CHAPTER 3

THERMAL UCP USING SIMULATED ANNEALING AND TABU SEARCH TECHNIQUES

3.1 INTRODUCTION

This chapter discusses the characteristics of Simulated Annealing and Tabu Search. Simulated Annealing (SA) is a powerful general-purpose stochastic search technique to solve hard-constrained optimisation problems. This Simulated Annealing algorithm and Tabu Search are titled under the classification of statistical method of training. The important novel feature of selecting this algorithm is, it overcomes the problems like entrapment of local minima, and it gives us the best optimum solution for the problem considered. Statistical methods differ from other algorithm by the way of accepting or rejecting the new feasible solution evolved during the training process. Here comes the concept of Boltzman distribution which determines the range of probability of acceptance and gives a procedure for selecting the optimum solution quickly. SA has been successfully applied to a range of combinatorial problems in electrical engineering, including power systems. Tabu search is a meta strategy for guiding known heuristics to overcome local optimality. The overall approach is to avoid entrainment in cycles by forbidding or penalizing moves, which take the solution, in the next iteration, to points in the solution space previously visited. The Tabu search has traditionally been used on combinatorial optimization problems. The technique is straightforwardly
applied to continuous functions by choosing a discrete encoding of the problem. Many of the applications in the literature involve integer programming problems, scheduling, routing, traveling salesman and related problems.

3.2 SIMULATED ANNEALING

3.2.1 Physical Concepts

SA is a powerful technique used to solve combinatorial optimization problems. Although the simulated annealing algorithm (SAA) has the disadvantage of consuming more CPU time, it also has some advantages. They are:

- It could find a high quality solution that does not entirely depend on the choice of the initial solution.
- It does not need a complicated model of the problem under study.
- It can start with any given solution and try to improve the solution. This feature could be utilized to improve a solution output from other sub optimal heuristic methods.
- It has been theoretically proved to converge to the optimal solution
- It does not need a large computer memory.

Annealing, in physical terms, refers to the process of heating up a solid to a high temperature followed by slow cooling achieved by decreasing the temperature of the environment in steps. At each step the temperature is
maintained constant for a period of time sufficient for the solid to reach thermal equilibrium (Mantawy et al 1998, Einar Stale Huse et al 1999, Purushothama and Lawrence Jenkins 2003). At equilibrium, the solid could have many configurations; each corresponding to different spins of the electrons and to specific energy levels. At equilibrium, the probability of a given configuration, $P_{\text{config}}$, is given by Boltzmann distribution in Equation (3.1).

$$P_{\text{config}} = K \exp\left(-\frac{E_{\text{config}}}{C_p}\right)$$  \hspace{1cm} (3.1)

Where, $E_{\text{config}}$ is the energy of the given configuration and $K$ is a constant, Metropolis et al, proposed a Monte Carlo method to simulate the process of reaching thermal equilibrium at a fixed temperature $C_p$. In this method, a randomly generated perturbation of the current configuration of the solid is applied so that a trial configuration is obtained. Let $E_c < E_t$, till a lower energy level has been reached and the trial solution has to be altered. If $E_c > E_t$, then the trial configuration is accepted as the current configuration with probability given in Equation (3.2).

$$\exp\left(\frac{(E_c - E_t)}{C_p}\right)$$  \hspace{1cm} (3.2)

Where $C_p$ ~ control parameter of the cooling schedule. The process continues where a transition to a configuration of higher energy level is not necessarily rejected. Eventually thermal equilibrium is achieved after a large number of perturbations, where the probability of a configuration approaches Boltzman Distribution. By gradually decreasing $C_p$ and repeating Metropolis simulation, new lower energy levels become achievable. As $C_p$ approaches zero, the least energy configurations will have a positive probability of occurring.
3.2.1.1 Applications of SA to Combinational Optimization Problems

By making an analogy between the annealing process and the optimization problem, a great class of combinatorial optimization problems can be solved following the same procedure of transition from equilibrium state to another, reaching the minimum energy of the system. This analogy can be stated as:

- Solutions in the combinatorial optimization problems are equivalent to states (configurations) of the physical system.
- The cost of a solution is equivalent to the energy of a state.
- Demand as a control parameter is introduced to play the role of temperature in the annealing process.

