CHAPTER 6

SA BASED SOLUTION PROCEDURE FOR CONDUCTOR GRADATION PROBLEM

6.1 INTRODUCTION

As discussed in chapter-4, in radial distribution feeders, the power flows in different sections vary and hence the grading of conductor sizes along the feeder mains further reduces the system cost. The problem was solved using a GA algorithm in chapter-4. This chapter presents a solution procedure to solve this problem using Simulated Annealing (SA) algorithm.

6.2 EXISTING SA BASED METHODOLOGIES

As discussed in chapter-2, this problem of conductor grading for radial feeders was first studied by M.Ponnavaikko and K.S.P Rao, using Dynamic Programming and Method of Local Variations methods. No solution procedures using Simulated Annealing Algorithmic approach, to solve this type of problems have been developed so far. In this research, a SA based solution procedure has been developed to solve this problem. The proposed Simulated Annealing Algorithmic approach is presented in this chapter.

6.3 PROPOSED SIMULATED ANNEALING ALGORITHMIC APPROACH

6.3.1 Mathematical Formulation of the Problem

The objective function of the problem as discussed in chapter-2 is stated as

Minimize

\[ Z = \sum_{i=1}^{ND} \left( \frac{K_4}{a_{ci}} + K_5 a_{ci} \right) \]  \hspace{1cm} (2.21)

Subject to,

\[ \sum_{i=1}^{ND} \left( \frac{K_2}{a_{ci}} \right) \leq \Delta v_m \]  \hspace{1cm} (2.22)
and \( a_i, a_{c1}, ..., a_{c_N} \) \( (2.23) \)

Where, \( K_{2i}, K_{3i}, K_{4i} \) and \( K_{5i} \) as described in Appendix-C are reproduced below.

\[
\Delta v'_m = \frac{\Delta v_m}{(1 + g)^M} - \sum_{i=1}^{ND} K_{3i} 
\]

\( (2.24) \)

\[
K_{2i} = \sqrt{3} l_{m_i} \ell_i \rho \cos \theta 
\]

\( (2.25) \)

\[
K_{3i} = \sqrt{3} l_{m_i} \ell_i x \sin \theta 
\]

\( (2.26) \)

\[
K_{4i} = 26.28 \rho \ell_i l_{m_i}^2 \left[ \sum_{k=1}^{M} \frac{(1 + g)^{2k} (LLF_k) C_{ek}}{(1 + u)^k} \right] + (1 + g)^{2M} \sum_{k=M+1}^{NLF} \frac{(LLF_k) C_{ek}}{(1 + u)^k} 
\]

\( (2.27) \)

\[
K_{5i} = f' \ell_i 
\]

\( (2.28) \)

The analogy between the SA procedure used in thermodynamic systems [75, 76, 80, 85] and the combinatorial problem is straightforward and can be brought out as follows:

Current state of the thermodynamic system \( \leftrightarrow \) current solution

Energy equation given by \( E \) for the thermodynamic system \( \leftrightarrow \) objective function \( C \)

Ground state \( \leftrightarrow \) global minimum (current solution at the end of SA process at \( T''_{\text{min}} \))

Temperature \( T' \) \( \leftrightarrow \) control parameter that gives a measure of tolerance allowed between two successive function values for accepting a worse point.

Cooling time that must be allowed to achieve quasi equilibrium state at a temperature \( \leftrightarrow \) Number of iterations performed at each temperature, \( N_t \)
To define the problem within the framework of SA, we have to construct a coding for solution. In electrical terms this solution consists of a set of elements representing the feeder sections in terms of conductor sizes. This set of elements has to be of either integers or binary coded integers for problems of above nature, whose variables happen to be the discrete ones. Hence, a solution to this problem is a string of integers in SA terms. The problem considers different conductor sizes for the feeder segments of a given radial distribution feeder which follows a tree configuration. As discussed and illustrated using the feeder shown in fig-4.1 of chapter-4, a tree consists of ND consumer nodes fed from a source node (substation node). Thus a tree has a total number of feeder segments equal to ND. Hence, a solution string has ND integers as shown in fig-6.1. Each solution string is decoded to obtain the value of the cost function. That solution string which results in a minimum value of the cost function is selected as the optimum solution.

\[ c_1, c_2, \ldots, c_3, c_4, \ldots, c_{13} \]

\begin{array}{ccccccccccc}
3 & 3 & 3 & 3 & 3 & 3 & 2 & 2 & 2 & 1 & 1 & 1
\end{array}

**Fig-6.1** A solution string of conductor gradation problem

Simple power flow algorithm using moment method is run at the beginning of the algorithm to determine \( I_{m_i}, i = 1, \ldots, ND \). Flow in the infeed edge of the tail node is the same as its load demand. Flow in all other edges is the sum total of the flows in infeed edges of its children nodes. Nodes that have no children are identified as tail nodes and the tree search routine starts updating the flow variables when it encounters these nodes.

