Thermal unit commitment using genetic algorithms

D. Dasgupta
D.R. McGregor

Indexing terms: Unit commitment, Genetic algorithms

Abstract: Unit commitment is a complex decision-making process because of multiple constraints which must not be violated while finding the optimal or near-optimal commitment schedules. The paper discusses the application of genetic algorithms to determine the short-term commitment order of thermal units in power generation. The objective of the optimal commitment is to determine the on/off states of the units in the system to meet the load demand and spinning reserve requirements at each time period, such that the overall cost of generation is minimised, while satisfying various operational constraints. The paper examines the feasibility of using genetic algorithms, and reports preliminary results in determining a near-optimal commitment order of thermal units in a studied power system.

1 Introduction

In power industries fuel expenses constitute a significant part of the overall generation cost. There exist different types of thermal power unit, using different fuels (e.g. coal, natural gas, oil) with different production costs, generating capacities and characteristics. Fig. 1 shows the block diagram of a simple power system which couples the generating units and different end users. The system usually operates under continuous variation of consumer load demand. This demand for electricity exhibits such large variations between weekdays and weekends, and between peak and off-peak hours, that it is not economical to keep all the generating units continuously online. So the demand and the reserve requirement impose global constraints in coupling all active generating units, while the different operating characteristics of each unit constitute local constraints. Thus determining which units should be kept online and which ones should not constitutes a difficult problem for operators seeking to minimise the system operational cost.

Thus the unit-commitment (UC) problem belongs to the class of complex combinatorial optimisation problem. Several mathematical programming techniques have been proposed [14, 4] to solve these time-dependent unit-commitment problems. They typically include complete priority ordering (CPO) and heuristic methods, dynamic programming (DP) [10, 13], the method of local variations, mixed integer programming, Lagrangian relaxation [2, 15, 23], the branch and bound method [4], bender decomposition [1] etc.

Among all these methods, dynamic programming methods based on a priority list have been used most extensively throughout the power industry. However, different strategies to select a set of units from a priority list have been adopted with dynamic programming to limit the search space and execution time. They include DP-SC (dynamic programming-sequential combination) [20], DP-TC (dynamic programming-truncated combination) [19], DP-STC [20] which is a combination of the DP-SC and DP-TC approaches, DP-VW (variable window-truncated dynamic programming) [17], and a neural-based method DP-ANN [18].

Recently some researchers have suggested techniques based on artificial intelligence to supplement the limitation of mathematical programming methods. These include simulated annealing [22], expert systems [16], heuristic rule-based systems [21] and neural networks [18]. These hybrid approaches have demonstrated some improvement in solving unit-commitment problems. However, heuristic and expert system based mathematical approaches require a lot of operator interaction, which is troublesome and time-consuming for even a medium-sized utility [9].

The paper presents an adaptive search method, called the genetic algorithm (GA), for the optimal or near-optimal commitment order of thermal units in power generation. Genetic algorithms are different from the above-mentioned classical methods in three ways [7]: they work with a coding of the parameter set rather than with actual parameters, and work equally with discrete and continuous functions; they search from a population of points; and they use probabilistic transition rules. These differences ensure that the success of a genetic algorithm is usually not related to the semantics of any particular problem.

The first author gratefully acknowledges the support given by the Government of Assam (India) for awarding a State Overseas Scholarship. We also acknowledge the valuable discussions with our colleagues in the Department, and the reviewers of the paper for their comments.
Our motivation to apply a genetic algorithm to the unit-commitment problem is that it can completely replace classical mathematical programming methods, and it is easy to implement as a search procedure.

The paper will first give a brief description of genetic algorithms, and will then explain their implementation in the problem of optimising the commitment order of thermal units in (single-area) power systems.

## 2 Genetic algorithms

Genetic algorithms [11] (GAs) represent a class of general-purpose stochastic search techniques which simulate natural inheritance by genetics and the Darwinian 'survival of the fittest' principle. Genetic algorithms are increasingly being applied in a variety of search, optimisation and scheduling problems across a wide spectrum of disciplines [7].

A genetic algorithm has four components that must be designed to solve the problem as contemplated. These components are: the syntax of the chromosome; its interpretation; its evaluation; and the set of operators to work on the chromosomes. A genetic algorithm works with a population of candidate solutions (chromosomes). The fitness of each member of the population (the point in search space) is computed by an evaluation function that measures how well an individual performs with respect to the problem domain. The constraints can be incorporated in the evaluation function in the form of penalty terms. The better-performing members are rewarded, and individuals showing poor performance are punished or discarded. So, starting with an initial random population, the genetic algorithm exploits the information contained in the present population, and explores new individuals by generating offspring in the next generation using genetic operations. By this selective breeding process it eventually reaches a near-optimal solution with a high probability.

