Cloud Computing—Effect of Evolutionary Algorithm on Load Balancing

Shahrzad Aslanzadeh, Zenon Chaczko and Christopher Chiu

Abstract In cloud computing due to the multi-tenancy of the resources, there is an essential need for effective load management to ensure an efficient load sharing. Depends on the structure of the tasks, different algorithms could be applied to distribute the load. Workflow scheduling as one of those load distribution algorithms, is specifically designed to schedule the dependent tasks on available resources. Considering a job as an elastic network of dependent tasks, this paper describes how evolutionary algorithm, with its mathematical apparatus, could be applied as workflow scheduling in cloud computing. In this research, the impact of Generalized Spring Tensor Model on workflow load balancing, in context of mathematical patterns have been studied. This research can establish patterns in cloud computing which can be applied in designing the heuristic workflow load balancing algorithms to identify the load patterns of the cloud network. Furthermore, the outcome of this research can help the end users to recognize the threats of tasks failure in processing the e-business and e-since data in cloud environment.

Keywords Cloud computing · Load balancing · Resource management

Introduction

Rapid improvement in todays technologies enabled businesses to grow more quickly. Cloud computing as one of the new emerging technology provides real time services for enterprises without binding them to their organizations. Accessing to the Internet with any devices that can connect to the Internet authorized the businesses to

S. Aslanzadeh (✉) · Z. Chaczko · C. Chiu
Faculty of Engineering & IT, University of Technology, Sydney, Australia
e-mail: Shahrzad.Aslanzadeh@uts.edu.au

Z. Chaczko
e-mail: Zenon.Chaczko@uts.edu.au

C. Chiu
e-mail: Christopher.Chiu@uts.edu.au

access their information at any time [1]. As an example Dropbox is one of these popular services that will accredit the access to the information while it can be easy synchronized and updatable at anytime [2]. By applying the virtualization techniques, cloud providers can minimize the costs of resource management process. As Creeger [3] highlights, cloud users can pay to access multiple resources at any time specially in peak hours, which will help them to grow more quickly, and rolling out among their competitors.

Cloud computing is the enhanced generation of grid computing. It refers to clusters of computers with the ability of dynamic provisioning in geographically distributed networks which is customisable on users requirements. Despite of the mentioned characteristics, cloud computing has one more advantages over grid computing. The ability of virtualization enables elasticity and scalability in cloud computing which can be considered as prominence of that over grid computing. Therefore according to virtualization concept, cloud computing can be explained as the collaboration of scalable and elastic virtualized resources that can be provisioned dynamically over the internet. Moreover cloud computing was named as utility computing. Cloud computing can offer variety of services on infrastructure, platform and software. It can bring profits for businesses by saving more money on their IT infrastructures. Cloud computing is a fifth utility after water, gas, electricity and telephone. It allows users to use its services according to their demands and without any constraint [4].

The overall aim of this research is to visualize the magnitude and direction of the load between workflow tasks and jobs in cloud computing. The visualization will mainly help in monitoring the cloud load to increase the availability of the resources while minimizing the response time. Moreover the visualization will be useful in terms of identifying the anomalies and threats in workflow applications which will lead to effective decision making.

In this research a Evolutionary workflow scheduling algorithm will be investigated to identify the load patterns within cloud network. The patterns to be investigated shall highlight the interconnectivity between tasks and jobs and shall enhance the recovery plan upon failure.

Reviewing the literature variety of load balancing algorithms have been proposed to balance the load in cloud computing by representing the static and dynamic movement of the tasks. But still there is a shortage of effective visualization tool to capture the anticipatory behaviour of the workflow tasks which can project the interactions and dependencies between workflow tasks.

