International Journal of Distributed and Cloud Computing
9 (1) 2021, 18-27
http://www.publishingindia.com/ijdcc

Amazon EC2 Locations for Spot Pricing - A Hierarchical Clustering Approach

Veena Khandelwal[1*] and Shantanu Khandelwal[2]
[1]Department of Computer Science & Engineeing, SRM Institute of Science and Technology, India.
Email: vn.khandelwal@gmail.com
[2]Manager, KPMG Services Pte. Ltd., Singapore. Email: shantanukhandelwal@protonmail.com
*Corresponding Author

Abstract: EC2 provides cloud services through several regions and availability zones. Spot instances are offered by Amazon EC2 through spot pricing. Amazon launched it simplified spot pricing model at Reinvent: 2017. The prices change less frequently now and are more predictable. To ensure availability, spot instances can be acquired from diversified locations. The biggest concern to the user is the identification of regions and availability zones to choose for optimizing cost and maximizing availability. The study attempts to find similarity in cost per instance among different EC2 locations according to Ward's Hierarchical Clustering Algorithm. The analysis uses past 60 days history traces from last week of September to November 2019 of four different compute instance types across all Amazon EC2 regions and availability zones. Results suggest a significant amount of similarity in terms of cost of instance in spot pricing across different locations. This raises user's confidence in adopting spot market.

Keywords: Amazon EC2 regions, Availability zones, Hierarchical clustering, Spot instances.

I. Introduction

Cloud computing delivers different types of services in terms of resource capacities, irrespective of their physical location. These services are governed by an important agreement called Service Level Agreements. This agreement governs both the parties including cloud computing service provider and cloud end user at different pricing models. Amazon Web Services spot instances, Microsoft Azure's low-priority virtual machines and Google Cloud Engine offer pre-emptible virtual machines without any availability guarantee.

Amazon Elastic Compute Cloud (EC2) offers three different pricing models, namely Reserved Pricing, On-demand Pricing and Spot Pricing. Customers find spot pricing attractive as spot instances are offered at a very cheap price. Excess computing capacity is sold by Amazon as spot instances. Spot instances are offered by Amazon EC2 through spot pricing at a substantially low price without any availability guarantee, as spot instances can be interrupted by the cloud service provider at any time.

Amazon provides Infrastructure as a Service (IaaS) from various diversified locations comprising several regions and availability zones. The entire spot pricing model is built around bidding on excess resources at deep cost savings for executing a varied workload in a specified region and availability zone.

Earlier spot prices spiked higher than the on-demand price multiple times a day on every weekday. Spot prices were non uniformly distributed and exhibited seasonal nature. There were price variations up to 77% during different weekdays and hours [1].

In November 2017, Amazon simplified its spot pricing policy [2]. Spot prices are now stable and have lower volatility. After the change, spot prices hover between a small range without much volatility. The new spot instance market features significantly more stable with far fewer price events and longer times between price events. They are now predictable for longer period of time with lesser price events. The seasonal nature of the spot market has changed. The prices in the new model are updated far less frequently and are uniformly distributed throughout the week.

Amazon has also changed its spot instance termination policy. Spot instance market price is dynamically decided by Amazon and does not depend on the user submitted bids. With the new spot pricing model, when the demand exceeds the capacity, the price remains relatively the same, but some instances are reclaimed automatically by Amazon EC2, not for price, but for the capacity-oversubscribed. Spot instances can now be terminated when there is an increase in demand or decrease in supply due to more demand of reliable on-demand instances. Instances cannot be purchased at the published price because that does not reflect the actual supply and demand curve for the excessive resources. However, in order to control costs, maximum spot price can be set by the users depending upon their budget and time of executing jobs on spot instances.

In order to address this, Amazon provides a 2 minute termination notice. Depending on the needs, user can specify whether they want to hibernate, stop, or terminate spot instances when they are interrupted during the notice period [3]. Spot instances with bid prices higher than market price may be terminated as well. Bidding higher prices no more guarantees higher reliability or execution duration. Users can no longer control termination. Users no more bid exorbitantly for the spot instances i.e., even more than the on-demand prices when prices are jumping highly. Instead, they pay the current spot price. Users no more now get free lunch also which was earlier utilized for executing small jobs of duration less than one hour at no cost.

New spot pricing policy degrades the reliability of spot instances as well as it excludes the use of spot price prediction techniques. With the new pricing model, we have essentially traded one problem for another: predictability of getting and keeping an instance for the predictability of price. With the new spot pricing it has become much easier to get started with as the prices are now steady and there are fewer price adjustments than earlier. Also Hibernation feature introduced with the new spot pricing policy prevents data loss and makes execution of high performance computing jobs much easier. The new pricing policy has led to generally higher prices [4].

With this new spot pricing model, a couple of counter-intuitive things can and do occur:

- Bidding for spot instances at the current price does not result in acquiring an instance.
- An existing instance may be reclaimed even though the bid price was higher than the current price.