A pseudo-code outlining the genetic algorithm is as follows:

```plaintext
Initialise P(t = 0);
/* P(0) = initial population */
Evaluate members of P(t);
While (not termination condition)
  Generate P(t + 1) from P(t) as follows:
    select individuals from P(t) on basis of fitness;
    Perform genetic operation on those selected;
    t = t + 1;
    evaluate members of P(t);
Several different genetic operators have been developed, but usually two (crossover and mutation) are extensively used to produce new chromosomes. The crossover operator exchanges portions of information between the pair of chromosomes. The mutation operator is a secondary operator which randomly changes the values at one or more genes of a selected chromosome in order to search for unexplored space. These operations are shown in Fig. 2. The workings of simple GAs have been described in detail elsewhere [7].

Genetic algorithms are suitable to solve problems where the domain knowledge is limited to evaluation procedures that can only measure the quality of any given point in the search space. If the problem space is encoded properly, the genetic search eventually converges to the globally optimal or a near-optimal solution. However, the quality of the solution and the length of time it takes to find the solution mainly depend on the nature of the problem and the use of GA parameters.
```

![Fig. 2 One-point crossover and simple mutation operations](image)

## 3 Problem description

Operating in the present competitive environment, unit commitment has become increasingly important in the power industry, because significant savings can be accrued from sound commitment decisions. To do this a scheduler must indicate which of the generating units are to be committed (on or off) in every time interval during the scheduling time horizon. This decision must take into account load forecast information and the economic implications of the start-up or shut-down of various units. The transition between their commitment states must satisfy the operating (minimum up and down time) constraints.

Therefore, there are two types of constraint associated with unit-commitment problems. The first type is the global one which results from load requirements that couple all the generating units during each time period. The second type of constraint is local, and it represents the different operating restrictions of the individual unit.

To maintain a certain degree of reliability, some standby capacity is necessary, that can immediately take over when a running unit breaks down or an unexpected load occurs. Hence the amount of spinning reserve is an important factor in ensuring an uninterrupted power supply. There are several spinning reserve policies that have usually been adopted: a fixed percentage of the forecast peak demand at every time period; a variable reserve; a reserve slightly greater than the output of the most heavily loaded unit; or a probabilistic reserve constraint known as unit-commitment risk may be used to ensure better system reliability. However, there should be no significant excess of reserve capacity in an economic commitment. However, spinning reserve constraints only provide the lower bound, since the total capacities of the committed units may not exactly match the load and spinning reserve.

### 3.1 Objective function

The objective (or cost) function of the unit-commitment problem is to determine the state of each unit $u_i$ (0 or 1) at each time period $t$, where the unit number $i = 1, \ldots, U_{\text{max}}$, and time periods $t = 1, \ldots, T_{\text{max}}$, so that the overall cost is a minimum within the scheduling time horizon.

$$
\min \sum_{i=1}^{U_{\text{max}}} \sum_{t=1}^{T_{\text{max}}} \left[ \alpha_i(AFLC) + u_i(1 - u_i) \right] S_i(x)
+ u_i^* (1 - u_i) \alpha_i \quad (1)
$$

For each committed unit, the cost involved is the start-up cost ($S_i$) and the average full-load cost (AFLC) per MWh.
according to the unit’s maximum capacity, such that

\[ \frac{t_\text{pre}}{t_\text{res}} \geq R' + L' \]  \hspace{1cm} (2)

where \( R' \) is the maximum output capacity of unit \( i \), \( L' \) is the demand and \( R' \) is the spinning reserve in time period \( t \). The above objective function should satisfy the minimum up-time and down-time constraints of the generating units.

The start-up cost is expressed as a function of the number of hours \( (x) \) the unit has been down, and the shut-down cost is considered as a fixed amount \( (D_i) \) for each unit per shut-down; these state transition costs are applied in the period when the unit is committed or taken off-line, respectively [10].