Load Balancing in Cloud Computing

Cloud computing is composed of several different resources, interconnected to each other to form a network or a grid. These resources should be flexible and dynamic in terms of usage and allocations. In cloud computing, load balancing is one of the major techniques that has a dramatic impact on resource availability. The term availability, was always a main concern in cloud- computing. Fundamentally, availability explains the ubiquitousness of the network information in case of resource scaling. Load balancing could be illustrated, as proper strategy for task scheduling that will lead to balanced load distribution in cloud networks. It is an important key to improve the network performance. Moreover load balancing algorithm can minimize the response time while utilizing the resource usage. The lack of proper load management can create traffics due to the long waiting time for accessing the resources. Today most of the cloud vendors are trying to use automated load balancer to enable the users, scale the numbers of their resources automatically. Promoting the availability and performance of the cloud system highlights the main goal of the automated load balancers. To design an effective load balancing algorithm, Dillon [5] suggested the following strategies:

- Load balancing algorithm should be smart enough to make load balancing decisions in a right time;
- Depends on behaviour of the application load balancer should be able to gather the information locally and globally;
- Load balancer should be designed in a centralized or distributed pattern. If the load balancer is centralized then there is a less opportunity for scalability purposes;
- Local load balancers are costing less, but the information provided by global load balancer is more accurate.

Reviewing the literature, different load balancing algorithms have been proposed to utilize the available resources. Efficient load balancing algorithm should be robust, and simple enough to be compatible with variety types of applications. The following points are defining a standard framework to design an effective load balancing algorithm [5]:

- *Complexity*: The algorithm should not be too complex as complexity will add more overhead on the system;
- *Scalability*: The algorithm should be scalable enough to manage all the existing services, if the network scaled up/down;
- *Fault tolerance*: The algorithm should be able to manage the load, even if any failure occurs in the network;
- *Performance and makespan*: The load balancing should be able to optimize the response time to enhance performance.

Load balancing methodologies are categorised into two main groups:

1. Static load balancing This method is mainly designed for homogenous and stable environments. Static load balancing algorithm cannot handle the dynamic load changes and thats why it cannot be used in real time systems.
2. Dynamic load balancing With this algorithm load balancer will manage the load dynamically at the run time. These algorithms are more flexible and will consider different attributes of the system before managing the load. Each of the static or dynamic algorithms could be divided into 4 different categories:

    a. *Centralized versus Distributed*: In centralized model, scheduler has information about all the resources. In this model generally there is more controlling over the resources and the implementation is much easier. However, in case of scalability and fault tolerance, centralized load balancing is not fully efficient. In distributed load balancing, there is no central controller for monitoring the nodes. Multiple schedulers can be used to help in scheduling the tasks. Distributed load balancing is suitable for scalable networks and it will support elasticity.

b. *Preemptive algorithm versus non-preemptive*: Preemetive algorithm will allow jobs to be interrupted. As an example, if a low priority job changes to be a high priority job, then preemetive algorithm could be so practical. On the other hands, in non-preemtive task scheduling methods, no interruption is allowed until all the scheduled processes are completed.
c. *Mediate versus batch mode*: In immediate mode, jobs will be assigned as soon as they arrive. So there is no waiting time for them. In batch mode, jobs will be grouped base on mapping criteria then each group will be assigned to proper resources for processing.
d. *Independent versus workflow*: Workflow tasks are describing the tasks with some sort of dependencies. As an example finishing time of one task can be the start time of the other tasks. Most of the workflow tasks are represented with DAG graphs or Petri nets, and other language modelling tools such as XPDL and XML. On the other hand with independent scheduling approach, tasks could be assigned independently without considering the prerequisites for implementing a specific task.

Therfore we can summarise the load balancing benefits as follows:

- Load will be distributed evenly
- Processing time will be minimized
- Resource utilization will be maximized
- Availability of the system will be increased
- Performance will improve
- Resource will be more utilized
- Resource consumption will be minimized

Considering the benefits of the load balancing, there are some challenges that need to be addressed within load balancing concept.

Throughput Most of the loads balancing algorithms are trying to complete the highest numbers of tasks in given period of time. High throughput is an essential component to ensure the better performance of the system.

Response time The time that the load balancer needs to assign the tasks on available resources, for smaller response time, the performance is higher.

