Unit Commitment using Dynamic Programming

Project 7 Presentation

Presented by:

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

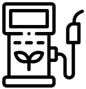
Date of the Presentation: October 13th, 2020

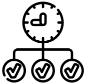
Overview

Unit Commitment

Unit Commitment

Introduction to Unit Commitment:





Credits: Freepik

- Most economical and cost-efficient way for power generation
- Considers power consumption at different times and different constraints of the power plants
- Predicts sequence of enabling and disabling the generating units intelligently
- Minimize the cost of generating units
- Reduces the amount of fuel consumption and hence is more eco-friendly
- Increase the efficiency of the entire process

Unit Commitment

Different methods to achieve Unit Commitment:

- Static method
- Solve for all possible activation schemes
  - Enumeration
  - Priority list

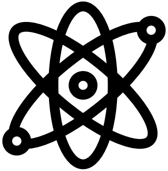
- Dynamic method
- Finding the minimum cost among all the activation schemes
  - Dynamic programming

Dynamic Programming

Dynamic Programming

What is dynamic programming?

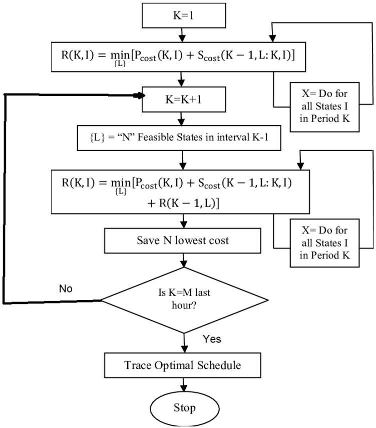
- Conventional algorithm process for optimization
- Divide into multiple steps or stages
- Use recursive solution procedure



Credits: Freepik

Dynamic programming

Algorithm



Unit Commitment using DP

Goals:

- Optimal operating schedule (activation and generation) for a set of three power generating units
- Minimum and maximum power constraints have to be fulfilled
- Solve the unit commitment problem for varying nominal load
- Performance compared to unit commitment using priority lists

Credits: Freepik

Approach:

Dynamic programming has a recursive solution procedure:

$$J_{n^*}(s) = min_{x_n}(c(s,x_n) + J_{n+1^*}(x_n))$$

The unit commitment problem:

$$minimize_{\alpha_{i,k},P_{i,k}} \Sigma_{k=1T}\ \Sigma_{i=1N}\ \alpha_{i,k} c_i(P_{i,k})$$

The load balance equation:

$$P_{load,k} - \Sigma_{i=1N}\ \alpha_{i,k} P_{i,k} = 0$$

The power generated by each plant:

$$\alpha_{i,1} P_{i,min} \leq P_{i,1} \leq \alpha_{i,1} P_{i,max}$$
$$for\ k=1,\ldots,T\ and\ i=1,\ldots,N$$

The equation to solve the EDP of the first state:

$$minimize_{,P_{i,1}} \Sigma_{i=1N}\ \alpha_{i,1} c_i(P_{i,1})$$
Subject to $P_{load,k} - \Sigma_{i=1N}\ \alpha_{i,1} P_{i,1} = 0$
For $\alpha_{i,1} P_{i,min} \leq P_{i,1} \leq \alpha_{i,1} P_{i,max}$

The general formula for the *k* number of stages:

$$J_k(\alpha_k) = EDP\ (\alpha_k) + J_{k+1}(\alpha_{k+1})$$

Matlab code for forward dynamic programming:

```matlab
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

```

Matlab code for dispatch problem:

```matlab
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
```

Matlab code for 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
```

Matlab code for feasible activation schemes:

```matlab
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
```

Matlab code for permutation and combination:

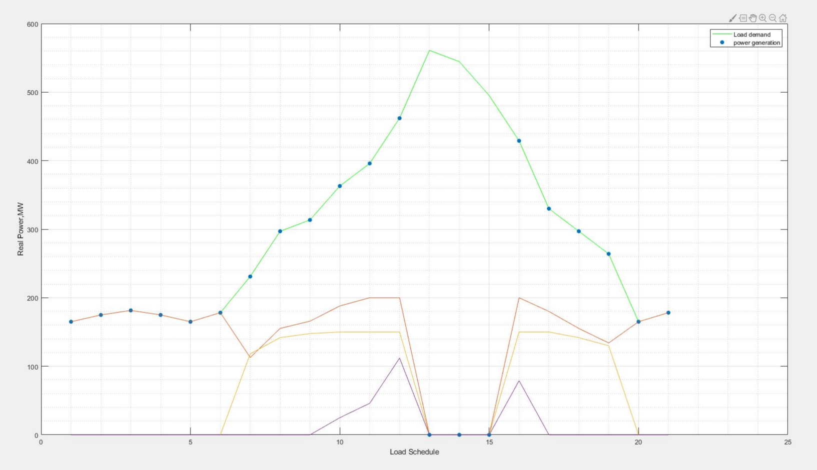
```
function [M, I] = permn(V, N)  % PERMN(V,N) - return all permutations
  nV = numel(V); % returns the number of elements in V
  if N == 1
    M = V(:);   % Assigning values of V to M with one column and many rows
    I = (1:nV)'; % giving indexes
  else
    [Y{N:-1:1}] = ndgrid(1:nV);   % 1*N matrix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

Matlab code for graph plotting:

