Table of Contents

1. Abstract: 1

1.1 Background 1

1.2 Objective: 1

1.3 Methodology 1

1.4 Contribution 1

1.5 Expected Results 1

1.6 Conclusion 1

Keywords: 1

2. Introduction 2

3. Problem statement: 3

4. Datasets 4

5. Literature Review 5

5.1 Cloud resource auto scaling (Reactive mechanism) 5

5.2 Predictive based Cloud Resource Scaling Strategy 6

5.3 Hybrid auto scaling mechanism 7

6. Current best solutions in cost aware Auto scaling mechanism 8

7. Limitation of current best solution 9

8. Proposed Project Plan: 10

9. Expected Outcome 10

10. Conclusion 10

11. References: 11

# Abstract:

1.1 Background

It is possible to scale up or down any virtual computational resources to achieve cost-effective outcomes and availability in an on demand cloud environment. However, actual automated elasticity and effectiveness of cost optimization in pay-per-use cloud model using Spared Instance type has not been researched much.

## Objective:

Primary objective is to achieve a practical minimum-cost flow model for building a fault tolerant application in current cloud infrastructures using spared instance (spot instance in AWS).

## Methodology

Identifying problems in currently available solutions and compare the variables like fault-tolerance (FT), Cloud Infrastructure costs (CIC), Cloud resources (CR) used in various models.  
  
Perform Load / Stress test on suggested solution deployed in AWS to compare the factors such as level of fault tolerance achieved, infrastructure costs etc. with current solutions.

## Contribution

Providing a complete fault tolerance mechanism using spared instance against any kind of server load / stress in minimum-cost flow cloud architecture which current solutions does not offer.

## Expected Results

The proposed system will be more fault tolerant in less cloud infrastructure costs using an intelligent cloud resources combination techniques.

## Conclusion

Achieve fault tolerance in cloud computing with minimum costs and thus saving millions of dollars each year for hundreds of businesses worldwide.

## Keywords:

## Spare Instance, Spot Instance Service, Cloud Auto Scaling, Fault tolerance, DDOS, YoYo attack, Cloudwatch, AWS

# Introduction

Software is now very fundamental part of daily life in current world and we need to perform a great deal of effort to make this software functional, operational and available to users. User needs to interact with software whether they use their phones or computers, withdraw funds from ATM, gets services from medical clinic or cross the road using traffic lights. This research compares two approaches to scale cloud computational resources in terms of minimum-cost flow. It also suggests a completely automated scaling approach and explains why it is more preferable.

Fault-tolerance is the ability to run software system smoothly even if the traffic usage goes beyond the limit or if some resource systems used to build the software service, fails.

It is very crucial that certain types of software need to available to users all the times, such as educational portal, medical services, government facilities, e-commerce websites etc. When software system/server fails due to high traffic loads, DDOS attack or crash in any other computational parts of the system, according to current best solution server administrator is notified immediately and they allocate new resources to support increased load / pressure to resolve the problem. However, this procedure has some minor downtime and server administrator again needs to remove / downgrade the unused cloud resources when user traffic load is reduced again.

In this research, a complete automated approach is suggested combining some cloud infrastructure resources such as Spot Instance, cloud watch, load balancer, auto scale group etc. which may help business to avoid downtime in software system in minimum costs and thus avoid huge costs and trusts of the customers.

# Problem statement:

AWS or any cloud provider platform has the mechanism for solving fault tolerance or auto scaling approach based on the traffic/application load demand. However, our target is not only to solve fault-tolerance problem, but to do it in a very minimal cost approach.

Cloud computing providers actually needs to keep additional computing resources ready to provide when their customers’ demands. This on-demand scaling is the fundamental of cloud offering. However, these additional resources when not being used, just remains idle. So, they came up with an idea of offering these spare computing resources in a very cheap price but without any guarantee that long-term continuity. They can just terminate these instances anytime they want (based on their clients’ demand) but will provide a warning before 2 minutes prior to termination.

AWS named them Spot Instance. Most cloud users use them for one-time short computing events such as data analysis, scientific computing events where high computing ability is needed in cheaper price.

However, we have come up with the idea of using these Spot spared computing instances in business application and any backend service which actually needs to sustain for long-time and run without any disruption.

