A Reinforcement Learning Algorithm to Economic Dispatch Considering Transmission Losses

Jasmin.E.A., T.P.Imthias Ahamed and V.P.Jagathiraj[1]

***Abstract*: - Reinforcement Learning (RL) refers to a class of learning algorithms in which learning system learns which action to take in different situations by using a scalar evaluation received from the environment on performing an action. RL has been successfully applied to many multi stage decision making problem (MDP) where in each stage the learning systems decides which action has to be taken. Economic Dispatch (ED) problem is an important scheduling problem in power systems, which decides the amount of generation to be allocated to each generating unit so that the total cost of generation is minimized without violating system constraints. In this paper we formulate economic dispatch problem as a multi stage decision making problem. In this paper, we also develop RL based algorithm to solve the ED problem. The performance of our algorithm is compared with other recent methods. The main advantage of our method is it can learn the schedule for all possible demands simultaneously.**

***Index Terms*—Economic Dispatch, Multi-stage Decision Making Problem, Reinforcement Learning, Q learning**

I. INTRODUCTION

In this paper, we present a Reinforcement Learning based algorithm for solving the Economic Dispatch problem. Reinforcement Learning is a strategy of learning which relies on experience [1, 2]. At any particular state of the system, an action is performed and goodness of that action is measured and stored for decision making for the same state occurring in future. It is a good technique of solution especially when the system environment is random or ill defined.

Reinforcement Learning has been successfully applied for several difficult problems such as control of inverted pendulum, playing Backgammon and other computer games and elevator control problems [3 -5]. It has also been used as a solution strategy for Automatic Generation Control of Power system [6] and Unit Commitment problem [7] of power generation.

A Power system consists of a large number of units having a variety of characteristics. The load demand is not at all a constant throughout, varies from time to time. Meeting the load, with minimum cost of generation, while satisfying all the constraints associated with the system, is an important problem in a power system. A variety of efforts have been made to solve this optimization problem incorporating the different types of constraints or multiple objectives. In [8] a review of algorithms developed prior to 1990 is given. The conventional methods include lambda iteration method [9], base point participation factor method [10] etc. These methods are not suitable to solve problems with non convex fuel cost functions, which is the most often the one in practical systems. Even though Dynamic Programming is a good solution strategy for Economic Dispatch problem without imposing any restrictions on the fuel cost curves [9], it suffers from curse of dimensionality.

For handling the non-convex cost functions, many soft computing techniques have been successfully employed. Hopfield Neural Networks are being used for solution in [11] [12] and radial basis functions in [13]. A solution based on Genetic Algorithm is developed by Chen in [14]. Another stochastic search technique based on Simulated Annealing has been proposed in [15] to solve this problem of scheduling. A Partition Approach algorithm, which is a well suited for non convex optimization problems is developed in [16]. [17] and [18] put forward Evolutionary Programming based approaches for solving Economic Dispatch problem.

One major limitation of all the soft computing

[1] *E.A.Jasmin, Lecturer, Dept. of Electrical & Electronics,Govt Engg.College,Thrissur,Kerala, email : eajasmin@gmail.com*

Dr. T.P.Imthias Ahamed, Sel.grade Lecturer, Dept. of Electrical& Electronics, T.K.M. College of Engineering, Kollam,Kerala email : imthiasa@gmail.com

Dr.V.P.Jagathiraj, Professor, School of Management Studies, Cochin University of Science and Technology,Kerala email: jagathi@cusat.ac.in

techniques is the requirement of large computation time. As far as Economic Dispatch is concerned, the total load demand during various hours is different. Hence, we have to learn the schedule for all possible demands. Almost all the soft-computing techniques compute the load distribution for a specified load. Hence the computational effort has to be repeated for all possible load demand. One advantage of RL method is information learned while learning for a particular total load demand is used to learn optimum schedule for other load demands. In other words, the computational effort for learning the dispatch for all possible load demands is almost same as the effort required to learn for one particular load demand.

This paper is organized as follows. Mathematical formulation of Economic Dispatch problem is explained in next section. Technique of learning through RL and the different notations used in the learning algorithm are explained in section III. Section IV gives the formulation of Economic Dispatch as a multi stage decision making problem and the algorithm neglecting transmission losses. Then the transmission losses are considered and the algorithm is extended in Section V. Section VI explains the simulation results and conclusion is given in section VII.

