CHAPTER 5

CLUSTERING BASED UNIT COMMITMENT
EMPLOYING COMBINED GENETIC ALGORITHM-
SIMULATED ANNEALING

5.1 Introduction

Unit commitment determines when to start and shut down units such that the load plus spinning reserve requirements within a specific period is satisfied at all times. Proper commitment of available generating units could result in fuel cost savings. Due to system operating requirements, capacity in excess of load must be committed. The amount of excess capacity or reserve is governed by predetermined spinning reserve requirements and some form of reliability measures. Because of the tremendous expense involved in unit commitment, the electrical utility must determine which generators are the most economical to operate and the combinations of units that should be committed to meet a given load demand. Problems associated with unit commitment have generally been difficult to solve because of the uncertainty of particular aspects of the problems. For instance, the availability of fuels, imprecise load forecasts, variable costs affected by the loading of generating units of different fuels and losses caused by reactive flows are some of the unpredictable issues. In order to reach a feasible solution to this economic problem, different constraints must be considered such as spinning reserves, thermal unit constraints, must run fuel units, fuel constraints and other operating constraints.

Fuel cost savings can be obtained by proper commitment of available generating units. This chapter describes a new approach to the unit commitment problem through classification of units into various clusters based on hybrid technique of genetic algorithm and simulated annealing. This classification is carried out in order to reduce the overall operating cost and to satisfy the minimum up/down constraints easily. Unit commitment problem is an important optimizing task in daily operational planning of power systems which can be mathematically formulated as a large scale nonlinear mixed-integer minimization problem. A new methodology employing the concept of cluster algorithm called as additive and divisive hierarchical clustering has been employed based on hybrid technique of genetic algorithm and
simulated annealing in order to carry out the technique of unit commitment. Proposed methodology involves two individual algorithms. While the load is increasing, additive cluster algorithm has been employed while divisive cluster algorithm is used when the load is decreasing. The proposed technique is tested on a 10 unit system and the simulation results show the performance of the proposed technique.

Inspired by the results of GA method and to overcome the general difficulties in GA approach, a hybrid technique comprising of both GA and simulated annealing has been proposed. This combined technique along with the application of cluster algorithms has been proposed in this thesis. The method employs Additive and Divisive Cluster Algorithms. The proposed methodology can be unfolded in to three stages. In the first stage, four clusters are formed namely base load, intermittent load, semi-peak load and peak load clusters. All the generating units of the plant are segregated into corresponding clusters based on operating costs. The operating costs are obtained by GA, based on the operating costs of units clusters are formed and also useful for preparing the priority list. In the second stage, UC solution is obtained by developing Additive Cluster algorithm for increasing load pattern. Finally in the third stage a Divisive Cluster algorithm is developed for decreasing load pattern.

5.2 Simulated Annealing

This method was independently described by Scott Kirkpatrick. Based on the annealing process in the statistical mechanics, the simulated annealing (SA) was introduced for solving complicated combinatorial optimization problems. SA is effective in network reconfiguration problems for large - scale systems and its searching capability becomes more significant as the system size increases. Moreover, the objective function with a smooth strategy enables the SA to escape more easily from local minima and to reach rapidly to the vicinity of an optimal solution.

Simulated annealing mimics the annealing process to solve the optimization problem. It uses a temperature parameter that controls the search. The temperature parameter typically starts off high and is slowly “cooled” or lowered in every iteration. At each iteration a new point is generated and its distance from the current point is proportional to the temperature. If the new point has a better function value, it replaces the current point and the iteration counter is incremented. It is possible to
accept and move forward with a worst point. The probability of doing so directly depends on the temperature. This unimuitive step some time helps to identify a new search region in hope of finding a better minimum.