### 6.3.2 Problem Formulation under the Framework of SA

The following definitions and tasks of the GA procedure discussed in section-4.3 for solving the conductor gradation problem is adopted for solving the problem using SA procedure also.

- Mathematical formulation of the problem as discussed in section-2.2.2 of chapter-2
- The Data Structure and feeder Construction (Appendix-K)
- Tree Encoding as discussed section-4.3.3 of chapter-4 except that a solution called as ‘Chromosome’ in GA terminology is referred to as ‘Solution String’ in SA procedure and ‘LCHROM’ as ‘LSTG’

- Decoding as discussed section-4.3.4 of chapter-4

- Generation of initial population as discussed section-4.3.5.1 of chapter-4. Since SA algorithm is a point-by-point method, one initial solution string only is generated.

The initial solution can be either given as input or generated at random. The initial solution is the current solution at the first transition and the current best solution at the first iteration of the initial temperature $T_0'$. The initial temperature $T_0'$ is obtained from the average of differences of objective function values of 20 random transitions resulting in uphill moves (usually between 20 and 100), so as to pick up samples from all the regions of the solution space that contains all feasible solutions. As the infeed edges of ND consumer nodes constitute a solution string, ND transitions are scheduled in order, in an iteration of a temperature. A transition changes (increments or decrements) the value of the randomly selected feeder section/node to a conductor size next to its current size after verifying the feasibility so as not to disturb the ordering of the conductor sizes towards both the ends of the feeder. Let us consider the feeder and a solution of the feeder shown in fig-4.1 and fig-4.2 respectively. The element $c_5 = 3$ cannot be decremented to 2 while $c_6$ can be decremented to 2. To state another example, $c_{11} = 1$ cannot be incremented to 2 while $c_{10}$ can be incremented to 2. Whenever a transition is attempted at a gene, code representing a conductor size of the infeed edge of a node is changed to one of the following.

- bigger than or at the least equal to that of its child of maximum size and
- lower than or at the most equal to that of its parent.

Thus, though the transitions are performed randomly, the above stated limitations help produce feasible solution strings. A transition produces a neighbor solution to the current solution. Each solution string is evaluated using (2.21) to obtain its objective (cost) function value. If the objective (cost) function value of neighbor solution is less than that of the current solution, the neighbor solution is accepted and updated as the current solution. Otherwise the neighbor solution is either accepted or rejected based on the Metropolis criterion. The temperature, which is the control parameter used to
determine this decision is reduced at a constant rate after searching through \( N_t \) number of iterations at a particular value. ND transitions constitute an iteration. The process of annealing is repeated till one of the following convergence criteria is met:

- Temperature reaches a specified minimum level.
- Until the current best solution is not improved for prespecified number of consecutive temperatures (NEPS).

Best values of the SA parameters, \( N_t \) and the temperature reduction factor \( \beta \) are obtained. Investigations on the average of objective function values of the test solutions and of the average solution time over certain number (10 runs) of runs gives the best values of these parameters. Best values are the ones that produce reasonably good solutions (sub-optimal) at lesser solution times. Several of runs provide a poor savings due to random behavior of the algorithm. Analyses on the results obtained for these parameters and the implementation aspects are discussed in section-6.4 and presented to highlight the working of the SA algorithm.

