International Journal of Computer Applications (0975 – 8887)
Volume 180 – No.41, May 2018

Enhanced Resource Provisioning Strategies for Scientific Workflows in Cloud Environment: A Survey

S. Sridevi
Assistant Professor
Velammal Engineering College, Chennai, India

Jeevaa Katiravan
Associate Professor
Velammal Engineering College, Chennai, India

ABSTRACT

Efficient resource provisioning is the key challenge that brings the best quality of service which is beneficial both to the users and CSPs. Since cloud computing aims at providing adaptive provisioning as pay-per-use basis, dynamic resource provisioning is a critical research issue. The on-demand provisioning and resource availability in cloud computing make it ideal for executing scientific workflow applications. To ensure better performance, there is a need for auto-scaling. The problem of assigning resources to tasks and orchestrating their execution to preserve the dependencies of workflows is NP-complete. [2]Hence, no optimal solution can be found in polynomial time. NP-complete problems are often addressed by using heuristic or meta-heuristics approaches. This survey presents the existing auto-scaling techniques and meta-heuristics approaches for VM placements of cloud workflows.

Keywords

Workflows, resource provisioning, auto-scaling, meta-heuristics

1. INTRODUCTION

"Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g. networks, servers, storage application and services) that can be provisioned and released with minimum management effort or service provider interaction."[1]

From the past decade, cloud computing paradigm has witnessed a rapid development for delivering services to users over a network. A huge number of commercial cloud service providers (CSPs) are delivering various cloud computing services. The working models of cloud computing are: *Deployment models and Service models.* Deployment models define the type of access to the cloud which can be of four types: Public, Private, Community and Hybrid. The cloud model is composed of three different service offerings as:

- *Infrastructure-as-a-Service (IaaS)* is the delivery of technology infrastructure as an on-demand scalable service. It provides access to fundamental resources such as physical machines, virtual machines, virtual storage, etc.
- *Platform-as-a-Service (PaaS)* provides the runtime environment for applications, development & deployment tools, etc.
- *Software-as-a-Service (SaaS)* model allows using software applications as a service to end users. SaaS is a software delivery methodology that provides licensed multitenant access to software and its functions remotely as a Web-based service.

Efficient resource provisioning is the key challenge that brings the best quality of service which is beneficial both to the users and CSPs. Resource provisioning comprises of resource discovery, scheduling and allocation processes.

Cloud provisioning is the allocation of a cloud provider's resources to a customer. When a cloud provider accepts a request from a customer, it must create the appropriate number of virtual machines (VMs) and allocate resources to support them. The process is conducted in several different ways: advance provisioning, dynamic provisioning and user self-provisioning. With advance provisioning, the customer contracts with the provider for services and the provider prepares the appropriate resources in advance of start of service. The customer is charged a flat fee or is billed on a monthly basis. With dynamic provisioning, the provider allocates more resources as they are needed and removes them when they are not. The customer is billed on a pay-per-use basis. With user self-provisioning, the customer purchases resources from the cloud provider through a web form, creating a customer account and paying for resources with a credit card. The provider's resources are available for customer use within hours, if not minutes.

2. SCIENTIFIC WORKFLOWS

Two types of applications are considered for cloud migration: bag-of-tasks (BoTs) which consists of many independent tasks and workflows in which tasks are interconnected through dependencies. "Scientific workflow is a specification of a scientific process, which represents, streamlines, and automates the analytical and computational steps that a scientist needs to go through from dataset selection and integration, computation and analysis, to final data product presentation and visualization. "Workflow organizes the task sequence of execution within the deadlines and recognizes inter-task dependency. Tasks in the workflow have a relation like parent and child i.e. the execution of parent task should be completed before the child task starts its execution. The tasks in the workflow and the dependencies among them can be represented as a Directed Acyclic Graph (DAG) G = (T, E) in which T is the set of tasks and information about data dependencies among them is represented by the set of edges E.