Cheap price and warning prior to 120 seconds of termination is our focal point.

We have planned to design a mathematical algorithm (plus practical system) which can auto load balance and auto scale up new instances utilising termination notice event and run without any disruption.

Practical example: On a particular peak hour, our target application requires 15 virtual instances of specific configuration (quad core processor, 8GB ram etc.) to meet the traffic/computing demand. Now, if we use general computing virtual instances (such as EC2 from Amazon) it will cost us around 15 \* 120 USD. But if we can use Spare Spot computing instances cost will be just around 15 \* 8.7 USD. Huge savings but problem is we need to design this with fault tolerance and availability that can run without interruption.

1. Datasets

**Real-time data from AWS**

1) Current prices of the spare spot instances through Spot Instances

2) Instances termination event through Spot Fleet API.

3) Our current application load/ traffic usages from Cloudwatch monitoring system.

4) Hardware resources utilisation level (like 67% cpu usage, 80% memory usage) of any particular instance using Cloudwatch system.

5) Price prediction graph for Spot Instances

**Publicly available datasets:**  
*Link to AWS Spot Instance Pricing History from Kaggle:*

<https://www.kaggle.com/noqcks/aws-spot-pricing-market>

# Literature Review

This section surveyed all the closely related state-of-the-art work in the field of fault-tolerance using scaling of cloud resources. There are currently three main strategies for addressing fault tolerance by scaling cloud resources for software systems.

## 5.1 Cloud resource auto scaling (Reactive mechanism)

This approach does not consider the future needs. It is also often referred as threshold based rule or elasticity rule which is pre-defined by the cloud infrastructure providers.

The decision to scale up or scaling down the software system / server are determined by the last values that are monitored in the system. Amazon Web Services (AWS) auto scaling [4] mechanism and some other cloud infrastructure providers such as Azure [6], RightScale [7] offer rule based auto scaling strategy which basically allow system administrator to add or remove computational resources at a particular given time. For example: Run 7 EC2 instances from 9:00 AM to 6PM every day because it is peak business time and 2 EC2 instances for other time.

However, this auto scaling strategy is simple and convenient when system administrator can predict software system computational resource requirements really well which is very difficult at times. This also requires manual intervention by the system administrator.

(J Jiang, J Lu & G Zhang, 2013).

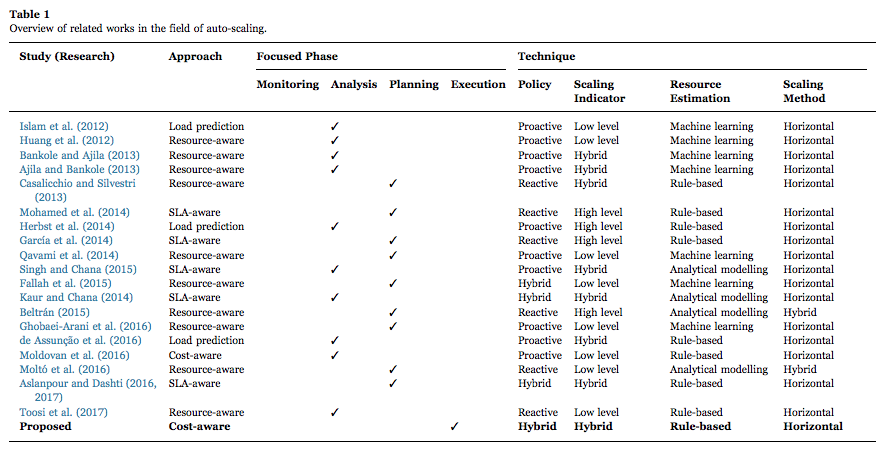
In addition, lack of anticipation of the system administrator in reactive approach may affect the performance of auto scaling of the system in great extent. When there is sudden increase in traffic load or in an event of DDOS attack, it may take several minutes to initialize of new VM instance and scaling up could happen too late. (Jingqi Yang & Chuanchang Liu, 2016).

