CHAPTER 6

PARTICLE SWARM OPTIMIZATION ON WORKFLOW MANAGEMENT IN CLOUD

6.1 INTRODUCTION

Particle swarm optimization is a computational method that optimizes a problem by iteration to improve a workflow management or task scheduling with minimized task completion time (finishing time) in cloud environment. PSO optimizes a problem with the help of workflow models or particles which move around in the search-space in accordance with simple formulae to achieve the above objective.

Each model’s (particles) movement is influenced by its local best known workflow, but it is also guided towards the best known positions in the search-space which is updated as better positions founded by other particles. This is expected to move the swarm towards the best solutions with an indication as described by Zhang et al (2008). In terms of problems, the complex applications can be divided into two classes namely computing demanding and data demanding. As far as the data demanding application, the scheduling strategy should decrease the data movement which means decrease the transferring time. But the computing demanding tasks, the scheduling strategy will schedule the data to the high performance computer.

In case of cloud computing, scientific workflow will gain a more utilizations. Even though, facing a lot of new challenges, such as resource and task scheduling are one. This work, focuses on minimizing the total finishing cost, total completion time and transferring time. In order to reduce the completion time, it is needed to schedule the computing demand tasks to the
high performance computer. As the tasks scheduling is a NP-complete problem, some heuristic algorithms have been used to resolve this kind of problems. Thus, to achieve this kind of task scheduling called particle swarm optimization method is used as described by Kennedy & Eberhard (1995). The main contributions are as follows, to formulate a model for task scheduling in cloud computing is to minimize the overall completion time and transmitting time and to design a PSO algorithm to solve task scheduling based on the proposed model, compare and analyze with other algorithm based on PSO.

6.2 TASK SCHEDULING PROBLEM FORMULATION

Denoting the task scheduling by a Task Interaction Graph (TIG) is for end. The TIG can be represented as G (V, E) as like CE (CH, CR), where V= {Ch1, Ch2, …, n} represents the tasks of an application and E= {Cij} indicates the information exchange between these tasks. The edge weight eij between node i and j denotes the information exchange between these pair of tasks. The node is defined for processor centers. The node weigh w corresponds to the work capacity of the node. Figure 6.1 shows an example of the task scheduling in a heterogeneous environment.

![Figure 6.1 A TIG example on heterogeneous system](image)

Figure 6.1 A TIG example on heterogeneous system
Consider the task scheduling with the following states. The processors in the cloud computing are heterogeneous and they have different processing ability which depend on their amount units of memory and performance of CPU capacity. A task’s processing cost will be varying according to the task being assigned to different processors. On the other hand, the communication cost between two cloudlets will be changing because between two different cloudlet shave different bandwidth and changing over time. Aim of this work is how to minimize the communication time and finishing cost. In order to formulate the task scheduling, consider \( J_x = \{1, 2, 3, \ldots, n\} \) as \( n \) independent tasks permutation and \( P_y = \{1, 2, 3, \ldots, m\} \) as \( m \) computing resources and \( B_{xy}, x, y = \{1, 2, 3, \ldots, z\} \) as the bandwidth between two cloudlets and \( z \) is the number of node.

\[
\sum_{z=1}^{m} a_{xz} = 1, \forall x = 1, 2, \ldots, n \tag{6.1}
\]

\( a_{xz} = 1 \) if task \( x \) is assigned to processor \( z \), and \( a_{xz} = 0 \), otherwise; \( b_{xyzu} = 1 \) if \( z \neq u \) and task \( x \) is assigned to processor \( z \) and task \( y \) is assigned to processor \( u \), and \( b_{xyzu} = 0 \) otherwise; \( n \) is the number of tasks; \( m \) is the number of processors.

\[
\sum_{z=1}^{m} \sum_{u=1}^{m} b_{xyzu} = 1, \quad \forall x, y = 1, 2, \ldots, n, z \neq u \tag{6.2}
\]

\[
a_{xz}, b_{xyzu} \in \{0, 1\}, \forall x, y, z, u \tag{6.3}
\]

\( D E_{xz} \) is the amount of data that the \( x \) task assigning to the processor \( z \) and \( P_m \) and \( P_c \) are the processor's memory and CPU's capacity. \( D T_{xy} \) is the interchange data amount between task \( x \) and task \( y \).

\[
C_{fin}(M) = \sum_{x=1}^{n} \sum_{z=1}^{m} a_{xz} * \frac{D E_{xz}}{P_m * P_c} \tag{6.4}
\]
\[ C_{\text{int}}(M) = \sum_{x=1}^{n-1} \sum_{j=t+1}^{n} b_{x,y,z,u} \cdot \frac{DT_{x,z}}{B_{x,y}} \]  