With the introduction of new spot pricing model, spot instances are suitable for executing applications such as high performance computing jobs, short-span workloads, development environments, large scale simulations such as IPL-like traffic pattern required by research community, high concurrency events, financial modelling and analysis, big data analytics, paid web services, distributed applications, testing and developing, web scraping and many others that do not require Reserved Instances. For use cases such as scientific research and computing, high concurrency events, large scale simulations where cost is an important aspect, there is a great amount of flexibility required for the adoption of spot instances. Finance folks are interested in same number of servers at lesser cost and developers are interested in same cost but more performance. To optimize cost and reduce the impact of spot instance volatility, customers can execute their jobs at diversified locations across multiple regions and availability zones. For acquiring spot instances at diversified locations and building spot price aware scheduling algorithms, the biggest concern to the user is the identification of regions/availability zones to choose to optimize cost and availability.

The objective of this study is to help developers design high availability solutions with failover at low cost using spot instances for applications that have significantly long execution time. The study identifies EC2 locations that have most similar hourly spot price variations to reduce the impact of volatility. By computing similarity between hourly spot prices of several compute instance types across different locations, this work illustrates the locations that have most similar price changes. We find that prices across availability zones within a region i.e., availability zones and prices in some regions also are very similar.

With the earlier spot pricing policy, spot instances were only suitable for execution of short jobs due to highly volatile nature of spot instances. We study new spot pricing policy across 16 different Amazon EC2 regions and 64 availability zones to design high availability solutions with failover using spot prices for long running sudden tasks at a low cost.

The rest of this paper is structured as follows. Section II of this paper examines related work. Section III examines the spot history data, Amazon infrastructure and different compute instances. Section IV focuses of Hierarchical Clustering technique used in this paper. Section V shows hierarchical clustering results of our analysis. Section VI will perform discussions on our results to glean any interesting insights. Finally Section VII will summarise and give our overall conclusions.

II. Related Work

Very few works are based on this new spot pricing model. Although there are several papers published since the new spot pricing model was introduced, most of them analyze spot history traces before 2017. The extensive work done earlier is now no longer applicable as spot price dynamics have changed significantly. Very few authors study new spot pricing model. We discuss all papers based on the old and new spot pricing model in the related work.

Prior to November 2017, it was believed that spot prices were market driven, but actually it was not so. Very high peaks which correspond to spot prices much higher than on-demand prices were observed. An unreliable instance cannot be acquired by any user by bidding at bid price higher than on-demand price. According to [5] there must be some hidden underlying mechanism governing spot price by Amazon EC2. When the spot prices were low during off-peak hours/off-peak days, spot instances were suitable for executing small and medium sized jobs of length 4 to 12 hours while optimizing execution cost. With frequent price changes, acquiring spot instances required forecasting spot prices, determining bidding strategy suitable for the job type, integration of check-pointing and fault-tolerance policies to complete job execution before the deadline. Much work has been done in these areas to optimize cost, reduce execution time and increase reliability of spot instances.

Authors of [6] model spot instance lifetime using a Markov Chain. [7] employ normal approximation to model the spot price distribution. [8] assume spot prices are normally distributed and focus on achieving availability. [9] make use of discrete semi-Markovian chain to model the spot price variations. [10]

perform analysis of spot price predictability. [11] make use of predictive model based on artificial neural networks. [5] perform reverse engineering on how prices are set. [12] show that Gaussian distribution shows better fit. Authors perform statistical analysis of spot instances using a mixture of Gaussian distribution to model inter-price time of each spot instance and determine the time correlation in terms of hour-in-day and day-of-week. Authors of [13] use Gradient Descent algorithm to predict one-day-ahead and one-week-ahead spot prices. Authors of [14] make use of several moving average statistical methods to predict the next hour spot prices. [15] use Random Regression Forest to predict one-day and one-week-ahead spot prices.

Authors of [16] study Amazon EC2 new spot pricing policy. Authors study spot prices in 4 different regions and conclude that location plays a critical role in spot instance pricing. Spot prices still differ depending on the granularity of that location i.e., AWS region and availability zone. According to [17] with the new spot pricing policy, frequency of price events has reduced substantially. Hourly and weekly price trends have also normalized. According to [4] since November 2017, when market driven pricing policy changed to retail pricing policy, spot prices are continually increasing.

However, some recent works in cloud computing are also based on security issues in adopting cloud computing, Big Data security and improving performance. Authors of [18] discuss various security issues and challenges associated with Cloud computing environment and current solution space. Authors of [19] choose optimized service broker routing policy based on different parameters to increase the overall performance of the cloud computing systems. Authors of [20] use cloud computing to operate with the Big Data to establish an architecture relaying on the security of the network in order to improve the security issues. Authors of [21] improve the multi-user data sharing mechanism in hybrid cloud using public cloud's cost savings and elasticity with the private cloud's security and customisation. Authors of [22] propose an efficient algorithm for advanced scalable Media-based Smart Big Data (3D, Ultra HD) on Intelligent Cloud Computing systems.

III. Spot History Data

The study involves 60 days spot price traces beginning from 24-09-2019 to 23-11-2019. Spot prices history traces were obtained using AWS Command Line Interface from AWS website directly and stored in csv file.