II PROBLEM FORMULATION

Consider a power system having N generating units. Let P_T be the power demand to be satisfied with these N units at any slot of time and let P_L be the total transmission loss in the system. Economic Dispatch is to find an optimum schedule of power allocation among these N units. The allotment should be in such a way that the cost of generation should be minimum as far as possible. At the same time, generating unit power constraints should also be met. Therefore Economic Dispatch can be treated as a constrained optimization problem**.**

The objective function of Economic Dispatch problem C_T is the total cost for supplying the load demand including the transmission loss. The problem is to minimize C_T subject to the constraint that the total generated power meets the demanded load and the transmission losses. At the same time the power constraints on all units are being met. That is,

$$C_T = C_1(P_1) + C_2(P_2) + C_3(P_3) + \ldots + C_N(P_N)$$

$$C_T = \sum_{i=1}^{i=N} C_i(P_i) \quad \ldots\ldots(1)$$

Where P_i - Power allotted to i^{th} unit and C_i-Cost function of i^{th} unit.

The constraints to be met are

$$P_{load} + P_L - \Sigma (P_i) = 0 \quad \ldots\ldots(2)$$

$$P\min_i \le P_i \le P\max_i \quad \text{for } i = 1 \text{ to } N \quad \ldots\ldots(3)$$

P_L – transmission loss in the system

Transmission loss in the system can be calculated using well known B matrix loss formula [9] as

$$P_L = \sum_{i=1}^{i=N} \sum_{j=1}^{j=N} P_i B_{ij} P_j + \sum_{i=1}^{i=N} B_{0i} P_i + B_{00},$$

where

B_{ij} is the ij^{th} element of the loss coefficient square matrix, B_{0i} is the i^{th} element of the loss coefficient vector and B_{00} is the loss coefficient constant. Thus, the problem is to minimize the cost function C_T subject to the constraints given in (2) and (3).

III REINFORCEMENT LEARNING

Reinforcement Learning is one effective method of solving multi stage decision making problems. A MDP involves making a decision or taking an action **"a_k"** at each stage based on the current state, **"x_k"** of the system. On taking action $\mathbf{a_k}$, the system moves to a new state x_{k+1}. Solution of MDP involves finding a sequence of actions $a_0, a_1, a_2, \cdots a_N$ so that the specified objective is realized. We formulate Economic Dispatch problem as a MDP and propose an algorithm based on Q learning technique to solve the problem.

Q learning involves finding the so-called Q-values. Q-values are defined for all state action pairs, say (***x, a***). It gives a measure of goodness of selecting action ***a*** in state ***x***. In our context, if $Q(a^*, x_k) \le Q(a_k, x_k)$ for all possible actions, then a^* is the best action. A comprehensive study of Q learning can be found in [2].

The algorithm starts with an initial estimate for Q values. The learning process proceeds as follows: Each time the decision making agent interacts with the environment and chooses an action from the action set (The procedure for selecting an action at each step is given in the next paragraph). In Q learning method, following the response of the environment to the selected action, the learning process updates the Q value associated with the corresponding state-action pair. On performing the action, the system reaches a new state which provides a new set of actions. A new action from the current action set is then selected. As the algorithm proceeds, the Q values of different state- action pairs seem to converge to optimum so that in the successive iterations, best action will be performed at each state.

Now, let us see how to choose an action at each step. At any discrete step, there will be one action whose estimated Q value up to that moment is best. That action is called as greedy action. But the current estimate may

be wrong. There may be a better action. Therefore the solution strategy should exploit the goodness achieved for the greedy action, at the same time explore other possibilities. For the selection of an action from the given action set one method is *ε-greedy* method. In this method the greedy action is chosen with probability of (1-ε) and a random action from the remaining action set with probability 'ε'. 'ε' can take any value up to unity. Smaller values of 'ε' decreases the exploration of the action set while values closer to unity increase the rate of exploration at each step. Usually in the first few iterations, a large value of 'ε' is used so that all actions are given sufficient chance to be performed and as the algorithm progresses ε may be reduced since the greedy action itself turn to be the best action. This action selection drastically reduces the computational overhead, at the same time avoiding trapping to local minima.

