CHAPTER 6

HYBRID PROPOSED PARTICLE SWARM OPTIMIZATION METHOD FOR SOLVING MULTI AREA UNIT COMMITMENT PROBLEM

6.1 INTRODUCTION

Particle Swarm Optimization (PSO) is a population based stochastic optimization technique developed by Eberhart (1995) and Kennedy, inspired by the social behavior of bird flocking or fish schooling. PSO shares many similarities with evolutionary computation techniques such as GA. The system is initialized with a population of random solutions and searches for optima by updating generations. However, unlike the GA, the PSO has no evolution operators, such as crossover and mutation. In PSO, the potential solutions called particles fly through the problem space by following the current optimum particles.

6.2 BACKGROUND OF PARTICLE SWARM OPTIMIZATION

The particle swarm optimization concept consists of, at each time step, changing the velocity of (accelerating) each particle toward its $p_{best}$ and $l_{best}$ locations (local version of PSO). Acceleration is weighted by a random term, with separate random numbers being generated for acceleration toward $p_{best}$ and $l_{best}$ locations.

Each particle keeps track of its coordinates in the problem space which are associated with the best solution (fitness) it has achieved so far.
(fitness value is also stored). This value is called $P_{\text{best}}$. Another "best" value that is tracked by the particle swarm optimizer is the best value, obtained so far by any particle in the neighbors of the particle. This location is called $l_{\text{best}}$. When a particle takes all the population as its topological neighbors, the best value is a global best and is called $g_{\text{best}}$. The PSO concept consists of at each step, changing the velocity of each particle towards its $P_{\text{best}}$ and $l_{\text{best}}$ locations. Acceleration is weighted by a random term, with separate random numbers being generated for acceleration towards $P_{\text{best}}$ and $l_{\text{best}}$ locations. In past several years, PSO has been successfully applied in many research and application areas. It is understood that PSO gets better results in a faster, cheaper way compared with other methods.

Another reason for PSO's attractive nature is that it has only few parameters to adjust. PSO could be used across a wide range of applications.

PSO optimizes an objective function by undertaking a population-based search. The population consists of potential solutions, named particles, which are a metaphor of birds in flocks. These particles are randomly initialized and freely fly across the multidimensional search space. During flight, each particle updates its own velocity and position based on the best experience of its own and the entire population. All particles have fitness values, which are evaluated by the fitness function to be optimized, and have velocities, which direct the flying of the particles. The particles fly through the problem space by following the current optimum particles. PSO is initialized with a group of random particles (solutions) and then searches for optima by updating generations. In every iterations, each particle is updated by following two "best" values. The first one is the best solution (fitness) it has achieved so far. This value is called $P_{\text{best}}$. Another "best" value that is tracked by the particle swarm optimizer is the best value, obtained so far by
any particle in the population. This best value is a global best and called $g_{\text{best}}$. After finding the two best values, the particle updates its velocity and positions according to Equations (6.2) and (6.3). The position of each particle is updated in each iteration. This is done by adding the velocity vector to the position vector, as shown in Equation (6.3).

$$V_{id} = V_{id} + C_1 \text{rand}( ) \left( P_{id} - X_{id} \right) + C_2 \text{rand}( ) \left( P_{gd} - X_{id} \right)$$ (6.1)

$$V_{id} = W V_{id} + C_1 \text{rand}( ) \left( P_{id} - X_{id} \right) + C_2 \text{rand}( ) \left( P_{gd} - X_{id} \right)$$ (6.2)

$$X_{id} = X_{id} + V_{id}$$ (6.3)

where

- $V_{id}$ is the velocity of the particle
- $X_{id}$ is the current particle solution
- $W$ is the inertia weight
- $P_{id}$ is $p_{\text{best}}$. (Particle best)
- $P_{gd}$ is $g_{\text{best}}$ (global best)

$\text{rand}( )$ is a random number between (0,1)

