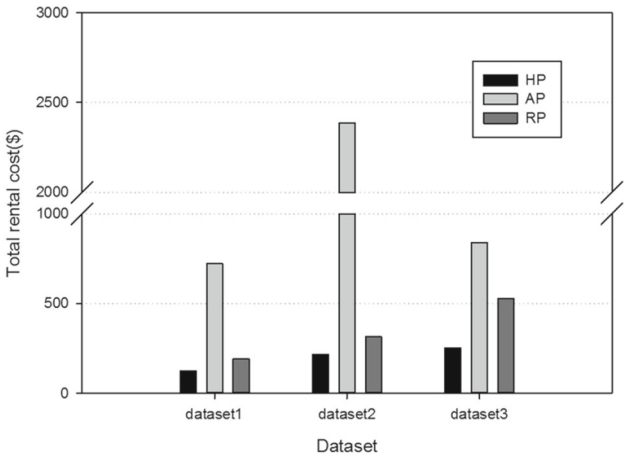


Fig. 9 Impact of the bidding strategy on the total rental cost. The figure illustrates the total rental cost of all spot virtual machine purchasing approaches under different spot price datasets

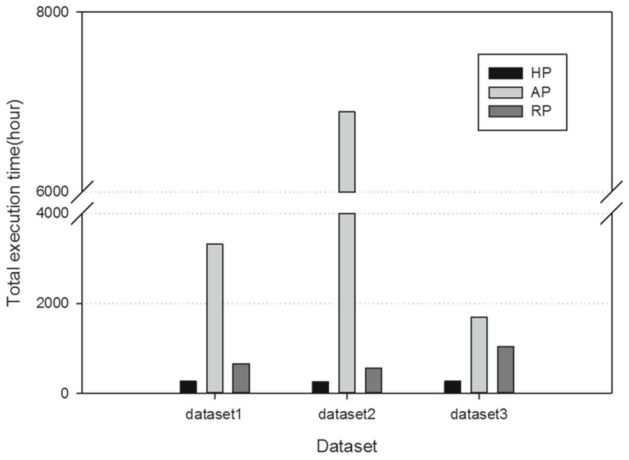


Fig. 10 Impact of the bidding strategy on the total execution time. The figure illustrates the total execution time of all spot virtual machine purchasing approaches under different spot price datasets

6.3 Impact of the task number

The rental cost and the execution time are also influenced by the task number. To study the impact of the task number on the bidding results, we vary the task number from 10 to 40 with a step value of 10 in our experiment. No reliability enhancement strategy is employed. The performance of all the approaches is studied. Figures 11, 12, 13, 14, 15 and 16 show the performance of the approaches. The experimental results of Figs. 11, 12, 13, 14, 15 and 16 show that: (1) the rental cost and the execution time are significantly influenced by the task number; (2) HP still outperforms AP and RP consistently; (3) although the rental cost saved from a single task is not significant, the saved rental cost is proportional to the number of tasks. Therefore, much rental cost is saved when there are a large number of tasks.

6.4 Impact of the checkpointing strategy

In this section, we study the impact of the checkpointing strategy on the total rental cost and the total execution time.

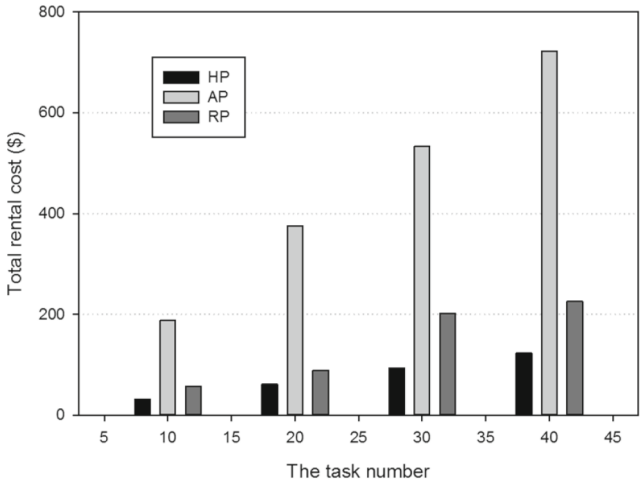


Fig. 11 The total rental cost of all spot virtual machine purchasing approaches under dataset 1