5.2.1 Overview of Algorithm

This process of simulated annealing simulates the slow cooling of molten metal to achieve the minimum function value in a minimization problem. The cooling phenomenon is simulated by controlling the temperature-like parameter introduced with the concept of Boltzmann probability distribution. According to the Boltzmann probability distribution [6], a system in thermal equilibrium at a temperature $T$ has its energy distributed probabilistically according to Eqn. (4.1)

$$P(E) = \exp \left( -\frac{E}{kT} \right)$$ (5.1)

Where $k$ is the Boltzmann constant. It can be seen from Eqn.(4.1) that a system at a high temperature has almost uniform probability of being at any energy state but at low temperature it has a small probability of being at a high energy state. Therefore, by controlling the temperature $T$ and assuming that the search process follows the Boltzmann probability distribution, the convergence of the algorithm can be controlled. According to Metropolis [6], if at any instant the current point is $X_t$ and the function value at that point is $E(t) = f(x_t)$, then the probability of the next point being at $X_{t+1}$ depends on the difference in the function values at these two points i.e. $\Delta E = E(t+1) - E(t)$ and it is calculated by using the Boltzmann probability distribution given by Eqn. (4.2)

$$P(E(t+1)) = \min \left[ 1, \exp \left( \frac{-\Delta E}{kT} \right) \right]$$ (5.2)

If $\Delta E \leq 0$, this probability is one and the point $X_{t+1}$ is accepted. But if $\Delta E > 0$, it implies that $X_{t+1}$ is worse than that of $X_t$. In this case according to Metropolis algorithm, there is some finite probability of selecting the point $X_{t+1}$ even
though it is worse than $X_t$. This probability depends on relative magnitude of $AE$ and $T$. If the parameter $T$ is large, this probability is high for points with large function values. Thus any point is acceptable for large values of $T$. On the other hand, if the parameter $T$ is small, the probability of accepting an arbitrary point is small. Thus for small values of $T$, the points with only small deviation in function value are accepted.

5.2.2 Implementation of simulated Annealing method

**Step-1:** Initialize all initial variables i.e. gain of integral controller of both areas $k_i = (k_1, k_2) n, \varepsilon, T$ and set $t = 0$

**Step-2:** Calculate a neighborhood point $k_{i+1}$ which is usually random number.

**Step-3:** Using the $k_i$ and $k_{i+1}$ values simulate the block diagram and find the performance index ($f$) values.

**Step-4:** If $\Delta f = f(k_{i+1}) - f(k_i) < 0$. Then $t = t + 1$ go to **Step 5**. Otherwise

Create random number ‘r’ in the range (0, 1) if $r \leq \exp\left(-\frac{\Delta f}{T}\right)$

Then $t = t + 1$; go to **Step 5**

Otherwise, Go to **Step 2**

**Step-5:** If $|k_{i+1} - k_i| < \varepsilon$ and $\varepsilon$ and $T$ is very small then stop otherwise.

If $t$ mode $n = 0$ then lower the value of $T$ and go to **Step 2**

Otherwise go to **Step 2**

Fig. 5.1 shows the flowchart representation of concept of Simulated Annealing applied to optimize the fuel cost of unit commitment problem. Based on the conditions explained in the flowchart of Fig 5.1 the Simulated Annealing method is performed until the termination criterion is met and the optimal value of gain of integral controller in both areas is obtained. The values of $\varepsilon, T, n$ considered for this technique for obtaining the optimal value of gain of integral controller in both areas are given in Table 5.1.

**Table 5.1: Parameters of Simulated annealing method**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Termination parameter, $\varepsilon$</td>
<td>0.0001</td>
</tr>
<tr>
<td>Initial Temperature, $T(^\circ C)$</td>
<td>4000</td>
</tr>
<tr>
<td>Number of iterations, $n$</td>
<td>25</td>
</tr>
<tr>
<td>Temperature Reduction parameter ($^\circ C$)</td>
<td>20</td>
</tr>
</tbody>
</table>
Start

Initialize all initial variables $x_i = (x_1, x_2, ..., x_n)$ and set $t = 0$

Calculate a neighborhood point $x_{t+1}$ which is usually a random number

Using the $x_t$ and $x_{t+1}$ values, simulate the block diagram and find the performance index ($f$) values

Is $\Delta f = f(x_{t+1}) - f(x_t) < 0$?