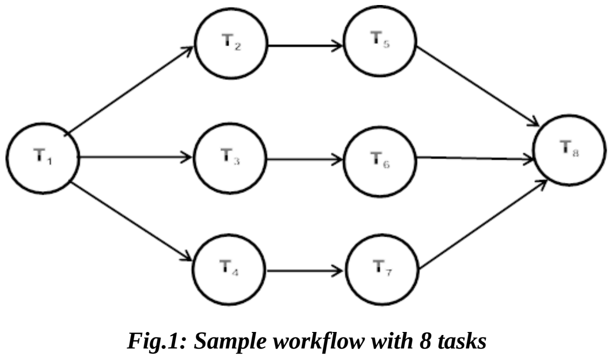


Fig.1: Sample workflow with 8 tasks

Fig.1 shows the tasks and their sequence of execution in a workflow graph G. The tasks T_2,T_3, and T_4 start their execution only after the completion of the task T_1. Hence T_1 is the parent task and T_2T_3, T4are the child tasks. Similarly, the task T_8 can be executed only after the completion of T_5, $T_6$and T_7. T_1 act as the entry node and T_8 as the exit task.

2.1 Why Scientific Workflows?

Scientific workflows are a flexible representation to declaratively express applications with data and control dependencies, and are mainstream in domains such as astronomy, physics, climate science, earthquake science, biology, and others.

2.2 Pegasus Workflow Management System

The Pegasus project encompasses a set of technologies that help workflow-based applications execute in a number of different environments including desktops, clusters, grids and clouds. It bridges the scientific domain and the execution environment by automatically mapping high-level workflow descriptions onto distributed resources.

Pegasus has been used in a number of scientific domains including astronomy, bioinformatics, earthquake science, gravitational wave physics, ocean science, limnology, and others.

2.3 Synthetic Workflows

CyberShake: The Cybershake workflow is used by the Southern California Earthquake Center (SCEC) to characterize earthquake hazards in a region using the Probabilistic Seismic Hazard Analysis (PSHA) technique.

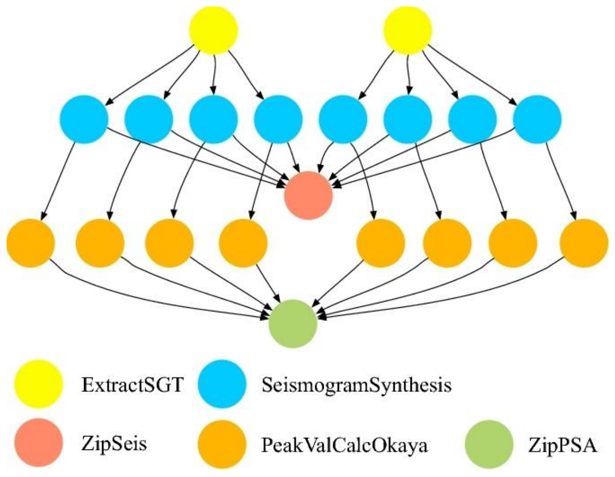


Fig.2: Structure of CyberShake

Epigenomics: The Epigenomics workflow is essentially a data processing pipeline that uses the Pegasus Workflow Management System to automate the execution of the various genome sequencing operations. The DNA sequence data generated by the Illumina-Solexa Genetic Analyzer system is split into several chunks that can be operated on in parallel. This workflow is being used by the Epigenome Center in the processing of production DNA methylation and histone modification data.

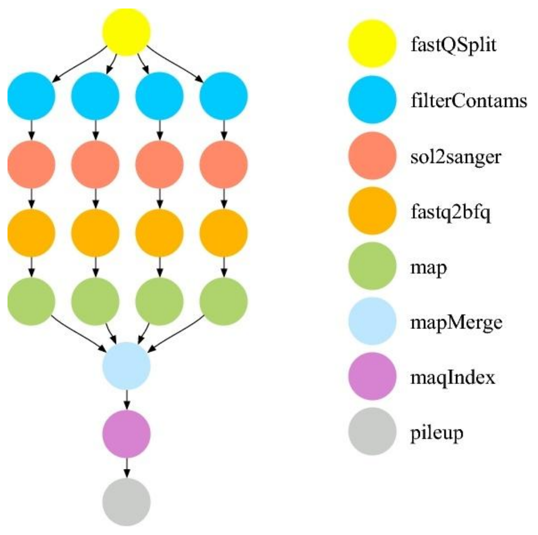


Fig.3: Structure of Epigenomics

Ligo: The Laser Interferometer Gravitational Wave Observatory (LIGO) is attempting to detect gravitational waves produced by various events in the universe as per Einstein’s theory of general relativity. The LIGO Inspiral Analysis Workflow is used to analyze the data obtained from the coalescing of compact binary systems such as binary neutron stars and black holes.