6.3.3 SA Algorithm

Definition of SA Parameters and Variables

\[
\begin{align*}
\text{LSTG} & \quad \text{- length of the solution string} \\
N_t & \quad \text{- number of iterations at a temperature} \\
\beta & \quad \text{- temperature reduction factor} \\
T_0 & \quad \text{- initial temperature} \\
T & \quad \text{- current temperature} \\
T_{MIN} & \quad \text{- minimum temperature} \\
\text{BIGM} & \quad \text{- a large enough constant value} \\
[\text{INITX}]_m & \quad \text{- initial solution, } m=1 \text{ to LSTG} \\
[\text{CURRENTX}]_m & \quad \text{- solution at the current transition, } m=1 \text{ to LSTG} \\
[\text{TRIALX}]_m & \quad \text{- neighbor solution of the current solution, } m=1 \text{ to LSTG} \\
[\text{OPTIMX}]_m & \quad \text{- best solution obtained thus far in the SA process, } m=1 \text{ to LSTG} \\
\text{INITF} & \quad \text{- objective function value } C \text{ of the initial solution} \\
\text{CURRENTF} & \quad \text{- objective function value } C \text{ of the current solution}
\end{align*}
\]
TRIALF - objective function value $C$ of the neighbor solution
OPTIMF - minimum objective function value obtained thus far in the SA process
Ndown - number of downhill moves at a temperature
Nup - number of uphill moves at a temperature
Nrej - number of uphill moves rejected at a temperature
Nnew - number of moves that fetched new minimum values at a temperature
FSTAR - counter to vector FSTAR, $i = 1, \ldots, NEPS$
- a vector of final function values used to decide upon termination and of size NEPS
iter - counter for $N_t$
iNode - counter to the solution string, $iNode = 1$ to $ND$
EPS - error tolerance for termination
METROPOLIS = 1 if a worse move is accepted. 0 otherwise.

Step 1: Obtain the initial temperature $T_0'$ by computing $\Delta C$ of random uphill moves i.e. a transitions that results in $\Delta C = C(t + 1) - C(t) > 0$. Initialize the SA parameters and static data of the system.

Step 2: Read the feeder and the load data and construct the feeder:
Initialize each node,
- with load data
- with its infeed edge data, its parent and children nodes

Step 3: Read conductor data - size-id, area of cross section, cost, reactance per km etc.

Step 4: Initialize
$$T' \leftarrow T_0'$$
Naccp $\leftarrow 0$
$$FSTAR_i \leftarrow 1 \times 10^{20}; \quad i = 1 \text{ to } NEPS$$
$$BIGM \leftarrow 1 \times 10^{20}$$

Step 5: Generate initial solution string $[INITX]_m$.

Step 6: Decode $[INITX]_m$ and obtain INITF.
- get the decision variables $b_i, i = 1 \text{ to } m$. 
- register the size of feeder segments coded into each of them into the data modules of the respective nodes.
- compute investment cost and loss cost of lines and feeder voltage drop
- compute objective function value $C$ using equation (2.21)

$\text{INITF} \leftarrow C$

**Step 7:** If $0 < \text{INITF} < \text{BIGM}$ then,

Assign

$$[\text{CURRENTX}]_m \leftarrow [\text{INITX}]_m$$

$$[\text{OPTIMX}]_m \leftarrow [\text{INITX}]_m$$

$\text{CURRENTF} \leftarrow \text{INITF}$

$\text{OPTIMF} \leftarrow \text{INITF}$

$FSTAR_{1i} \leftarrow \text{INITF}$ and go to next step.

Else, Go to Step-5.

**Step 8:** Set $Nup \leftarrow 0$; $Ndown\leftarrow 0$;

$Nnew\leftarrow 0$; $Nrej \leftarrow 0$;

**Step 9:** Set $\text{iter} \leftarrow 1$;

**Step 10:** Initialize $\text{iNode} \leftarrow 0$

**Step 11:** $\text{iNode} \leftarrow \text{iNode} + 1$

If $\text{iNode} > \text{ND}$ then go to Step-17.

Else go to next step.

**Step 12:** *Transition to neighbor solution:*

Generate $[\text{TRIALX}]_m$ a neighbor solution to $[\text{CURRENTX}]_m$ by introducing a random change $\text{iNode}^{\text{th}}$ element, $[\text{CURRENTX}]_{\text{iNode}}$ as discussed in section-6.3.2 of this chapter.

