Comparison of Resource Optimization Algorithms in Cloud Computing

R. Yamini and M. Germanus Alex

1Research and Development Center, Bharathiar University, Coimbatore.

Department of Computer Science Engineering, SRM University, Kattankulathur, Tamil Nadu, India.

rs_yamini@yahoo.com, yamini.r@ktr.srmuniv.ac.in

2Department of Computer Science, Kamarajar Govt. Arts & Science College, Tirunelveli, Tamil Nadu, India.

mgalxus@yahoo.com

Abstract

Today cloud computing plays a vital role in information and communication technology. We can hire the required infrastructure, platform and applications from private and public clouds. The efficiency of cloud computing relies on energy consumption, cost and utilization of resources. The proposed algorithm is optimized based on three aspects such as energy efficiency, cost reduction and maximum utilization of resources. This paper compares the existing technique with proposed technique using various metrics. Based on experimental results the proposed technique is the best method.
1. Introduction

In order to provide the best services to the cloud users, various resource optimization algorithms are used in cloud computing\(^1\) to optimize the resources\(^3\). In this paper existing four algorithms were analyzed and compared with the proposed Tri-objective resource optimization algorithm. Genetic Algorithm(GA), Ant colony optimization Algorithm(ACO), Particle Swarm Optimization Algorithm(PSO) and Bacterial Foraging Optimization(BFO) Algorithm were analyzed in this paper.

**Genetic Algorithm(GA)**

This is based on evolutionary model for complex systems. This algorithm is a global, efficient and parallel searching technique. It is like a natural evolution method. Therefore, GA is well suitable for solving the resource optimization problem. Multiple point searching is an enhanced feature of genetic algorithm. The search space is reduced by using GA techniques in resource optimization. In this approach, the candidates solutions are represented in terms of chromosomes which are mended in each stage. GA uses the selection methodology to find the survival of fitness function. For each and every stage the highest fitness value is found and those values are inherited to next generation.

**Ant Colony Optimization (ACO) Algorithm**

It is a randomized probabilistic technique, and used to find an optimal route which is similar to the ants to find the route for their food. Generally ants produce a pheromones when they find a route. This route is attracted by more ants that directs to find the optimal route. In cloud computing environment a cloud service provider provides different cloud resources on different virtual machines. Here ACO plays a vital role for optimization of cloud resources. Virtual machines are considered as nodes, ants are assumed as agents and resources are similar to the food. Ant travels among the nodes and assign the requested jobs to the cloud resources.

**Particle Swarm Optimization(PSO) Algorithm**

It is a computational method which belongs to the super set of swarm intelligence. In this algorithm is self adaptive\(^9\) iterative procedure is used to optimize a problem. The candidate solutions are iteratively processed with respect to a given measure of quality. In PSO, population of candidate solutions is referred to as ‘particles’. The optimization in PSO involves movement of these particles around in the search-space. The movement is guided by simple mathematical formulae over the particle's velocity and position. Each particle’s local best known position influences its movement. In addition to this it is also guided toward the best known positions in the search-space, which are updated as better positions are found by other particles. The swarm is supposed to converge towards the optimal solution by these particle movements. Like Ant colony algorithm, Particle swarm algorithm is also a Metaheuristic algorithm. A
simple form of algorithm works by starting a population of the candidate solutions (the Swarm). Each candidate solution is analogous to a particle in the system. The system is supposed to be the solution space. These particles are then moved around in the search-space on basis of some mathematical formulae with the aim to converge to an optimum solution.

**Bacterial Foraging Optimization (BFO) Algorithm**

Bacterial foraging algorithm is based on hyper-heuristics method. In this algorithm, partial result is denoted by bacterium and the motion of the bacterium as heuristics. The optimization in bacterial foraging algorithm lies on process like chemo taxis, swarming, reproduction, elimination and dispersal. The aim of the algorithm is how the animal finds the food is similar to that the power consumption per time (P/T) is increased.

Chemo taxis: This method is based on the motion of the Escherichia Coli bacteria through drowning and breaking down via flagella. If the bacterium is going in the similar way for a time period. The movement of the bacteria towards the foods but it can either be pulled or declined. The bacterium will attract the different kinds of bacteria throughout the times.

Reproduction: The second step is the reproduction. Here the physical fitness value of the bacterium is computed and it will be arranged in sorted order. The healthier bacterial will survive longer and breaks in to further two.

Evacuation and dispersal: To avoid local optima the evacuation and dispersal step is performed.

2. **Proposed Tri-Objective Resource Optimization Algorithm**

This algorithm uses the cost function of Energy Conscious Task Consolidation (x value) and Maximum Utilization (y value). The outcomes are combined to construct a point P(x,y) in search space. This algorithm builds, for each resource, the corresponding point in the solution set E and checks the following:

- If the newly obtained result dominates the current result, then it updates the best result and reset the equivalent solution in Subset A.
- If two results are equal, then it updates A with the newly obtained result if suitable.
- Parameter ρ is the sum of two objectives of Energy conscious Task consolidation and Maximum Utilization model.