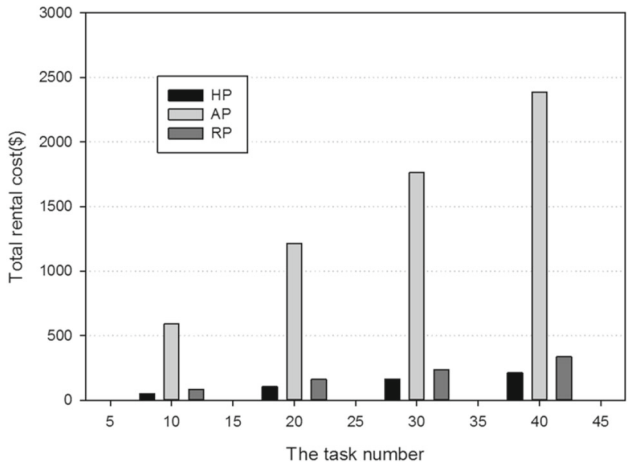


Fig. 12 The total rental cost of all spot virtual machine purchasing approaches under dataset 2

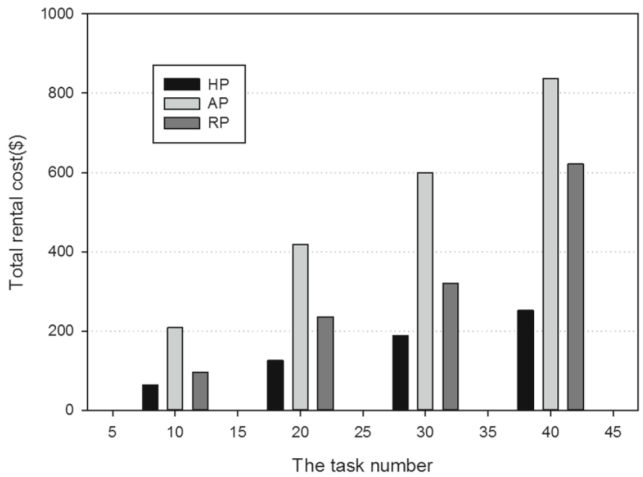


Fig. 13 The total rental cost of all spot virtual machine purchasing approaches under dataset 3

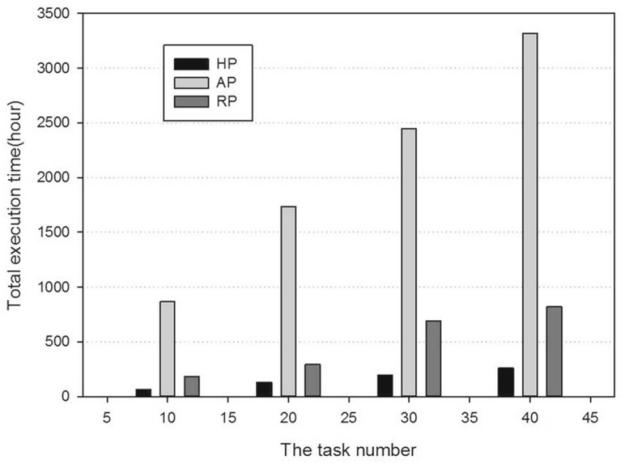


Fig. 14 The total execution time of all spot virtual machine purchasing approaches under dataset 1

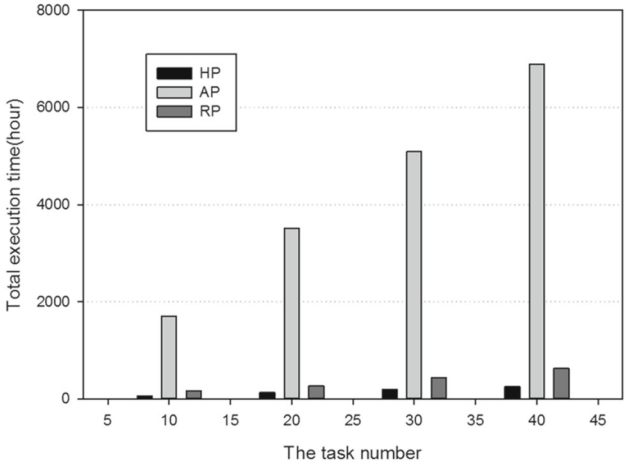


Fig. 15 The total execution time of all spot virtual machine purchasing approaches under dataset 2