Table I: Amazon EC2 Regions, Availability Zones and Instance Types used for Study

Region Asia Pacific		*10*	*eu-west-2 (London)*
1	*ap-northeast-1 (Tokyo)*	AZ26	eu-west-2a
AZ0	ap-northeast- 1a	AZ27	eu-west-2b
AZ1	ap-northeast- 1c	AZ28	eu-west-2c
AZ2	ap-northeast- 1d	*11*	*eu-west-3 (Paris)*
2	*ap-northeast-2 (Seoul)*	AZ29	eu-west-3a
AZ3	ap-northeast- 2a	AZ30	eu-west-3b
AZ4	ap-northeast- 2b	AZ31	eu-west-3c
AZ5	ap-northeast- 2c	*Region South America*	
3	*ap-south-1 (Mumbai)*	*12*	*sa-east-1 (Sao Paulo)*
AZ6	ap-south-1a	AZ32	sa-east-1a
AZ7	ap-south-1b	AZ33	sa-east-1b
AZ8	ap-south-1c	AZ34	sa-east-1c
4	*ap-southeast-1 (Singapore)*	*Region US-East*	
AZ9	ap-southeast-1a	*13*	*us-east-1(N. Virginia)*
AZ10	ap-southeast-1b	AZ35	us-east-1a
AZ11	ap-southeast-1c	AZ36	us-east-1b
5	*ap-southeast-2 (Sydney)*	AZ37	us-east-1c
AZ12	ap-southeast-2a	AZ38	us-east-1d
AZ13	ap-southeast-2a	AZ39	us-east-1f
AZ14	ap-southeast-2c	*14*	*us-east-2 (Ohio)*
Region Canada		AZ40	us-east-2a
6	*ca-central-1 (Central)*	AZ41	us-east-2b

AZ15	ca-central-1a	AZ42	us-east-2c
AZ16	ca-central-1b	*Region US-West*	
Region EU		*15*	*us-west-1 (Oregaon)*
7	*eu-central-1 (Frankfurt)*	AZ43	us-west-1a
AZ17	eu-central-1a	AZ44	us-west-1c
AZ18	eu-central-1b	*16*	*us-west-2 (N. California)*
AZ19	eu-central-1c	AZ45	us-west-2a
8	*eu-north-1 (Stockholm)*	AZ46	us-west-2b
AZ20	eu-north-1a	AZ47	us-west-2c
AZ21	eu-north-1b	AZ48	us-west-2d
AZ22	eu-north-1c	*Instance Types*	
9	*eu-west-1 (Ireland)*	1	c5.2xlarge
AZ23	eu-west-1a	2	c5.4xlarge
AZ24	eu-west-1b	3	c5.9xlarge
AZ25	eu-west-1c	4	c5.18xlarge

Spot prices history consists of spot prices from 16 Amazon EC2 regions shown as bold and numbered, consisting of 49 availability zones numbered AZ0 to AZ48 and 4 compute instance types, namely c5.2xlarge, c5.4xlarge, c5.9xlarge and c5.18xlarge as shown in Table I. Spot instances in availability zone sa-east-1b started on 16-10-2019. As spot price are more predictable and change less frequently in the new spot pricing model, the study considers same spot price for sa-east-1b availability zone for time period beginning from 24-09-2019 to 15-10-2019. The study uses python 3.7.3 to implement hierarchical clustering using Ward's Method by importing scipy.cluster.hierarchy module.

IV. Methodology

A. Hierarchical Cluster Analysis

Cluster analysis is a statistical method that refers to the methods used to group different data objects into groups (clusters) based on similarity/dissimilarity among the data such that the intra-cluster similarity is high and inter-cluster similarity is low. The resulting partitioning into different sub classes reveals internal structures if any, and aids in improving our understanding of the data. Hierarchical clustering begins by considering each point as an individual cluster and at each step; clusters that are similar to each other are merged. Similarity at each step is determined based on the proximity metrics of the clusters. Split and merge process of the clusters is recorded in a tree like structure called Dendrogram. In Spot Pricing, hierarchical clustering is used to identify distinct groups of potential regions and availability zones so that, for example, bidding decision can be appropriately targeted.

B. Hierarchical Agglomerative Methods

1) Single Linkage

Single Linkage finds distance between closest clusters. Distance between two clusters *x* and *y* is given as:

$$D(x,y) = \sum_{i=1}^{S_x} \sum_{j=1}^{S_y} Min\{d\,(x_i, y_i)\} \tag{1}$$

where, S_x and S_y denote cluster sizes. Proximity between two clusters is the proximity between their two closest objects [23]. Cluster merging of the two closest clusters takes place in each step. The method produces clusters of irregular shapes. We avoid single linkage for spot pricing cluster analysis as it does not take into account cluster structure and results in forming long chains of the data. Single Linkage can sometimes form clusters with objects in different cluster when objects in other cluster are nearer than to the objects within its own cluster. The edges in the clusters start connecting and very soon all the data falls into one cluster. We need to avoid this and find different clusters.