## Predictive based Cloud Resource Scaling Strategy

In predictive scaling approach, historical data of cloud computational resources are analysed carefully to construct a mathematical scaling model to anticipate future demand of cloud computational resources by the software system. (P. Kokkinos, T.A. Varvarigou, 2013). Auto scaling is performed in advance in this approach. Few research has been done in this view of point. Histogram techniques has been used in [1] [11] to predict traffic load. The historical values neural network input or several linear regression equations. S. Rathnayake & D. Loghin, 2017 suggested a computational resource prediction model (for CPU and memory use) following the double exponential smoothing of the utilization and has compared the result with simple mean and weighted moving average of computational resource usage factors. Z. Shen, S. Subbiah, X. Gu, and J. Wilkes (2011) has applied auto-regression to predict traffic request rate and discovered that sensitivity of the algorithm performance can be easily determined by the previous history window data. N. Roy, A. Dubey, and A. Gokhale (2011) has used second order auto regressive moving average (ARMA) to predict traffic load and cloud resource usages. In addition, Z. Shen, S. Subbiah, X. Gu, and J. Wilkes (2011) has tried to classify the repeated patterns demand of the cloud computational resources.  
Z. Shen, S. Subbiah, X. Gu, and J. Wilkes (2011) has suggested that repeating patterns in cloud resource ( memory, CPU, IO and network ) usages can be identified by FFT as well and they have compared it with auto-regression, auto-correlation and histogram.

## Hybrid auto scaling mechanism

Finally, W. Iqbal, M. N. Dailey, D. Carrera, and P. Janecek (2011) has suggested hybrid cloud resource scaling strategy that use reactive approach for scaling up and regression based mechanism for scaling down. Z. Shen, S. Subbiah, X. Gu, and J. Wilkes (2011) presented a method for predicting the demand of the cloud resources. They have also presented model for prediction of error handling to accomplish adaptive cloud computational resource allocation where any prior information about the application running inside the cloud is not necessary.

