International Journal of Contemporary Research in Computer Science and Technology (IJCRCST) e-ISSN: 2395-5325
Volume2, Issue 8 (August '2016)

A RESOURCE MIGRATION ON WEB APPLICATION CLUSTER TO PUBLIC CLOUDS

Dr.P.Sumitra,
Assistant Professor,
Department of Computer Science and Applications,
Vivekanandha College of Arts and Science College for Women, (Autonomous),
Elayampalayam, Namakkal, Tamilnadu

P.Kaladevi,
M.Phil Scholar,
Department of Computer Science and Applications,
Vivekanandha College of Arts and Science College for Women, (Autonomous),
Elayampalayam, Namakkal, Tamilnadu

Abstract: Web applications are accessed by millions of users over the internet via common web browser software. Cloud based virtualized services are used to support resources for web applications. Web application workloads are migrated to cloud environment to utilize the resources.web/application server, load-balancer and database are transferred from the local data center to the selected cloud infrastructure. Cloud service provider's offers computational services and Virtual Machine (VM) images for information systems. CloudGenius framework is constructed to handle process migration from web applications into public cloud resources. CloudGenius provides migration support for multi- component web applications. Evolutionary migration process for web application clusters is distributed over multiple locations. Parallel Genetic Algorithm (PGA) is applied to select migration solutions. CumulusGenius is an implementation support for CloudGenius framework.

KEYWORDS:- *CloudGenius, CumulusGenius, Analytical Hierarchy Process, Parallel Genetic Algorithm, Virtual Machine*

I. INTRODUCTION

Cloud computing is a large-scale distributed computing paradigm in which a pool of computing resources is available to users via the Internet. Computing resources, e.g., processing power, storage, software, and network bandwidth, are represented to cloud consumers as the accessible public utility services [2] . Infrastructure- as-a-Service (IaaS) is a computational service model widely applied in the cloud computing paradigm. In this model, virtualization technologies can be used to provide resources to cloud consumers. The consumers can specify the required software stack, e.g., operating systems and applications; then package them all together into virtual machines (VMs). The hardware requirement of VMs can also be adjusted by the consumers. Finally, those VMs will be outsourced to host in computing environments operated by third-party sites owned by cloud providers. A cloud provider is responsible for guaranteeing the Quality of Services (QoS) for running the VMs. Since the computing resources are maintained by the provider, the total cost of ownership to the consumers can be reduced.

The steadily increasing domination of Cloud computing in the software market means that existing applications may need to migrate to this environment in order to reap the benefits of reduced infrastructural costs and dynamic access to computational resources. Cloud service providers offer users efficient and scalable data storage services with a much lower marginal cost than traditional approaches. Cloud applications, which implement critical business functionalities for many enterprises, demand highly available infrastructures at affordable costs. This high availability can be provided through the allocation of different redundancy mechanisms to components of the cloud infrastructures. As a consequence, a major issue is related to the suggestion of a feasible IT infrastructure according to dependability and cost requirements [9].

II. PROBLEM STATEMENT

A migration from an organization-owned data center to a cloud infrastructure service implies more than few trivial steps. First, an appropriate cloud infrastructure service, or IaaS offering, is selected. This demands a well-thought decision to be made that considers all relevant factors, price, Service Level Agreement (SLA) level, network latency, data center location, availability and support quality. Second, the existing web application and its execution platform, i.e., a web/application server, a load-balancer and a database, are transferred from the local data center to the selected cloud infrastructure service [3]. Next, a migration strategy needs to be defined and applied to make the transition from the local data center to the cloud infrastructure service.

The key problem in mapping web application server components to cloud data centers, is selecting the best collection of VM images and compute services to ensure that a system's QoS targets are met. Another challenge is to satisfy conflicting selection criteria related to software and computer services. To address the complexities when migrating multi component web application server clusters, we expand the migration framework CloudGenius. The CloudGenius

framework translates cloud service selection steps into multi criteria decision-making problems using $(MC_2)^2$ and the Analytic Hierarchy Process (AHP). CloudGenius originally provides a framework that guides through a cloud migration process and offers a model and method to determine the best combined and compatible choice of VM images and compute services for a single web server.