However, unit-commitment decisions based solely on unit AFLC usually do not provide sufficient information about the impact of system load conditions on how efficiently (e.g. fully) the committed units are being utilised while determining the optimal or a near-optimal commitment [12]. An index is used to measure the utility of each commitment decision:

\[ \text{utility factor} = \frac{\text{load} - \text{reserve requirements}}{\text{total committed output}} \]

while satisfying the global constraint (eqn. 2). This factor helps to compensate for the deficiency associated with over-committed decisions based solely on the classical AFLC of units. Therefore, during the performance evaluation, commitment decisions with a low utility factor are penalised accordingly.

4 Implementation details

The implementation proposed here uses a simple genetic search technique to determine the optimal (a least total cost solution) or a near-optimal commitment schedule for a given study period.

The short-term commitment is considered with a 24 hour time horizon, which may be repeated using the load profile of each day. Since the system load varies substantially over a 24 hour period and the cost of operation over this time span depends on the timing and frequency of the unit’s start-ups and shut-downs, this commitment problem is generally viewed as a multiperiod problem in which the commitment horizon is divided into a number of periods of shorter length (usually a 1 hour commitment interval).

In order to use a genetic algorithm for this problem, the first step is to encode the commitment space. If the whole planning horizon (24 hours) is encoded in a chromosome by the concatenation of commitment spaces of all time periods, it appears to be possible to determine a complete commitment order for the whole span at a time by performing a global search using the genetic algorithm. However, this approach makes the problem hard for the genetic algorithms to solve, for the following reasons.

First, with the increased number of generating units, the length of the encoded chromosome increases in a higher ratio; for example, for a system with 20 generating units, the chromosome length becomes 480 bits when the scheduling span is 24 hours. However, with a longer encoded string, genetic algorithms find difficulty in reaching a near-optimal solution, since a genetic search exploits schemata which represent hyperplanes, and an increase in the size of encoding increases the amount of space the algorithm needs to explore to find good schemata [11]. Secondly, since the optimal population size is a function of the string length for better schema processing [8], so with the increased string length a bigger size of population is needed. However, this will result in both memory and computational overhead. Thirdly, in addition to the above drawbacks, such encoding makes the problem space highly epistatic (a biological term that states the amount of interdependency among genes encoding the chromosome). When the epistasis is extremely high, gene values are so dependent on each other that unless a complete set of unique values is found simultaneously, no substantial fitness improvements can be seen [6]. In this case, since the commitment decisions of previous hours have a strong effect on the decisions in successive hours, so encoding these dependent decisions in a single string makes the representation highly epistatic. This increases the complexity of the search space and makes it difficult for a genetic search to find a near-optimal solution in a reasonable time.

In this work, the problem is considered as a multi-period process, as in practice, and a simple genetic algorithm is used for commitment scheduling. Each chromosome is encoded in the form of a position-dependent gene (bit string) representing the number of thermal units available in the system, and the allele value at loci give the state (on/off) of the units as a commitment decision at each time period, as shown in Fig. 3.

\[ u_i = 0 \text{ or } 1 \text{ according to the unit and } i \text{ is unit index} \]

\[ \text{on/off} \]

Fig. 3 Chromosomal representation of unit-commitment decisions

Unit-commitment decisions which satisfy a load-reserve requirement and the operating constraints of units are regarded as feasible solutions, and any violation of the constraints is penalised through a penalty function. So the raw fitness function is formulated here by using a weighted sum of the objective function, and values of the penalty function based on the number of constraints violated and the extent of these violations. By choosing suitable weights for the penalty function it is possible to find a near-optimal solution to the problem. In our case, fitness function

\[ = \text{objective function} + \text{penalty function} \]

\[ (L - R, \text{utility factor, min up, min down}) \]

where \( L - R \) is the load-reserve requirement, and min up and min down are minimum-up and minimum-down time constraints of the units.

In our implementation the following scaling technique is used to normalise fitness and to produce a non-negative figure of merit:

scaled fitness \( (I) = (M - I)/(M - N) \)

where \( I \) is the individual raw fitness, and \( M \) and \( N \) are the relative maximum and minimum fitnesses, respectively, among all individuals in the current generation. The goal is to maximise the scaled fitness in order to minimise the cost (or objective) function. The scaled fitness is used to determine the probability of selection of members in the population for breeding.

The flow chart of the implementation of the genetic algorithm for unit commitment is shown in Fig. 4. For a better understanding of the diagram, some of the flow lines connected to the repeatedly used program module.
(GA routines) and domain information (look-up tables) are numbered in accordance with the sequence of execution.