2) Complete Linkage

Complete Linkage defines distances as the maximum instance distance over all possible pairs in two given clusters [24], which is a non local measure. Complete linkage has the opposite effect of single linkage. It can keep similar objects in separate clusters.

$$D(x,y) = \sum_{i=1}^{S_x} \sum_{j=1}^{S_y} Max\{d\,(x_i, y_i)\} \tag{2}$$

Complete Linkage finds distance between the farthest elements in the clusters. It forces well balanced, tighter spherical clusters than single linkage with consistent diameter i.e., equal size and also does not take into account cluster structure. This method is suitable for compact regular clusters. It is extremely sensitive to outliers.

3) Average Linkage Method

Average linkage method is a compromise between nearest neighbor distance computation of Single linkage and farthest neighbor computation of complete linkage method [25]. It tends to maximize the coherency of the joined clusters. This approach to calculate dissimilarity between two clusters takes all observations from each groups, computes their dissimilarity and calculates the average of the dissimilarity [26]. As all instance distances influence the distance calculation, this algorithm is robust towards outliers than complete linkage. It forms uniform sized, convex shaped clusters. It can also break large sized clusters into smaller ones.

$$D(x,y) = \frac{1}{S_x . S_y} \sum_{i=1}^{S_x} \sum_{j=1}^{S_y} d\,(x_i, y_i) \tag{3}$$

Mathematically, similarity can be written as:

$$sim(C1, C2) = \sum D(x,y)/|C1| \times |C2| \tag{4}$$

where, $C1$&$C2$ are clusters and $x_i \in C1$& $x_j \in C2$.

4) Ward's Method

This approach of calculating the similarity between two clusters is an alternative approach to single linkage clustering. It is robust against outliers but biased towards formation of spherical clusters. It is computationally expensive but performs fewer computations. The clusters formed are of fairly equal size, are less than optimal clusters [27]. At each step Ward's method joins the two clusters to make one cluster that minimizes variance measured by Sum of Squared Error (SSE) at each step. SSE is a measure of the distance between a cluster's instance attribute value and the cluster mean value. Instead of looking at the diameter of the result, it looks for the deviation of the result. It then merges the results with the smallest deviation.

$$SSE = \sum_{i}^{S_x} \sum_{j}^{S_y} |X_{ij} - Y_i|^2 \tag{5}$$

$$D(x,y) = SSE(x,y) - \big(SSE(x) + SSE(y)\big) \tag{6}$$

$$sim(C1, C2) = \sum_{1}^{p} (D(x,y)^2\,)/|C1| \times |C2| \tag{7}$$

where, p=61 days for which spot prices were collected in this study, number of points to be merged into clusters is 49 which refer to the different availability zones and $C1$ & $C2$ are clusters and $x_i \in C1$ & $x_j \in C2$. Hierarchical clustering performed for spot pricing uses Ward's method for calculating similarity between two clusters. For spot pricing cluster analysis, our basic requirement is that the price variations within a cluster should beminimum. This would help in identifying EC2 locations that have similar spot prices with minimum spot price variations. Similarity is calculated by finding the sum of the squared distances between any two points. Hierarchical clustering algorithm extracts data from spot history traces. Dendrograms are used to represent cluster merging process.

- On the x axis there are labels that specify indices of price traces in data.
- On the y axis we see the distances (of the 'Ward' method).

V. Hierarchical Clustering Results

After performing Hierarchical clustering using Ward's method, we show dendrograms of spot prices for four different compute instance types as mentioned in Table I to illustrate which clusters join at each level and the distance between clusters formation. Cluster formation is done using Ward's method based on the spot pricing history traces of 4 compute instance types in all 49 availability zones. When there is a large difference in the distance as calculated using between clusters at subsequent levels, it suggests that at lower level, clusters that are relatively close together were joined whereas, at the following level, clusters that were joined were distant. This is shown in dendrograms using three different horizontal lines at y=16, y=26 and y=45. The optimum number of clusters in spot pricing is the number present just before that large jump in distance shown by different horizontal lines in the dendrograms.

A. c5.2xlarge Instance Type

From 16 Amazon EC2 regions, initially 6 clusters are formed based upon their spot price similarity calculated using Ward's method. This is shown by line y=15 in Fig. 1. Cluster 1 is formed by AZ0 to AZ8 which belong to regions ap-northeast-1, ap-northeast-2 and ap-south-1. Cluster 2 is formed by AZ9 to AZ16 which belong to regions ap-southeast-1, ap-southeast-2 and ca-central-1. Cluster 3 formed by AZ17 to AZ25 which belong to regions eu-central-1, eu-north-1 and eu-west-1. Cluster 4 is formed by AZ26 to AZ31 which belong to regions eu-west-2 and eu-west-3. Cluster 5 is formed by AZ32 to AZ39 which belong to regions sa-east-1 and us-east-1. Cluster 6 is formed by AZ40 to AZ48 which belong to regions us-east-2, us-west-1, us-west-2. These 6 clusters are formed at the least vertical distance, i.e., have the lowest variations and hence highest similarity in prices. Hierarchical merging of clusters for different regions at line y=16 are shown in Table II and Table III.