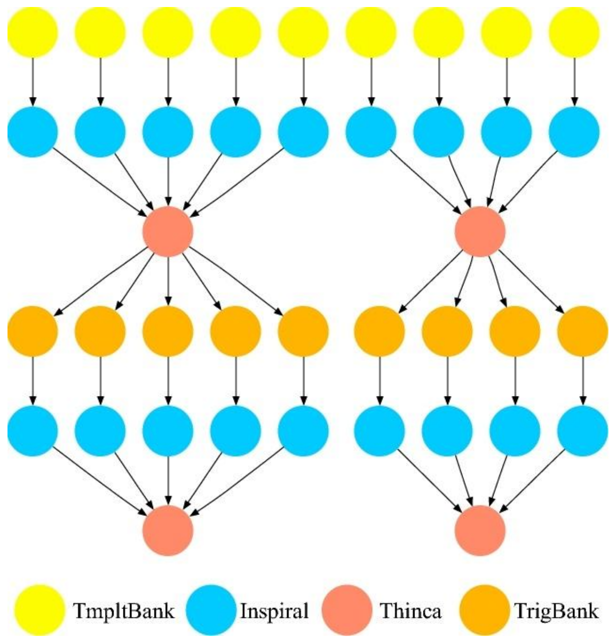


Fig.4: Structure of LIGO

Montage: Montage has been created by the NASA/IPAC Infrared Science Archive as an open source toolkit that can be used to generate custom mosaics of the sky using input images in the Flexible Image Transport System (FITS) format. During the production of the final mosaic, the geometry of the output is calculated from the geometry of the input images. The inputs are then re-projected to be of the same spatial scale and rotation.

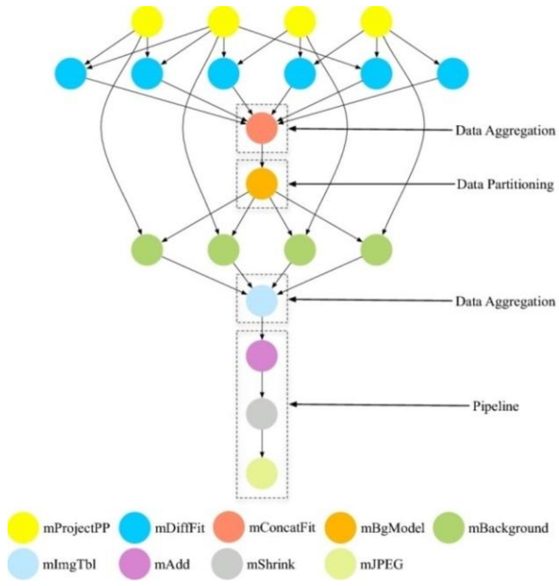


Fig.5: Structure of Montage

Sipht: The SIPHT workflow from the bioinformatics project at Harvard is used to automate the search for untranslated RNAs (sRNAs) for bacterial replicons in the NCBI database.