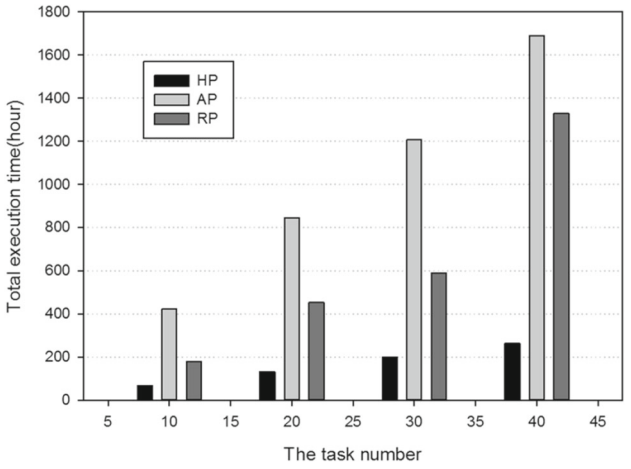


Fig. 16 The total execution time of all spot virtual machine purchasing approaches under dataset 3

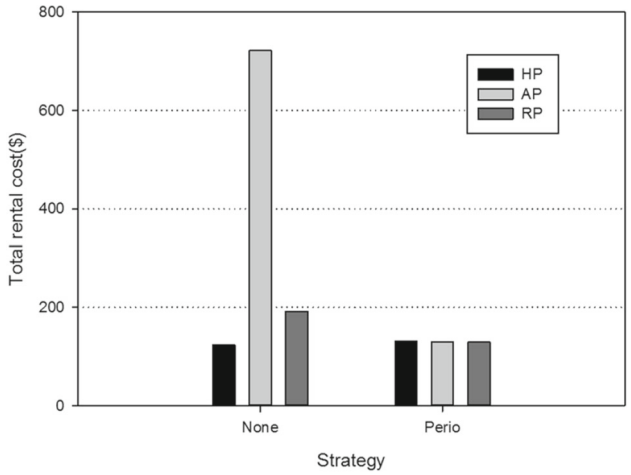


Fig. 17 Impact of the checkpointing strategy on the total rental cost. This figure illustrates the total rental cost of all spot virtual machine purchasing approaches under dataset 1

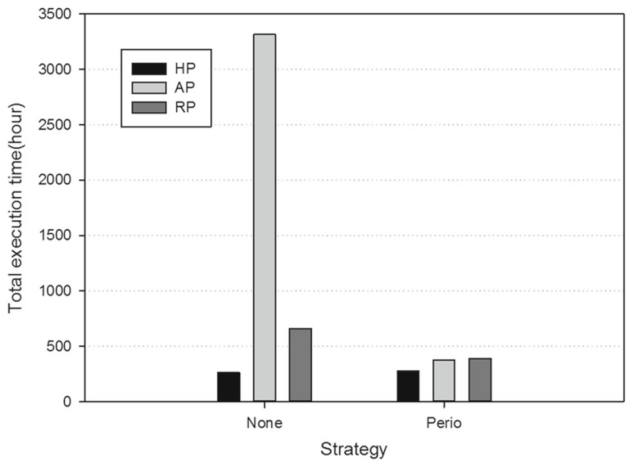


Fig. 18 Impact of the checkpointing strategy on the total execution time. This figure illustrates the total execution time of all spot virtual machine purchasing approaches under dataset 1

The performance of all the approaches is studied. Figure 17 shows the performance of all approaches on the total rental cost. Fig. 18 shows the performance of all approaches on the total execution time. The experimental results of Figs. 17 and 18 show that: (1) the rental cost and the execution time are significantly influenced by the checkpointing strategy; (2) in AP and RP, the cost rental cost and the total execution time significantly decrease when the checkpointing strategy is employed; (3) in HP, the cost rental cost and the total execution time increase a little when the checkpointing strategy is employed.