To find the best result, the output of the two cost functions must be normalized. For a given task $j_i$, the utilization of a chosen resource $s_i$ is based on the speed and the time needed to complete the task. The maximum value ($a_{\text{max}}$) returned by Energy Conscious Task Consolidation may varies from 0 to $a_{\text{max}}$. $a_{\text{min}}$ and $a_{\text{max}}$ as the lower and upper bound of the Energy Conscious Task Consolidation...
unit range, the normalized metric on the unit scale \((b_{\text{min}} \text{ to } b_{\text{max}})\) is given by 
\[
\text{norm}_x = \frac{(f_{ij} - a_{\text{min}})(b_{\text{max}} - b_{\text{min}})}{(a_{\text{max}} - a_{\text{min}})}.
\]
Here \(a_{\text{min}}\) is set to 0. The normalized unit range \(b_{\text{min}}\) is set to 1 and \(b_{\text{max}}\) is set to 100. Hence \(f_x\) is the normalized cost function of Energy Conscious Task Consolidation \(f_x = b_{\text{max}} - \text{norm}_x\) value returned by the Maximum Utilization then the co-ordinate point \(P_i\) for the selected resource \(s_i\) is \(P_i = (f_x, f_y)\) and distance \(d = |f_x - f_y|\).

Normalized complement of distance \(d\) is denoted by \(\rho_i = b_{\text{max}} - d\). If the value of \(d\) is less then we will get the best solution. Hence the value of \(d\) determined the best solution. This algorithm identifies the best result in the result set \(E\) and its subset \(A\). Thus this algorithm finds and assigns the most power effective resource\(^{11,12}\) for the requested task.

Algorithm 1 Tri-Objective Resource Optimization

Input:
- \(j_i\) is the set of tasks that belongs to the set \(J\).
  \[j_i \in J = \{ j_0, j_1, j_2, \ldots, j_n \}\]
- \(S\) is the set of resources that forms the set \(S = \{ s_0, s_1, s_2, \ldots, s_m \}\)

Output:
- Select the green resource within a set of Resources. \(s^* \in S\)

start
  \(s^*, \text{ best and } A = \emptyset\)
  for \(s \in S\) do
    \(x = f_x, \ y = f_y, \ \rho = (b_{\text{max}} - |x - y|), \ solution = (x, y)\)
    if \(\text{solution} > \text{best}\) then
      \(best = \text{solution}\)
      \(s^* = S\)
      \(A = A \cup (\text{solution}, s, \rho)\)
    endif
    ifequals(solution, best) then
      \(A = A \cup (\text{solution}, s, \rho)\)
    endif
  endfor
  for \((f, s, \rho) \in A\) do
    if \((f_x \neq \text{best}_x) \text{ and } (f_y \neq \text{best}_y)\) then
      if \(\rho > \rho_{\text{best}}\) then
        \(best = f\)
        \(\rho_{\text{best}} = \rho\)
        \(s^* = S\)
      else
        if \((f_x + f_y) > (\text{best}_x + \text{best}_y)\) then
          \(best = f\)
          \(\rho_{\text{best}} = \rho\)
          \(s^* = s\)
        endif
      endif
    endif
  endfor
stop
TROA consists the following features which are not present in other algorithms. The main objectives are:
- Maximum Utilization of Resources with Minimum Power Consumption
- Minimum cost
- Less power consumption leads to less carbon-di-oxide emission

3. Comparison of GA, ACO, PSO, BFO and TROA

In this section, the features of Genetic Algorithm (GA), Ant Colony Optimization, Particle Swarm Optimization (PSO), Bacterial Foraging Optimization (BFO) were compared with the proposed Tri-Objective Resource Optimization Algorithm.