In applying the SAA to solve the combinatorial optimization problems, the basic idea is to choose a feasible solution at random and then get a neighbor to this solution. A move to this neighbor is performed if it has a better lower objective value or, in case the neighbor has a higher objective function value, if \( \exp\left(-\frac{\Delta E}{C_p}\right) \geq U(0,1) \), where \( \Delta E \) is the increase in objective value in the neighbor. The effect of decreasing \( C_p \) is that the probability of accepting an increase in the objective function value is decreased during the search. The most important part in the SAA is to have a good rule for finding a diversified and intensified neighborhood so that a large amount of the solution space can be explored. Another important part is how to choose the initial value of \( C_p \) and how \( C_p \) should decrease during the search. SA means a simulation of the annealing process of metal. If the temperature is lowered carefully from a high temperature in the annealing process, the melted metal will produce the
crystal at 0°K. This algorithm (Mantawy et al 1998) finds the near optimal solution by substituting the random movement of solution for the fluctuation of particle of system in the annealing process and by making the objective function value correspond to the energy of the system, which decreases, with the descent of temperature. Here taking temperature and the demand as the control parameter refines the basic SA algorithm. Hence the solution quality is improved. The flowchart of SA algorithm is shown in Figure 3.1.

3.2.2 SA Algorithm for UCP

Step (0): Find the Initial Feasible Solution By Optimum Allocation.

Step (1): Demand and Temperature are taken as the Control Parameter.

Step (2): Generate the Trial Solution.

Step (3): Check for the stopping criterion.

   (a) If satisfied, go to the Next Hour for Checking the Same.

   (b) Else, Decrement the System Peak Demand for that instant and again generate the Trial Solution.

Step (4): Get the Optimal Schedule and

   (a) Assuming the Fuel Cost to be Constant per hour, equate the total power demand to the total no of units switched ON.

   (b) From the Total Fuel Cost subtract the constant cost function C. Equate the Remaining cost Value to the ON Units Equally.
Figure 3.1 Flowchart of SA for UCP
(c) Assume the Initial Temperature of the turbine as 660 degrees and a generation of 210 MW.
(d) If the Demand Decreases/Increases, then the temperature should Decrease/Increase by an amount given by the Equation (3.3).

\[ C_{p_{k+1}} = C_{p_k} / ((1 + (C_{p_k} \cdot \ln(1 + \delta)) / 3\sigma C_{p_k})) \]  

(3.3)

Where,
- \( C_p \) ~ Control Temperature in Degrees.
- \( \delta \) ~ Distance Parameter.
- \( \sigma \) ~ Standard Deviation of the Cost Value Generated.

Step (5): Calculate the Total Operating Cost \( F_t \) as the summation of running cost and Start up and Shut down Cost by Equation (2.3).

3.3 TABU SEARCH

3.3.1 Physical Concepts

In general terms, tabu search (TS) is an iterative improvement procedure that starts from some initial feasible solution and attempts to determine a better solution in the manner of a greatest – decent algorithm. However, TS is characterized by an ability to escape local optima by using a short-term memory of recent solutions. Moreover, TS permits backtracking to previous solutions, which may ultimately lead, via a different direction, to better solutions. The main two components of a TSA are the tabu list (TL) restrictions and the aspiration level (AL) of the solution associated with these restrictions (Mantawy et al 1998, Gerald B. Sheble and George N. Fahd 1994).
3.3.1.1 Significance

TS is a powerful optimisation procedure that has been successfully applied to a number of combinatorial optimisation problems. It has the ability to avoid entrapment in local minima. TS employ a flexible memory system (in contrast to ‘memory less ’ systems, such as Simulated Annealing and Genetic Algorithm and rigid memory system as in branch and bound). Specific attention is given to the short-term memory component of TS, which has provided solutions superior to the best obtained by other methods for a variety of problems. TS also has many advantageous features including:

❖ Finding a high quality solution that does not entirely depend on the choice of the initial solution.

❖ Not needing a complicated mathematical model of the problem under study. Hence problems that are difficult to formulate can be easily implemented.

❖ Ability to start with any given solution and attempt to improve it. This feature could be utilized to improve a solution output from other sub optimal or heuristic methods.

The unit commitment problem can be solved using the Tabu Search method (Mantawy et al 1998, Gerald B. Sheble and George N. Fahd 1994, Xaiomin Bai and Shahidepour 1996). New rules for randomly generating feasible solutions are also applied. The combinatorial optimisation sub problem is solved using the TSA while an economic dispatch problem is solved by a quadratic programming routine. Different criteria for the UCP are implemented...
and compared. Some examples are cited to test the developed computer model and the required results are obtained.

3.3.1.2 Tabu List Restrictions

TS may be viewed as a 'meta-heuristic' superimposed on another heuristic. The approach is designed to transcend local optimality by the strategy of forbidding certain moves. The purpose of classifying certain moves as forbidden is basically to prevent cycling. Moves that hold tabu status are generally a small fraction of those available, and a move loses its tabu status and becomes accessible after a relatively short time. The choice of appropriate types of tabu restrictions list, TL, depends on the problem under study. A fraction that utilizes historical information from the search process, extending up to ‘Z’ iterations in the past, where ‘Z’ can be fixed or variable depending on the application or stage of search. The TL restrictions could be stated directly as a given change of variables or indirectly as a set of logical relationships or linear inequalities. The use of these two approaches depends on the size of TL for the problem under study.