**Step 13:** *Decode $[\text{TRIALX}]_m$ and obtain $\text{TRIALF}$.***

- get the decision variables $b_i, i = 1$ to $m$.
- register the size of feeder segments coded into each of them into the data modules of the respective nodes.
- compute investment cost and loss cost of lines and feeder voltage drop
- compute objective function value $C$ using equation (2.21)

$\text{TRIALF} \leftarrow C$
Step 14: If TRIALF \leq CURRENTF and TRIALF > 0 accept TRIALX

\begin{align*}
&[\text{CURRENTX}]_m \leftarrow [\text{TRIALX}]_m \\
&\text{CURRENTF} \leftarrow \text{TRIALF} \\
&\text{Ndown} \leftarrow \text{Ndown} + 1 \\
&\text{Naccp} \leftarrow \text{Naccp} + 1 \\
\end{align*}

Display the new solution TRIALX, TRIALF and go to next step.

Else, Go to Step-16.

Step 15: If TRIALF \leq OPTIMF and TRIALF > 0 then

\begin{align*}
&[\text{OPTIMX}]_m \leftarrow [\text{TRIALX}]_m \\
&\text{OPTIMF} \leftarrow \text{TRIALF} \\
&\text{Nnew} \leftarrow \text{Nnew} + 1 \\
\end{align*}

Display the new optimum solution OPTIMX and OPTIMF.

Go to Step-11.

Step 16: Neighbor solution TRIALX is an uphill move. Hence use Metropolis criterion (go to METROPOLIS subroutine) to decide on acceptance.

If (METROPOLIS = 1), then accept TRIALX.

\begin{align*}
&[\text{CURRENTX}]_m \leftarrow [\text{TRIALX}]_m \\
&\text{CURRENTF} \leftarrow \text{TRIALF} \\
&\text{Nup} \leftarrow \text{Nup} + 1 \\
&\text{Naccp} \leftarrow \text{Naccp} + 1 \\
\end{align*}

Else reject TRIALX

\text{Nrej} \leftarrow \text{Nrej} + 1 and Go to Step-11.

Step 17: \text{iter} \leftarrow \text{iter} + 1

If \text{iter} > N_t go to next step

Else go to Step-8.

Step 18: Check termination criteria:

\begin{align*}
&\text{QUIT} \leftarrow \text{FALSE} \\
&\text{FSTAR}_t \leftarrow \text{CURRENTF} \\
\text{If } (\text{OPTIMF} - \text{FSTAR}_t \leq \text{EPS}) \text{ then, } \text{QUIT} \leftarrow \text{TRUE} \\
\text{If } (|\text{CURRENTF} - \text{FSTAR}_i| > \text{EPS}) \text{ for } i = 1, \ldots, \text{NEPS}, \text{ then QUIT} \leftarrow \text{FALSE} \\
&\text{T}' \leftarrow \text{T}' \ast \beta \\
\text{If } \text{T}' \leq \text{T}'_{\text{MIN}} \text{ then QUIT} \leftarrow \text{TRUE} \\
\end{align*}

Step 19: Terminate SA if appropriate:

If (QUIT = TRUE) then [\text{CURRENTX}]_m \leftarrow [\text{OPTIMX}]_m
Else (i.e. if termination criteria is not met, prepare for another loop)

\[ FSTARI_i \leftarrow FSTARI_{i-1} \quad i = NEPS, \ldots, 2 \]
\[ CURRENTF \leftarrow OPTIMF \]
\[ [CURRENTX]_m \leftarrow [OPTIMX]_m \] and go to Step-9.

**Step 20:** Display the following:

Minimum function value (optimum): OPTIMF
Optimum solution: [OPTIMX]_m
Total number of moves: Ndown + Nup + Nrej
Number of downhill moves: Ndown
Number of uphill moves: Nup
Number of uphill moves rejected: Nrej

**Step 21:** STOP

/*****************************/

METROPOLIS decision making on uphill move

**Step 1:** METROPOLIS \leftarrow 0.

**Step 2:** Compute \[ P = e^{\frac{(TRIALF - CURRENTF)}{T}} \]

**Step 3:** Generate a random number \( PP \) between 0 and 1

**Step 4:** If \( PP < P \) then \( METROPOLIS \leftarrow 1 \)

**Step 5:** STOP

/*****************************/
Flow Chart of the Proposed SA Algorithm

START

Initialize SA parameters: $\beta$, $N$, $T'_{MIN}$, $T'_0$, EPS, NEPS, vector $FSTAR$, Naccp etc.
Initialize system parameters: MAX_NEIGH, ND etc.

Read the feeder-conductor data. (area of c.s., cost, parameters etc.)