Fig. 4 Flow diagram for unit commitment using genetic algorithms

The genetic-based unit-commitment program starts with a random initial population (at $T = 1$), and it computes the fitness of each individual (commitment decision) using the forecasted load demand at each period, and the operating constraints of the units (using look-up tables). Each time the genetic optimiser is called, it runs for a fixed number of generations or until the best individual remains unchanged for a long time (here 100 successive generations).

Since the unit-commitment problem is time-dependent, these piecewise approaches of working forward in time and retaining the best decision cannot be guaranteed to find the optimal commitment schedule. The reason for this is that a decision with significantly higher costs during the early hours of scheduling could lead to significant savings later and may produce a lower overall cost-commitment schedule.

To make the genetic-based unit-commitment program robust in finding near-optimal solutions, a number of feasible commitment decisions (less than or equal to a predefined value $S$) with smaller associated costs are saved at each time period. These strategies (a strategy is a sequence of commitment decisions from the starting period to the current period, with its accumulated cost) determine how many possible alternative paths are available at each period to find the overall operation cost. Fig. 5 illustrates the different paths available, where one path converges to the other midway and another stops because it could not find a path within the allocated resources. The selection of $S$ is effective in economical scheduling (in finding an optimal solution), memory requirement and computation time. To save computation time, the same strategies are carried forward to the next period, if the load remains unaltered or varies slightly in the current period such that load–reserve requirements are satisfied by all strategies. If a strategy cannot meet

Fig. 5 Alternative paths (strategies) for finding a near-optimal commitment order

the demand of present period, the genetic optimisation process is performed for the period to find a feasible successor path with smaller cost. This approach increases the likelihood of finding the path of minimum cumulative cost.

These temporary commitment strategies are used to update the status information of the units (up-time/down-time counter) to keep track of the units in service or shut-down for a number of successive hours. In the next time period half of the population is replaced by randomly generated individuals to introduce diversity into the population so that the search for a new commitment strategy can proceed according to the load demand. The purpose of keeping half of the previous population is that in most situations the load varies slightly in some successive time intervals and the previous better individuals (commitment strategies) are likely to perform well in the current period. However, if there is a drastic change in load demand, newly generated individuals can explore the commitment space to find the best solution. The iterative process continues for each period in the scheduling horizon, and the accumulated cost associated with each commitment strategy gives the overall cost for the commitment path.

In a time period, if a unit is to be decommitted owing to a decrease in load demand, and because of that its minimum-up time constant is violated, then the unit is considered to remain committed (in banking state) until its minimum-up period is completed. In addition, to tackle sharp rises in load demand in the next period, a look-ahead mechanism is incorporated which decides that the unit will remain committed (in the banking state) even though it represents an uneconomic decision for the current period.

During these multiperiod optimisation processes, if in a particular period no feasible solution (strategy) is found, the process is repeated so that at least one feasible solution is found before shifting to the next period. However, in our test example such repetitions are required only in a few occasions in later periods.

5 Experimental results

The unit-commitment program based on a genetic algorithm is applied to an example problem which consists of 10 thermal units. The capacities, costs and operating constraints vary greatly among the various generating units in these test systems. Different types of load profile are tested, which represent typical operating circumstances in the studied power system. We have considered short-term scheduling where the time horizon is 24 hours and scheduling for an entire day is done in advance, which may be repeated by using the load profile of each day for long-term scheduling.

In these experiments the spinning reserve requirement is assumed to be 10% of the expected hourly peak load. A program to implement the algorithm has been run on a SUN (Sparc) workstation under a UNIX 4.11 operating system. The experiment is conducted with a population size of 250 using different crossover and mutation rates. For the result reported here (shown in tabular form), a crossover probability of 78% and a mutation rate of 15% were used, along with a stochastic remainder selection scheme [3] for reproduction. We also used an elitist scheme which passes the best individual unaltered to the succeeding generation. Each run was allowed to continue up to 500 generations and the strategy path with minimum cumulative cost gives a near-optimal commitment for the whole scheduling period.

For this example, Table 1 gives the characteristics of, and the initial states of, the generating units. Table 2 gives the commitment schedules for two cases, which were run independently. In first case, one best solution is saved at each time period, and in second case, multiple least-cost strategies are saved to determine the minimum-cost path. A comparison shows that a substantial reduction in overall cost can be achieved when the best commitment schedule is determined from multiple least-cost strategies. In this Table, the second column gives the hourly load demand, the third column shows the total requirement after adding the spinning reserve, and the other columns give the total output capacity (in MW) of the committed units and the state of the units in each case, where '1' is used to indicate that a unit is committed, and '0' to indicate that a unit is decommitted.