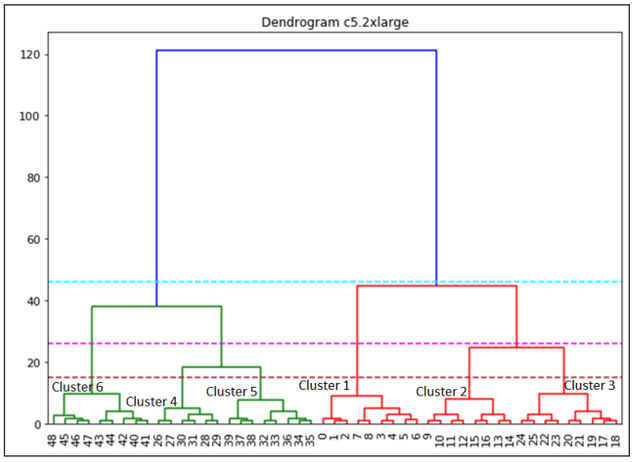

Fig. 1: Cluster Formation using Ward's Method for c5.2xlarge Instance Type

B. c5.4xlarge Instance Type

From 16 Amazon EC2 regions, initially 6 clusters are formed based upon their spot price similarity calculated using Ward's method. This is shown by line y=15 in Fig. 2. Cluster 1 is formed by AZ0 to AZ5 which belong to regions ap-northeast-1 and ap-northeast-2. Cluster 2 is formed by AZ9 to AZ14 which belong to regions ap-south-1, ap-southeast-1 and ap-southeast-2. Cluster 3 formed by AZ15 to AZ25 which belong to regions ca-central-1, eu-central-1, eu-north-1 and eu-west-1. Cluster 4 is formed by AZ26 to AZ36 which belong to regions eu-west-2, eu-west-3 and sa-east-1 and availability zones 1a and 1b of region us-east 1. Cluster 5 formed by AZ37 to AZ39 which belong to regions availability zones 1c, 1d, 1f of region us-east-1, us-east-2, us-west-1 and us-west-2. These 5 clusters are formed at the least distance, i.e., have the lowest variations and hence highest similarity in prices. At line y=26, clusters 1, 2 merge. These clusters are less similar to each other as they are formed at a distance of 26. At line y=46, clusters 3, 4 merge to form one cluster. These clusters are most dissimilar to each other as they have joined far apart.

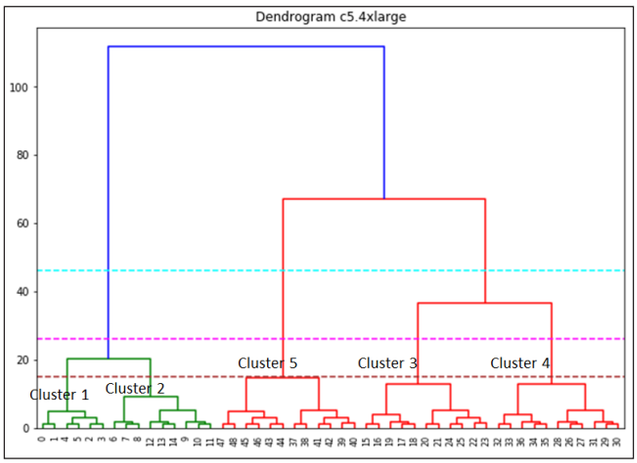

Fig. 2: Cluster Formation using Ward's Method for c5.4xlarge Instance Type

C. c5.9xlarge Instance Type

From 16 Amazon EC2 regions, initially 6 clusters are formed based upon their spot price similarity calculated using Ward's method. This is shown by line y=15 in Fig. 3. Cluster 1 is formed by AZ0 to AZ8 which belong to regions ap-northeast-1, ap-northeast-2 and ap-south-1. Cluster 2 is formed by AZ9 to AZ16 which belong to regions ap-southeast-1, ap-southeast-2 and ca-central-1. Cluster 3 formed by AZ17 to AZ25 which belong to regions eu-central-1, eu-north-1 and eu-west-1. These 6 clusters are formed at the least distance, i.e., have the lowest variations and hence highest similarity in prices.

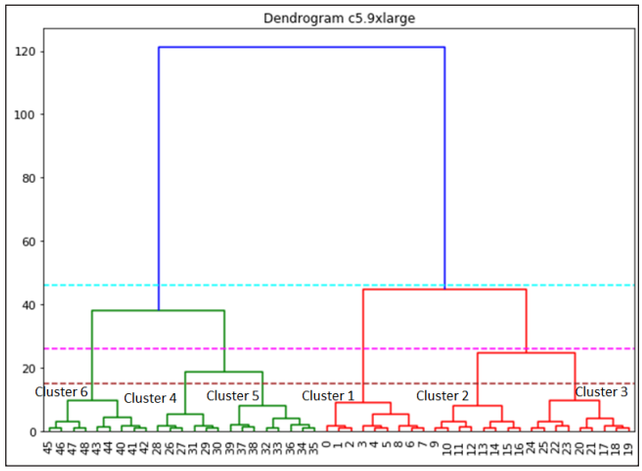

Fig. 3: Cluster Formation using Ward's Method for c5.9xlarge Instance Type

The cluster formation in c5.9xlarge instance types at line y=15 is exactly as c5.2xlarge. This suggests that spot price changes of both instance types are very similar in all regions and availability zones. At line y=26, clusters 2 and 3 merge. These clusters are less similar to each other as they are formed at a distance of 26. At line y=46, clusters 1, 2, 3 merge to form one cluster. These clusters are most dissimilar to each other as they have joined far apart. A slight variation in merging of clusters is observed for these regions with respect to c5.2xlarge instance type. But, these does not make any significant impact to our study as we are interested in most similar regions, i.e., lowest level merge and not most dissimilar regions, corresponding to highest level merge.