Read node data (load, infeed edge, parent and children nodes). Initialize the nodes and construct the feeder.

Generate initial solution $[INITX]_m$

Decode $[INITX]_m$. Record the conductor size of each feeder section in the data module of respective nodes and compute objective function $C$.
TRIALF $\leftarrow C$

NO

Is $[INITX]_m$ feasible?

YES

Record INIT values as CURRENT value, OPTIM value and as first element of final-function-value vector $FSTAR$. i.e. as $FSTAR_1$
130

A

B

C

Generate neighbor solution $[\text{TRIALX}]_m$ at $iNode$

Decode $[\text{TRIALX}]_m$ and obtain $\text{TRIALF}$

Downhill move?

Is uphill move acceptable as per *Metropolis* criterion?

Record $\text{TRIAL}$ values as $\text{CURRENT}$ values

Is $\text{TRIALX}$ a new optimum?

Record $\text{TRIAL}$ values as $\text{OPTIM}$ values.
Display the new optimum

iter $\leftarrow 1$

$iNode \leftarrow 1$

NO

YES

NO

YES

NO

YES

NO

YES

NO

YES
Assign OPTIM values to CURRENT values. Update last 3 elements of FSTAR_i vector with their preceding elements.

Display optimum cost OPTIMF and optimum solution [OPTIMX]m

STOP
6.4 APPLICATION

The proposed SA procedure for the Conductor Gradation Problem was tested on the same feeder shown in fig-4.1 of chapter-4, which was used to test the proposed GA procedure. It is a 11kV primary distribution feeder with 13 consumer points taken from Tiruchirappalli district, Tamil Nadu. Length of the feeder is 21.89 km and is operated with a total peak load of 1070.9 kW at a power factor of 0.8. All the feeder segments are of ‘Rabbit’ conductors. Three different conductor types namely ‘Rabbit, Weasel, Squirrel’ have been considered (see Table-b.1 of Appendix-B for conductor data). Power loss of the feeder before grading was found to be 49.85kW and the energy loss over a planning period of 15 years will be 8443.07kWh. Voltage drop along the feeder has been 7.31%. Other constants specific to the problem are taken from the works reported in [38] and [40].

The initial temperature $T_0$, for the problem was obtained as $T_0 = 176616.21$. Most suitable values for the number of transitions at a temperature level $N_t = 25$ and the temperature reduction factor $\beta = 0.65$ are obtained. The problem is solved ten times for each of the following values of $\beta$: 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 0.95 and 0.99 fixing the other parameters at $N_t = 25, T_0 = 176616.21$ and $\text{NEPS} = 4$. The best values of $N_t$ and $\beta$ are justified in terms of both the solution time and quality of solution, i.e. the average of the final solutions obtained over 10 runs with $\beta = 0.65$ was very close to the optimum value and was obtained in an average solution time lesser than that needed for other values of the parameter. For values of $\beta$ less than 0.65, the gap between the average of solutions obtained and the optimum solution was larger. For values of $\beta$ between 0.65 and 0.8, the average cost obtained was close to the optimum value but the average solution time increases slowly. For values of $\beta$ larger than 0.8, solution time increases exponentially. Some of the runs resulted in solutions whose distances are 0.33% to 0.5% from the optimal solution and obtained in a solution time of 195 seconds. Optimal solution was obtained consistently for values of $\beta$ above 0.8 in a solution time greater than over 500 seconds.

To get the best value for $N_o$, the problem is solved ten times for each of the following values of $N_o$: 10, 15, 20 25, 30, 40, 50 and 60, fixing the other parameters at $\beta = 0.65, T_0 = 176616.21$ and $\text{NEPS} = 4$. As expected the average solution time increases linearly with $N_o$. For values of $N_o$ ranging from 25 to 60, the average cost was close to the
optimum cost. Hence, a value \( N_r = 25 \) is justified in terms of both solution time and quality of solution.

Power loss of the feeder after grading increased to 51.23 kW (an increase of 2.8% with respect to the loss before grading) and the energy loss over a planning period of 15 years increased to 8676.01 kWh. Voltage drop along the feeder after conductor gradation has increased to 7.91% from 7.31% but has resulted in a cost saving of Rs. 87989.78. The feeder of fig-4.1 with optimal conductor sizes is shown in fig-6.2. Cost of the optimum solution is Rs. 925530.95.