TL’s are managed by recording moves in the order in which they are made. Each time a new element is added to the bottom of a list, the oldest element on the list is dropped from the top. A TL is designed to ensure the elimination of cycles of length equal to the TL size. Empirically, TL sizes that provide good results often grow with the size of the problem and stronger restrictions are generally coupled with smaller lists. The way to identify a good TL size for a given problem, the class and choice of tabu restrictions is by simply watching for the occurrence of cycling when the size is too small and
the deterioration in solution quality when the size is too large. The best sizes lie in an intermediate range between these extremes. In some applications, a simple choice of ‘Z’ in a range centred on 7 seems to be quite effective.

3.3.1.3 Aspiration Criteria

Another key issue of TS arises when the move under consideration has been found to be tabu. The tabu status of a move is not absolute, but can be overruled if certain conditions are met, expressed in the form of aspiration level (AL). If an appropriate aspiration criterion is satisfied, the move will be accepted in spite of the tabu classification. Different forms of aspiration criteria are used in the literature. The one we use in this study is used to override the tabu status if the tabued move yields a solution, which has better objective function than the one obtained earlier for the same move. Consequently, the AL associated with each move in the TL is equal to the value of the objective function obtained when performing that move. The effectiveness of using aspiration criterion is to add some flexibility in the tabu search by directing the search towards the attractive moves.

3.3.2 TS Algorithm for UCP

In solving the UCP, two types of variables need to be determined. The unit’s status variables U and V, which are integer variables and the unit, have output power variables P that are continuous variables. The problem can then be disintegrated into two sub problems, a combinatorial problem in U and V and a non linear optimisation problem in P. TS is used to solve the combinatorial optimisation while the non-linear optimisation is solved via a
quadratic programming routine. Here the basic TS algorithm is modified in order to improve the solution quality and the flowchart of the TS algorithm is shown in Figure 3.2. The algorithm (Mantawy A.H. et al 1998) contains three major steps:

- First, generating randomly feasible trial solutions
- Second, calculating the objective function of the given solution by solving the EDP
- Third, applying the TS procedures to accept or reject the solution in hand.

Step(0): Assume that the fuel costs to be fixed for each hour and all the generators share the loads equally.

Step(1): By optimum allocation find the initial feasible solution \((U_i, V_i)\) where \(U_i\) and \(V_i\) are integer variables denoting the unit status

Step(2): Demand is taken as the control parameter.

Step(3): Generate the trial solution.

Step(4): Calculate the total operating cost, \(F_i\), as the summation of running cost and start up – shut down cost.

Step(5): Tabulate the fuel cost for each unit for every hour.
Figure 3.2 Flowchart of TS for UCP
3.4 GENERATING TRIAL SOLUTION

The neighbors should be randomly generated, feasible, and span as much as possible the problem solution space. Because of the constraints in the UCP this is not a simple matter. The most difficult constraints to satisfy are the minimum up/down times. The implementation of new rules to obtain randomly feasible solutions faster is described in (Mantawy et al 1998).

3.5 GENERATING AN INITIAL SOLUTION

The TS algorithm requires a starting feasible schedule, which satisfies all the system and units constraints. This schedule is randomly generated. The algorithm given in (Mantawy et al 1998) is used for finding this starting solution.

3.6 OPERATING COST CALCULATION

Once a trial solution is obtained, the corresponding total operating cost is determined. Since the production cost is a quadratic function, the EDP is solved using a quadratic programming routine. The start – up cost is then calculated for the given schedule.

3.7 STOPPING CRITERIA

There may be several stopping criteria for the search. For this implementation, the search is stopped if the following conditions are satisfied:
The load balance constraints are satisfied.
The spinning reserve constraints are satisfied.

3.8 IMPLEMENTATION

To solve for the UCP using SA and TS, software in Turbo C package is developed. The software provides interactive approach in dealing with the various data input required for solving the UCP from the user.

3.9 CONCLUSION

Tabu Search has the ability to escape local optima, which usually causes simple descent algorithms to terminate, by using a short-term memory of recent solutions, which may lead via different directions, to better solutions. Tabu search has a flexible memory system. The short-term memory component of TS provides solutions superior to the best obtained by other methods. SA escapes from local minima by employing a probability function. SA has been theoretically proven to converge to the optimal solution with a probability of one. SA can find a high quality solution that does not depend on the choice of the initial solution. SA does not need a large memory. Both SA and TS are easy to implement. Both the algorithms do not need a complex mathematical model. They can start with any initial solution and try to improve on it. On comparing the results obtained from SA and TS methods, TS is better than SA with respect to cost and computing time.