Overhead For each load balancing algorithm there is an associated overhead which is created by the process communications, tasks allocation and processor operation. The large overhead can impact the performance of the system.

Fault tolerance The load balancing algorithm should be able to find any node failure while allocating the tasks on available resources. Scalability As one of the specification of the cloud computing is related to its scalability, therefore each of the load balancing algorithm should be able to scale updown base on the status of the network.

Proposition of Evolutionary Algorithms in Cloud Computing

Elastic Workflow Scheduling Model

Elasticity could be highlighted as one of the main characteristics of the cloud. In load balancing, elasticity could be explained as the ability of the system to allocate and reallocate the resources dynamically. Similarly in workflow load scheduling, elasticity could be interpreted in context of resource allocations for dependent tasks. In our proposed workflow load balancing, inspired from Fig. 16.1, tasks could be seen as an elastic network of mass spring model, in which each task is connected to its dependent with a spring. The rigidity of the springs will capture a standard pattern that could explain the impact of the task dependencies level on load balancing [6]. Hooks law, advised in 17th century by Robert Hook, elaborates that the expansion of a spring is proportional with the force that was imposed on it. The constant factor K highlights the elasticity ratio of the materials. In Hooks law elasticity is referring to the ability of the elastic body to return to its original status, after it was bent, stretched or squeezed [7]. If force is greater than the elasticity limitation of the material, it will cause a permanent deformation. Figure 16.2 is illustrating the spring behavior under force [8].

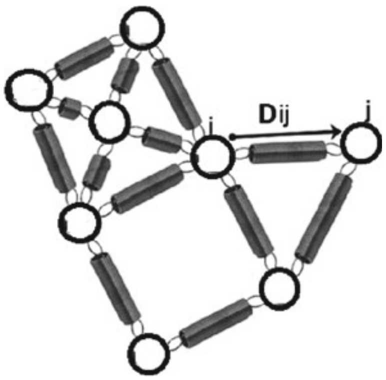

Fig. 16.1 Network of mass spring model

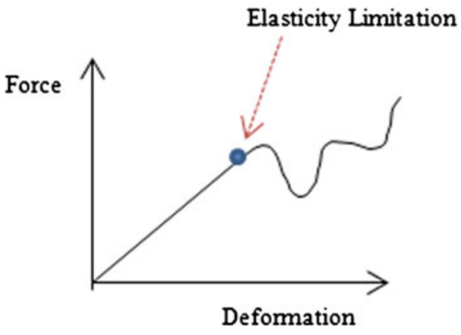

Fig. 16.2 Elasticity behavior

Generalized Spring Tensor Model in Cloud Computing

Various computational algorithms have been suggested to analyze the dynamics and complexity of the cloud. Among these algorithms, Coarse-gained algorithms could be highlighted as one of the important methods, developed to study the complication of the elastic networks. Elastic Network Model (ENM) is one of the coarse-gained models that particularly used in science to study the fluctuation and magnitudes of the proteins movement. Anisotropic network model (ANM) and Gaussian Network Model (GNM) are two different types of ENM algorithm [9, 10]. GNM is designed to analyze the B-factor values in network of proteins. B-Factor explains the protein dynamics by reflecting the magnitude of the protein fluctuation in related position. Moreover the fluctuation of the atoms is considered as Gaussian distributed along X, Y, and Z vector [10]. Furthermore, ANM model is able to evaluate the direction of the fluctuation in network of elastic nodes. In contrast with GNM model, ANM cannot explore the magnitude of the proteins fluctuations. In theory, ANM is complying with Generalized Hooks law which is featuring the relation between the stress and strain in 3 dimensions. Based on Hooks principal, imposing force P can cause deformation on elastic bodies. Inspiring from Hooks law, ANM removed the required energy minimization, and proposed a simpler version of Hooks law which could calculate the fluctuation direction of the load. As the fluctuation in this case is only limited to longitude axis of nodes i and j, therefore the magnitude of this fluctuation is senseless [11]. To overcome the above mentioned limitations of ANM and GNM algorithms, generalized spring tensor (STeM) was considered by Bahar [9] who explored the fluctuation and direction of the proteins simultaneously. Using Go-like algorithm as the potential model, Generalized spring tensor (STeM), a coarse-gained algorithm, is able to manage the network complexity by analyzing the magnitude and direction of the load [6]. In this model the interaction between two nodes i and j will be investigated but not in a linear structure. Next stage explains the mathematical apparatus of the STeM algorithm in more details.