![Solution feeder with optimally graded conductors](image)

**Fig-6.2 Solution feeder with optimally graded conductors**

**Numerical computation:**

The step-by-step results obtained during a typical run of proposed SA algorithm, at an intermediate temperature level, applied onto the feeder shown in fig-4.1, and also the computation of initial temperature are presented below.

i) **Initial temperature:** Cost of the solution strings of about 20 random uphill transitions are manually computed for obtaining the initial temperature \( T_0 \), a key parameter to be input for the SA procedure.

1. Cost of solution string-1 = 1193724.88 (Rs)  
   Cost of neighbor solution = 1131353.43  
   Difference of the transition = 62371.45
2. Cost of solution string-2 = 940118.42
   Cost of neighbor solution = 956812.71  Difference of the transition = 16694.29
3. Cost of solution string-3 = 1185127.74
   Cost of neighbor solution = 1219372.42  Difference of the transition = 34244.68
4. Cost of solution string-4 = 1454021.83
   Cost of neighbor solution = 1540037.23  Difference of the transition = 86015.40
5. Cost of solution string-5 = 970225.81
   Cost of neighbor solution = 984567.27  Difference of the transition = 14341.46
6. Cost of solution string-6 = 1008073.01
   Cost of neighbor solution = 1020166.19  Difference of the transition = 12093.18
7. Cost of solution string-7 = 1315693.00
   Cost of neighbor solution = 1451911.40  Difference of the transition = 136218.40
8. Cost of solution string-8 = 961141.27
   Cost of neighbor solution = 975482.73  Difference of the transition = 14341.46
9. Cost of solution string-9 = 939281.68
   Cost of neighbor solution = 946486.23  Difference of the transition = 7204.55
10. Cost of solution string-10 = 933788.81
    Cost of neighbor solution = 937101.68  Difference of the transition = 3312.87
11. Cost of solution string-11 = 933032.87
    Cost of neighbor solution = 947687.91  Difference of the transition = 14655.04
12. Cost of solution string-12 = 1131353.43
    Cost of neighbor solution = 1191614.45  Difference of the transition = 60261.02
13. Cost of solution string-13 = 983857.02
    Cost of neighbor solution = 991817.49  Difference of the transition = 7960.47
14. Cost of solution string-14 = 967741.58
    Cost of neighbor solution = 971054.45  Difference of the transition = 3312.87
15. Cost of solution string-15 = 954459.87
    Cost of neighbor solution = 971154.16  Difference of the transition = 16694.29
16. Cost of solution string-16 = 970225.82
    Cost of neighbor solution = 1011410.30  Difference of the transition = 41184.48
17. Cost of solution string-17 = 987575.91
    Cost of neighbor solution = 1009541.40  Difference of the transition = 21965.49
18. Cost of solution string-18 = 959781.10
    Cost of neighbor solution = 967741.58  Difference of the transition = 7960.48
19. Cost of solution string-19 = 990899.02
Cost of neighbor solution = 997310.38  Difference of the transition = 6411.36

20. Cost of solution string-20 = 926665.06
Cost of neighbor solution = 933491.43  Difference of the transition = 6826.37

Average of Differences of the transitions = 28703.48. Assuming $A_0 = 0.85,$

$$T_0' = \frac{\Delta C^-}{\ln 1/A_0} = 176616.21$$

After 5 temperature cycles, $T_0'$ reduces to $T' = 20492.61,$ if reduced at the rate of $\beta = 0.65.$ Therefore the temperature for the 6th temperature cycle will be,

$$T' = T' \times 0.65 = 10391.57 \times 0.6 = 13320.20$$

Initialize $T'_{\text{min}} = 0.00$

ii) Initialize EPS = 0.01, NEPS = 4

iii) The optimum cost obtained thus far in the process is Rs.991817.49. Then,

OPT $\leftarrow$ 991817.49

iv) Initialize the vector FSTAR of size 4 with a larger constant initially. The vector is filled with the current solution obtained at the end of $N_t$ iterations of each temperature by FIFO (First In First Out) order. The vector is filled during the last four temperature cycles of the SA process as shown below:

<table>
<thead>
<tr>
<th>FSTAR$_1$</th>
<th>997310.38</th>
</tr>
</thead>
<tbody>
<tr>
<td>FSTAR$_2$</td>
<td>991817.49</td>
</tr>
<tr>
<td>FSTAR$_3$</td>
<td>1013253.91</td>
</tr>
<tr>
<td>FSTAR$_4$</td>
<td>1109522.57</td>
</tr>
</tbody>
</table>

**Stepwise results of SA process at a temperature level**

**Step1:** The current solution, SOL1 is:

SOL1 $\leftarrow$ 33333222211111
The objective (cost) function value of SOL1 obtained (2.21) = 971054.45

\[ C \leftarrow 971054.45 \]

\[ \text{[CURRENTX]_m} \leftarrow \text{SOL1} \]

The current cost is less than the optimum value obtained thus far in the run. Therefore, update optimum value also.

\[ \text{OPT} \leftarrow 971054.45 \]

\[ i\text{Node} \leftarrow 1 \]

**Step2:** Attempts to introduce changes of g1 to g4 are not successful, since they resulted in infeasible solutions. \(i\text{Node}\) is incremented at each step.

**Step3:** The neighbor solution (SOL2) generated by attempting a random change in

\[ \text{[CURRENTX]_m}, m = i\text{Node} \] i.e. g5 is,

\[ \text{SOL2} \leftarrow 3333222211111 \]

The cost of SOL2 = 995130.37

\[ C \leftarrow 995130.37 \]

The cost of SOL2 is greater than that of SOL1. i.e. \(\Delta C > 0\). Hence, Metropolis criterion is used to decide on the new solution. Compute

\[ \Delta C = 995130.37 - 971054.45 = 24075.92 \]

\[ P(\text{SOL2}) = e^{-\frac{\Delta C}{kT}} = e^{-\frac{24075.92}{1x13320.20}} = 0.164068 \]

A random number generated between (0-1) is \(r = 0.5\). Since \(P(\text{SOL2}) < r\), the solution SOL2 is not accepted. Therefore, proceed to the next step.

\[ i\text{Node} = i\text{Node} + 1 \]

**Step4:** The neighbor solution (SOL3) generated by attempting a random change in the

\[ \text{[CURRENTX]_m}, m = i\text{Node} \] i.e. g6 which incremented the code representing the conductor size from 2 to 3 is,

\[ \text{SOL3} \leftarrow 3333332211111 \]

The cost of SOL3 = 958961.27

\[ C \leftarrow 958961.27 \]

The cost of SOL3 is less than that of SOL1. i.e. \(\Delta C < 0\). Hence, SOL3 is accepted.
The current cost is less than the optimum value obtained thus far in the run. Therefore, update optimum value also.

\[ \text{OPT} \leftarrow 958961.27 \]

\[ iNode = iNode + 1 \]

**Step5:** Attempt to introduce change of \( g7 \) is not successful, since it resulted in infeasible solution. (random change decremented the code representing conductor size from 2 to 1).

\[ iNode = iNode + 1 \]

**Step6:** Attempt to introduce change of \( g8 \) is not successful, since it resulted in infeasible solution (random change incremented the code representing conductor size from 2 to 3).

\[ iNode = iNode + 1 \]

**Step7:** The neighbor solution (SOL4) generated by attempting a random change in the \([\text{CURRENTX}]_m, m = iNode\) i.e. \( g9 \) which incremented the code representing the conductor size from 1 to 2 is,

\[ \text{SOL4} \leftarrow 333332221111 \]

The cost of \( \text{SOL4} = 955648.39 \)

\[ C \leftarrow 955648.39 \]

The cost of \( \text{SOL4} \) is less than that of \( \text{SOL3} \). i.e. \( \Delta C < 0 \). Hence, \( \text{SOL4} \) is accepted.

\[ [\text{CURRENTX}]_m \leftarrow \text{SOL4} \]

The current cost is less than the optimum value obtained thus far in the run. Therefore, update optimum value also.

\[ \text{OPT} \leftarrow 955648.39 \]

\[ iNode = iNode + 1 \]

**Step8:** The neighbor solution (SOL5) generated by attempting a random change in the \([\text{CURRENTX}]_m, m = iNode\) i.e. \( g10 \) which incremented the code representing the conductor size from 1 to 2 is,
SOL5 ← 3333332222211
The cost of SOL5 = 947687.92
C ← 947687.92
The cost of SOL5 is less than that of SOL4. i.e. ΔC < 0. Hence, SOL5 is accepted.