Therefore Reinforcement Learning basically requires a set of states and associated actions from which to select, numerical measure of the goodness of a particular action (reward) which is used to update the Q value associated with state-action pair. If x_{k+1} is the state reached from a state x_k on performing an action a_k, Q value, Q (x_k, a_k) is updated using the equation,

$$Q^{n+1}(x_k, a_k) = Q^n(x_k, a_k) + \alpha * [g(x_k, a_k, x_{k+1}) + \gamma * min_{a' \varepsilon A}(Q(x_{k+1}, a')) - Q^n(x_k, a_k)],$$

where A is the action set, $g(x_k, a_k, x_{k+1})$ is the reward function associated with the state-action pair, α is the learning parameter and γ is the discount factor for accommodating the future effects.

IV RL ALGORITHM FOR ECONOMIC DISPATCH NEGLECTING LOSSES

We will first formulate Economic Dispatch as a multi stage decision making problem and then Reinforcement Learning algorithm for solving the same without including losses is developed.

In the Economic Dispatch problem, the problem is to find out the optimum schedule of generation for the N generating units. We will treat this as a multi stage decision making task having *N* stages (0 to N-1). State of the system at any stage can be represented by a tuple *(k, RP_k), k* being the stage number and RP_k being the power to be allotted among the remaining *N-k* units. At each stage of the problem, there is a set of possible states

$\chi_k = \{(k, D_{mink}), \ldots\ldots (k, D_{maxk})\}$ where

D_{mink} – Minimum allocation possible with the remaining N-k units

D_{maxk} – Maximum allocation possible with the remaining N-k units.

At each stage k, an action (a_k) is selected from the action set (A_k) which is the power allocation for one of the units ($unit_k$). Therefore, as the learning system goes through stages 0 to N-1, power allocation is made to all the N units.

The action set at any stage k is defined as:

$A_k = \{a_{mink}, \ldots\ldots a_{maxk}\}$, where a_{mink} and a_{maxk} being the minimum and maximum power allocation possible to the k^{th} unit. These values depend on the minimum and maximum power generation possible with k^{th} unit and also on the minimum and maximum power generation possible with the remaining *N-k* units. The number of elements of the action set (A_k) also depends on the discretisation step size.

Initially we will be having N units and the demanded power P_T MW to be dispatched. This can be taken as $stage_0$ with $RP_0 = P_T$. The state of the system at k=0 is the tuple, $x_0 = (0, RP_0)$. The learning system takes a decision on how much amount to be dispatched to 0^{th} unit. This is treated as action a_0. On applying this action, the system proceeds to the next stage or $stage_1$ with the remaining power, $RP_1 = RP_0 - a_0$ MW to be dispatched among the remaining N-1 units. The new state is denoted as $(1, RP_1)$, Then an action a_1 is selected from the action set A_1 which makes the system transition to $stage_2$. This action selection and stage transition continues up to $stage_{N-1}$. In the last stage ($stage_{N-1}$) when the state is $(N-1, RP_{N-1})$, there is no choice of action, the only action possible is to allocate the remaining power.

i.e., $a_{N-1} = RP_{N-1}$.

For selecting an action from the action set, we use ε-greedy method where the greedy action is selected with a probability of 1-ε, while one of the actions from the remaining action set is selected with a probability of ε. On each action performance, the learning system gets back a reward. In this Economic Dispatch problem, the reward at any stage k can be defined as the cost of generating power a_k by k^{th} unit. Therefore,

$$r_k = g(x_k, a_k, x_{k+1}) = C_k(a_k) \qquad \ldots\ldots(4)$$

For each of the stages from 0 to N-2, the Q value of the state- action pair is then updated using the equation,

$$Q^{n+1}(x_k, a_k) = Q^n(x_k, a_k) + \alpha * [g(x_k, a_k, x_{k+1}) + \gamma * min_{a' \varepsilon A}(Q(x_{k+1}, a')) - Q^n(x_k, a_k)] \qquad \ldots\ldots\ldots(5)$$

where α – *step size of learning*

γ - *discount factor*

When stage has reached N-1, since there is no future stages, Q value is updated using the equation,

$$Q^{n+1}(x_k\ a_k) = Q^n(x_k, a_k) + \alpha * [g(x_k, a_k, x_{k+1}) - Q^n(x_k, a_k)] \qquad \ldots\ldots\ldots(6)$$

Our algorithm has two phases: *learning phase and retreiving phase.* During the learning phase, we learn the

Q-values for all possible combinations of state action pair. To achieve this, we run the algorithm with randomly generated values of demanded power sufficient number of times. The entire algorithm for learning phase is given in Algorithm I.

