Otto-von-Guericke-University Magdeburg
Faculty of Electrical Engineering and Information Technology
Chair for Automation/Modelling

Power Systems Control and Optimization

Project 7 Report

Unit Commitment using Dynamic Programming

Zohaib Ahmed Masood (224366)
Shubhajit Biswas (224028)
Arman Ahmed Khan (230601)
Akshat Parasher (223818)

Introduction

In a power system, there are several numbers of generating units that are used to produce electricity. In any instance, the total load of the system is met by different generating units in different power plants. A power system has a continuously changing power consumption and loads at different times. These loads can vary throughout the day and has various peak values on different days. These changing load requirements are to be managed and optimized by committing a specific number of generating units to the system, to optimize the cost of the power system.

Economic dispatch is the process of power plants to produce energy at the lowest cost to satisfy consumers while Unit Commitment (UC) is a process which looks for the most economical and cost-efficient way for power generation when the power consumption at different times and different constraints of the power plants are considered. Unit Commitment is one of the most important parts of every power system planning project.

For a varying load during a day, different combinations of the generating units are to be enabled to meet the load requirement. It all has to be decided in advance that which generating units are to be engaged when the load increases. Similarly, when the load decreases, disengaging the generating units is also to be known in advance. The sequence of enabling and disabling the generating units is to be decided in such a way that the cost incurred is minimum. This optimum sequence of enabling and disabling generating units is called Unit Commitment. The Unit Commitment problem is also Economic Dispatch over a day. The period taken into account may vary too.

One of the solutions to meet the load requirement is to commit enough generating units and keep them enabled. However, it will be very costly when the load is not large enough. To reduce the cost, some of the generating units are to be shut down when not required. Unit Commitment is a very difficult optimization problem since there is a huge number of possible combinations of enabled and disabled generating units in the power system. There are many ways to solve the Unit Commitment problem. In this task, we will be using Dynamic Programming to solve the Unit Commitment problem.

Dynamic Programming

Dynamic programming (DP) is a conventional algorithm used to solve the deterministic UC problem. Dynamic programming is a methodical procedure in which different decisions of turning on and off the generating units is divided into multiple steps or stages. It systematically evaluates a large number of possible decisions cost-effectively. Each stage is considered to be a point in time and when the load varies (different point in time), it has to decide to jump to a different stage bearing the minimum cost.

The whole algorithm of Dynamic Programming is based on dividing the complete problem into stages and taking a path to the next stage (different load setting) in the most cost-effective manner and hence optimizing the Unit Commitment problem. At each stage, a policy decision is needed. When a policy decision is made at each state, let's say stage i, it takes it to the next state *i*+1. Every state is an independent state and does not depend on any of the previous stages (Principal of optimality). Dynamic programming has a recursive solution procedure and can be given as follows in Eq.1:

$$J_{n*}(s) = min_{xn}(c(s,x_n) + J_{n+1*}(x_n)) \text{ (Eq.1)}$$

At any stage *n*, we have to look for the minimum cost incurred at that stage *n* and account for the cost incurred in the next stage as well. This recursive solution can be expanded to calculate the cost incurred at each stage. For example, a problem having four stages and starting at point A can be expanded as follows:

$$J_{1*}(A) = min_{x1}(c(A,x_1) + J_{2*}(x_1))$$
$$J_{2*}(x_1) = min_{x2}(c(x_1,x_2) + J_{3*}(x_2))$$
$$J_{3*}(x_2) = min_{x3}(c(x_2,x_3) + J_{4*}(x_3))$$
$$J_{4*}(x_3) = min_{x4}(c(x_3,x_4) \text{ (Eq.2)}$$

Dynamic programming can be done both ways, forward and backward. In both cases, the shortest and the optimal path will always be the same. The unit commitment problem can also be given as follows in Eq.3:

$$minimize_{\alpha i,k\ ,Pi,k} \Sigma_{k=1T} \Sigma_{i=1N}\ \alpha_{i,k} c_i(P_{i,k}) \text{ (Eq.3)}$$