In summary, we can conclude from the above experiments that SpotCloudSim can help to highlight the advantages and shortcomings of the spot virtual machine purchasing approaches.

7 Conclusion

In this paper, we present a CloudSim-based cloud simulator named SpotCloudSim for simulation of spot virtual machine purchasing approach. SpotCloudSim consists of the new features: (1) rental fee charging, (2) virtual machine pricing, (3) spot virtual machine allocation, (4) purchasing decision-making, (5) out-of-bid event trigging, (6) coping strategy calculation (7) experimental results displaying. We illustrate the new features of SpotCloudSim by using three spot virtual machine purchasing approaches. The experimental results show that SpotCloudSim can illustrate the advantages and shortcomings of each approach. We will provide an easy-to-use user interface for the researchers in our future work.

Acknowledgements This work is supported by NSFC (61602054), NSFC (61472047), and Beijing Natural Science Foundation (4174100). This work is supported by Macao FDCT-MOST Grant 001/2015/AMJ and Macao FDCT Grant 104/2014/A3.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

Ethical approval All procedures performed in studies involving human participants were in accordance with the ethical standards of the institutional and/or national research committee and with the 1964 Helsinki declaration and its later amendments or comparable ethical standards.

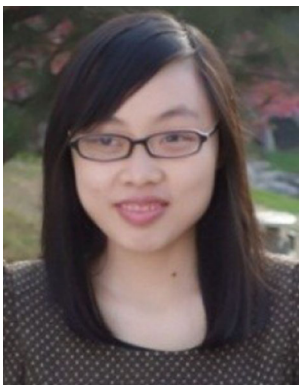
Research involving animal and human rights This article does not contain any studies with animals performed by any of the authors.

Informed consent Informed consent was obtained from all individual participants included in the study.

References

1. Shabeera, T. P., Madhu Kumar, S. D., Salam, S. M., Murali Krishnan, K.: Optimizing VM allocation and data placement for data-intensive applications in cloud using ACO metaheuristic algorithm. Engineering Science and Technology, an International Journal, published online, pp. 1-13(2016)
2. Wang, S., Zhou, A., Hsu, C.H., Xiao, X., Yang, F.: Provision of data-intensive services through energy- and QoS-aware virtual machine placement in national cloud data centers. IEEE Trans. Emerg. Top. Comput. **4**(2), 290–300 (2016)
3. Dastjerdi, A.V., Buyya, R.: Compatibility-aware cloud service composition under fuzzy preferences of users. IEEE Trans. Cloud Comput. **2**(1), 1–13 (2014)
4. Xiao, Z., Song, W., Chen, Q.: Dynamic resource allocation using virtual machines for cloud computing environment. IEEE Trans. Parallel Distrib. Syst. **24**(6), 1107–1117 (2013)
5. Fox, A., Griffith, R., Joseph, A., Katz, R., Konwinski, A., Lee, G., Patterson, D., Rabkin, A., Stoica, I.: Above the Clouds: A Berkeley View of Cloud Computing, vol. 28, pp. 13. Dept. Electrical Eng. and Comput. Sciences, University of California, Berkeley, Rep. UCB/EECS, (2009)