$C_1$ and $C_2$ are learning factors usually $C_1 = C_2 = 2$
6.2.1 Pseudocode for PSO

For each particle
Initialize particle
End
Do
For each particle
Calculate the fitness value
If the fitness value is better than the best fitness value (p\text{best}) in history, set the current value as the new p\text{best}
End
Choose the particle with the best fitness value of all the particles as the g\text{best}
For each particle
Calculate the particle velocity according to Equation (6.2)
Update the particle position according to Equation (6.3)
End

While the maximum iterations or minimum error criteria are not attained, the particles velocities on each dimension are clamped to a maximum velocity $V_{\text{max}}$. If the sum of accelerations causes the velocity on that dimension to exceed $V_{\text{max}}$, it is a parameter specified by the user, then the velocity on that dimension is limited to $V_{\text{max}}$. The detailed flow chart for PSO is shown in Figure 6.1.
Figure 6.1 Basic flowchart of the PSO method
6.2.2 Modification of Search Point in the PSO

Each particle tries to modify its position using the following information:

(a) The current position \((x, y)\)

(b) The current velocities \((V_x, V_y)\)

(c) The distance between the current positions, \(p_{\text{best}}\) and \(g_{\text{best}}\)

This modification can be represented by the concept of velocity. Using equation (6.2), a certain velocity, which gradually gets close to \(p_{\text{best}}\) and \(g_{\text{best}}\) can be calculated. The current position (searching point in the solution space) can be modified by equation (6.3). Figure 6.2 shows a concept of modification of a search point by PSO, where

![Figure 6.2 A concept of modification of a search point by PSO method](image)
$V_{\text{org}}$ is the original velocity of the particle

$S^k$ is the position of the particle at $k^{\text{th}}$ iteration

$S^{k+1}$ is the position of the particle at $(k+1)^{\text{th}}$ iteration

$V_{\text{mod}}$ is the modified velocity of the particle

### 6.2.3 PSO Parameters

From the above case, it is clear that there are two key steps when applying PSO to optimization problems namely the representation of the solution and the fitness function. One of the advantages of PSO is that PSO takes real numbers as particles. It is not like GA, which needs to change to binary encoding, or special genetic operators have to be used. For example, to find the solution for $f(x) = x_1^2 + x_2^2 + x_3^2$, the particle can be set as $(x_1, x_2, x_3)$, and fitness function is $f(x)$. Then use the standard procedure to find the optimum. The searching is a repeat process, and the stop criterion is that the maximum iteration number is reached or the minimum error condition is satisfied.

There are not many parameters need to be tuned in PSO. Here is a list of the parameters and their typical values. The role of the inertia weight $W$ is considered very important in PSO convergence behavior. The inertia weight is employed to control the impact of the previous history of velocities on the current velocity. In this way, the parameter $W$ regulates the trade-off between the global and local exploration abilities of the swarm. A large inertia weight facilitates global exploration (searching new areas), while a small one tends to facilitate local exploration, i.e., fine-tuning the current search area. A suitable value for the inertia weight $W$ usually provides
balance between global and local exploration abilities and consequently a reduction on the number of iterations required to locate the optimum solution.

The typical range is 20 - 40. Actually for most of the problems 10 particles are large enough to get good results. For some difficult or special problems, take 100 or 200 particles as well. It is also determined by the problem to be optimized. It determines the maximum change one particle can take during one iteration. Usually set the range of the particle as the $V_{\text{max}}$. For example, the particle $(x_1, x_2, x_3)$ $x_1$ belongs $[-10, 10]$, then $V_{\text{max}} = 20$. $C_1$ and $C_2$ usually equal to 2. However, other settings were also used in different papers. But usually $C_1$ equals to $C_2$ and ranges from 0 to 4. The maximum number of iterations is set to 2000. This stop condition depends on the problem to be optimized.