Here we are minimizing the cost of power generation by selecting the units having minimum costs and the number of plants is denoted by *N.* The number of stages or the time of the next stage is denoted by *T*. Activation schemes of the power plants is represented with α. The cost is directly proportional to the generated power *P.* The load balance equation can be given as follows in Eq.4:

$$P_{load,k} - \Sigma_{i=1N}\ \alpha_{i,k} P_{i,k} = 0 \text{ (Eq.4)}$$

Here the equation shows that the generated power should be equal to the load demand. And the power generated by each plant can be shown as follows:

$$\alpha_{i,1} P_{i,min} \leq P_{i,1} \leq \alpha_{i,1} P_{i,max}$$

$$for\ k=1,\ldots,T\ and\ i=1,\ldots,N$$

When we are starting from a state which can be considered as the initial state or the first state, we still have to solve the Economic dispatch problem (EDP). Which means that we still have to

consider the cost of the first stage and minimize it. The equation to solve the EDP of the first state can be given as follows in Eq.5:

$$\boldsymbol{minimize}_{,P_{i,1}} \sum_{i=1}^{N} \alpha_{i,1} c_i(P_{i,1}) \textbf{ (Eq.5)}$$

Subject to $P_{load,k} - \sum_{i=1}^{N} \alpha_{i,1} P_{i,1} = 0$

For $\alpha_{i,1} P_{i,min} \leq P_{i,1} \leq \alpha_{i,1} P_{i,max}$

So whenever we move from the initial state to the next state, we consider the cost of the new stage and add the cost of the initial stage. The general formula for the *k* number of stages can be shown by the following equation Eq.6:

$$J_k(\alpha_k) = EDP\ (\alpha_k) + J_{k+1}(\alpha_{k+1}) \textbf{ (Eq.6)}$$

Dynamic Programming can be very challenging when the number of stages is increased since every node at a stage is a combination of multiple generating units. A power system can have an astronomical number of combination of generating units making it a very complex calculation which can take a lot of time and memory in getting an optimal solution. To reduce the complexity of the problem, we can add different constraints to simplify the calculations. This can be done by different methods e.g. excluding activation schemes. There are other ways to reduce the complexity of the problem like evaluating only the n-best activation schemes and saving only the m-best strategies but these solutions may introduce sub-optimality and reduce the chances of finding an optimal solution.

Project MATLAB Code

The codes are explained by commenting on each line of the code.

1. Dynamic Programming

```
M = 3; % number of units

Pload = 280*[0.5 0.53 0.55 0.53 0.5 0.54 0.7 0.9 0.95 1.1 1.2 1.4 1.7 1.65 1.5 1.3 1.0 0.9 0.8 0.5 0.54]';   %
N = length(Pload); %number of time steps

Pmin = [50; 37; 25];      %minimum power of generation units
Pmax = [200; 150; 140];   %maximum power of generation units
%% Forward dynamic programming
aopt = zeros(M,N);  % M*N various combinations of unit selection initialization
Popt = zeros(M,N);  % M*N optimised power initialization
tic                 % start counting the time of execution
k = 0;              % start counting the number of steps in execution
for i = 1:N         % N represents the total number of time-steps
    a = feasible_activation_schemes(Pload(i),Pmin,Pmax);  % a represents the matrix
    %containing activation schemes of generation units with respect to the load demands
    Jmin = Inf;     % initializing Jmin with empty value
    for j = 1:size(a,2) % go through all the activation schemes
        [J,P]= dispatch(a(:,j),Pmin,Pmax,Pload(i));  % getting the values of J cost at P power for each load
        k = k + 1; % counter
        if J<Jmin
            Jmin = J;  % finding the minimum cost
            aopt(:,i) = a(:,j); % activated schemes associated with the Jmin, minimum cost
            Popt(:,i) = P;      % power generated by the activated units
        end
    end
end
t = toc; % end of the timer
```