The genetic-based unit-commitment system has been tested in different operating conditions to evaluate the algorithm's performance.

It is observed that the scheduling which produces optimal power output does not always give the overall minimum-cost scheduling; also, the minimum cost scheduling is very sensitive to the system parameters and the operating constraints of the generating units.

6 Conclusions

In this paper we have discussed the application of a genetic algorithm to solve short-term unit-commitment problems, i.e. deciding the commitment order of units for an entire day in advance. When a set of smaller cost

| Unit | Maximum capacity | Minimum up-time | Minimum down-time | Initial status | Start-up cost $b_1$ | Shut-down cost $b_2$ | AFC
|------|-----------------|----------------|-----------------|----------------|-------------------|-------------------|-----|
| 1    | 60              | 3              | 1               | -1             | 85                | 20.588            | 0.2 | 15 15.3
| 2    | 80              | 3              | 1               | -1             | 101               | 20.584            | 0.2 | 26 16
| 3    | 100             | 4              | 2               | -1             | 114               | 22.57             | 0.2 | 40 20.2
| 4    | 120             | 4              | 2               | 5              | 94                | 10.65             | 0.18| 32 20.2
| 5    | 150             | 5              | 3               | -7             | 113               | 18.639            | 0.18| 29 25.6
| 6    | 280             | 6              | 2               | 2              | 176               | 27.588            | 0.15| 42 30.5
| 7    | 520             | 8              | 4               | -5             | 267               | 34.749            | 0.09| 75 32.5
| 8    | 150             | 4              | 2               | 3              | 282               | 45.749            | 0.09| 49 26.0
| 9    | 320             | 6              | 2               | -6             | 187               | 39.617            | 0.130|70 25.9
| 10   | 200             | 5              | 2               | -3             | 227               | 26.641            | 0.11|62 27.0

- indicates unit is down for hours, and positive otherwise

We used start-up cost $b_1 = [1 - \frac{1}{\sqrt{1 + \left(\frac{b_2}{x}\right)^2}}] + b_2$.  

**Table 1: Characteristics and initial state of the thermal units**

Table 2: Unit-commitment schedules determined by the genetic algorithm

<table>
<thead>
<tr>
<th>Case 1 When only best strategy is saved at each hour</th>
<th>Case 2 Best of five least-cost strategies saved</th>
</tr>
</thead>
<tbody>
<tr>
<td>Commitment output</td>
<td>State of units</td>
</tr>
<tr>
<td>-------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>h</td>
<td>MW</td>
</tr>
<tr>
<td>1</td>
<td>1489.00</td>
</tr>
<tr>
<td>2</td>
<td>1372.00</td>
</tr>
<tr>
<td>3</td>
<td>1289.00</td>
</tr>
<tr>
<td>4</td>
<td>1280.00</td>
</tr>
<tr>
<td>5</td>
<td>1271.00</td>
</tr>
<tr>
<td>6</td>
<td>1314.00</td>
</tr>
<tr>
<td>7</td>
<td>1372.00</td>
</tr>
<tr>
<td>8</td>
<td>1314.00</td>
</tr>
<tr>
<td>9</td>
<td>1271.00</td>
</tr>
<tr>
<td>10</td>
<td>1240.00</td>
</tr>
<tr>
<td>11</td>
<td>1197.00</td>
</tr>
<tr>
<td>12</td>
<td>1182.00</td>
</tr>
<tr>
<td>13</td>
<td>1154.00</td>
</tr>
<tr>
<td>14</td>
<td>1138.00</td>
</tr>
<tr>
<td>15</td>
<td>1124.00</td>
</tr>
<tr>
<td>16</td>
<td>1096.00</td>
</tr>
<tr>
<td>17</td>
<td>1066.00</td>
</tr>
<tr>
<td>18</td>
<td>1073.00</td>
</tr>
<tr>
<td>19</td>
<td>993.00</td>
</tr>
<tr>
<td>20</td>
<td>978.00</td>
</tr>
<tr>
<td>21</td>
<td>943.00</td>
</tr>
<tr>
<td>22</td>
<td>1022.00</td>
</tr>
<tr>
<td>23</td>
<td>1081.00</td>
</tr>
<tr>
<td>24</td>
<td>1459.00</td>
</tr>
</tbody>
</table>

Cumulative scheduling cost = 940103.65
Cumulative scheduling cost = 877854.32

The difference in cost is approximately 7% in these two cases

7 References