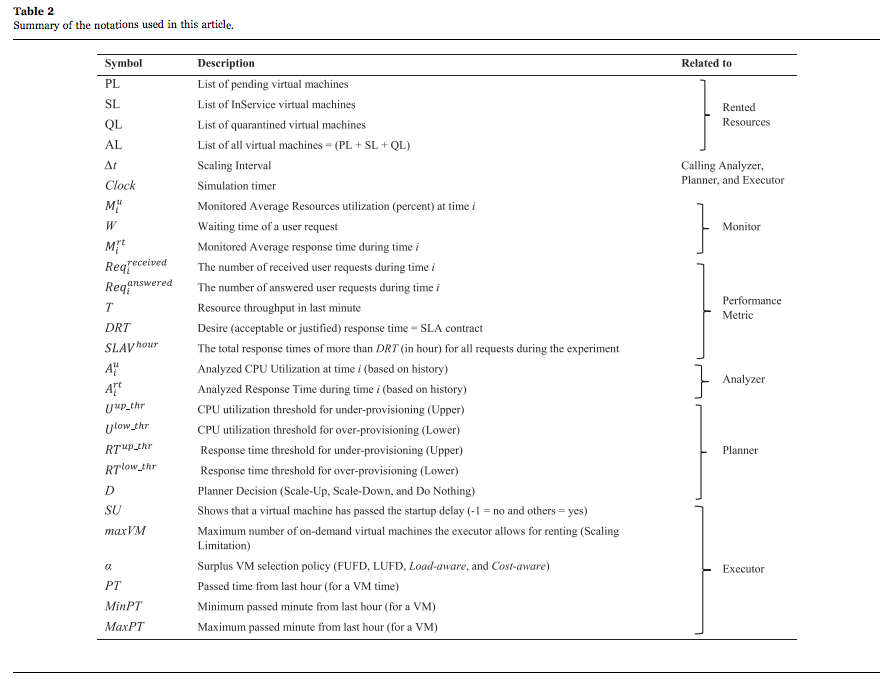


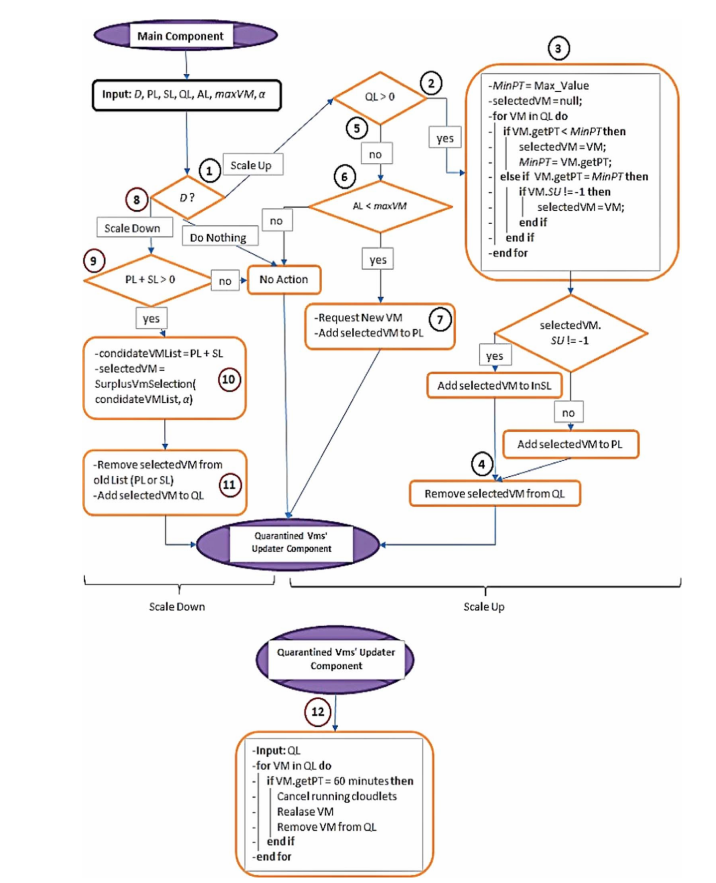
Islam et al. (2012) has suggested a model focused on analysis of resource utilization using neural network and linear regression for resource provisioning.

Toosi et al. (2017) has suggested an auto-scaling mechanism with reactive approach based on threshold-based systems to configure a load balancing formula for utilization of renewable energy.

# Current best solutions in cost aware Auto scaling mechanism

Zhang et al. (2016), Kllapi et al. (2011) proposed an auto scaling schedule based framework where optimized time and cost of independent cloud workflow execution is considered.   
According to Zhang’s scaling strategy, types of multiple cloud resources and cloud configuration combined of those resources are given high priority when designing the auto-scaling mechanism.  
  
Aslanpour el at. (2017) has proposed a solution for scaling of application equipping Suprex executor mechanism. This approach handles its operation following a MAPE concept with a strategy to save costs and limit exploitations of surplus resources.