Yes: $t = t + 1$

No: Create random number $r$ in the range $(0, 1)$

Is $r \leq \exp\left(-\frac{\Delta f}{T}\right)$?

Yes: $t = t + 1$

No: Lower the value of $T$

Is $|x_{t+1} - x_t| < \epsilon$? And $t$ very small

Yes: Stop

No: Is $(t \mod n) = 0$?

Yes: Stop

Fig 5.1: Flow chart of Simulated Annealing Method
5.3 Problem Formulation

Subject to the minimization of the cost-objective function in the unit commitment problem, certain units are stated to be as ‘ON’ and remaining as ‘OFF’. The following are the various notations considered during the implementation of the problem:

\( N \) : Number of generating units in the plant;
\( t \) : Scheduling period in hours (h);
\( i \) : Index of Unit \( i = 1,2, \ldots, N \);
\( t \) : Index of time \( t = 1,2, \ldots, T \);
\( I_i(t) \) : \( i \)th unit status at \( t \)th hour;
\( P_i(t) \) : Generation of \( i \)th unit at \( t \)th hour;
\( P_{r,m} \) : Minimum output power (MW) of \( i \)th unit;
\( P_{r,m} \) : Maximum output power (MW) of \( i \)th unit;
\( D(t) \) : Demanded power at \( t \)th hour;
\( R(t) \) : System reserve at \( t \)th hour;
\( T_{i,m} \) : Minimum up time of \( i \)th unit;
\( T_{i,off} \) : Minimum down time of \( i \)th unit;
\( x_i(t) \) : Duration during which \( i \)th unit is continuously ON;
\( X_i^{off} \) : Duration during which \( i \)th unit is continuously OFF;
\( SC_i(t) \) : Start-Up cost of \( i \)th unit;
\( FC_i(t) \) : Fuel cost of \( i \)th unit;
\( TC \) : Total Cost of generation;
\( HC(i) \) : Hot start cost of \( i \)th unit;
\( CC(i) \) : Cold start cost of \( i \)th unit;
\( CS(i) \) : Cold start hour of \( i \)th unit;
\( a_i, b_i, c_i \) : Fuel cost coefficients
Objective Functions

The objective function of UC problem is the minimization of the TC which has the components of FC and SC and is given by:

\[ \text{Min } (TC) = \sum_{t=0}^{T} \sum_{i=1}^{N} (FC_i(t) + SC_i(t)) \tag{5.3} \]

Where Fuel cost of \( i \)th unit: \( FC_i(t) = a_i + h_i P_i(t) + c_i P_i^2(t) \tag{5.4} \)

and Start-up cost

\[ SC_i(t) = H(i) : \text{if } T_i^{off} \leq X_i^{off}(t) \leq H_i^{off} \tag{5.5} \]

where \( H_i^{off} = H_i^{off} + CS(i) \tag{5.6} \)

System Constraints

The constraints, which must be considered during the optimization process of the UC problem (1), are given below.

Load Demand

All the committed units must generate total power equal to load demand as:

\[ D(t) = \sum_{i=1}^{N} P_i(t) \tag{5.7} \]

Spinning Reserve

To maintain system reliability for sudden variation of loads, system should have adequate amount of spinning reserve capacity. In this paper 10\% of the load demand is taken and which satisfies:

\[ \sum_{i=1}^{N} I_i(t) \cdot P_i^{max} \geq D(t) + R(t) \tag{5.8} \]
Generated Power Limits
The power output of each unit should satisfy:

\[ P_{i}^{\text{min}} \leq P_{i}(t) \leq P_{i}^{\text{max}} \]  \hspace{1cm} (5.9)

Minimum Up/Down Time

Once the unit is committed there is a minimum time before it is de-committed and viz. \( T_{i}^{\text{on}} \leq X_{i}^{\text{on}}(t) \) or \( T_{i}^{\text{off}} \leq X_{i}^{\text{off}}(t) \)  \hspace{1cm} (5.10)