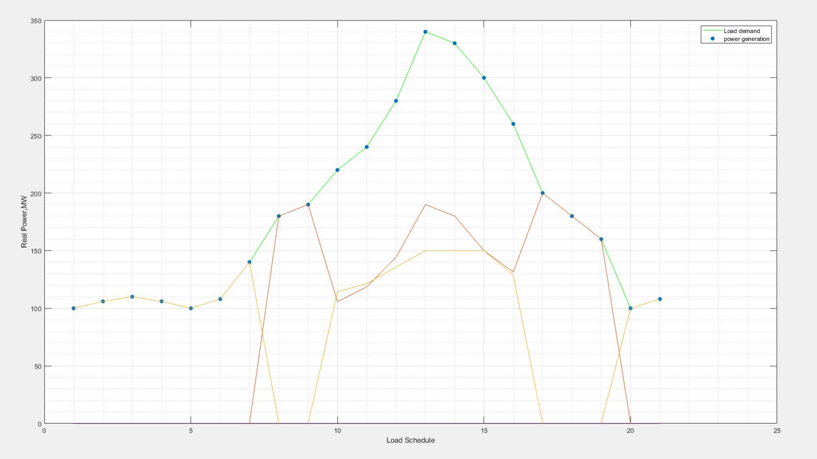
```
figure(1)

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

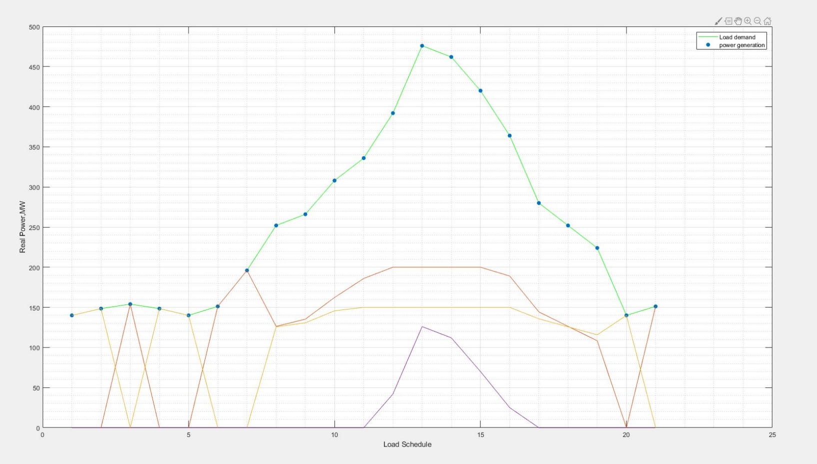
Graph for nominal load of 330MW:



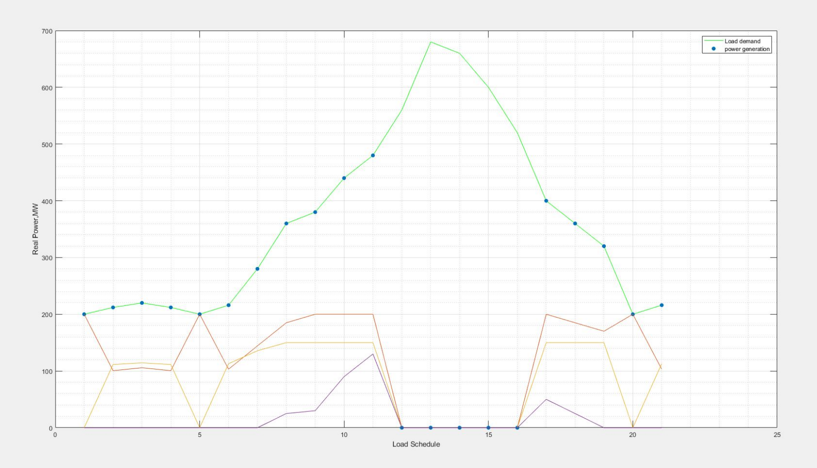
Graph for nominal load of 200MW:



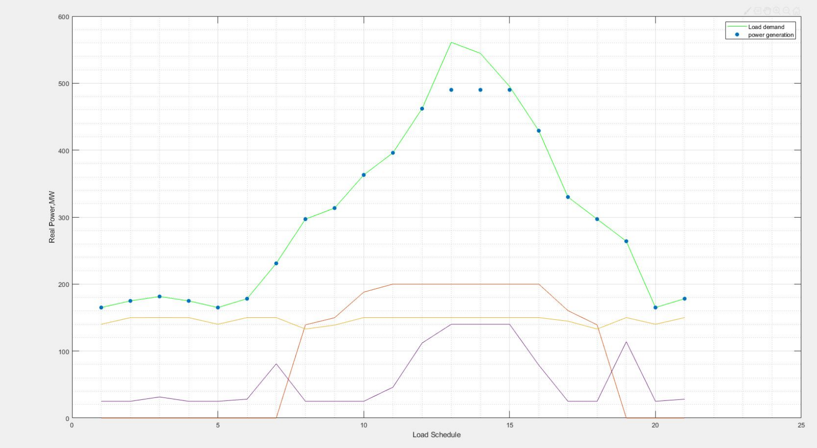
Graph for nominal load of 280MW:



Graph for nominal load of 400MW:



Graph for priority list:



Conclusion

Conclusion

- For nominal loads of 280 MW and 200 MW the system was working without any problems
- If the load demand exceeded the maximum power generating capacity, the system had shut down all the generating unit to maintain the optimality (In case of nominal load is 330 MW and 400 MW)
- Whereas, in case of priority list the power generating units were not shut down even though the load demands were not met hence this method did not guarantee optimality

Thank you for your attention!

Are there any questions?