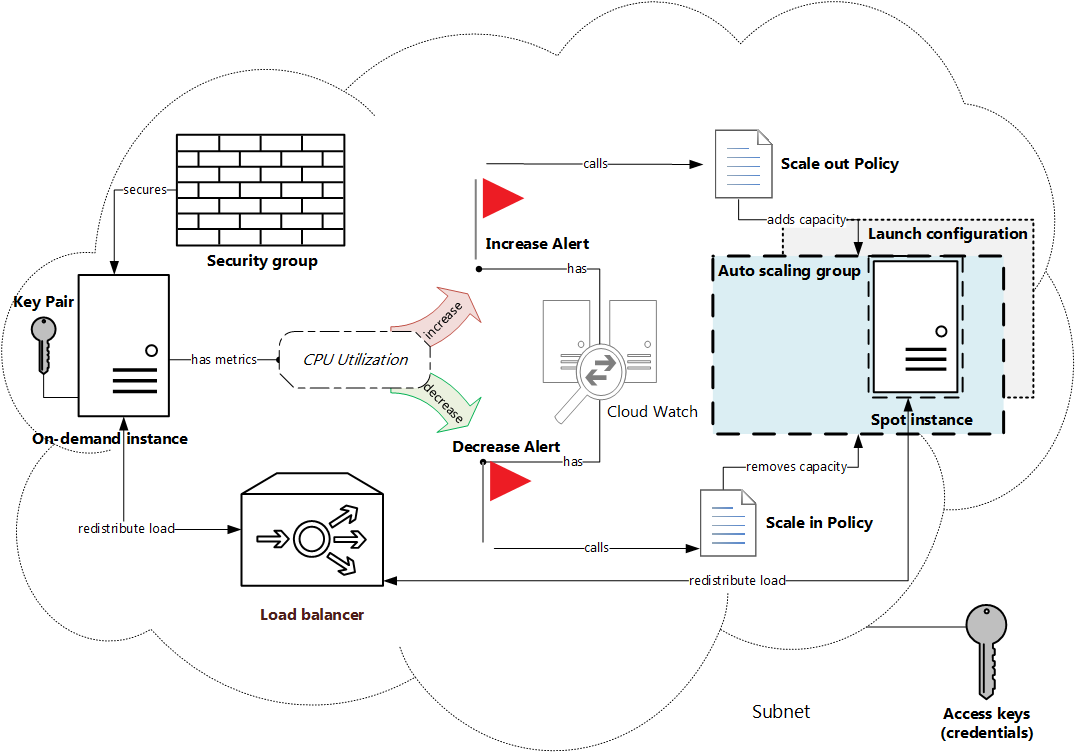


# 7. Limitation of current best solution

On-demand virtual instances are very expensive cloud computing resource. Auto-scaling with these instances based on traffic load / resource usage patterns incur huge costs.

Although schedule based cloud workflow execution introduces unique optimization strategy for scaling supporting multiplication of EC2 based on the demand of load balancing and destroying them when needed, is not price efficient in occasion of new types of DDOS attack like Yo-Yo or very irregular traffic load pattern issues.

New kind of DDOS attack (Yo-Yo) or any irregular traffic pattern incur huge costs in current best solution. So, although this auto scale mechanism can make system fault tolerant it does not provide efficient cost-flow mechanism.

****

# 

# Proposed Project Plan:

Step 1: Developing the theoretical Model

Step 2: Mathematical Justification

Step 3: Practical model build up in AWS  
  
Step 4: Load test / result data analysis  
  
Step 5: Preparing report and paper

# 9. Expected Outcome

* Working solution of a practical and fully fault tolerant software system which survives in extreme load test.
* Achieving automated Fault tolerance (FT) in minimum Cloud Infrastructure Costs (FIC) compared with currently available semi-auto server solutions for the software systems.
* Automated Cloud Infrastructure (CI) which requires no or less human intervention once configured correctly.
* Higher fault tolerance (FT) with less infrastructure costs (FIC) compared with current server solutions.

# 10. Conclusion

Irregular traffic load or new DDOS attack like Yo-Yo can cause serious performance and economic degradation in current auto scaling solution. Periodic bursts of traffic overload cause auto scaling mechanism to oscillate between scaling up and scaling down process.  
  
However, Spared instance services are very cheap to use and they can scale really fast as they are stateless and requires only simple launch configuration. So it’s possible to meet on-demand traffic loads or sudden traffic surge with our intelligent and automated scaling approach using Spot instance.

# 

# 11. References:

* 1. R. Han, L. Guo, M. Ghanem, and Y. Guo, “Lightweight resource scaling for cloud applications,” in Cluster, Cloud and Grid Computing (CCGrid), 12th IEEE/ACM International Symposium on, 2012, pp. 644–651.
  2. Gandhi, A., Dube, P., Karve, A., Kochut, A., & Zhang, L. (2014, June). Adaptive, Model-driven Autoscaling for Cloud Applications. In *ICAC* (Vol. 14, pp. 57-64).

* 1. Roy, N., Dubey, A., & Gokhale, A. (2011, July). Efficient autoscaling in the cloud using predictive models for workload forecasting. In Cloud Computing (CLOUD), 2011 IEEE International Conference on (pp. 500-507). IEEE.
  2. “Amazon auto scaling,” <http://aws.amazon.com/autoscaling/>