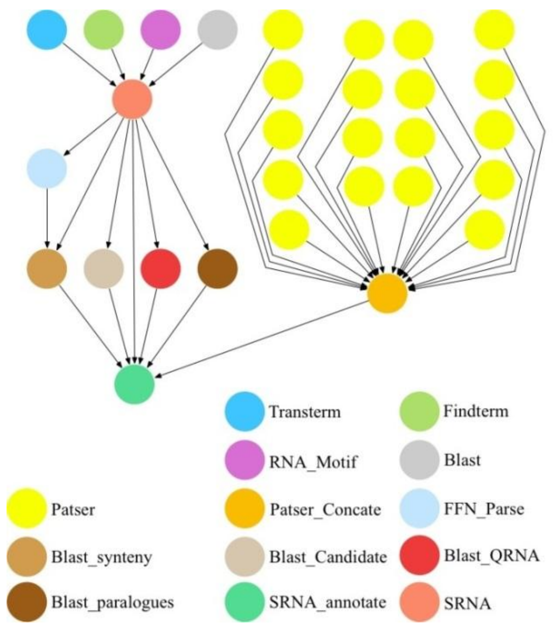


Fig.6: Structure of SIPHT

3. SIMULATORS FOR CLOUD WORKFLOWS

3.1 CloudSim[4]

CloudSim is a holistic framework for modeling cloud computing environments and performance testing application services. The main advantages of using CloudSim are: *time effectiveness, flexibility and applicability*. It offers the following novel features: (i) support for modeling and simulation of large-scale cloud computing environments. (ii) a self-contained platform for modeling Clouds, service brokers, provisioning, and allocation policies. (iii) support for simulation of network connections among the simulated system elements; and (iv) facility for simulation of federated Cloud environment that inter-networks resources from both private and public domains.

3.2 WorkflowSim[5]

WorkflowSim is an extension of the existing CloudSim simulator by providing a higher layer of workflow management.

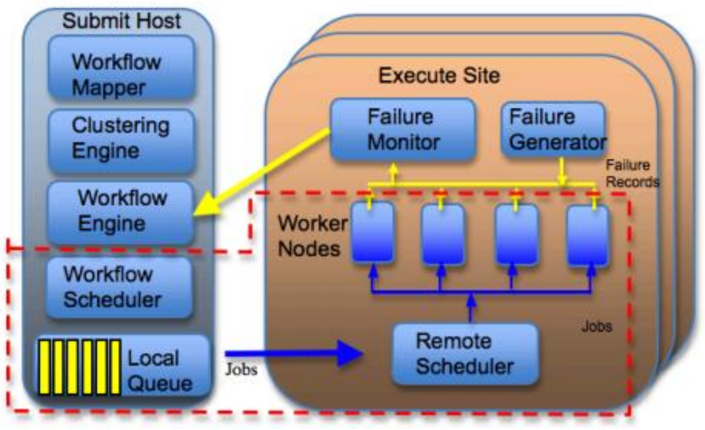


Fig 7: WorkflowSim overview. The area surrounded by red lines is supported by CloudSim

The model of WMS contains a Workflow Mapper to map abstract workflows to concrete workflows, a Workflow Engine to handle the data dependencies; and a Workflow Scheduler to match jobs to resources.

3.3 ElasticSim [6]

ElasticSim is an extension of the existing CloudSim simulator by adding support for resource runtime auto-scaling and stochastic task execution time modeling. Most of the existing workflow scheduling algorithms are static and are based on deterministic task execution times.

The ElasticSim consists of workflow modeling, resource modeling, scheduling and executing layers. The Workflow modeling layer is used to handle the dependencies of workflows and to model stochastic task execution times. The Task-Resource mapping layer is used to make runtime resource auto-scaling plan and to schedule workflows. A Cloud Resource layer is used to model VMs with interval based pricing models.