There are two versions of PSO namely global and local version. Global version is faster but might converge to local optimum for some problems. Local version is little-bit slower but not easy to be trapped into local optimum. Use global version to get quick result and use local version to refine the search.

The role of the inertia weight $W$ is considered critical for the PSO’s convergence behavior. The inertia weight is employed to control the impact of the previous history of velocities on the current velocity. In this way, the parameter $W$ regulates the trade-off between the global (wide-ranging) and local (nearby) exploration abilities of the swarm. A large inertia weight facilitates global exploration (searching new areas), while a small one tends to facilitate local exploration, i.e., fine-tuning the current search area. A suitable value for the inertia weight $W$ usually provides balance between global and local exploration abilities and consequently results in reduction of number of
iterations required to locate the optimum solution. Initially, the inertia weight was constant. However, experimental results indicated that it is better to initially set the inertia to a large value, in order to promote global exploration of the search space, and gradually decreases it to get more refined solutions. Thus, an initial value around 1.2 and a gradual decline towards 0.4 can be considered as a good choice for $W$. The parameters $C_1$ and $C_2$ are not critical for PSO’s convergence. However, proper fine-tuning may result in faster convergence and alleviation of local minima. As default values, $C_1 = C_2 = 2$ were proposed, but experimental results indicate that $C_1 = C_2 = 0.5$ might provide even better results. Recent work reports that it might be even better to choose a larger cognitive parameter, $C_1$, than a social parameter, $C_2$, but with $C_1 + C_2 \leq 4$. The random numbers are used to maintain the diversity of the population, and they are uniformly distributed in the range $(0, 1)$.

6.2.4 **Comparison between the PSO and the GA**

The strength of the GA's is in the parallel nature of their search. A GA implements a powerful form of hill climbing that preserves multiple solutions, eradicates unpromising solutions and provides reasonable solutions. Through genetic operators, even weak solutions may continue to be part of the makeup of future candidate solutions. The genetic operators used are central to the success of the search. All GAs require some form of recombination as this allows the creation of new solutions that have by virtue of their parent’s success and a higher probability of exhibiting a good performance. In practice, crossover is the principal genetic operator whereas mutation is used much less frequently. Crossover attempts to preserve the beneficial aspects of candidate solutions and to eliminate the undesirable components while the random nature of mutation is probably more likely to
degrade a strong candidate solution than to improve it. Another source of the algorithm’s power is the implicit parallelism inherent in the evolutionary metaphor. By restricting the reproduction of weak candidates, the GA eliminates not only that solution but also all of its descendants. This tends to make the algorithm likely to converge towards high quality solutions within a few generations.

Most of the evolutionary techniques have the following procedure

1. Random generation of an initial population
2. Reckoning of a fitness value for each subject. It will directly depend on the distance to the optimum.
3. Reproduction of the population based on fitness values.
4. If the requirements are met, then stop. Otherwise go back to step 2.

From the procedure, one can learn that the PSO shares many common points with the GA. Both algorithms start with a randomly generated population; both have fitness values to evaluate the population. Both update the population and search for the optimum with random techniques. Both systems do not guarantee success. However, the PSO does not have genetic operators like crossover and mutation. Particles update themselves with the internal velocity. They also have memory which is important to the algorithm. Compared with Genetic Algorithms (GA's), the information sharing mechanism in the PSO is significantly different. In GA's, chromosomes share information with each other. So the whole population moves like one group towards an optimal area. In the PSO, only g_best (or l_best) gives out the information to others. It is a one-way information sharing mechanism. The
evolution only looks for the best solution. Compared with GA all the particles tend to converge to the best solution quickly even in the local version in most cases.

Particle Swarm Optimization shares many similarities with evolutionary computation techniques in general, and GA’s in particular. All three techniques begin with a randomly generated population and all utilize a fitness value to evaluate the population. They all update the population and search for the optimum with random techniques. A large inertia weight facilitates global exploration (search in new areas) while a small one tends to assist local exploration. The main difference between the PSO approach compared to the EC and the GA is that the PSO does not have genetic operators, such as crossover and mutation.