(6.5)

Equation \( C_{\text{fin}}(M) \) and \( C_{\text{int}}(M) \) respectively represent the finishing cost and the transforming time. Suppose the processing time is known for task \( x \) executing on processor \( y \) and the communication time is known for transmitting the data from \( x \) node to \( y \) node. Also map all the tasks to the processors make the total time and cost reduced, which in turn the \( Total(M) \) as per equation (6.6) value is minimized.

\[ Total(M) = C_{\text{fin}}(M) + C_{t}(M) \]  

(6.6)

### 6.3 IMPLEMENTATION OF PSO

PSO provides an optimized solution in task scheduling process. In PSO, each single solution is a model or workflow, identically termed as "particle", in the search space generated for cloud environment. All particles contain fitness values (i.e. completion time) which are evaluated by fitness function and velocities (i.e. resource utilization using reservation cluster) which direct the flying of the particles. The particles pass through the problem space by following the current optimum particles as the best solution stated by Schute & Groenwold (2005).

PSO is initialized with a group of random particles (solutions) and then it searches for optima by updating generations. Each particle is updated by following two "best" values in every iteration. The first one is the best solution (fitness of obtaining minimum completion time) which has been achieved so far. This fitness value is called “pbest”. Another "best" value named global best “gbest” is tracked by the particle swarm optimizer by any particle in the population. The best value called local best “lbest” is obtained when a particle takes part in the population of its topological neighbors. In each generation the
velocity and the position of particle will be update in light of $v_i^{k+1}$ and $x_i^{k+1}$ based on equations (6.7) and (6.8) respectively.

$$v_i^{k+1} = \omega v_i^k + c_1 rand_1 * (pbest_i - x_i^k) + c_2 rand_2 * (gbest - x_i^k)$$ (6.7)

$$x_i^{k+1} = x_i^k + v_i^k$$ (6.8)

Their parameters are listed below,

- $v_i^k$ - Velocity of particle i at iteration k
- $v_i^{k+1}$ - Velocity of particle i at iteration k+1
- $x_i^k$ - Position of particle i at iteration k
- $x_i^{k+1}$ - Position of particle i at iteration k+1
- $\omega$ - Inertia weight
- $c_1, c_2$ - Acceleration coefficients
- $rand_1, rand_2$ - Random number between 0 and 1
- $pbest_i$ - Best position of particle i
- $gbest$ - Best position of entire particles in a population

### 6.3.1 Particle representation on PSO

To resolve the task scheduling, so map each underlying solution to a particle. Every particle as n dimensions vector response to the n tasks. An element delegates a task and the element is an integer value between 1 and n. The particle represents one of the task scheduling.
Illustrative example for the task assignment to PSO particle mapping is shown below,

<table>
<thead>
<tr>
<th>Task 1</th>
<th>Task 2</th>
<th>Task 3</th>
<th>Task 4</th>
<th>Task 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Processor 3</td>
<td>Processor 1</td>
<td>Processor 2</td>
<td>Processor 1</td>
<td>Processor 3</td>
</tr>
</tbody>
</table>

6.3.2 Initial Swarm generation

The initial particle population is constructed randomly for PSO algorithm with idea of Lizheng et al (2012). The initial position and velocity of each particle is produce according to the following formula,

\[ x_i^1 = x_{\min} + (x_{\max} - x_{\min}) \cdot rand \] (6.9)

\[ v_i^1 = v_{\min} + (v_{\max} - v_{\min}) \cdot rand \] (6.10)

Where \( x_{\max} = v_{\max} = 4.0 \ x_{\min} = v_{\min} = -0.4 \) and rand is a random value between 0 and 1. As the velocity is a continuous value and the task scheduling is a discrete permutation in PSO algorithm, transform the continuous value to discrete permutation. The Small Position Value (SPV) rule \( v_i^1 \) which borrowed from the random key representation to solve the task assignment can convert the continuous value to discrete permutation. Use the SPV rule transform a continuous position vector, \( x_k^i = [x_1^i, x_2^i, ..., x_n^i] \) to a dispersed value permutation vector, \( s_k^i = [s_1^i, s_2^i, ..., s_n^i] \). In order to counting the processing time, map each element of the vector \( s_k^i \) into processor’s vector, \( p_k^i = [p_1^i, p_2^i, ..., p_n^i] \). The converting operation is defined as following equation:

\[ p_i^k = s_i^k \ mod \ m + 1 \] (6.11)
6.3.3 PSO Algorithm for task scheduling

Algorithm begins with k random particle vector and each particle is of n dimensions. Every particle vector is a candidate solution of the underlying problem. The particles are the task that are to be assigned and the dimensions of the particle are the number of the special tasks in a workflow. Then, each particle moves by the direction on the pbest and gbest until the maximal number of iterations is reached like flow represented in Figure 6.2. When the algorithm execution is over, the gbest and fitness value are the corresponding task scheduling and the minimal cost of the optimal strategy.