III. RELATED WORK

In cloud computing, a resource provisioning mechanism is required to supply cloud consumers a set of computing resources for processing the jobs and storing the data. Cloud providers can offer cloud consumers two resource provisioning plans, namely short-term on-demand and long-term reservation plans. Amazon EC2 [1] and GoGrid [2] are, for instances, cloud providers which offer IaaS services with both plans. In general, pricing in on-demand plan is charged by pay-per-use basis. In this paper, Bu-Sung Lee, Dusit Niyato proposed minimizing both under provisioning and over provisioning problems under the demand and price uncertainty in cloud computing environments is our motivation to explore a resource provisioning strategy for cloud consumers. In particular, an optimal cloud resource provisioning (OCRP) algorithm is proposed to minimize the total cost for provisioning resources in a certain time period. The algorithms include the proposed OCRP, expected-value of uncertainty provisioning (EVU), maximum advance reservation provisioning (MaxRes), and nonreservation provisioning (NoRes) algorithms [9]. Many mechanisms have been proposed to allow not only a data owner itself but also a public verifier to efficiently perform integrity checking without downloading the entire data from the cloud, which is referred to as public auditing. In these mechanisms, data is divided into many small blocks, where each block is independently signed by the owner; and a random combination of all the blocks instead of the whole data is retrieved during integrity checking [3].

Vasilios Andrikopoulos, Steve Strauch, Frank Leymann providing decision support for the migration of applications to the Cloud is reduced to the selection of the appropriate IaaS provider that can support the best application performance for the least cost.As a consequence, B. J. S. Chee and C. Franklin, overcome a major issue is related to the suggestion of a feasible IT infrastructure according to dependability and cost requirements [7]. The major contribution of this paper is a modeling strategy based on a hierarchical and heterogeneous modeling for cloud infrastructure planning our public auditing mechanism. W. Kuo and M. Zuo contributions are models based on state-space models such as SPN and combinatorial models such as RBD for representing cloud applications. These heterogeneous models are hierarchically combined to depict the dependencies between components of the cloud infrastructure [8]. Alexandrov, A., Folkerts, E., Sachs, K., Iosup, A., Markl, V., and Tosun are contribution of this paper is to identify these decision points and analysis tasks, and define their dependencies. K. Ren, C. Wang, and Q. Wang of traditional approach for checking data correctness are to retrieve the entire data from the cloud, and then verify data integrity by checking the correctness of signatures of the entire data [9].

IV. PROPOSED WORK

Cloud services are deployed to provide resources and service components. CloudGenius framework is adapted to handle workload migration for web applications [2]. CloudGenius is enhanced to support migration under hybrid cloud environment. Component dependency analysis, customization and middleware service integration features are added to the system. The CloudGenius framework is enhanced to support migration under public and private cloud environment. User selection criteria based migration scheme is integrated with the system. migrating web application services to public clouds. Control flow and data dependency analysis mechanism is integrated with the migration system. Middleware services are adapted to support migration tasks.

CloudGenius originally provides a framework that guides through a cloud migration process and offers a model and method to determine the best combined and compatible choice of VM images and compute services for a single web server. With enhancements to the framework, we provide comprehensive support (HMOHM) approach. For migrations of distributed, multi-component web application clusters while factoring in data flow dependencies of components raising network traffic costs [5]. As the solution space for the problem grows exponentially, we developed a Hadoop and Genetic Algorithm (GA) based approach to cope with computational complexities in a growing market of cloud service offerings. By combining a GA with AHP, we created a Hybrid Multi-Goal Optimization Heuristic Method (HMOHM).

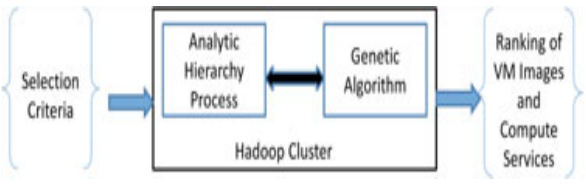

Figure 1. Hybrid Multi-goal Optimization Heuristic Method

A new challenge resulting from the combination of GA and AHP is to transfer subjective opinions stated once into an AHP-based fitness function. We developed a novel approach for AHP-based fitness functions in GAs. AHP requires pair-wise comparisons among all alternatives to normalize values

for an absolute scale. To this end, we implemented a parallel version of the HMOHM over hadoop clusters

A. Cloud Migration Process

Migrations of web applications to the cloud are expected to be linear transitions from the state of "not migrated" to "migrated". Web applications and related software stacks introduce complexity. Migrations should involve multiple repetitions and reconsiderations of past actions. CloudGenius accommodates the vicissitudes within a cloud migration by embedding decision support methods into an evolutionary migration process model. Also, existing migration strategies can be employed within the process model. CloudGenius' migration process model describes the states of a web application as depicted in a finite state machine [4]. From an initial state a web application is migrated by an application engineer into the cloud with an optional cancellation path. From its migrated state a web application can be migrated within the cloud, be it between providers or services, or due to changes in the software stack.