2. Dispatch problem

```
function [Jopt,Popt] = dispatch(a,Pmin,Pmax,Pload) % defining a function that returns optmised cost and optimised power
  ind = find(a);       % finding the index of active Power plants in a matrix
  M = length(a);       % finding the length of matrix a

  pmin = Pmin(ind); % reduce to a vector containing only one active unit with minimum power

  pmax = Pmax(ind);  % reduce to vector containing only active unit with maximum power
  Aload = ones(1,length(ind)); % fill matrix by ones of size 1*length of active schemes

  J = @(p) cost(a,p); % value function for specific
                      % activation scheme a

  %% solve the scheduling problem for given activation scheme
  p0 = pinv(Aload)*Pload; % initial guess, fulfills equality
                          % constraint

  opt = optimset('Algorithm','interior-point','Display','off','MaxIter',1e5,'MaxFunEvals',1e5);
  [popt,Jopt] = fmincon(J,p0,[],[],Aload,Pload,pmin,pmax,[],opt); % fimincon returns the minimum of constrained nonlinear multivariable function
  Popt = zeros(M,1); % initiate the optimal power matrix
  Popt(ind) = popt; % getting the optimal power in reference to the activated units

end
```

3. Cost function

```
function [J] = cost(a,p) % cost function with inputs power and activated units and output with cost J
    c(1,:) = [0.005 11 200]; % cost function matrix
    c(2,:) = [0.009 10 180]; % cost function matrix
    c(3,:) = [0 15 230];     % cost function matrix
    M = length(a);  % total number of power generating units
    P = zeros(M,1); % initiating power vector
    P(find(a)) = p; % P containing optimal power according to the activated units only
    J = 0;          % initiating cost =0
    for i = 1:M     % looping through all the available power generation units
        if a(i) == 1 % if found any active power generating unit
          J = J + polyval(c(i,:),P(i)); % adding the cost of that activated power generation unit
        end
    end
end
```

4. Feasible activation schemes

```
function a = feasible_activation_schemes(Pload, Pmin, Pmax) % this function returns the combination of power generating units w.r.t load requirements
M = length(Pmin);    % nnumber of power generating units
all = permn([0 1],M); % permutations and combinations with values 0,1 having M columns

N = size(all,1);   % number of rows in 'all' matrix
a = [];            % initializing 'a' matrix
for i = 1:N
    if all(i,:)*Pmin<=Pload && all(i,:)*Pmax>=Pload         % checking which combination of power plants satisfy the given Pmin and Pmax constraints
        a = [a all(i,:)'];                                  % allowing only satisfying combinations out of P1,P2,P3 to generate power as per the requirements of the load
    end
end
end
```

5. Permutation and Combination

```
function [M, I] = permn(V, N)  % PERMN(V,N) - return all permutations
    nV = numel(V); % returns the number of elements in V
    if N == 1
      M = V(:);   % Assigning values of V to M with one column and many rows
      I = (1:nV)'; % giving indexes
    else
      [Y{N:-1:1}] = ndgrid(1:nV);   % 1*N matrix
      I = reshape(cat(N+1, Y{:}), [], N); % R(number of rows)*N, I matrix containing [1,2] values
      M = V(I);  % R(number of rows)*N matrix containing [0,1] values
    end
end
```

6. Graph plotting

```
figure(1)

plot([1:N],Pload,'g')              %represents load demands as given
hold on
scatter([1:N],sum(Popt),'filled')  %total power provided by the sum of generating units
plot([1:N],Popt)                   %represents the optimal power provided by each generation units
hold off
legend('Load demand','power generation')
xlabel('Load Schedule')
ylabel('Real Power,MW')

grid on
grid minor
```

Results:

1. Nominal Load = 330 MW

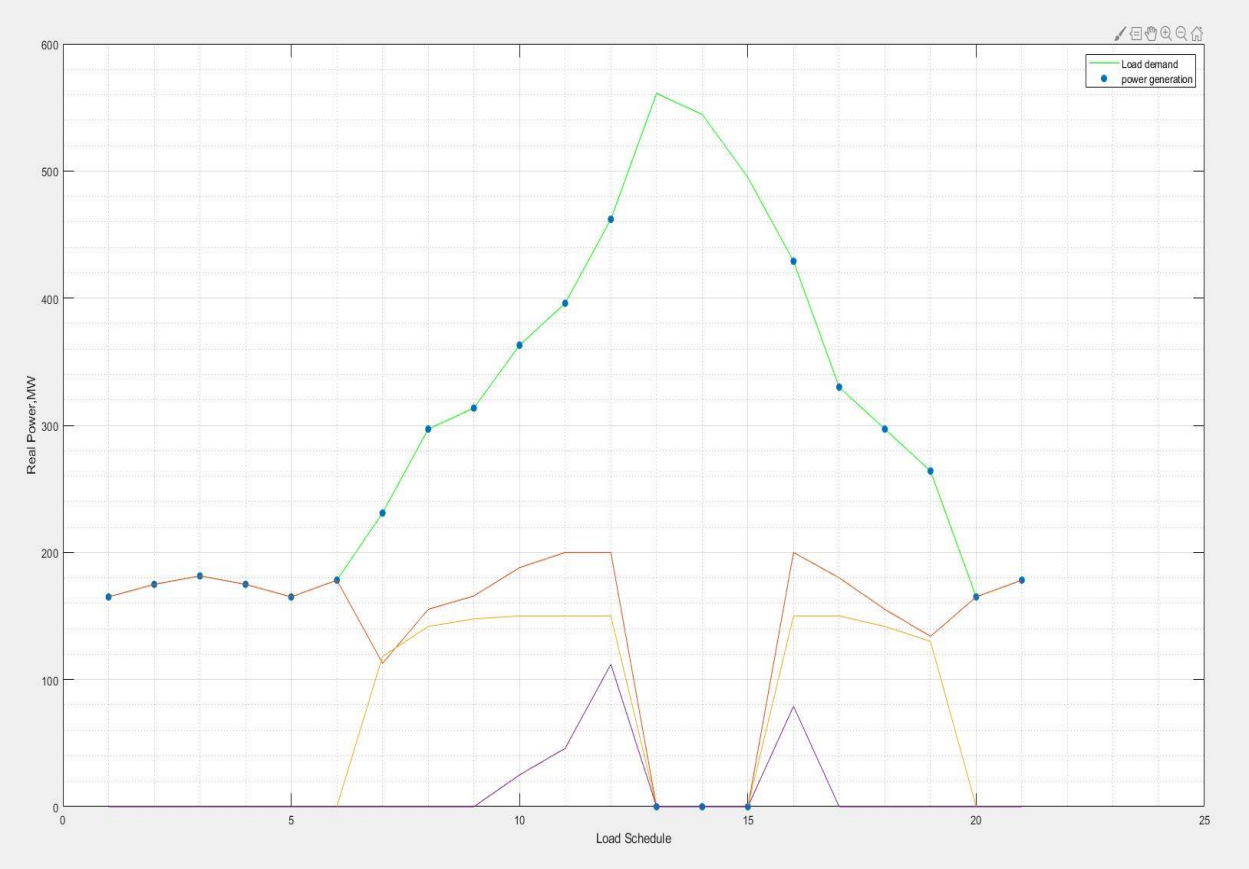