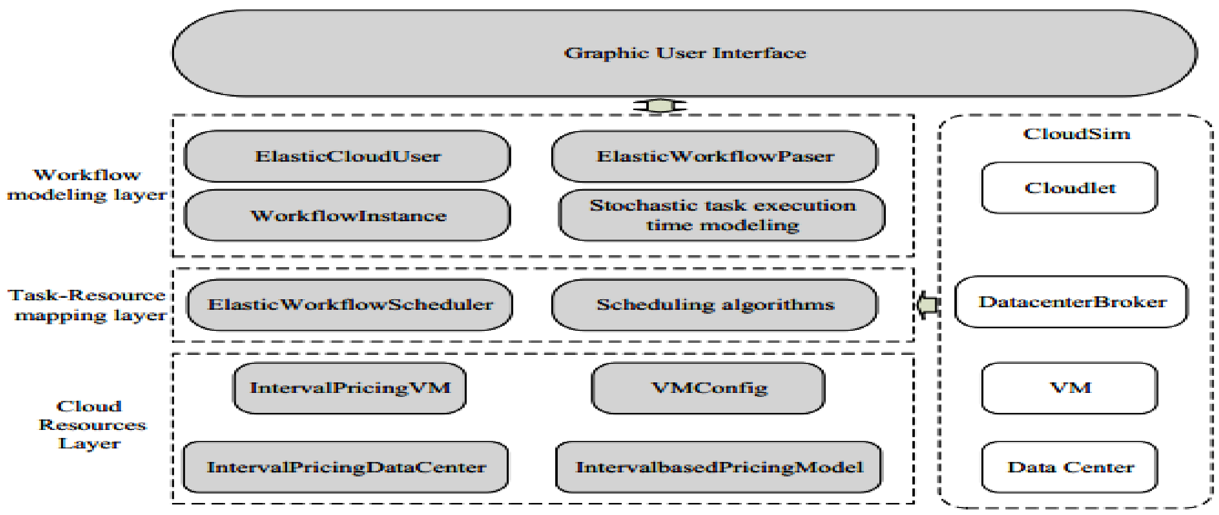


Fig 8: Layers of ElasticSim

4. CLOUD PROVISIONING STRATEGIES

Since the problem of assigning resources to tasks and orchestrating their execution to preserve the dependencies of workflows are NP-complete [2], no optimal solution can be found in polynomial time. Cloud computing offers adaptive resource provisioning by pay-per-use basis. Hence the survey will focus on the enhanced resource provisioning techniques. To ensure better performance dynamically, there is a need for auto-scaling.

Auto-scaling can be done in two ways: horizontal-scaling which adds or removes the number of VMs and vertical-scaling which controls the size of the VM.[3]Vertical scaling yields better cost effective results than horizontal scaling.

The best resources from the resource pool are selected by applying heuristic approaches. The selected resources are given to the auto-scaling decider. Fig 2 shows the model of a scaling planner. This planner is implemented dynamically so that the system can give the optimal and elastic workflow schedule.

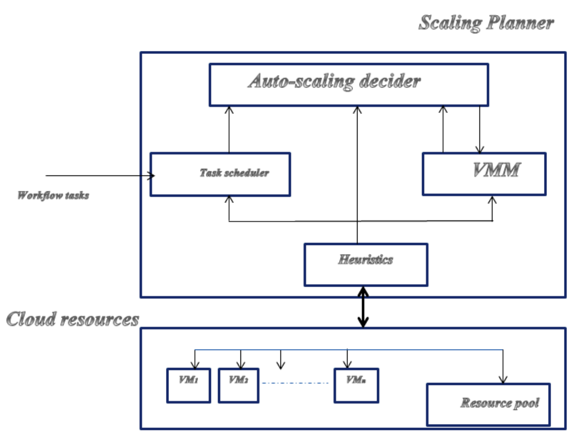


Fig.9: Scaling Planner

5. AUTO-SCALING TECHNIQUES : A LITERATURE REVIEW

The core of auto-scaling is the resource estimation as it determines the efficiency of resource provisioning.

5.1 Rule-based resource estimation for cloud workflows: [7]

Simple rule-based approaches involve no accurate resource estimation. Though it is very easy to implement, it requires understanding of the application characteristics and expert knowledge to determine the thresholds and proper actions.

5.2 Structure aware resource estimation: [8]

This approach overcomes the problems of over-provisioning and under-provisioning. In this approach the number of tasks in the workflow and the arrangement of tasks in the workflow (structure) are used to determine the number of Virtual Machines required executing the workflow.

5.3 Fuzzy inference based resource estimation for cloud workflows: [9]

Fuzzy clustering is an effective method to divide resources. The resources have five different characteristics:

i. *Processing capacity:* the computation cost in unit time of the processing unit in resource network, it is the weight of processing unit.

ii. *Average communication capacity:* the mean value of the communication costs of the links that is connected with the processing unit, the capacity is the average weight value of the edges connected this processing unit.

iii. *Maximum communication rate:* the link that has the fastest transmission speed is connected with this processing unit.

iv. *Network location:* the variance of the distance from a processing unit to the center processing unit. It shows the position of the processing unit in the network.

v. *The number of links:* the number of the edges connected this processing unit.