Eventually, a web application might be replaced or needs to be disposed and moves to an inactive state. CloudGenius provides decision support in the transition phases to a migrated web application, except reactivation. The migration process model describes the steps within a transition and embeds the decision support. A model for clusters of web applications in Business Process Model and Notation (BPMN) 2.0.

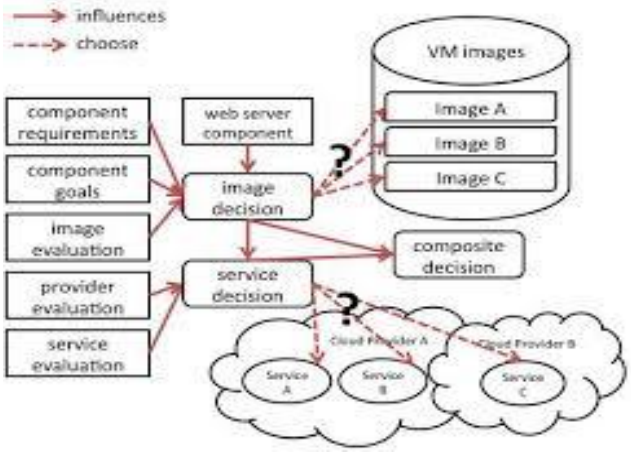

Figure 2. Cloud Migration Process Model

B. Cluster Modeling

Web application clusters comprise load balancer, database server, and potentially inter-connected web and application server components. Web applications might be divided into inter-connected front-end, business logic, and data layers to allow layer-independent scaling. In practice, web servers and application servers have merged, for example the Apache Tomcat and Red Hat JBoss Application Servers. Application servers not only provide an execution environment for web applications, but also contain a web server to handle http requests [4]. Our approach, thus, focuses on application servers in the cluster modeling. In an initial step, an application engineer has to model the application as a cluster. All components of the cluster setup must be added as elements c_h to set C and interconnections must be defined in form of component pairs in set I. The extended CloudGenius approach is capable of finding a best solution for any combination of inter-connected components.

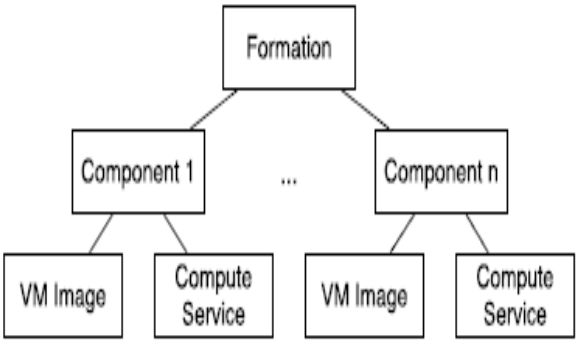

Figure 3. Structure of Cluster Model

C. Parallel Genetic Algorithm

Meta-heuristics allow coping with exponentially growing computational complexities. In particular, gargantuan search spaces of discrete solutions can be searched even without knowledge about the search space's structure. Thereby, meta-heuristics give the option to trade waiting time for accuracy. When parallelizing meta-heuristics, computation can be divided into multiple processing units which can be run on multiple server instances.The proposed algorithm facilitates parallel computations and resembles a population-based genetic algorithm meta heuristic due to the discrete nature of the search problem. The algorithm consists of four major steps: (1) create initial population, (2) assignment of fitness values, (3) selection of elite, and (4) evolution of a population. Step 1 only occurs once at the beginning of the algorithm, while steps 2-4 repeat until a termination criterion is fulfilled. Either a certain number of generations have been evolved or a certain amount of time has passed.

The algorithm expects a database of cloudVM images and compute services including their compatibilities. Structure of a cluster solution candidate by CloudGenius model. In addition to the model, three parameters must be set in the beginning to adjust the GA: (a) population size, (b) elite size and (c) maximum computation time.

1. Candidates and Populations

Genetic algorithms search over populations in multiple iterations referred to as generations. Every population consists of a predefined amount of solution candidates, each representing a viable and discrete solution to the problem. We propose candidates to be the cluster solutions F_i as described

in depict the structure of a cluster solution candidate. To initiate the GA, an initial population must be generated. Our algorithm picks cluster solutions randomly ignoring duplicates and adds them to the population until it has grown to the expected population size (a). After this step the algorithm is ready to evolve the initial population.

2. Fitness Value of Candidate

Every population is evaluated to determine the fitness values of its candidates. Our genetic algorithm employs the evaluation function j(,) for each candidate F_i. Since the AHP determines values of candidates in a comparison matrix, the whole population must be known and evaluated together. In particular, for parallel genetic algorithms this is of special interest since every parallel evaluation task must be given the whole set of candidates in a population. For implementations of the algorithm this leads to a limitation on the population size imposed by the available memory size of tasked computation units.