6. Li, H., Dong, M., Ota, K., Guo, M.: Pricing and repurchasing for big data processing in multi-clouds. IEEE Trans. Emerg. Top. Comput. **4**(2), 266–277 (2016)
7. Liu, Z., Wang, S., Sun, Q., Zou, H., Yang, F.: Cost-aware cloud service request scheduling for SaaS providers. Comput. J. **57**(2), 291–301 (2014)
8. Zhang, W.Z., Xie, H.C., Hsu, C.H.: Automatic memory control of multiple virtual machines on a consolidated server. IEEE Trans. Cloud Comput. **5**(1), 2–14 (2017)
9. Li, H., Dong, M., Ota, K.: Radio access network virtualization for the social internet of things. IEEE Cloud Comput. **2**(6), 42–50 (2015)
10. Dong, M., Li, H., Ota, K., Yang, L.T., Zhu, H.: Multicloud-based evacuation services for emergency management. IEEE Cloud Comput. **1**(4), 50–59 (2014)
11. Agmon Ben-Yehuda, O., Ben-Yehuda, M., Schuster, A., Tsafrir, D.: Deconstructing amazon EC2 spot instance pricing. ACM Trans. Econ. Comput. **1**(3), 16 (2013)
12. Stokely, M., Winget, J., Keyes, E., Grimes, C. and Yolken, B.: Using a market economy to provision compute resources across planet-wide clusters. Parallel and Distributed Processing, 2009. IPDPS 2009. IEEE International Symposium on, IEEE, pp. 1–8 (2009)
13. Wang, Q., Ren, K., Meng, X.: When, cloud meets eBay: Towards effective pricing for cloud computing. INFOCOM, 2012 Proceedings IEEE, IEEE, pp. 936–944, (2012)
14. Chen, J., Wang, C., Zhou, B. B., Sun, L., Lee, Y. C. and Zomaya, A. Y.: Tradeoffs between profit and customer satisfaction for service provisioning in the cloud. Proceedings of the 20th international symposium on High performance distributed computing, pp. 229–238. ACM, New York (2011)
15. Zhou, A., Sun, Q., Sun, L., Li, J. and Yang, F. 'Maximizing the profits of cloud service providers via dynamic virtual resource renting approach', EURASIP Journal on Wireless Communications and Networking, Vol.2015, No.1, pp.71(2015)
16. Guo, W., Chen, K., Wu, Y., Zheng, W.: Bidding for highly available services with low price in spot instance market. Proceedings of the 24th International Symposium on High-Performance Parallel and Distributed Computing, pp. 191–202. ACM, New York (2015)
17. He, X., Shenoy, P., Sitaraman, R. and Irwin, D.: Cutting the cost of hosting online services using cloud spot markets. The 25th International ACM Symposium on High-Performance Parallel and Distributed Computing (HPDC), pp. 1–12 (2015)
18. Legrand, A., Marchal, L., Casanova, H.: Scheduling distributed applications: the SimGrid simulation framework. CCGrid 2003. 3rd IEEE/ACM International Symposium on Cluster Computing and the Grid, 2003. Proceedings., pp. 138–145 (2003)
19. Buyya, R., Murshed, M.: GridSim: a toolkit for the modeling and simulation of distributed resource management and scheduling for grid computing. Concurr. Comput. Pract. Exp. **14**(13–15), 1175–1220 (2002)
20. Calheiros, R. N., Ranjan, R., De Rose, C. A., Buyya, R.: Cloudsim: A novel framework for modeling and simulation of cloud computing infrastructures and services, arXiv preprint arXiv:0903.2525 (2009)
21. Calheiros, R.N., Ranjan, R., Beloglazov, A., De Rose, C.A., Buyya, R.: CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. Softw. Pract. Exp. **41**(1), 23–50 (2011)
22. cloud2sim. "https://sourceforge.net/projects/cloud2sim/."
23. Chen, W., Deelman, E.: WorkflowSim: a toolkit for simulating scientific workflows in distributed environments. 2012 IEEE 8th International Conference on E-Science. pp. 1–8 (2012)
24. AuctionSim. "http://www.cloudbus.org/cloudsim/CloudAuctionV2.0.zip."
25. Bux, M., Leser, U.: Dynamiccloudsim: simulating heterogeneity in computational clouds'. Proceedings of the 2nd ACM SIGMOD Workshop on Scalable Workflow Execution Engines and Technologies(SWEET), pp. 1–12. ACM, New York (2013)
26. Gupta, S.K.S., Banerjee, A., Abbasi, Z., Varsamopoulos, G., Jonas, M., Ferguson, J., Gilbert, R.R., Mukherjee, T.: GDCSim: a simulator for green data center design and analysis. ACM Trans. Model. Comput. Simul. **24**(1), 1–27 (2014)
27. Tighe, M., Keller, G., Bauer, M., Lutfiyya, H.: DCSim: a data centre simulation tool for evaluating dynamic virtualized resource management. 2012 8th International Conference on Network and Service Management (cnsm) and 2012 workshop on systems virtualiztion management (svm), pp. 385–392 (2012)
28. Zafer, M., Song, Y., Lee, K.-W.: Optimal bids for spot VMs in a cloud for deadline constrained jobs. Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on IEEE, pp. 75–82 (2012)
29. Javadi, B., Thulasiram, R.K., Buyya, R.: Characterizing spot price dynamics in public cloud environments. Future Gener. Comput. Syst. **29**(4), 988–999 (2013)
30. Yi, S., Andrzejak, A., Kondo, D.: Monetary cost-aware checkpointing and migration on amazon cloud spot instances. IEEE Trans. Serv. Comput. **5**(4), 512–524 (2012)
31. Jung, D., Chin, S., Chung, K., Yu, H., Gil, J.: An efficient checkpointing scheme using price history of spot instances in cloud computing environment. Network and Parallel Computing, pp. 185–200. Springer, Berlin (2011)