5.4 Hybrid Genetic Algorithm-Simulated Annealing

Genetic algorithms are procedures based on the principles of natural selection and natural genetics that have proved to be very efficient in searching for approximations to global optima in large and complex spaces in relatively short time. The basic components of GA are:

- Representation of problem to be solved
- Genetic operators (selection, crossover, mutation)
- Fitness function
- Initialization procedure

GA starts by using the initialization procedure to generate the first population. The members of the population are usually strings of symbols (chromosomes) that represent possible solutions to the problem to be solved. Each of the members of the population for the given generation is evaluated and according to its fitness value, it is assigned a probability to be selected for reproduction. Using this probability distribution, the genetic operators select some of the individuals. By applying the operators to them, new individuals are obtained. The mating operator selects two members of the population and combines their respective chromosomes to create offspring.

The performance of GA can be improved by introducing more diversity among the strings so that pre-mature convergence can be eliminated. This can be achieved by replacing weaker strings i.e. the strings having low fitness value with better strings i.e. strings having higher fitness value. Simulated Annealing may be used for this purpose. This method follows the cooling process of molten metals through annealing. At high temperature, the atoms can move freely and these movements get restricted to shape gradually.
Initialize all initial variables $p$, $p_r$, $p_m$, $n_{\text{max}}$, $T$ and set $n=0$

Initialize random population of string of size $l$

$n=n+1$

Is $n > n_{\text{max}}$?

No

Create the clusters based on fitness value

Yes

Convert strings considered row wise into corresponding values of $p$, and calculate the operating cost which forms the objective

Perform the operations of GA which include selection, crossover and mutation and obtain new child strings.

Obtain the operating cost $f(x_i)$ of each string

Select a neighbourhood value $x_{i+1}$ and also calculate the operating cost of $f(x_{i+1})$

Is $\Delta f = f(x_{i+1}) - f(x_i) < 0$?

Yes

Copy the new string into population

No

Copy the old string into population

Create a random number between 0 and 1

Is $r \leq \exp\left(-\frac{\Delta f}{T}\right)$?

Yes

No

Fig 5.2 Flow chart for cluster based hybrid GA-SA to UC
The performance of GA can be improved by introducing more diversity among the strings so that pre-mature convergence can be eliminated. This can be achieved by replacing weaker strings i.e. the strings having low fitness value with better strings i.e. strings having higher fitness value. Simulated Annealing may be used for this purpose. This method follows the cooling process of molten metals through annealing. At high temperature, the atoms can move freely and these movements get restricted to shape gradually the performance of GA can be improved by introducing more diversity among the strings so that pre-mature convergence can be eliminated. This can be achieved by replacing weaker strings i.e. the strings having low fitness value with better strings i.e. strings having higher fitness value. Simulated Annealing may be used for this purpose. This method follows the cooling process of molten metals through annealing. At high temperature, the atoms can move freely and these movements get restricted to shape gradually a structure of crystal as the temperature is reduced slowly.

If the temperature is not reduced properly, the system may end up in a polycrystalline state, which may have higher energy state than the crystalline state. Based on this annealing process, this algorithm begins with an initial point and a high temperature $T$. A second point is created at random in the vicinity of the initial point and the difference in the function values $\Delta f$ at these two points is calculated and compared. If in case the second point has a smaller function value, the point is accepted; otherwise the acceptance of that point is measured by Boltzmann probability distribution.

In order to incorporate SA into GA, the strings resulted from GA after performing the mutation operation in every generation are modified by SA. New strings are generated from current strings via a perturbation mechanism. A new string is generated in the vicinity of the old string and both the operating costs are calculated. The string with least operating cost is copied into the next iteration. If the new string fails to get copied into the next iteration then the acceptance of the new string is determined by Boltzmann probability distribution shown in Fig 5.2. The flowchart of hybrid GA-SA applied to clustered based unit commitment is shown in Fig 5.2.
5.5 Division of Clusters based on Hybrid GA-SA

The proposed methodology can be unfolded into three stages.

Stage-1: In this stage objective cost function of each unit is obtained by using hybrid technique of GA-SA. Priority list of units is prepared based on the minimum objective cost functions and clusters are formed.