6.3 IMPLEMENTATION OF PROPOSED PARTICLE SWARM OPTIMIZATION METHOD FOR MAUC PROBLEM

A way of solving the multi-area unit commitment problem is to solve it in two phases. In the first phase, the MAUC problem is formulated taking into some of the constraints and unit commitment is done by simple priority list method. In the next phase, the economic dispatch problem is solved by using proposed particle swarm optimization method. The various steps of the algorithm are given below.

Step 1: Read the generating unit data and demand profile.

Step 2: Perform the simple priority list method to get the initial commitment schedule for each area.
Step 3: Initialization of particle. The initial particle of size $N_p$ is generated randomly for the committed unit in each area:

(a) Calculate the initial particle population

$$I_p = [(P_1^{kp}, \ldots, P_2^{kp}); 1, 2, 3, 4; p_1, \ldots, N_p] \quad (6.3)$$

(b) Calculate the fuel cost for each particle using equation (6.1)

$$FC_p = [(a(P_i^{kp})^2 + b(P_i^{kp}) + c); k_1, k_2, k_3, k_4; p_1, \ldots, N_p] \quad (6.4)$$

(c) Calculate the start up cost of each particle

(d) Calculate the production cost

$$\text{Production Cost} = FC_p + SC_p \quad (6.5)$$

$$F_p = FC_p + SC_p + k(\sum_{i=1}^{N_p} P_i^{kp} - D_i^{kp}) \quad (6.6)$$

(e) To calculate the $P_{best}$ by using the fitness function values, if the current value is better than the previous $P_{best}$, then set the $P_{best}$ value equal to the current value and compute the $g_{best}$; if the current value is better than the $g_{best}$ then reset the current particles.

Step 4: Updating the Velocity
The velocity is updated by considering the current velocity of the particles, the best fitness function value among the particles in the swarm using the following equation. (6.7)

\[ V_{i}^{p+1} = \omega V_{i}^{p} + C_{1} \text{rand}((P_{bi}^{kp} - P_{i}^{kp}) + C_{2} \text{rand}((P_{gi}^{kp} - P_{i}^{kp})) \]

(6.7)

where \( \omega \) is weight factor. The weight \( \omega \) is computed using equation.

\[ \omega = \omega_{\text{max}} - (\omega_{\text{max}} - \omega_{\text{min}}) / \text{iter}_{\text{max}} \times \text{iter} \]  

(6.8)

**Step 5:** Updating the particle position

The position of each particle is updated by adding the updated velocity with the current position of the individual in the swarm.

\[ P_{i}^{kp} = P_{i}^{kp} + V_{i}^{p+1} \]  

(6.9)

The steps described in sub sections 3 to 5 are repeated, until a criterion is met usually a sufficiently good fitness the maximum generation count is reached.

**Step 6:** Optimum generation schedule is obtained for four areas using the gbest particle. Check area generation with the local demand.

**Step 7:** Areas with lower fuel cost may export the excessive generation to areas with higher fuel cost (deficiency areas) with the tie line limit.
6.3.1 Applications of the PSO

The PSO has been successfully applied in many areas, such as function optimization, artificial neural network training, fuzzy system control and other areas, where the GA can be applied. The various application areas of Particle Swarm Optimization include:

- Power Systems operations and control
- NP-Hard combinatorial problems
- Job Scheduling problems
- Vehicle Routing Problems
- Mobile Networking
- Modeling optimized parameters
- Batch process scheduling
- Multi-objective optimization problems
- Image processing and Pattern recognition problems and so on.

Currently, several researchers are using particle swarm optimization and hence, the application area also increases tremendously.