To run the algorithm, we have to choose ε, α and γ. The value of ε is chosen as 0.5 in the first few iterations for providing sufficient rate of exploration and is reduced as the learning proceeds. The learning parameter α decides on how much the Q value is modified on each iteration of learning. Smaller values will make the convergence slower while larger values make it oscillatory. In this problem, by trial and error we choose a value of 0.1. In the context of Reinforcement Learning problems, a reward received at a future stage may not be having the same effect as the same reward received at present state. Discount factor incorporates the extent of discounting the present reward in view of future reinforcements. In the case of Economic Dispatch problem, the cost incurred in a later stage has the same significance as the cost incurred in the current stage, the discount factor is taken as 1.

After learning is completed, for getting the dispatch corresponding to a demanded power, actions corresponding to minimum Q value (greedy action) at different stages are found out and this will give the optimum allocation schedule. To get the allocation schedule for a particular value of load demand P_T, we take the starting state as the tuple $(0, RP_0)$ where $RP_0 = P_T$. Then find the greedy action for 0^{th} stage as a_0 which will give the allocation P_0 for $unit_0$ and proceeds to the next stage with tuple $(1,RP_1)$. This proceeds until greedy action (optimum power) for all the N stages are found out. The algorithm for getting the schedule is given in Algorithm II.

Algorithm I *(Learning phase)*

Get Unit parameters
Initialize learning parameters
For all the stages, Identify possible state vectors, χ_k
Evaluate minimum and maximum demands permissible
Initialize Q (x, a) with zero
For (n =1 to max_iteration)
Do
P_T = rand ($Dmin_0$, $Dmax_0$)
$RP_0 = P_T$
For k=0 to N-2
Do
State tuple $x_k = (k, RP_k)$
Identify the action space A_k
Select an action a_k from action set A_k using ε- greedy method.
Apply action a_k and allot power to k^{th} unit,
$RP_{k+1} = RP_k - a_k$
Calculate the reward function $g(x_k, a_k, x_{k+1})$ using eqn.(4)
Update Q^n to Q^{n+1} using eqn (5)
End do
$a_{N-1} = RP_{(N-1)}$
Calculate the reward function $g(x_k, a_k, x_{k+1})$ using eqn(4)
Update Q^n to Q^{n+1} using eqn (6)
Update learning parameter ε[2]
Enddo

AlgorithmII *(retrieving phase)*

Get the demand power P_T to be scheduled
$RP_0 = P_T$
For (k=0 to N-1)
Do
State tuple $x_k = (k,RP_k)$
Find $a_g = argmin_a (Q(x_k,a))$
Allotted power $P_k = a_g$
$RP_{k+1} = RP_k - P_k$
Enddo

V ALGORITHM FOR ECONOMIC DISPSTCH CONSIDERING TRANSMISSION LOSSES

We now extend our previous RL algorithm in order to incorporate the losses in the transmission system. We use the B matrix loss formula for calculating the loss MW at each step. We find the dispatch of the possible load values considering transmission losses also using Algorithm III which is described below.

We execute Algorithm I to learn the Q values. The optimum generation schedule corresponding to a load demand P_T is obtained by running the algorithm II. Then from the optimum allocation obtained, corresponding loss is calculated using B coefficients. The demanded power is then modified as $P_{T(new)} = P_{T(old)} + Loss$. Then the dispatch is again obtained for the power $P_{T(new)}$ using algorithm II. This is continued and in general, the amount of power to be allocated in $(n+1)^{th}$ step is given as, $P_{T(n+1)} = P_{T(n)} + Loss_n$, $Loss_n$ being the MW loss corresponding to the power schedule obtained in the n^{th} step. This allocation is continued until the change in demanded power in two successive steps is negligibly small. The above procedure is continued for all the load values in suitable steps. The entire algorithm is given below:

[2] ε is reduced by a value of 0.04 after every (max_iteration/10) iterations

Algorithm III *(Considering Transmission Losses)*

Get unit parameters including B- coefficient matrix of the system
Learn the Q values using Algorithm I
Initial load $P_T = D_{min}$
Do
{
Initialize P_{loss} *and Prev_loss to zero*
Initialize final loss tolerance to a suitable value μ
Initialize change in loss (Δ) *to a suitable value*
Do
{
$Prev\_loss = P_{loss}$
Find the allocation corresponding to P_T *using Algorithm II*
Find the loss using B coefficients as P_{loss}
Update $P_T = P_T + P_{loss}$
Compute change in loss $\Delta = P_{loss} - prev\_loss$
}
while ($\Delta > \mu$)
$P_T = P_T + step$[3]
}
While ($P_T <= D_{max}$)

VI SIMULATION RESULTS

For testing the efficacy of our Algorithm and to compare with recent techniques, we consider an IEEE 6 bus system [15], [16]. The fuel cost curve of the units is represented by a third order polynomial function. The associated fuel cost coefficients and B-matrix parameters are given in Table I. We find the dispatch for various load values ranging from minimum demand possible to maximum value possible. In this simulation, we choose a discretisation step size of 2MW. Hence there could be a maximum difference of 2MW between power generation and power demand including losses. This remaining power of the demanded value (less than 2MW), which is negligibly small compared to the total demand is randomly assigned to one of the units without exceeding maximum limit. The same steps are carried out for finding the dispatch for all the possible load values. as given in Algorithm III. The total time taken to obtain all the schedule is 9.87 seconds. The schedule and the total cost rounded of to the nearest integer is given in Table II. For other methods, training or learning phase is to be repeated for each of the required demand.

For comparing the computational efficiency with other recent methods, the schedule is obtained for a single load value of 1200 MW. The method gives an optimal cost of $5676.22 and the power loss calculated is 43.72MW. The obtained schedule is tabulated and comparison is made with Simulated Annealing given in [15] and Partition Approach Algorithm in [16] in Table III. From the table, we see that the obtained cost is less than those obtained through simulated annealing. It is slightly more than that obtained through Partition approach Algorithm, but the difference is only $5.16. The dispatch schedule obtained is also comparable. Transmission loss in all the case is nearly 43MW. The time of execution of the proposed algorithm is found to be less (9.68sec) compared to simulated annealing (27.25) in [15] and Partition Approach algorithm (16.82). Also when we need the schedule for all the load values, the other two methods require approximately 16 times the time given in Table III.

Thus the proposed algorithm is much faster than other recent techniques. Moreover, the total time taken by our method to find the schedule for 16 different loads is considerably less than the time taken by other methods to find the dispatch for one load.

Table I

Unit No	1	2	3
	Generator data		
a_i	*11.20*	*-632*	*147.144*
b_i	*5.102*	*13.01*	*4.28997*
c_i	*-2.6429E-3*	*-3.05714E-2*	*3.0845E-4*
d_i	*3.33333E-6*	*3.3333E-5*	*-1.7677E-7*
$P_{gi,min}$	*100*	*100*	*200*
$P_{gi,max}$	*500*	*500*	*1000*

B Coefficients

17.5E-5	5.0E-6	7.5E-6
25.0E-6	1.5E-5	1.0E-5
37.5E-6	1.0E-5	4.5E-5

Table II – Schedule for many possible loads

P_D(MW)	P_{g1}	P_{g2}	P_{g3}	Cost	Loss(MW)
400	100	100	204	1958	4
500	100	100	306	2383	6
600	100	100	409	2800	9
700	261	100	373	3246	14
800	298	100	420	3696	18
900	330	100	495	4140	25
1000	338	100	591	4737	29
1100	341	100	691	5200	32
1200	344	100	800	5464	44
1300	414	100	835	6002	49
1400	446	156	851	6474	53
1500	453	229	874	6896	56
1600	461	300	899	7438	59
1700	469	302	992	7772	64
1800	455	321	991	8320	67
1900	487	492	996	8801	75

[3] Step is taken as 100MW in this simulation.