D. c5.18xlarge Instance Type

From 16 Amazon EC2 regions, initially 5 clusters are formed based upon their spot price similarity calculated using Ward's method. This is shown by line y=15 in Fig. 4. Cluster 1 is formed by AZ0 to AZ11 which belong to regions ap-northeast-1, ap-northeast-2, ap-south-1 and ap-southeast-1. Cluster 2 is formed by AZ9 to AZ19 which belong to regions ap-southeast-2 and ca-central-1 and eu-central-1. Cluster 3 formed by AZ20 to AZ31which belong to regions eu-north-1, eu-west-1, eu-west-2 and eu-west3. Cluster 4 is formed by AZ32 to AZ38 which belong to regions sa-east-1 and us-east-1. Cluster 5 is formed by AZ39 to AZ48 which belong to availability zone us-east-1a

of region us-east 1 and regions us-east-2, us-west-1, us-west-2. These 5 clusters are formed at the least distance, i.e., have the lowest variations and hence highest similarity in prices. The cluster formation in c5.18xlarge instance types is similar to c5.2xlarge to a great extent. This suggests that spot price changes of both instance types are very similar in all regions and availability zones.

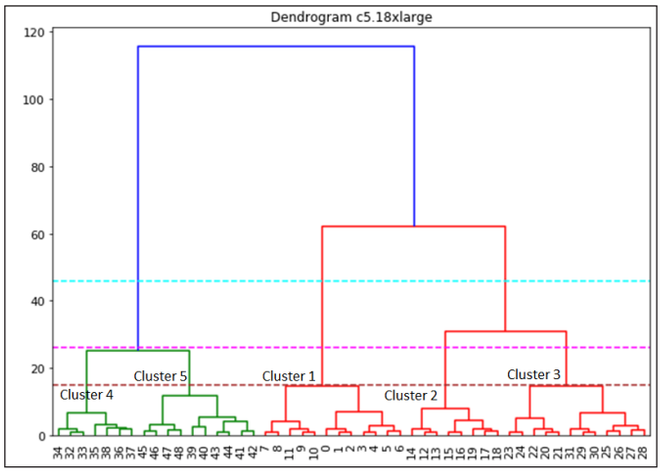

Fig. 4: Cluster Formation using Ward's Method for c5.18xlarge Instance Type

At line y=26, clusters 2, 3 merge and clusters 4, 5 merge. These clusters are less similar to each other as they are formed at a distance of 26. At line y=46, clusters 1, 2, 3 merge to form one cluster. These clusters are most dissimilar to each other as they have joined far apart. A slight variation in merging of clusters is observed for these regions with respect to c5.2xlarge instance type. Again, this does not make any significant impact to our study as we are interested in most similar regions, i.e., lowest level merge and not most dissimilar regions corresponding to highest level merge.

Hierarchical merging of clusters for different availability zones is shown in Table IV. The table shows formation of clusters 1 to 6 in different availability zones. From Cluster 1 results, we find that zones AZ0 to AZ8 form one cluster for most of the instance types. From cluster 2 results we find that zones AZ19 to AZ14 form one cluster. From cluster 3 results, we find that zones AZ17 to AZ25 form one cluster for most of the instance types. From cluster 4 results, we find that zones AZ26 to AZ31 form one cluster for most of the instance types. From cluster 5 results, we find that zones AZ32 to AZ39 form one cluster for most of the instance types. From cluster 6 results, we find that zones AZ40 to AZ48 form one cluster for most of the instance types.

Table II: Hierarchical Clustering for Cluster 1, Cluster 2 and Cluster 3 for Different Compute Instance Types Showing Different Regions that Merge due to Similarity in Spot Price for Different Compute Instance Types at Line y=15

Instance Type	Cluster 1 (Regions)	Cluster 2 (Regions)	Cluster 3 (Regions)
c5.2xlarge	ap-northeast-1	ap-southeast-1	eu-central-1
	ap-northeast-2	ap-southeast-2	eu-north-1
	ap-south-1	ca-central-1	eu-west-1
c5.4xlarge	ap-northeast-1	ap-south-1	ca-central-1
	ap-northeast-2	ap-southeast-1	eu-central-1
		ap-southeast-2	eu-north-1
			eu-west-1
c5.9xlarge	ap-northeast-1	ap-southeast-1	eu-central-1
	ap-northeast-2	ap-southeast-2	eu-north-1
	ap-south-1	ca-central-1	eu-west-1
c5.18xlarge	ap-northeast-1	ap-south-1	ap-southeast-2
	ap-northeast-2	ap-southeast-1	ca-central-1
	ap-south-1		eu-central-1