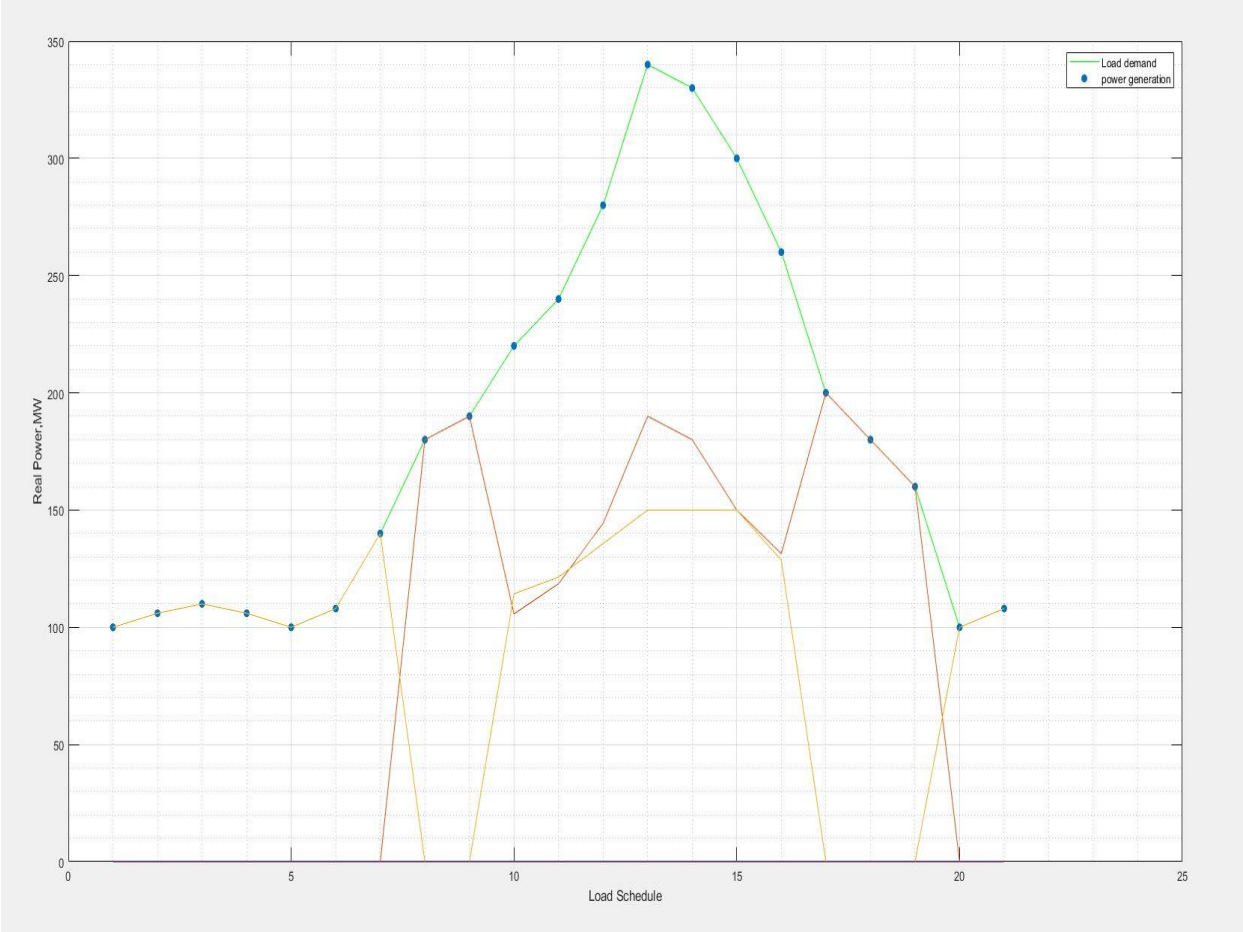