The FCBWTS[9] algorithm is an application scheduling algorithm for the heterogeneous environment, which has three phases: (1) a resource clustering phase for dividing the processors; (2) a task prioritizing phase for computing the priorities of all tasks; and (3) a processor allocation phase for scheduling tasks on the suitable processor.

5.4 Turnaround time driven auto-scaling: [10]

In the dynamic scaling mechanism, a turnaround time monitor and an event alarm is used to trigger the relative scaling policy and execute it. In the schedule-based auto scaling module, the Pre-configuration Maker can automatically conduct serial testing with the help of elastic load tester.

5.5 Machine Learning based resource estimation: [11]

The first stage is the prediction method which gets the task, virtual machine type, cloud and generates the runtime parameter value using a ML method. In the second stage, the predicted runtime parameters are used together with pre-runtime parameters to predict the execution time of the task.

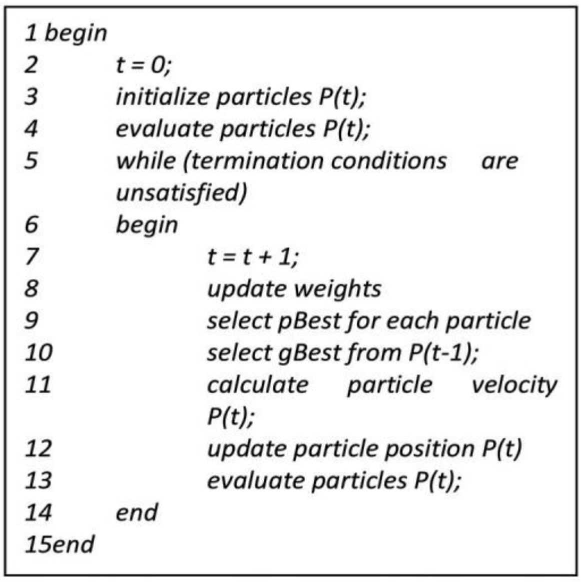
6. STRATEGIES FOR VIRTUAL MACHINE SELECTION & PLACEMENT: A Review

The output of auto-scaling decider is the number of VMs required to execute the workflow tasks. Selection of that required number of VMs from the resource pool and placing them to make optimal execution is the next phase of work.

Virtual Machine placement problem has been solved as heuristic, meta-heuristic, deterministic and approximation algorithm based on the angle of VMP issue. The majority of works uses meta-heuristic algorithms since it provides good quality solutions. This survey will discuss about the existing meta-heuristic approaches for VMP. A meta-heuristic is a higher-level procedure designed to generate a sufficiently good solution to an optimization problem, especially with incomplete or imperfect information or limited computation capacity.

6.1 Particle Swarm Optimization: [12]

PSO is an evolutionary computational technique introduced by Kennedy and Eberhart in 1995 motivated by social behavior of the particles. Each particle is allied with position and velocity and moves through a multi-dimensional search space. In each iteration, each particle adjusts its velocity based on its best position and the position of the best particle of the whole population. PSO combines local search methods with global search methods trying to balance exploration and exploitation. PSO has been used in a scheduling heuristic which dynamically balances the task mappings when resources are unavailable.



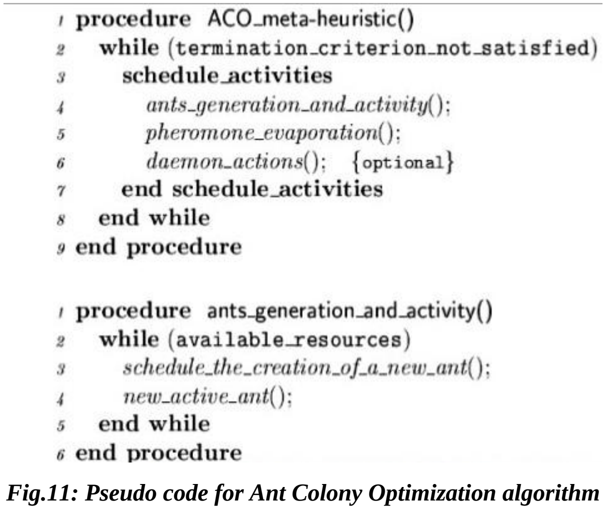
```
begin
    t = 0;
    initialize particles P(t);
    evaluate particles P(t);
    while (termination conditions are unsatisfied)
    begin
        t = t + 1;
        update weights
        select pBest for each particle
        select gBest from P(t-1);
        calculate particle velocity P(t);
        update particle position P(t)
        evaluate particles P(t);
    end
end
```