* 1. Mao, M., Li, J., & Humphrey, M. (2010, October). Cloud auto-scaling with deadline and budget constraints. In Grid Computing (GRID), 2010 11th IEEE/ACM International Conference on (pp. 41-48). IEEE.
  2. “Azure AutoScale”, <https://azure.microsoft.com/en-us/features/autoscale/>.
  3. “RightScale AutoScale”, http://docs.rightscale.com/faq/What\_is\_auto-scaling.html
  4. Zhang, Q., Cheng, L., & Boutaba, R. (2010). Cloud computing: state-of-the-art and research challenges. Journal of internet services and applications, 1(1), 7-18.
  5. Lorido-Botran, T., Miguel-Alonso, J., & Lozano, J. A. (2014). A review of auto-scaling techniques for elastic applications in cloud environments. Journal of Grid Computing, 12(4), 559-592.
  6. Kokkinos, P., Varvarigou, T. A., Kretsis, A., Soumplis, P., & Varvarigos, E. A. (2013, June). Cost and utilization optimization of amazon ec2 instances. In Cloud Computing (CLOUD), 2013 IEEE Sixth International Conference on (pp. 518-525). IEEE.
  7. H. Goudarzi, M. Ghasemazar, and M. Pedram, “Sla-based optimization of power and migration cost in cloud computing,” in Cluster, Cloud and Grid Computing (CCGrid), 2012 12th IEEE/ACM International Symposium on, 2012, pp. 172–179.
  8. Lorido-Botrán, T., Miguel-Alonso, J., & Lozano, J. A. (2012). Auto-scaling techniques for elastic applications in cloud environments. Department of Computer Architecture and Technology, University of Basque Country, Tech. Rep. EHU-KAT-IK-09-12.
  9. Rathnayake, S., Loghin, D., & Teo, Y. M. (2017, August). CELIA: Cost-time Performance of Elastic Applications on Cloud. In Parallel Processing (ICPP), 2017 46th International Conference on (pp. 342-351). IEEE.
  10. Yang, J., Liu, C., Shang, Y., Cheng, B., Mao, Z., Liu, C., ... & Chen, J. (2014). A cost-aware auto-scaling approach using the workload prediction in service clouds. Information Systems Frontiers, 16(1), 7-18.
  11. Fox, A., Griffith, R., Joseph, A., Katz, R., Konwinski, A., Lee, G., ... & Stoica, I. (2009). Above the clouds: A Berkeley view of cloud computing. Dept. Electrical Eng. and Comput. Sciences, University of California, Berkeley, Rep. UCB/EECS, 28(13), 2009.
  12. Varia, J. (2010). Architecting for the cloud: Best practices. Amazon Web Services, 1, 1-21.
  13. Vaquero, L. M., Rodero-Merino, L., & Buyya, R. (2011). Dynamically scaling applications in the cloud. *ACM SIGCOMM Computer Communication Review*, *41*(1), 45-52
  14. N. Roy, A. Dubey, and A. Gokhale, “Efficient autoscaling in the cloud using predictive models for workload forecasting,” in Cloud Computing (CLOUD), 2011 IEEE International Conference on. IEEE, 2011, pp. 500–507.
  15. Z. Gong, X. Gu, and J. Wilkes, “Press: Predictive elastic resource scaling for cloud systems,” in Network and Service Management (CNSM), 2010 International Conference on, 2010, pp. 9–16.
  16. Z. Shen, S. Subbiah, X. Gu, and J. Wilkes, “Cloudscale: elastic resource scaling for multi-tenant cloud systems,” in Proceedings of the 2nd ACM Symposium on Cloud Computing. ACM, 2011, p. 5.
  17. W. Iqbal, M. N. Dailey, D. Carrera, and P. Janecek, “Adaptive resource provisioning for read intensive multi-tier applications in the cloud,” Future Generation Computer Systems, vol. 27, no. 6, p. 871, 2011.
  18. Islam, S., Keung, J., Lee, K., Liu, A., 2012. Empirical prediction models for adaptive resource provisioning in the cloud. Future Gener. Comput. Syst. 28, 155–162
  19. Toosi, A.N., Qu, C., de Assunção, M.D., Buyya, R., 2017. Renewable-aware geographical load balancing of web applications for sustainable data centers. J. Netw. Comput. Appl
  20. Z. Li, H. Zhang, L. OBrien, S. Jiang, Y. Zhou, M. Kihl, R. Ranjan, Spot Pricing in the Cloud Ecosystem: A Comparative Investigation, Journal of Systems and Software, 114:1–19, 2016.
  21. H. Kllapi, E. Sitaridi, M. M. Tsangaris, Y. Ioannidis, Schedule Optimization for Data Processing Flows on the Cloud, Proc. of SIGMOD International Conference on Management of Data, pages 289–300, 2011.
  22. Aslanpour, M. S., Ghobaei-Arani, M., & Toosi, A. N. (2017). Auto-scaling web applications in clouds: a cost-aware approach. Journal of Network and Computer Applications, 95, 26-4.