Table III: Hierarchical Clustering for Cluster 4, Cluster 5 and Cluster 6 for Different Compute Instance Types Showing Different Regions that Merge due to Similarity in Spot Price for Different Compute Instance Types at Line y=15

Instance Type	Cluster 4 (Regions)	Cluster 5 (Regions)	Cluster 6 (Regions)
c5.2xlarge	eu-west-2	sa-east-1	us-east-2
	eu-west-3	us-east-2	us-west-1
			us-west-2
c5.4xlarge	eu-west-2		us-east-1
	eu-west-3		us-east-2
	sa-east-1		us-west-1
			us-west-2
c5.9xlarge	eu-west-2	sa-east-1	us-east-2
	eu-west-3	us-east-1	us-west-1
			us-west-2
c5.18xlarge	eu-north-1	sa-east-1	us-est-2
	eu-west-1	us-east-1	us-west-1
	eu-west-2		us-west-2
	eu-west-3		

Table IV: Hierarchical Clustering for Cluster 1, Cluster 2, Cluster 3, Cluster 4, Cluster 5 and Cluster 6 for Different Compute Instance Types Showing Different Availability Zones that Merge due to Similarity in Spot Price for Different Compute Instance Types at Line y=15

Instance Type	Cluster 1	Cluster 2	Cluster 3
c5.2xlarge	AZ0 to AZ8	AZ9 to AZ16	AZ17 to AZ25
c5.4xlarge	AZ0 to AZ5	AZ6 TO AZ14	AZ15 to AZ25
c5.9xlarge	AZ0 to AZ8	AZ9 TO AZ16	AZ17 to AZ25
c5.18xlarge	AX0 to AZ11	AZ12 TO AZ19	AZ20 to AZ31
Instance Type	Cluster 4	Cluster 5	Cluster 6
c5.2xlarge	AZ26 to AZ31	AZ32 to AZ39	AZ40 to AZ48
c5.4xlarge	AZ26 to AZ36		AZ37 to AZ48
c5.9xlarge	AZ26 to AZ31	AZ32 to AZ39	AZ40 to AZ48
c5.18xlarge	AZ32 to AZ38	AZ39 to AZ48	

Authors of [28] in their study confirm the lack of practical and deployable market-driven mechanism for using spot instances. The result of clustering analysis cannot be compared with any other study as this is the first study since Amazon changed it new spot pricing policy in November 2017.

VI. Conclusion

Academic community strongly advocated spot market whey it was introduced, but it is still not reflected by consumer behaviour. Consumers still doubt about Amazon's spot-pricing scheme. This is due to the lack of published information from Amazon EC2 regarding termination of spot instances. At some point of time even over-subscribed instances are terminated and at some other time even under-subscribed instances sustain. In order to play a promising role in the sustainability of EC2 IaaS exploitation at low cost, some availability guarantee is required that the job will not be interrupted frequently and will be completed within its deadline. Spot consumers prefer simple mechanisms without using complex procedures at the backend

such as spot price prediction, check pointing and migration which are a major obstacle in spot market adoption. This analysis identifies regions/availability zones with similar price variations which would help user in spot bidding decisions and designing high availability solutions at diversified EC2 locations with failover at low cost without falling back to on-demand instances when spot instances are not available. This study would fulfil consumer's real needs and raise their confidence in exploiting spot market and design practical and easily deployable spot mechanisms. The study explores spot prices of only 4 compute instance types. It could be extended to other instance types also. In future we would like to study on outliers analysis of the spot prices aiming to enhance customer confidence in the adoption of spot instances for lowering costs.

References

[1] K. Veena, C. Anand, and G. C. Prakash, "Temporal and spatial trend analysis of cloud spot instance pricing in Amazon EC2," *2016 IEEE 14th Intl Conf on Dependable, Autonomic and Secure Computing, 14th Intl Conf on Pervasive Intelligence and Computing, 2nd Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress(DASC/PiCom/DataCom/CyberSciTech),* Aug. 2016.

[2] R. Pary, New Amazon EC2 Spot Pricing Model: Simplified Purchasing without Bidding and Fewer Interruptions | Amazon Web Services. Accessed: Jan. 16, 2020. [Online]. Available: https://aws.amazon.com/blogs/compute/new-amazon-ec2-spot-pricing/

[3] Docs.aws.amazon.com, Spot Instance Interruptions - Amazon Elastic Compute Cloud. Accessed: Jan. 26, 2020. [Online]. Available: https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/spot-interruptions.html

[4] G. George, R. Wolski, C. Krintz, and J. Brevik, "Analyzing AWS spot instance pricing," *2019 IEEE International Conference on Cloud Engineering (IC2E),* Jun. 2019.

[5] O. A. Ben-Yehuda, M. Ben-Yehuda, A. Schuster, and D. Tsafrir, "Deconstructing Amazon EC2 spot instance pricing," *ACM Transactions on Economics and Computation*, vol. 1, no. 3, pp. 1-20, Sep. 2013.

[6] N. Chohan, C. Castillo, M. Spreitzer, M. Steinder, A. N. Tantawi, and C. Krintz, "See spot run: Using spot instances for mapreduce workflows," *HotCloud*. 2010, pp. 7-7.