STeM Algorithm and Workflow Load Balancing

Using STeM algorithm, the proposed workflow load balancing will highlight the direction and magnitude of the load fluctuations in cloud network. To manage the load efficiently the dependency ratio of the tasks, which was illustrated by the rigidity of the springs should be defined. Figure 16.3 is modeling two dependent tasks of i and j in 3D environment, where $D_i = (x_i, y_i, z_i)$ and $D_j = (x_i, y_i, z_i)$ are the positions of i and j in 3 Dimensions. Based on Hookeans principal, Force P could move an elastic body from its relaxed position by the value of d to the new state. The transition between original state and the new state could be impacted by constant

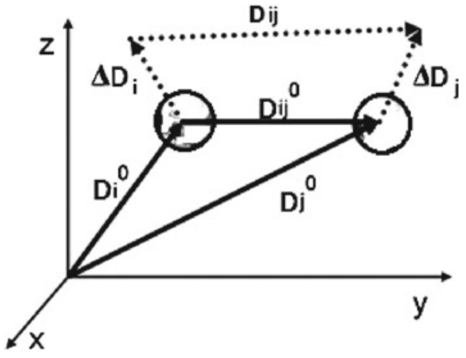

Fig. 16.3 Mass spring network

factor k. Based on the quality of the elastic body, the force could cause shearing, stretching or compression. According to Eq. 16.1 for some elastic bodies, if the directions of the stress and the strain are the same, then the magnitude of the force will be proportional [11, 12].

$$P = kd \tag{16.1}$$

However, if the force and displacement are not in a same direction the relation between stress and strain could be explained as Eq. 16.2:

$$P = k(\alpha d_1 + \beta d_2) \tag{16.2}$$

K is a second order tensor and α and β are real numbers with shifting values. Hooks law could also connect the stress and strain in three dimensional objects. In this sense, k could be explained as a 3×3 matrix that will be multiplied by movement of d in three dimensions to represent the applied force [13, 14]. Therefore the force vector will be depicted by Eqs. 16.3 and 16.4.

$$p = \begin{bmatrix} k_{11} & k_{12} & k_{13} \\ k_{21} & k_{22} & k_{23} \\ k_{31} & k_{32} & k_{33} \end{bmatrix} \times \begin{bmatrix} d_1 \\ d_2 \\ d_3 \end{bmatrix} \tag{16.3}$$

$$p_i = k_{i1}d_1 + k_{i2}d_2 + k_{i2}d_2 = \sum_{i=1}^{3} k_{ij}d_j \tag{16.4}$$

Adhering to the same principal, and inspiring from Go-Like model, STeM algorithm, will use Hessian matrix, which is composed of four 3×3 matrices. These matrices are resulted from contribution of bond bending, angel, torsional and non-local interaction between dependent task. Each of these hessian matrices could be calculated with Eq. 16.5.

$$H_{ij} = \begin{bmatrix} \frac{\partial^2 v_{1,(r,r_0)}}{\partial Xi \partial Xj} & \frac{\partial^2 v_{1,(r,r_0)}}{\partial Xi \partial Yj} & \frac{\partial^2 v_{1,(r,r_0)}}{\partial Xi \partial Xj} \\ \frac{\partial^2 v_{1,(r,r_0)}}{\partial Yi \partial Xj} & \frac{\partial^2 v_{1,(r,r_0)}}{\partial Yi \partial Yj} & \frac{\partial^2 v_{1,(r,r_0)}}{\partial Yi \partial Zj} \\ \frac{\partial^2 v_{1,(r,r_0)}}{\partial Zi \partial Xj} & \frac{\partial^2 v_{1,(r,r_0)}}{\partial Zi \partial Yj} & \frac{\partial^2 v_{1,(r,r_0)}}{\partial Zi \partial Zj} \end{bmatrix} \tag{16.5}$$