Stage-2: The pattern of load variation on the plant is a cycle of increasing and then decreasing takes place. Two separate algorithms are designed for load increasing pattern and for decreasing pattern. In this stage, an algorithm based on agglomerative clustering technique is developed for increasing load pattern.

Stage-3: This stage presents an algorithm for UC solution for the decreasing load condition. The algorithm is designed based on divisive clustering technique.

In this methodology hybrid GA-SA has been employed to form clusters. The operating cost of each plant is calculated and the units are clustered based on their fitness values. In this way hybrid GA-SA is employed to bring out the best clusters so that they can be employed to take up the load.

5.6 Characterization of the Units

Base load (BL) and Intermittent load (IL) units operate for long period in the day and they generate more number of units (MWH). Therefore, ideally speaking they should have minimum fuel cost, maximum generating capabilities but, can have high start-up costs and start-up times for the reason they are switched 'on' for the most of time. In addition, System reliability aspect is decided by the performance of these units. Semi-Peak Load (SPL) and Peak load (PL) units in contrast should have low start-up costs and start-up times as these units are rapidly switched 'on' and 'off' frequently. These units can have less generating capabilities and can have relatively high costs as they take up small loads above high base load and intermittent loads.
Preparation of priority list using hybrid GA-SA

In this stage the following steps are performed for preparing the priority list.

**Step-1:** The algorithm begins with the initialization of load patterns, unit characteristic along with the generation of initial population of string, where each string is corresponding to generation of a particular generator \( P_g \). The load pattern and unit characteristics are presented in table 5.2 & table 5.2.

**Step-2:** The binary coded strings are decoded into their corresponding decimal values of \( P_g \) by using linear mapping rule given by the equation 5.11.

\[
p_i = \frac{P_i^U - P_i^L}{2^l - 1} \times \text{decoded value of } S_i
\]

\( P_i^U \): Upper limit of generation of \( i^{th} \) unit; \( P_i^L \): Lower limit of generation of \( i^{th} \) unit.

\( S_i \): Sub-string of \( i^{th} \) unit; \( l_i \): Length of the string.

**Step-3:** These values are used to calculate the operating cost of each generator, which forms the objective function value. The operators of genetic algorithm perform in order to minimize the objective function.

**Step-4:** The operators of genetic algorithm like selection cross over and mutations are carried out and the child strings are generated for the first iteration. A neighborhood value of each string is generated and the operating cost of both the values is calculated. If the new string has less operating cost it is copied into the next iteration otherwise its acceptance is determined by Boltzmann probability distribution as shown in Fig 5.2.

This process continued until the termination criteria are met. The objective function values are brought out and given in table 5.4.

**Step-5:** Based on the generation cost functions, the closest cost function units are segregated into clusters as BL, IL, SPL and PL as given in table 5.5.

BL: Load upto 1000MW duration; IL: Load between 1000MW to 1200 MW; SPL: Load 1200MW to 1400 MW; PL: Load 1400MW to 1500MW.
The maximum limits for the four loads as:

BL-Max: 1000 MW; IL-Max: 1200 MW; SPL-Max: 1400 Mw and PL-Max: 1500 MW.

**Step-6:** For carrying out the agglomerative cluster algorithm, objective function values are stored in ascending order and for divisive cluster algorithm the objective function values are stored in the descending order as given in table 5.4. The closest values are divided into four clusters as BL, PL, Semi PL and IL.

**Step-7:** Read the load pattern as given in table 1 and unit characteristics in table 2.

**ED by Lambda-Iteration Method:**

**Step-1:** Set $\lambda$ value.

**Step-2:** Calculate $P_i$ for $i=1,2 \ldots n$. Where $n$ is the number of units in the cluster. $P_i$ is calculated subject to the minimization of objective function (1) under the constraints (5.7)-(5.10).

**Step-3:** Calculate error $\epsilon$ value (difference between demanded load and sum of generations).

**Step-4:** Check $\epsilon$ with tolerance value. If yes Go to main program to print UC results.

Else set new value of $\lambda$. Go To Step-2.