[7] A. Andrzejak, D. Kondo, and S. Yi, "Decision model for cloud computing under SLA constraints," *2010 IEEE International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems*, Aug. 2010.

[8] M. Mazzucco, and M. Dumas, "Achieving performance and availability guarantees with spot instances," *2011 IEEE International Conference on High Performance Computing and Communications*, Sep. 2011

[9] M. Zafer, Y. Song, and K.-W. Lee, "Optimal bids for spot VMs in a cloud for deadline constrained jobs," *2012 IEEE 5th International Conference on Cloud Computing*, Jun. 2012.

[10] H. Zhao, M. Pan, X. Liu, X. Li, and Y. Fang, "Optimal resource rental planning for elastic applications in cloud market," *2012 IEEE 26th International Parallel and Distributed Processing Symposium*, May 2012.

[11] R. M. Wallace, V. Turchenko, M. Sheikhalishahi, I. Turchenko, V. Shults, J. L. Vazquez-Poletti, and L. Grandinetti, "Applications of neural-based spot market prediction for cloud computing," *2013 IEEE 7th International Conference on Intelligent Data Acquisition and Advanced Computing Systems (IDAACS)*, Sep. 2013.

[12] B. Javadi, R. K. Thulasiram, and R. Buyya, "Characterizing spot price dynamics in public cloud environments," *Future Generation Computer Systems*, vol. 29, no. 4, pp. 988-999, Jun. 2013.

[13] V. K. Singh, and K. Dutta, "Dynamic price prediction for amazon spot instances," *2015 48th Hawaii International Conference on System Sciences*, Jan. 2015.

[14] J. L. Lucas-Simarro, R. Moreno-Vozmediano, R. S. Montero, and I. M. Llorente, "Cost optimization of virtual infrastructures in dynamic multi-cloud scenarios," *Concurrency and Computation: Practice and Experience*, vol. 27, no. 9, pp. 2260-2277, Dec. 2012.

[15] V. Khandelwal, A. K. Chaturvedi, and C. P. Gupta, "Amazon EC2 spot price prediction using regression random forests," *IEEE Transactions on Cloud Computing*, vol. 8, no. 1, pp. 59-72, Jan. 2020.

[16] N. Ekwe-Ekwe, and A. Barker, "Location, location, location: Exploring Amazon EC2 spot instance pricing across geographical regions," *2018 18th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID)*, May 2018.

[17] M. Baughman, S. Caton, C. Haas, R. Chard, R. Wolski, I. Foster, and K. Chard, "Deconstructing the 2017 changes to AWS spot market pricing," *Proceedings of the 10th Workshop on Scientific Cloud Computing - ScienceCloud'19*, 2019.

[18] Z. Gou, S. Yamaguchi, and B. B. Gupta, "Analysis of various security issues and challenges in cloud computing environment," *Identity Theft*, pp. 221-247.

[19] A. M. Manasrah, A. Aldomi, and B. B. Gupta, "An optimized service broker routing policy based

on differential evolution algorithm in fog/cloud environment," *Cluster Computing*, vol. 22, no. S1, pp. 1639-1653, Dec. 2017.

[20] K. E. Psannis, C. Stergiou, and B. B. Gupta, "Advanced media-based smart big data on intelligent cloud systems," *IEEE Transactions on Sustainable Computing*, vol. 4, no. 1, pp. 77-87, Jan. 2019.

[21] G. Gao, L. Wu, and Y. Yan, "A secure storage scheme with key-updating in hybrid cloud," *International Journal of High Performance Computing and Networking*, vol. 13, no. 2, p. 175, 2019.

[22] C. Stergiou, K. E. Psannis, B. B. Gupta, and Y. Ishibashi, "Security, privacy & efficiency of sustainable cloud computing for big data & IoT," *Sustainable Computing: Informatics and Systems*, vol. 19, pp. 174-184, Sep. 2018.

[23] F. Murtagh, and P. Contreras, "Algorithms for hierarchical clustering: An overview," *WIREs Data Mining and Knowledge Discovery*, vol. 2, no. 1, pp. 86-97, Dec. 2011.

[24] R. J. Jensen, G. Dunn, and B. S. Everitt, "An introduction to mathematical taxonomy," *Systematic Botany*, vol. 8, no. 1, p. 103, Jan. 1983.

[25] D. P. Berrar, W. Dubitzky, and M. Granzow (Eds.), "A practical approach to microarray data analysis," 2003.

[26] M. R. Anderberg, "Conceptual problems in cluster analysis," *Cluster Analysis for Applications*, pp. 10-24, 1973.

[27] J. H. Ward, "Hierarchical grouping to optimize an objective function," *Journal of the American Statistical Association*, vol. 58, no. 301, pp. 236-244, Mar. 1963.

[28] Z. Li, H. Zhang, L. O'Brien, S. Jiang, Y. Zhou, M. Kihl, and R. Ranjan, "Spot pricing in the cloud ecosystem: A comparative investigation," *Journal of Systems and Software*, vol. 114, pp. 1-19, Apr. 2016.