By combining the value of the four matrices, Eq. 16.6 is showing the potential correlation between two nodes:

$$(\Delta r_i) \cdot (\Delta r_i) = \frac{3k_\beta T}{\gamma}(H^{\top}_{3i-2,3j-2} + H^{\top}_{3i-1,3j-1} + H^{\top}_{3i,3j}) \tag{16.6}$$

As a future work, by using the correlation result of dependent tasks formulated in Matlab, we will highlight the areas with high dependencies percentage. This pattern could be used to predict better resource management plan and performance enhancement in cloud.

Conclusion and Future Work

This paper is proposing a new workflow load balancing algorithm that could be used in an elastic cloud. STeM algorithm has been suggested as the potential algorithm that could improve the load management by explaining the magnitude and direction of the fluctuation between dependent tasks. The model will help finding the level of the dependencies between each task by acknowledging the magnitude and direction of the load. Considering the behavior of the depended tasks, this approach will explain a pattern for managing the load balancing more efficiently. The expected benefits of the proposed algorithm will improve load balancing technique which could result in better performance rate. Moreover as the pattern will define the level of the tasks dependencies, a better fault tolerance and risk management could be predictable.

References

1. Khiyaita, A., Zbakh, M., El Bakkali, H., El Kettani, D.: Load balancing cloud computing: state of art. In: Network Security and Systems (JNS2), pp. 106–109 (2012)
2. Sawant, S.: A genetic algorithm scheduling approach for virtual machine resources in a cloud computing environment (2011)
3. Vöckler, J.-S., Juve, G., Deelman, E., Rynge, M., Berriman, B.: Experiences using cloud computing for a scientific workflow application. Condor **300**, 15–24 (2011)
4. Zhang, C., De Sterck, H., Jaatun, M., Zhao, G., Rong, C.: CloudWF: a computational workflow system for clouds based on Hadoop. Cloud Comput. **5931**, 393–404 (2009)
5. Galante, G., de Bona, L.C.E.: A survey on cloud computing elasticity. In: IEEE Fifth International Conference on Utility and Cloud Computing (UCC), pp. 263–270 (2012)

6. Lin, T.-L., Song, G.: Generalized spring tensor models for protein fluctuation dynamics and conformation changes. BMC Struct. Biol. **10**(Suppl. 1), S3 (2010)
7. HowStuffWorks 'Elasticity': http://science.howstuffworks.com/dictionaryphysics-terms/elasticity-info.htm
8. Hookes Law elasticity limitation: http://www.clickandlearn.org/physics/sph4u/hookeslaw.htm
9. Bahar, I., Rader, A.J.: Coarse-grained normal mode analysis in structural biology. Curr. Opin. Struct. Biol. **15**(5), 586–592 (2005)
10. Atilgan, A.R., Durell, S.R., Jernigan, R.L., Demirel, M.C., Keskin, O., Bahar, I.: Anisotropy of fluctuation dynamics of proteins with an elastic network model. Biophys. J. **80**(1), 505–515 (2001)
11. Relation, S.: Generalized Hooks Law (2009)
12. Yang, G., Kabel, J., Rietbergen, B.V.A.N., Odgaard, A., Huiskes, R.I.K., Cowin, S.C.: The Anisotropic Hooke's law for cancellous bone and wood. J. Elast. **2138**, 125–146 (1999)
13. Aweya, J., Ouellette, M., Montuno, D.Y., Doray, B., Felske, K.: An adaptive load balancing scheme for web servers. Int. J. Netw. Manag. **12**(1), 3–39 (2002)
14. Gaussian network model—Wikipedia, the free encyclopedia: http://en.wikipedia.org/wiki/Gaussiannetworkmodel