**Design of Divisive Clustering (DC) Algorithm for UC Problem**

This DC algorithm is proposed for UC when the load is decreasing after it stopped from increasing. The DCA starts at the point where some units in various clusters are already under 'on' condition. Now the requirement is to put some units under 'off' condition, so as to meet the present $D(t)$. The priority list is prepared based on the startup time/costs. The strategy is, to put off the unit with maximum generation cost.
**DC Algorithm:**

**Step-1:** Read the system load.

**Step-2:** De-Commit the next unit with maximum generation cost according to priority list.

**Step-3:** Commit the units in corresponding cluster by executing subroutine for Economic Dispatch (ED).

**Step-4:** Check the constraint: \( D(t) + R(t) \leq \text{sum of all generations} \). If condition is satisfied, go to main program. Else, go to step-2.

**Step-5:** Return.

### 5.7 Results and discussions

Table 5.2 and Fig 5.3 shows the daily load pattern on the plant and Table 5.3 shows the operating characteristics of all the units.

**Table 5.2: Daily Load Pattern On The Plant**

<table>
<thead>
<tr>
<th>Hour</th>
<th>Load (MW)</th>
<th>Hour</th>
<th>Load (MW)</th>
<th>Hour</th>
<th>Load (MW)</th>
<th>Hour</th>
<th>Load (MW)</th>
<th>Hour</th>
<th>Load (MW)</th>
<th>Hour</th>
<th>Load (MW)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>700</td>
<td>5</td>
<td>1000</td>
<td>9</td>
<td>1300</td>
<td>13</td>
<td>1400</td>
<td>17</td>
<td>1000</td>
<td>21</td>
<td>1300</td>
</tr>
<tr>
<td>2</td>
<td>750</td>
<td>6</td>
<td>1100</td>
<td>10</td>
<td>1400</td>
<td>14</td>
<td>1300</td>
<td>18</td>
<td>1100</td>
<td>22</td>
<td>1100</td>
</tr>
<tr>
<td>3</td>
<td>850</td>
<td>7</td>
<td>1150</td>
<td>11</td>
<td>1450</td>
<td>15</td>
<td>1200</td>
<td>19</td>
<td>1200</td>
<td>23</td>
<td>900</td>
</tr>
<tr>
<td>4</td>
<td>950</td>
<td>8</td>
<td>1200</td>
<td>12</td>
<td>1500</td>
<td>16</td>
<td>1050</td>
<td>20</td>
<td>1400</td>
<td>24</td>
<td>800</td>
</tr>
</tbody>
</table>

**Fig 5.3: Daily Load Pattern on the Plant**
Table 5.3: Operating characteristics of all the units

<table>
<thead>
<tr>
<th>Unit No. (i)</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p_{i,m}^a$ (MW)</td>
<td>455</td>
<td>455</td>
<td>162</td>
<td>130</td>
<td>130</td>
<td>80</td>
<td>85</td>
<td>55</td>
<td>55</td>
<td>55</td>
</tr>
<tr>
<td>$p_{i,m}^{max}$ (MW)</td>
<td>150</td>
<td>150</td>
<td>25</td>
<td>20</td>
<td>20</td>
<td>25</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>$a_i$ (kWh/hr)</td>
<td>1000</td>
<td>970</td>
<td>450</td>
<td>680</td>
<td>700</td>
<td>370</td>
<td>480</td>
<td>660</td>
<td>665</td>
<td>670</td>
</tr>
<tr>
<td>$b_i$ (kW/m²/hr)</td>
<td>16.19</td>
<td>17.26</td>
<td>19.7</td>
<td>16.5</td>
<td>16.6</td>
<td>22.26</td>
<td>27.74</td>
<td>25.92</td>
<td>27.27</td>
<td>27.79</td>
</tr>
<tr>
<td>$c_i$ (kW/m²/ hr)</td>
<td>0.00048</td>
<td>0.00031</td>
<td>0.00098</td>
<td>0.00211</td>
<td>0.002</td>
<td>0.00712</td>
<td>0.00079</td>
<td>0.00413</td>
<td>0.00222</td>
<td>0.00173</td>
</tr>
<tr>
<td>$T_i^{on}$</td>
<td>8</td>
<td>8</td>
<td>6</td>
<td>5</td>
<td>5</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$T_i^{off}$</td>
<td>8</td>
<td>8</td>
<td>6</td>
<td>5</td>
<td>5</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$HC(i)$ ($)</td>
<td>4500</td>
<td>5000</td>
<td>900</td>
<td>560</td>
<td>550</td>
<td>170</td>
<td>260</td>
<td>30</td>
<td>30</td>
<td>30</td>
</tr>
<tr>
<td>$CC(i)$ ($)</td>
<td>9000</td>
<td>10000</td>
<td>1800</td>
<td>1120</td>
<td>1100</td>
<td>340</td>
<td>520</td>
<td>60</td>
<td>60</td>
<td>60</td>
</tr>
<tr>
<td>$CS(i)$</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$Ini. State$</td>
<td>8</td>
<td>8</td>
<td>-6</td>
<td>-5</td>
<td>-5</td>
<td>-3</td>
<td>-3</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
</tr>
</tbody>
</table>