3. Selection of Elite

Based on the fitness values determined in the previous step, an elite of size (b) can be drawn from the current population. The elite is represented by the set of highest ranked solution candidates in the population. Any non-elite candidates are discarded in the next generation and replaced by evolved candidates. The size of the elite has been specified as a parameter in the beginning. With the termination of the algorithm a final elite is returned which includes a best solution candidate computed by the algorithm. The highest ranked elite candidate according to its value is the preferred cluster solution which is not guaranteed to be the globally best cluster solution in the search space.

4. Evolution of Population

While the elite candidates reside in the population, non-elite candidates are evolved to create a new generation and continue the search for best solutions. How the candidates are evolved depends on whether the algorithm applies a global or local search. Mutating candidates in only few attributes, e.g., changing the mapping of one component to a slightly different compute service or VM image, potentially leads to a local search. In contrast, substituting candidates with random cluster solutions from the search space that differ in a subset of attributes results in a global search.

5. Termination

Every genetic algorithm requires a termination rule that forces computations to stop and to accept the currently best solution candidate. Rules consider the current quality of the solution, or simply stop after a certain number of iterations or time limit. We propose multiple rules that lead to an eventual termination. First, the algorithm stops immediately after the sum of all uniquely discovered solution candidates passes the number of total individuals in the search space jf _ F_1, . . . , _ F_{ng}j = (jA_j $_{jSj}$)$_j$$C_j$. The algorithm halts when the number of generations exceeds the ratio. Second, a general stopping rule ends the genetic algorithm after a certain amount of time. Third, more sophisticated stopping rules can halt the algorithm when, for example, the value of the best solution has not improved over multiple generations or the actual population size undercuts the maximum population size, which means only few individuals have not been visited.

V. CONCLUSION

The main contributions of this paper are clearly identify the most important selection criteria, selection goals, and cloud service alternatives, considering the use case of migrating a web application cluster to public cloud services such as Amazon EC2 and GoGrid. We extend analytical formulations and models of our previous work for handling the migration of web server cluster components across multiple cloud data centers spanning over geographically distributed network boundaries. A hybrid decision making technique is proposed that combines multi-criteria decision making (AHP) and evolutionary optimization techniques (genetic algorithms) for selecting best compute service and VM image. A comprehensive experimental evaluation is carried out based on a realistic scenario for verifying the performance of the proposed decision making technique.

VI. REFERENCES

[1] Amazon Web Services. Web application hosting in the AWS cloud best practices [Online]. http://mediaamazonwebservices.com/AWS_Web_Hosting_Best_Practices.pdf.

[2] P. Mell and T. Grance, "The NIST definition of cloud computing, recommendations of the national institute of standards and technology," NIST Special Publication, Gaithersburg, MD, USA, 2011.

[3]. M. Menzel and R. Ranjan, "CloudGenius: Decision support for web server cloud migration," in Proc. 21st Int. Conf. World Wide Web, 2012, pp. 979–988.

[4] F.V. Louveaux, "Stochastic Integer Programming," Handbooks in OR & MS, vol. 10, pp. 213-266, 2003.

[5] Menzel, M. and Ranjan, R. (2012). Cloudgenius: decision support for web server cloud migration. In Proceedings of WWW '12, pages 979–988, New York, NY, USA. ACM

[6] B. Wang, B. Li, and H. Li, "Oruta: Privacy-Preserving Public Auditing for Shared Data in the Cloud," Proc. IEEE Fifth Int'l Conf. Cloud Computing, pp. 295-302, 2012.

[7] B. Wei, C. Lin, and X. Z. Kong, "Dependability modeling and analysis for the virtual data center of cloud computing," in *Proc. IEEE 13th Int. Conf. High Perform. Comput. Commun. (HPCC)*, Banff, AB, Canada, 2011, pp. 784–789.

[8] C. Clark, K. Fraser, S. Hand, J. G. Hansen, E. Jul, C. Limpach, I. Pratt, and A. Warfield, "Live migration of virtual machines," in Proc. 2nd Symp. Netw. Syst. Des. Implementation, May 2005, pp. 273–286.

[9] Sivadon Chaisiri, Bu-Sung Lee," Optimization of Resource Provisioning Cost in Cloud Computing", Ieee Transactions On Services Computing, Vol. 5, No. 2, April-June 2012

[10] Vasilios Andrikopoulos, Steve Strauch," Decision Support for Application Migration to the Cloud: Challenges and Vision" 3rd International Conference on Cloud Computing and Services Science, CLOSER 2013