[CURRENTX]_m ← SOL5
The current cost is less than the optimum value obtained thus far in the run. Therefore, update optimum value also.
OPT ← 947687.92

iNode = iNode + 1

Step9: The neighbor solution (SOL6) generated by attempting a random change in the
[CURRENTX]_m, m = iNode i.e. g11 is,
SOL6 ← 3333332222211
The cost of SOL6 = 954729.92
C ← 954729.92
The cost of SOL6 is greater than that of SOL5. i.e. ΔC > 0. Hence, Metropolis criterion is used to decide on the new solution. Compute

ΔC = 954729.92 - 947687.92 = 7042.00

\[ P(SOL6) = e^{-\frac{\Delta C}{kT}} = e^{-\frac{7042.00}{13320.20}} = 0.589388 \]

A random number generated between (0-1) is r = 0.4. Since P(SOL6) > r, the solution SOL6 is accepted.

[CURRENTX]_m ← SOL6
iNode = iNode + 1

Step10: The neighbor solution (SOL7) generated by attempting a random change in the
[CURRENTX]_m, m = iNode i.e. g12 is,
SOL7 ← 3333332222221
The cost of SOL7 = 961141.28
C ← 961141.28
The cost of SOL7 is greater than that of SOL6. i.e. ΔC > 0. Hence, Metropolis
criterion is used to decide on the new solution. Compute

$$\Delta C = 961141.28 - 954729.92 = 6411.36$$

$$\Pr(SOL7) = e^{-\Delta C / kT'} = e^{-6411.36 / 1 \times 13320.20} = 0.617963$$

A random number generated between (0-1) is $r = 0.9$. Since $\Pr(SOL7) < r$, the solution SOL7 is not accepted. Therefore, proceed to the next step.

$iNode = iNode + 1$

**Step11:** Attempt to introduce change of g13 is not successful, since it resulted in infeasible solution (random change incremented the code representing conductor size from 1 to 2).

$iNode \leftarrow 1$

**Step12:** Repeat Step2- Step11 for $N_t$ times.

**Step13:** Update the vector $FSTAR$ as given below:

<table>
<thead>
<tr>
<th>$FSTAR_1$</th>
<th>961141.28</th>
</tr>
</thead>
<tbody>
<tr>
<td>$FSTAR_2$</td>
<td>997310.38</td>
</tr>
<tr>
<td>$FSTAR_3$</td>
<td>991817.49</td>
</tr>
<tr>
<td>$FSTAR_4$</td>
<td>1013253.91</td>
</tr>
</tbody>
</table>

Update temperature $T' \leftarrow T' \times \beta$

**Step14:** Check the termination criteria.

If the cost of the current solution SOL6 differs from each of the four $FSTAR$ values by a value less than or equal to EPS (OR)

If the temperature $T' \leq T'_{min}$, go to Step-16.

Else go to next step.

**Step15:** SOL1 $\leftarrow$ SOL6

$[CURRENTX]_m \leftarrow$ SOL1 and go to Step-2.

**Step16:** STOP
The proposed SA procedure produces the same optimum solution as that obtained using GA approach for the test feeder. The solution string and the cost of the optimal solution have been obtained as ‘333333332111’ and Rs.925530.95 respectively. The algorithm halted when the FSTAR vector corresponding to the four (NEPS = 4) latest temperatures was obtained as given below.

<table>
<thead>
<tr>
<th>FSTAR_i</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>FSTAR_1</td>
<td>925530.95</td>
</tr>
<tr>
<td>FSTAR_2</td>
<td>925530.95</td>
</tr>
<tr>
<td>FSTAR_3</td>
<td>925530.95</td>
</tr>
<tr>
<td>FSTAR_4</td>
<td>925530.95</td>
</tr>
</tbody>
</table>

6.5 CONCLUSION

SA procedure yields the same solution for the test feeder as that obtained using GA approach. The SA approach was found to be effective in producing the optimum solution but at substantially higher solution times. The algorithm needs to be run several times to get consistent results for lesser values of $\beta$. For values of $\beta$ above 0.8, optimum was obtained consistently but the solution time increases exponentially. The best values are justified in terms of both the solution time and quality of solution. Best values for the SA parameters were obtained and presented.