Table 5.4 shows the values of operating costs of various generators which has been decoded from the randomly generated binary strings of hybrid GA-SA. Table 5.5 shows the priority order of various units corresponding to their operating costs with respect to additive clustering and divisive clustering and Table 5.6 shows the segregation of all the 10 units in order to take up the daily load pattern.

Table 5.4 : Minimum Cost Function of each Unit

<table>
<thead>
<tr>
<th>Unit No</th>
<th>Cost Function</th>
<th>Unit No</th>
<th>Cost Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.0008</td>
<td>6</td>
<td>0.0013</td>
</tr>
<tr>
<td>2</td>
<td>0.0010</td>
<td>7</td>
<td>0.0019</td>
</tr>
<tr>
<td>3</td>
<td>0.0021</td>
<td>8</td>
<td>0.0012</td>
</tr>
<tr>
<td>4</td>
<td>0.0014</td>
<td>9</td>
<td>0.0015</td>
</tr>
<tr>
<td>5</td>
<td>0.0010</td>
<td>10</td>
<td>0.0014</td>
</tr>
</tbody>
</table>

75
Table 5.5: Priority List Is Formed With Minimum Operation Cost For Ac & Dc Algorithms

<table>
<thead>
<tr>
<th>Priority Order</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>For Additive</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td>10</td>
<td>4</td>
<td>6</td>
<td>8</td>
<td>9</td>
<td>7</td>
<td>3</td>
</tr>
<tr>
<td>For Divisive</td>
<td>3</td>
<td>7</td>
<td>9</td>
<td>8</td>
<td>6</td>
<td>4</td>
<td>10</td>
<td>5</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 5.6: Division of Units into Clusters and Their Priority

<table>
<thead>
<tr>
<th>Cluster type</th>
<th>Base load</th>
<th>Intermittent load</th>
<th>Semi peak load</th>
<th>Peak load</th>
</tr>
</thead>
<tbody>
<tr>
<td>Priority units in the cluster</td>
<td>1,2</td>
<td>5,10,4</td>
<td>6,8</td>
<td>9,7,3</td>
</tr>
</tbody>
</table>

Table 5.7 and Fig 5.4 show the allocation of generation to various units based on the daily load pattern and based on the clusters. It can be observed from the table that the clusters only take up the load allotted to them while the other generators do not take up the load until it falls into the other category. The operating costs of the generators taking the load can be observed from the table.