<table>
<thead>
<tr>
<th>Features</th>
<th>GA</th>
<th>ACO</th>
<th>PSO</th>
<th>BFO</th>
<th>TROA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class</td>
<td>Evolutionary Algorithm</td>
<td>Swarm Intelligence</td>
<td>Swarm Intelligence</td>
<td>Bacterium Intelligence</td>
<td>Searching Algorithm</td>
</tr>
<tr>
<td>Aspiration</td>
<td>Aspired by the class of inheritance</td>
<td>Aspired by the behavior of ant to find the food</td>
<td>Aspired by the motion of birds or fishes</td>
<td>Aspired by the motion of the bacterium as heuristics</td>
<td>Aspired by the behavior of newly derived solution dominates the current solution</td>
</tr>
<tr>
<td>Initialization</td>
<td>population of strings encoded in the candidate solution</td>
<td>‘Ants’ represents the state of the problem</td>
<td>‘Particles’ represents the state of the problem</td>
<td>Bacteria represents the state of the problem</td>
<td>constructs the new solution corresponding to the evaluation space</td>
</tr>
<tr>
<td>Basic concept</td>
<td>Natural laws of evolution</td>
<td>Collective behavior of self organized</td>
<td>Collective behavior of self-organized or decentralized system which may be natural or artificial</td>
<td>Collective behavior of self organized</td>
<td>Searching and incremental behavior to find the optimal solution</td>
</tr>
<tr>
<td>Technique</td>
<td>AI technique where search heuristics are applied</td>
<td>Probabilistic technique to address computational tasks by searching shortest route using graphs</td>
<td>Computational iterative method to optimize the solution</td>
<td>Chemo taxis, swarming, reproduction, elimination and dispersal</td>
<td>Energy Conscious Task Consolidation and Maximum Utilization techniques</td>
</tr>
<tr>
<td>Rationale</td>
<td>Survival of the fittest</td>
<td>Pheromone from ants</td>
<td>particle moves in the search space with changing speed</td>
<td>Power consumption per time (P/T) is increased</td>
<td>Maximum Utilization of Resources with Minimum power consumption, Low cost, less CO2 emission</td>
</tr>
<tr>
<td>Method</td>
<td>selected chromosomes experiences mutation. Fittest carried over to next generation.</td>
<td>Pheromone contain the path.</td>
<td>Particle is moved in the search space and measured by certain parameters</td>
<td>Hyper-heuristics method</td>
<td>Heuristics method</td>
</tr>
</tbody>
</table>
Also the experiments were carried out using the parameters deadline, makespan, and resource utilization. Resource utilization is computed by finding the average CPU and memory utilization using the equation (1) and (2).

\[
\text{CPU Utilization} = \sum_{i=1}^{n} \left( \frac{C_i}{\text{total CPU}} \right) * 100
\]

\[
\text{Memory Utilization} = \sum_{i=1}^{n} \left( \frac{M_i}{\text{total MEMORY}} \right) * 100
\]

where \( n \) is the number of tasks. \( \text{totalCPU} \) and \( \text{totalMEMORY} \) are the total number of CPU and memory. The experiments used the parameters makespan, cost of user, which was different because of different scheduling methods and resources; deadline rate, which is the feedback effect of QoS because a perfect stable system requires feed- back to verify its performance; the fourth indicator is resource utilization.

### Makespan

Makespan is the total time of all tasks and is used to evaluate performance\(^5\). First, the experiment obtained the makespans of four algorithms. Fig. 1 shows that ACO is the best makespan of other 3 methods. This is because the main optimization goal of PBACO is to minimize the makespan. 200-1200 tasks were submitted 10 times repeatedly. The results output the execution time of each task, after which point makespan was calculated to get the mean value. There arrival rates of 50 per second.

![](image1.png)

Fig. 1: Make Span with Arrival Rate of 50

### Cost

Fig. 2 shows the cost comparison graph of GA, ACO, PSO and TROA. Other methods considered the deadline while TROA considered cost constraint\(^6,7\). When the total number of jobs is less there is no much difference with other methods. When the number of tasks are more that is about one lakh TROA method is the best one. It is reduced by around 40% of the total budget cost. Hence the proposed algorithm is cost effective one.
Resource Utilization

Five resources were selected and scheduled 100 times by the four methods. Each resources instance was used as the resource utilization for each scheduling. The results are shown in Fig. 3 shows the performance of resource utilization. TROA is the best technique of the four methods. TROA performance evaluation function evaluates and adjusts the quality of the solution in a timely manner and avoids falling into the local optimal solution.

Comparison based on Performance Indicators

Table 2 shows the comparison of 4 algorithms based on the performance characters such as power consumption, Resource Utilization, Cost, Makespan and Carbon-di-Oxide CO₂ emission.

<table>
<thead>
<tr>
<th>Characteristics</th>
<th>GA</th>
<th>ACO</th>
<th>PCO</th>
<th>TROA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Power Consumption</td>
<td>Maximum</td>
<td>Moderate</td>
<td>Moderate</td>
<td>low</td>
</tr>
<tr>
<td>Resource Utilization</td>
<td>Moderate</td>
<td>Moderate</td>
<td>Moderate</td>
<td>Maximum</td>
</tr>
<tr>
<td>Cost</td>
<td>Moderate</td>
<td>Moderate</td>
<td>High</td>
<td>Low</td>
</tr>
<tr>
<td>Makespan</td>
<td>Low</td>
<td>Best</td>
<td>Better</td>
<td>Better</td>
</tr>
<tr>
<td>CO₂ emission</td>
<td>Not Applicable</td>
<td>Moderate</td>
<td>Moderate</td>
<td>less</td>
</tr>
</tbody>
</table>

4. Conclusion

In this paper, we proved that the performance of the proposed Tri-objective Resource Optimization Algorithm (TROA) is the best algorithm based on the parameters of minimum or low power consumption of resources with maximum
utilization, minimum cost, better makespan and less CO$_2$ emission leads to the concept of environment sustainability. This paper analyzed various resource optimization algorithms and how they worked in cloud environment to attain the desired results. This analysis shows the unique features of each algorithm. Experiment results shows that the proposed Tri-objective Resource Optimization Algorithm is the best one to attain our primary three objectives.

References