Figure1 - The graph with a nominal load of 330MW

In Figure 1 we can see that the load curve has different values at different point in time. The load demand was met by the available generting units as the load demand kept rising. At time 13 (x-axis), the load demand exceeded the power generation capacity of the power plants and the power plants were shut down as they could not keep up with the rising load demand. When the load demand came down within the generating capacity range, that's when the plants were put back online at time 15 (x-axis).

2. Nominal Load = 200 MW



Figrure2 - The graph with a nominal load of 200 MW

In Figure 2 when the Nominal load was 200 MW, we can see that the generating units followed the load demand curve perfectly as the load demand did not exceed the generating capacity at any point in time.

3. Nominal Load = 280 MW

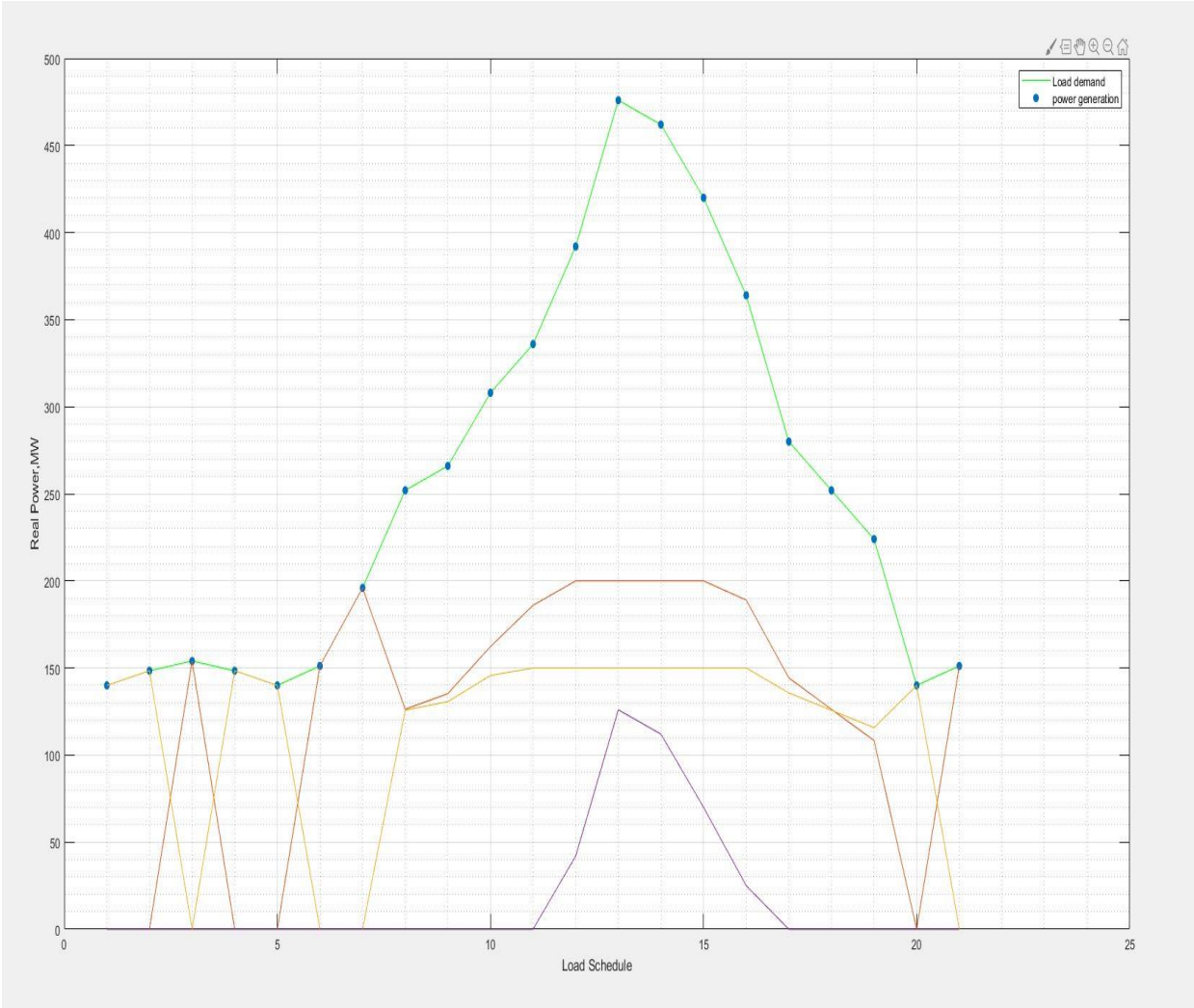


Figure3 - The graph with a nominal load of 280 MW

In Figure 3 when the Nominal load was 280 MW, the generating units also followed the load demand curve perfectly as the load demand here also did not exceed the generating capacity at any point in time.