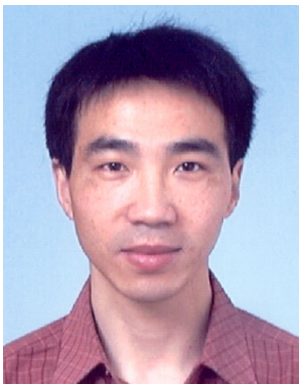


Ao Zhou is an assistant professor at the State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications. She received her Ph.D. degree in computer science at Beijing University of Posts and Telecommunications of China in 2015. Her research interests include cloud computing, service reliability.

Shangguang Wang is an associate professor at the State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications (BUPT). He received his Ph.D. degree at BUPT in 2011. His Ph.D. thesis was awarded as outstanding doctoral dissertation by BUPT in 2012. Dr. Wang is Vice Chair of IEEE Computer Society Technical Committee on Services Computing, 2015-2016 President of the Service Society Young Scientist Forum in China, General Chair of ICCSA 2016, Application Track Co-Chair of IEEE SCC 2015, Program Chair of the 2014 IOV, Program Chair of the 2014 SC2. He is currently serving as the Editor-in-Chief of the International Journal of Web Science. He has published more than 50 journal/conference papers such as the IEEE Transactions on Services Computing, the ACM Multimedia Computing, Communications, and Applications, the IEEE Transactions on Cloud Computing, the IEEE Transactions on Emerging Topics in Computing, and the IEEE Transactions on Automation Science and Engineering.

Qibo Sun received his Ph.D. degree in communication and electronic system from the Beijing University of Posts and Telecommunication in 2002. He is currently an associate professor at the Beijing University of Posts and Telecommunication in China. He is a member of the China computer federation. His current research interests include services computing, internet of things, and network security.

Jinglin Li is an associate professor at the Beijing University of Posts and Telecommunication, China. His current research interests include mobile internet, internet of things, internet of vehicles and convergence network service & security technologies.



Qinglin Zhao received the Ph.D. degree from the Chinese Academy of Sciences, Beijing, China, in 2005. From May 2005 to August 2009, he did research with The Chinese University of Hong Kong, Shatin, Hong Kong, and then with The Hong Kong University of Science and Technology, Sai Kung, Hong Kong. Since September 2009, he has been with the Faculty of Information Technology, Macau University of Science and Technology, Macau, China. His research interests include wireless communications and networking, next generation wireless LANs, software defined radio, software defined networking, software defined wireless networking, internet of things.

Fangchun Yang received his Ph.D. degree in communication and electronic system from the Beijing University of Posts and Telecommunication in 1990. He is currently a professor at the Beijing University of Posts and Telecommunication, China. He has published 6 books and more than 80 papers. His current research interests include network intelligence, services computing, communications software, soft switching technology, and network security. He is a fellow of the IET.