Table III – Comparison with other methods for P_T = 1200MW

Method	PAA	SA	RL
P_{g1}(MW)	362.2	342.8	343.72
P_{g2}(MW)	100	100	100
P_{g3}(MW)	781.4	801.4	800
Cost	5671.06	5682.32	5676.2
Loss(MW)	43.68	43.8	43.72
Time (sec)	16.802	27.253	9.68

VII CONCLUSION

We have developed a new approach to the solution of Economic Dispatch problem. RL approach is very powerful for solving such scheduling problems. It can accommodate any type of cost functions. We have tested the efficacy of the algorithm for the IEEE six bus system with third order cost functions and also considering transmission losses. Also from the comparison with other stochastic methods, we see that the new algorithm gives the optimum allocation with comparable cost, with much lesser time as compared to other methods. Also since with a single learning task, it is easy to retrieve schedule for any demand, it is more suitable for practical systems. It can be easily extended for other systems including more complex cost functions and unit operating conditions.

VII ACKNOWLEDGMENT:

The second author gratefully acknowledges the support of All India Council of Technical Education under the TAPTEC scheme (Ref. No. 8021/RID/NPROJ/TAP-139/2002-03). He also acknowledges the help and support rendered by T.K.M.College of Engineering authorities.

References:

[1] D.P.Bertsekas, J.N.Tsitsikilis. *Neurodynamic Programming,* Athena Scientific, Belmount,MA., 1996.

[2] R.S.Sutton, A.G.Barto. *Reinforcement Learning: An Introduction,* Cambridge, MA:MIT Press, 1998

[3] C.W.Anderson. *Learning to control an inverted pendulam using Neural Networks,* IEEE Control System Magazine, 9(1989) : 31 - 37

[4] G.J.Tasauro, TD gammon. *A self teaching backgammon program achieves master level play,* Neural Computation 6 (1994) : 215 – 219.

[5] RobortH.Crytes, AndrewG.Barto. *Elevator group control using multiple reinforcement learning agent,* Boston : Kulwer Academic, 2002.

[6] T.P.ImthiasAhamed, P.S.NagendraRao, P.S.Sastry. *A Reinforcement learning approach to Automatic Generation control*, Electric Power Systems Research, 63 (2002) : 9-26.

[7] T.P.Imthias Ahamed. *A Reinforcement Learning Approach to Unit Commitment Problem*, Proceedings of National Power System Conference 2006.

[8] B.H.Chowdhury, and SalfurRahman. *A Review of Recent advances in Economic Dispatch,* IEEE Transactions on Power Systems, 5 (1990) : 1248-1259.

[9] A.J.Wood, B.F.Woolenberg. *Power Generation and Control,* John Wiley Sons, 2002.

[10] C.L. Chen, S.C. Wang. *Branch and bound scheduling for thermal generating units,* IEEE Trans. Energy Convers. , 8 (1993) : 184 – 189

[11] M.R.Farooqui, P.Jain, K.R.Naizi. *Using Hopfield Neural Network for Economic Dispatch of Power systems,* National Power & Energy Conference proceedings, 2003 : 5-10.

[12] S.Senthikumar, V.Palnisamy. *A Dynamic Programming based fast computation Hopfield network for Unitcommitment and Economic Dispatch*, Electric Power & Energy systems (2006)

[13] P.Aravindababu, K.R.Nayer. *Economic Dispatch based on optimal lambda using radial basis function network,* Electrical Power and Energy systems, 21 (2002) : 551-556.

[14] P.H. Chen, H.C. Chang. *Large Scale Economic Dispatch by Genetic Algorithm,* IEEE Transactions on Power Systems 10, 4 (1995) : 1919 – 1926.

[15] K.P.Wong, C.C. Fung. *Simulated Annealing based Economic Dispatch algorithm*, IEE Proc.- C 140, 6 (1993) : 509 – 515.

[16] Whei Min Lin, Hong – Jey Gow. *A Partition Approach algorithm for Nonconvex Economic Dispatch,* Electric Power and Energy Systems (2007).

[17] T.Jayabarathi, K. Jayaprakash. *Evolutionary Programming techniques for different kinds of Economic Dispatch problems,* Electric Power Systems Research 73 (2005) : 169 – 176.

[18] P.Somasundaram, K.Kuppusamy. *Application of Evolutionary Programming to security constrained Economic Dispatch*, Electric Power & Energy systems, 27 (2005) : 343-351.

[19] A.Immanuel Selvakumar, K.Thaushikodi. *A New particle swarm optimization solution to non convex Economic Dispatch problems,* IEEE Transactions on Power systems 22, 1 (2007) : 42-51.