**Figure 6.2 Flowchart of PSO based Task Scheduling**
**Algorithm**

Step 1: Initialize particle position vector and velocity vector randomly according to $x_i^1$ and $v_i^1$. The vector’s dimension equal to the size of the special tasks.

Step 2: Convert the continuous position vector $(x_k^i = [x_1^i, x_2^i, ..., x_n^i])$ to discrete vector $(s_k^i = [s_1^i, s_2^i, ..., s_n^i])$ in light of SPV rule. Then, transforms to processor’s vector $(p_k^i = [p_1^i, p_2^i, ..., p_n^i])$ according to $p_k^i$. Last, calculating each particle’s fitness value as $Total(M)$.

Step 3: If one particle’s fitness value is better than current, setting current value replace previous pbest and as the new pbest.

Step 4: Selecting the best particle from all the particle as the gbest.

Step 5: For all particles update their position and velocity by $v_i^{k+1}$ and $x_i^{k+1}$.

Step 6: If reaching to the maximum iteration or getting the ideal result stops, otherwise repeating from Step 2.

**6.4 MAPPING SCHEDULE TOWARDS RESERVATION CLUSTER**

Each schedule’s log collects the cost and completion time. Based on the log, pbest estimated based on combination of minimum starvation time and completion time along with maximizing resource usage. Working mechanism of PSO with reservation cluster is shown in Figure 6.3.
Figure 6.3 Overview of PSO with Reservation Cluster Architecture

Each schedule planning’s, pbest is collectively to provide gbest solution with help of reservation cluster. Overview of PSO working along with two constraints of cost and time are represented in Figure 6.4.

Figure 6.4 Overview of PSO working with Reservation Cluster
6.5 RESULTS EVALUATION

6.5.1 Simulation Setup

Dual criteria scheduling with reservation cluster and PSO scheduling with reservation cluster in cloud environment were simulated using Cloudsim. Datacenter is usually composed of a set of hosts, each of which represents a physical computing node in the cloud. In simulation, 30 cloudlets with multiple tasks are created with heterogeneous configuration picked. To model cloud users, create application tasks that contain information related to execution details such as task processing requirements, disk I/O operations and the size of input files. 16 users are detailed in a cloud system, and each user with an exponentially distributed number of tasks.

Each task information consists of request time, starting time, bandwidth and other resources requirements. For the simulation hundred cloudlets are considered and the maximum size of the cluster is limited to N/5, where N is the total number of cloudlets used in the cloud system. In this case the reservation cluster size and reservation cluster of size twenty cloudlets. If the unscheduled tasks are more than the limit, dynamically the size or number of reservation clusters will be incremented. In this simulation, it is assumed to have a single reservation cluster. Implementation is carried out on CloudSim, because the rich set of simulation facilities empowers us to implement and evaluate the reservation cluster approach for heterogeneous distributed computing environments.

6.5.2 Evaluation of Task Scheduling through FIFO, DCS and PSO

Analysis has been done using the CloudSim simulator, in which the maximum cloudlets used in the cloud system is 100 and the maximum cloudlets that a cluster can accommodate is the total number of cloudlets divide by five
which have common communication ratio. Trials are performed in seven different task lists with reservation cluster following FIFO, DCS and PSO as stated and comparative results are represented in graphical in the Figure 6.5 based on Table 6.1.

Table 6.1 Analysis on Tangible Latest Completion time of FIFO, DCS and PSO

<table>
<thead>
<tr>
<th>Task List</th>
<th>CCR</th>
<th>FIFO</th>
<th>DCS</th>
<th>PSO</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.1</td>
<td>424603</td>
<td>364600</td>
<td>356028</td>
</tr>
<tr>
<td>2</td>
<td>0.2</td>
<td>424595</td>
<td>364570</td>
<td>352565</td>
</tr>
<tr>
<td>3</td>
<td>0.3</td>
<td>424586</td>
<td>364581</td>
<td>354580</td>
</tr>
<tr>
<td>4</td>
<td>0.1</td>
<td>424562</td>
<td>364551</td>
<td>355979</td>
</tr>
<tr>
<td>5</td>
<td>0.1</td>
<td>424560</td>
<td>364553</td>
<td>352552</td>
</tr>
<tr>
<td>6</td>
<td>0.2</td>
<td>424567</td>
<td>364557</td>
<td>354555</td>
</tr>
<tr>
<td>7</td>
<td>0.8</td>
<td>424540</td>
<td>364545</td>
<td>355974</td>
</tr>
</tbody>
</table>

While comparing the FIFO Completion time is higher than dual criteria scheduling and PSO. Comparatively PSO provides an efficient and minimum completion time than about 4.76 % of DCS and 15.23 % of FIFO.