4. Nominal Load = 400 MW

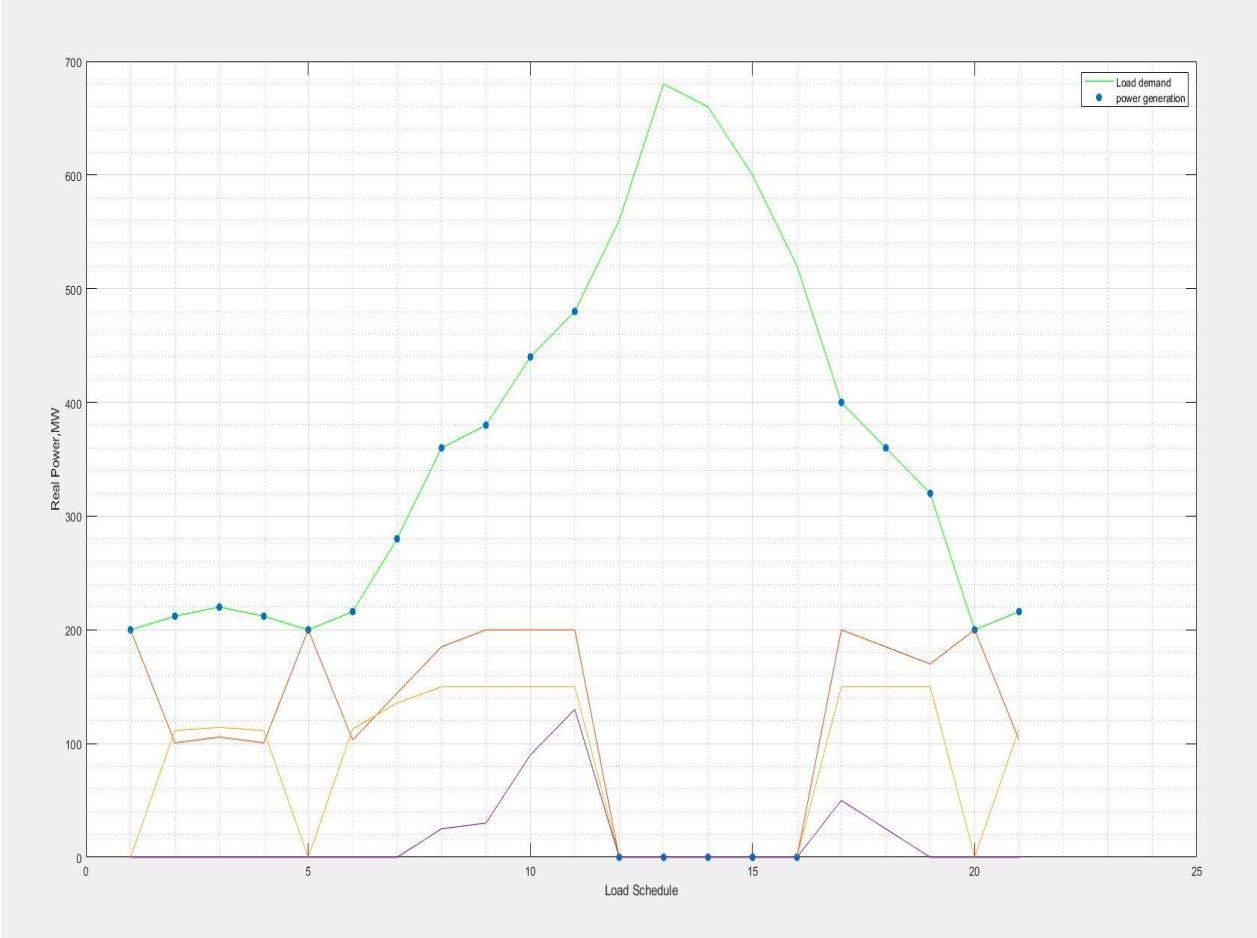


Figure4 - The graph with a nominal load of 400 MW

In Figure 4 the Nominal load is 400 MW and the generating units are following and meeting the load demand until time 12 (x-axis), where the load demand exceeded the power generation capacity of the power plants and the power plants were shut down. The plants were only able to be turned on at time 16 (x-axis) when the load demand came within the generating capacity.