![Graph showing operational cost of units for Hybrid GA-SA method](image)

**Fig 5.4**: Operational cost of units for Hybrid GA-SA method
### Table 5.7: Generation of Units in 24 Hour Schedule

<table>
<thead>
<tr>
<th>S.No</th>
<th>Load MW</th>
<th>Commitment Schedule</th>
<th>Operational cost ($)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>700</td>
<td>416.8</td>
<td>283.1</td>
</tr>
<tr>
<td>2</td>
<td>750</td>
<td>443.6</td>
<td>306.3</td>
</tr>
<tr>
<td>3</td>
<td>850</td>
<td>455</td>
<td>170</td>
</tr>
<tr>
<td>4</td>
<td>950</td>
<td>455</td>
<td>185</td>
</tr>
<tr>
<td>5</td>
<td>1000</td>
<td>455</td>
<td>235</td>
</tr>
<tr>
<td>6</td>
<td>1100</td>
<td>455</td>
<td>335</td>
</tr>
<tr>
<td>7</td>
<td>1150</td>
<td>455</td>
<td>385</td>
</tr>
<tr>
<td>8</td>
<td>1200</td>
<td>455</td>
<td>435</td>
</tr>
<tr>
<td>9</td>
<td>1300</td>
<td>455</td>
<td>455</td>
</tr>
<tr>
<td>10</td>
<td>1400</td>
<td>455</td>
<td>455</td>
</tr>
<tr>
<td>11</td>
<td>1450</td>
<td>455</td>
<td>455</td>
</tr>
<tr>
<td>12</td>
<td>1500</td>
<td>455</td>
<td>455</td>
</tr>
<tr>
<td>13</td>
<td>1400</td>
<td>455</td>
<td>455</td>
</tr>
<tr>
<td>14</td>
<td>1300</td>
<td>455</td>
<td>455</td>
</tr>
<tr>
<td>15</td>
<td>1200</td>
<td>455</td>
<td>435</td>
</tr>
<tr>
<td>16</td>
<td>1050</td>
<td>455</td>
<td>285</td>
</tr>
<tr>
<td>17</td>
<td>1000</td>
<td>455</td>
<td>235</td>
</tr>
<tr>
<td>18</td>
<td>1100</td>
<td>455</td>
<td>335</td>
</tr>
<tr>
<td>19</td>
<td>1200</td>
<td>455</td>
<td>435</td>
</tr>
<tr>
<td>20</td>
<td>1400</td>
<td>455</td>
<td>455</td>
</tr>
<tr>
<td>21</td>
<td>1300</td>
<td>455</td>
<td>455</td>
</tr>
<tr>
<td>22</td>
<td>1100</td>
<td>455</td>
<td>335</td>
</tr>
<tr>
<td>23</td>
<td>900</td>
<td>455</td>
<td>230</td>
</tr>
<tr>
<td>24</td>
<td>800</td>
<td>455</td>
<td>345</td>
</tr>
</tbody>
</table>

Total Operating cost **541254**
Fig 5.5 and Table 5.8 show the comparison in the total operating cost between the proposed method and GA based clustering method and simple GA based unit commitment. It can be observed that with the help of the proposed method the total operating cost becomes greatly reduced as compared with the existing techniques.

Table 5.8: Comparison of Total Operating Cost

<table>
<thead>
<tr>
<th>Technique employed</th>
<th>Total operating cost in $ per day</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proposed method</td>
<td>541254</td>
</tr>
<tr>
<td>GA based clustered UC</td>
<td>544156</td>
</tr>
<tr>
<td>Conventional GA based UC [38]</td>
<td>565825</td>
</tr>
</tbody>
</table>

Fig 5.5: Comparison of operational costs
5.8 Conclusions

A novel method based on combined GA-SA based clustering technique is proposed to solve the problem of Unit Commitment. The proposed method is more advantageous and less heuristic. Following load pattern, two individual algorithms based on Additive and Divisive cluster algorithms are proposed for increasing and decreasing load patterns. The operating cost of generation of units is obtained and based on these costs the units are segregated into clusters. Two separate priorities lists one for increasing and another for decreasing load conditions are prepared based on generation costs. Matlab code is developed for proposed method and tested on a 10-Unit test system. The simulation results show that the operating cost greatly reduces with this technique and the obtained results are also compared with [38]. An amount of $24571 per day is saved. The strategy employed proved to be quite effective and satisfactory as evident through simulation results.