Figure 6.5 Tangible Latest Completion time of FIFO, DCS and PSO
6.5.3 Evaluation of Task Scheduling through FIFO, DCS and PSO with Reservation Cluster

Table 6.2 Analysis on Tangible Latest Completion time of FIFO, DCS and PSO with Reservation Cluster

<table>
<thead>
<tr>
<th>Task List</th>
<th>FIFO RC</th>
<th>DCS RC</th>
<th>PSO RC</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>395589</td>
<td>342165</td>
<td>336541</td>
</tr>
<tr>
<td>2</td>
<td>389212</td>
<td>336668</td>
<td>327979</td>
</tr>
<tr>
<td>3</td>
<td>386373</td>
<td>334248</td>
<td>327490</td>
</tr>
<tr>
<td>4</td>
<td>397560</td>
<td>343845</td>
<td>338180</td>
</tr>
<tr>
<td>5</td>
<td>392958</td>
<td>339897</td>
<td>331105</td>
</tr>
<tr>
<td>6</td>
<td>393285</td>
<td>340176</td>
<td>333254</td>
</tr>
<tr>
<td>7</td>
<td>393580</td>
<td>340439</td>
<td>334855</td>
</tr>
</tbody>
</table>

Analysis has been done using the CloudSim simulator, in which the maximum cloudlets used in the cloud system is 100 and the maximum cloudlets that a cluster can accommodate is the total number of cloudlets divide by five. Trials are performed in seven different task lists with reservation cluster following first in first out scheduling, dual criteria scheduling and particle swarm optimization with reservation cluster are comparative results are represented in graphical in the Figure 6.6 based on Table 6.2. While comparing the FIFO with reservation cluster completion time is higher than dual criteria scheduling with reservation cluster and PSO with reservation cluster. Comparatively PSO with reservation cluster provides an efficient and minimum completion time than about 6.78% of DCS with reservation cluster and 17.86% of FIFO with reservation cluster.
Figure 6.6 Tangible Latest Completion time of FIFO, DCS and PSO with Reservation Cluster

6.5.4 Evaluating of Resource Failure in Cloud

Table 6.3 Analysis on Resource Failure among FIFO, DCS and PSO with and without Reservation Cluster

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>20</td>
<td>8</td>
<td>7</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>40</td>
<td>12</td>
<td>11</td>
<td>8</td>
<td>9</td>
<td>9</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td>60</td>
<td>17</td>
<td>15</td>
<td>10</td>
<td>14</td>
<td>13</td>
<td>6</td>
</tr>
<tr>
<td>4</td>
<td>20</td>
<td>16</td>
<td>14</td>
<td>10</td>
<td>10</td>
<td>9</td>
<td>6</td>
</tr>
<tr>
<td>5</td>
<td>40</td>
<td>12</td>
<td>11</td>
<td>7</td>
<td>9</td>
<td>8</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>60</td>
<td>17</td>
<td>15</td>
<td>10</td>
<td>14</td>
<td>12</td>
<td>6</td>
</tr>
<tr>
<td>7</td>
<td>20</td>
<td>14</td>
<td>12</td>
<td>8</td>
<td>9</td>
<td>8</td>
<td>5</td>
</tr>
</tbody>
</table>
Through above Table 6.3, Figure 6.7 graphical representation of FIFO, DCS and PSO with communication-to-computation ratio maintaining normalized resource usage and reservation cluster resource utilization are considered for task’s resource failure collection on each. Through reservation cluster on DCS, FIFO and PSO nearly 9.75% of resource failures than FIFO, DCS and PSO scheduling are overcome due to reservation cluster.

![Graphical Representation of Resource Failure among FIFO, DCS and PSO with and without Reservation Cluster](image)

**Figure 6.7** Graphical Representation of Resource Failure among FIFO, DCS and PSO with and without Reservation Cluster
6.6 SUMMARY

The DCS-RC algorithm achieves the objective with dynamic rescheduling technique and PSO with a construction of state space through rigorous evaluation of task completion time to identify the optimized solution. Based on the results obtained from the evaluation study of DCS-RC and PSO-RC, it is identified that the resource usage and scheduling scheme significantly improves resource utilization with minimizing task completion time in DCS-RC than PSO-RC. The dynamic rescheduling strategy incorporated into DCS-RC helps in ensuring the quality of schedules against performance drawbacks of cloud resources.