5. Priority list

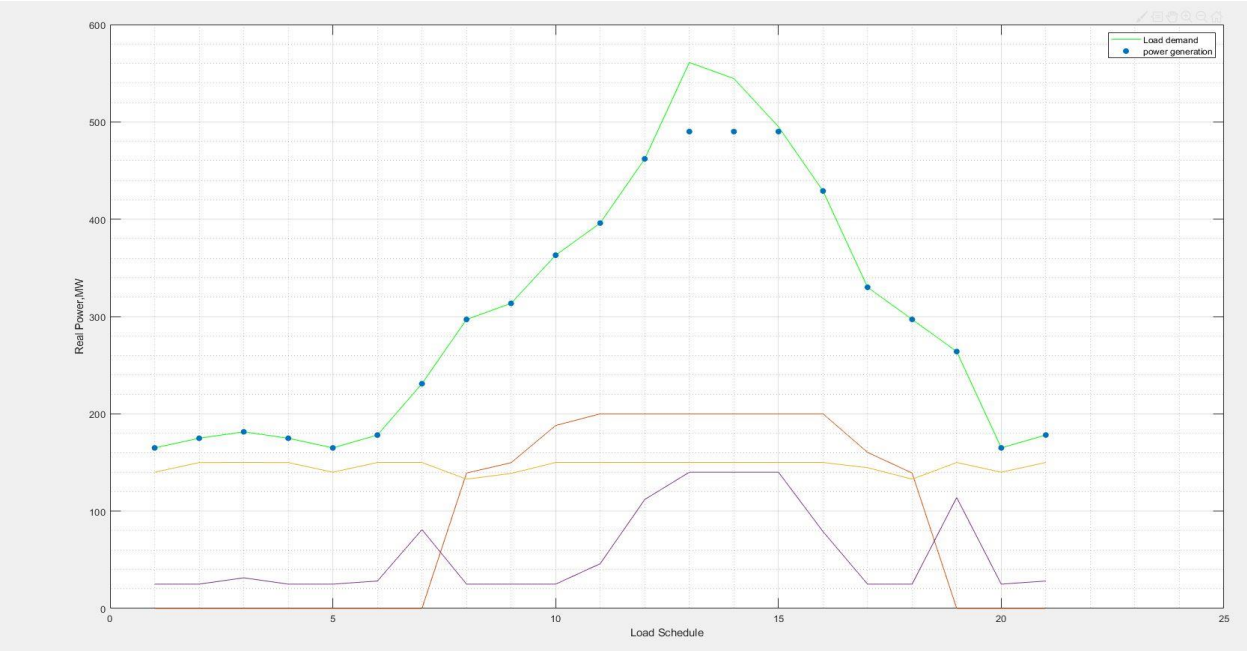


Figure5 - The Priority List graph with a nominal load of 330 MW

Conclusion

We have used dynamic programming to solve the Unit Commitment optimization problem and we have seen that for a nominal load of 330 MW the system has shut down all the power generating units at 13th load schedule point to save the costs. We needed more power generation units to meets the load demands. This is same for a nominal load of 400 MW where system was shutting down all the power generating units at 12th load schedule point.

For Nominal loads of 280 MW and 200 MW the system was working without any problems.

The comparison between Priority List method and Unit Commitment method could be seen from the graphs. In Priority List method the power generating units were not shut down even though the load demands were not met hence this method didn't guarantees optimality. Whereas in dynamic programming method all the three power generating units were shut down once the load demands were not met hence it guarantees optimality.