6.3.2 Hybridization with Other Methods

Evolutionary algorithms can be combined with more traditional optimization techniques. This is as simple as the use of a conjugate-gradient minimization, used after primary search with an evolutionary algorithm. It may also involve simultaneous application of algorithms like the use of evolutionary search for the structure of a model, coupled with gradient search for parameter values. Further, evolutionary computation can be used to
optimize the performance of neural networks, fuzzy systems, production systems, wireless systems and other program structures.

6.4 RESULTS AND DISCUSSION

The proposed MAUC algorithm has been implemented in a C++ environment and tested extensively. The test results of a multi-area system are presented in this section.

The proposed algorithm has been tested with four areas system and each area has 26 thermal generating units (Ouyang and Shahidpour 1991). The total number of units tested is 104, and their characteristics are presented in Appendix 2. There are identical units in each area. The load demand profile forecasts used in all four areas are presented in Appendix 3. The test are used to implement the proposed algorithm includes import/export capability and tie line capacity constraints. The following simulation parameters for the PSO method is population: 10, maximum iteration count: 500 and penalty factor 10000, Learning Factor \((c_1, c_2)\): 2.

<table>
<thead>
<tr>
<th>Hours (24)</th>
<th>Area-1 Cost ($)</th>
<th>Area-2 Cost ($)</th>
<th>Area-3 Cost ($)</th>
<th>Area-4 Cost ($)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>24034.21</td>
<td>20810.99</td>
<td>21812.12</td>
<td>22324.34</td>
</tr>
<tr>
<td>2</td>
<td>25641.05</td>
<td>19454.65</td>
<td>22327.65</td>
<td>18477.64</td>
</tr>
<tr>
<td>3</td>
<td>20961.50</td>
<td>19454.65</td>
<td>20455.78</td>
<td>18477.64</td>
</tr>
<tr>
<td>4</td>
<td>25615.50</td>
<td>17842.50</td>
<td>22327.65</td>
<td>18477.64</td>
</tr>
<tr>
<td>5</td>
<td>23893.09</td>
<td>17842.50</td>
<td>22127.65</td>
<td>18477.64</td>
</tr>
<tr>
<td>6</td>
<td>22141.94</td>
<td>17842.50</td>
<td>22127.65</td>
<td>18477.64</td>
</tr>
<tr>
<td>7</td>
<td>26339.20</td>
<td>21459.75</td>
<td>22274.06</td>
<td>25008.26</td>
</tr>
<tr>
<td>Hours (24)</td>
<td>Area-1 Cost ($)</td>
<td>Area-2 Cost ($)</td>
<td>Area-3 Cost ($)</td>
<td>Area-4 Cost ($)</td>
</tr>
<tr>
<td>-----------</td>
<td>-----------------</td>
<td>-----------------</td>
<td>-----------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>8</td>
<td>25198.44</td>
<td>25192.87</td>
<td>23933.21</td>
<td>23783.41</td>
</tr>
<tr>
<td>9</td>
<td>27865.84</td>
<td>27207.13</td>
<td>30106.5</td>
<td>24051.31</td>
</tr>
<tr>
<td>10</td>
<td>26390.88</td>
<td>23509.72</td>
<td>24870.30</td>
<td>27220.87</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Table 6.1 (Continued)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hours (24)</td>
<td>Area-1 Cost ($)</td>
<td>Area-2 Cost ($)</td>
<td>Area-3 Cost ($)</td>
<td>Area-4 Cost ($)</td>
</tr>
<tr>
<td>-----------</td>
<td>-----------------</td>
<td>-----------------</td>
<td>-----------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>11</td>
<td>25580.89</td>
<td>24692.87</td>
<td>29137.09</td>
<td>28673.89</td>
</tr>
<tr>
<td>12</td>
<td>32548.66</td>
<td>25282.66</td>
<td>24720.30</td>
<td>25430.86</td>
</tr>
<tr>
<td>13</td>
<td>22099.98</td>
<td>27181.34</td>
<td>27308.04</td>
<td>31917.42</td>
</tr>
<tr>
<td>14</td>
<td>21658.43</td>
<td>29222.61</td>
<td>25251.49</td>
<td>29103.34</td>
</tr>
<tr>
<td>15</td>
<td>19926.92</td>
<td>29080.01</td>
<td>24733.96</td>
<td>25230.32</td>
</tr>
<tr>
<td>16</td>
<td>21466.17</td>
<td>28418.96</td>
<td>26949.82</td>
<td>25185.44</td>
</tr>
<tr>
<td>17</td>
<td>24326.90</td>
<td>30720.94</td>
<td>28288.96</td>
<td>23953.68</td>
</tr>
<tr>
<td>18</td>
<td>24722.61</td>
<td>32656.24</td>
<td>32441.13</td>
<td>26633.47</td>
</tr>
<tr>
<td>19</td>
<td>26663.76</td>
<td>25768.56</td>
<td>25204.04</td>
<td>22762.45</td>
</tr>
<tr>
<td>20</td>
<td>27133.17</td>
<td>30396.82</td>
<td>26021.39</td>
<td>23902.28</td>
</tr>
<tr>
<td>21</td>
<td>25863.62</td>
<td>29124.79</td>
<td>26535.25</td>
<td>23910.40</td>
</tr>
<tr>
<td>22</td>
<td>21694.57</td>
<td>22514.97</td>
<td>24085.61</td>
<td>23953.68</td>
</tr>
<tr>
<td>23</td>
<td>20076.92</td>
<td>21259.75</td>
<td>26688.59</td>
<td>24773.24</td>
</tr>
<tr>
<td>24</td>
<td>20523.60</td>
<td>19454.65</td>
<td>25900.55</td>
<td>22189.14</td>
</tr>
<tr>
<td>Total cost ($)</td>
<td>582367.90</td>
<td>586392.40</td>
<td>605628.80</td>
<td>572396.00</td>
</tr>
</tbody>
</table>
Table 6.2 Comparison result of DP, EP and PSO method