Fig.10: Pseudo code for Particle Swarm Optimization algorithm

6.2 Ant Colony Optimization (ACO): [13]

ACO is a population-based meta-heuristic that can be used to find approximate solutions to difficult optimization problems. ACO is inspired by the behavior of real ants finding the shortest path between their colonies and a source of food. This approach was introduced by Dorigo in 1992. While walking amid their colony and the food source, ants leave pheromones on the ways they move. The pheromone intensity on the passages increases with the number of ants passing through and drops with the evaporation of pheromone. as the time goes on, smaller paths draw more pheromone and thus, pheromone intensity helps ants to recognize smaller paths to the food source.

ACO methods are useful for solving discrete optimization problems that need to find paths to goals such as traveling salesman problem, multidimensional knapsack problem, quadratic assignment problem, etc.



```
procedure ACO_meta-heuristic()
  while (termination_criterion_not_satisfied)
    schedule_activities
      ants_generation_and_activity();
      pheromone_evaporation();
      daemon_actions();   {optional}
    end schedule_activities
  end while
end procedure

procedure ants_generation_and_activity()
  while (available_resources)
    schedule_the_creation_of_a_new_ant();
    new_active_ant();
  end while
end procedure
```

Fig.11: Pseudo code for Ant Colony Optimization algorithm

6.3 Intelligent Water Drops Algorithm (IWD): [14]

The IWD is a population based algorithm proposed by Shah-Hosseini in 2007 for solving combinatorial problems. The IWD algorithm mimics the dynamics of river system and the behavior of water drops in the rivers. Each and every intelligent water drop is assumed to have two properties. One is the amount of soil each water drop carries, when they flow from one node to another. The second is the velocity of the water drop. The nature water drops find an optimal path to their destination. This process of finding an optimal path is used to solve many optimization problems.

```
1- Begin;
2- Initialize the value of the static parameters;
   $a_v, b_v, c_v, a_s, b_s, c_s, N_{IWD}, \rho_{IWD}, \rho_n, N, E$
3- Initialize the value of the dynamic parameters;
   $MaxSoil, MinSoil, MaxVel, MinVel$
4- Initialize the positions of the IWDs;
5- For each IWD;
   a. For the IWD residing in node i, choose the next node j, using (1),
      (2), (3);
   b. Compute the new velocity of water drop in node j using (4);
   c. Compute the soil that the IWD carriers from moving from node i to
      node j using (5), (6);
   d. After an IWD selects the node j, compute the soil of the edge
      between the two nodes and the soil of IWD using (7);
   Repeat for a specific number of iterations while all IWD construct their
   solution.
6- Find the iteration-best solution $T^{IB}$ from all the solution $T^{IWD}$ found by
   the IWDs, using (9)
7- Update the soils of paths included in the current iteration-best solution
   path using (10);
8- Go to step 3 unless the maximum number of iterations is reached;
9- End.
```

Fig.12: Pseudo code for Intelligent Water Drops algorithm

6.4 Biogeography Based Optimization (BBO): [15]

BBO is a population-based evolutionary algorithm that is based on the mathematics of biogeography. In BBO, problem solutions are analogous to islands, and the sharing of features between solutions is analogous to the migration of species.

```
Select $H_i$ according to immigration rate $\lambda_i$
if $H_i$ is selected
  for $j = 1$ to $N$
    Select $H_j$ according to emigration rate $\mu_i$
    if $H_j$ is selected then
      Replace SIV in $H_i$ with SIV in $H_j$
    end if
  end for
end if
```

```
for $i = 1$ to $N$ do
   Use $\lambda_i$ and $\mu_i$ to compute the probability $P_i$
   and mutation rate $m_i$
   for $SIV = 1$ to $D$ do
      Generate a random value $R$
      if $R < m_i$ then
         Generate a feasible value $F$ in searching space
         $H_i(\text{SIV}) = F$
      end if
   end for
end for
```