<table>
<thead>
<tr>
<th>Method</th>
<th>Area-1 Cost ($)</th>
<th>Area-2 Cost ($)</th>
<th>Area-3 Cost ($)</th>
<th>Area-4 Cost ($)</th>
<th>Total Cost ($)</th>
</tr>
</thead>
<tbody>
<tr>
<td>DP</td>
<td>590383.69</td>
<td>574583.80</td>
<td>560621.10</td>
<td>627072.10</td>
<td>2352660.80</td>
</tr>
<tr>
<td>EP</td>
<td>585740.10</td>
<td>588939.60</td>
<td>602163.30</td>
<td>572357.50</td>
<td>2349201.00</td>
</tr>
<tr>
<td>PSO</td>
<td>582367.90</td>
<td>586392.40</td>
<td>605628.80</td>
<td>572396.00</td>
<td>2346785.00</td>
</tr>
</tbody>
</table>

Table 6.2, presents the result obtained by the proposed particle swarm optimization method and it compared with DP and EP method. It can be observe that the operating cost of the four areas is reduced using PSO method. When compared to DP method would save 0.25 % and when it is compared to EP method would save 0.102%. Figure 6.3 shows the convergence characteristics for multi-area obtained using proposed methodology.

![Figure 6.3 Convergence characteristics of the PSO method](image)
6.5 SUMMARY

In this research, we proposed a more efficient approach based on PSO and it is an extremely simple algorithm that seems to be effective for optimizing a wide range of functions to the multi-area unit commitment problem. The proposed technique is used to improve the speed and reliability of the optimal search process. Instead of using iteration method in assigning power generation to each area, we used the Particle Swarm Optimization algorithm (PSO) to find the optimal power generation in each area and the entire system. Using the PSO algorithm in each area and the entire system, we can save time in performing the economic dispatch and operating cost.