Fig.13: Pseudo code for Biogeography Based Optimization algorithm

7. CONCLUSION

Resource estimation plays a crucial role when scheduling workflows in the cloud. An accurate resource estimation mechanism will reduce the cost and time of workflow execution. When the structure of the incoming workflow and the number of tasks and levels in the workflows are taken into consideration, an effective resource provisioning methodology can be made. Like these factors, Virtual Machine selection from the resource pool and their placement strategies made a drastic difference in the performance of the workflow execution. Several meta-heuristic approaches are reviewed in this paper. These approaches with implemented in hybrid can result in the better performance. These considerations are to be implemented in my future work.

8. REFERENCES

[1] P.Mell and T.Grance, "**The NIST Definition of Cloud Computing**," v.15, http://csrc.nist.gov/groups/SNS/cloud-computing.

[2] J.D.Ullman, "**NP-complete scheduling problems**", Journal of Computer and System Sciences, vol. 10, no. 3, pp. 384-393, 1975.

[3] E.Deelman, K.Vahi, G.Juve, M.Rynge, S.Callaghan, P.J.Maechling, R.Mayani, W.Chen, R.Ferreira da Silva, M.Livny, and K.Wenger, "**Pegasus: a Workflow Management System for Science Automation**", Future Generation Computer Systems, vol. 46, pp. 17 – 35, 2015.

[4] Rodrigo N.Calheiros, Rajiv Ranjan, Anton Beloglazov, Cesar A.F.De Rose and RajkumarBuyya, "**CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms**", Software – Practice and Experience, 2011; 41:23 – 50.

[5] Weiwei Chen, EwaDeelman, "**WorkflowSim: A Toolkit for simulating Scientific Workflows in Distributed Environments**", IEEE International Conference; 2012: 1-8.

[6] ZhichengCai, Qianmu Li and Xiaoping Li, "**ElasticSim: A Toolkit for simulating Workflows with Cloud Resource Runtime Auto-Scaling and Stochastic Task Execution Times**", Journal of Grid Computing (2017) 15: 257 – 272.

[7] ChenhaoQu, Rodrigo N.Calheiros and RajkumarBuyya, "**Auto-scaling web applications in clouds: A Taxonomy and Survey**", ACM Computing Surveys, 2016.

[8] K.Kanagaraj and S.Swamynathan, "**Structure aware resource estimation for effective scheduling and execution of data intensive workflows in cloud**", Future Generation Computer Systems (2017), http://dx.doi.org/10.1016/j.future.2017.09.001

[9] FengyuGuo, Long Yu, ShengweiTian and Jiong Yu, "**A workflow task scheduling algorithm based on the resources' fuzzy clustering in cloud computing environment**", Int. J. Commun. Syst. (2014), Wiley Online Library. DOI: 10.1002/dac.2743

[10] Xiaolong Liu, Shyan-Ming Yuan, Guo-HengLuo, Hao-Yu Huang and Paolo Bellavista, "**Cloud Resource Management with Turnaround Time Driven Auto-Scaling**", IEEE Access, vol 5, 2017.

[11] Thanh-Phuong Pham, Juan J.Durillo, and Thomas Fahringer, "**Predicting Workflow Task Execution Time in the Cloud using a Two-Stage machine learning approach**", IEEE Transactions on Cloud Computing. DOI 10.1109/TCC.2017.2732344

[12] Kennedy J, Eberhart R, "**Particle Swarm optimization**", Proc int conf neural networks, vol. 4. IEE; 1995. 1942 – 8.

[13] Dorigo M, Stutzle T, "**Ant colony optimization**", MIT press, 2004.

[14] Hamed Shah-Hosseini, "**Problem Solving by Intelligent Water Drops**", IEEE Congress on Evolutionary Computation. Swissotel, The Stamford, 2007.

[15] Haiping Ma, Simon D, "B**iogeography-based optimization: A 10-year Review**", IEEE Transactions on Emerging Topics in Computational Intelligence, Vol.1, No.5, October 2017.