2009 International Conference on Advances in Computing, Control, and Telecommunication Technologies

Reinforcement Learning solution for Unit Commitment Problem through pursuit method

Dr.Jasmin.E.A, , *Dept. of Electrical& Electronics, Govt Engg.College,Thrissur, Kerala, eajasmin@gmail.com*
Dr. Imthias Ahamed T.P., *Dept.of Electrical &Electronics,T.K.M.College,Kollam, imthiasa@gmail.com*
Dr.Jagathy Raj V.P., *Reader, School of Mangement studies, CUSAT, jagathy@cusat.ac.in*

***Abstract*—Unit commitment is an optimization task in electric power generation control sector. It involves scheduling the ON/OFF status of the generating units to meet the load demand with minimum generation cost satisfying the different constraints existing in the system. Numerical solutions developed are limited for small systems and heuristic methodologies find difficulty in handling stochastic cost functions associated with practical systems. This paper models Unit Commitment as a multi stage decision task and Reinforcement Learning solution is formulated through one efficient exploration strategy: Pursuit method. The correctness and efficiency of the developed solutions are verified for standard test systems.**

***Index Terms*— Unit Commitment, reinforcement learning, Q learning.**

I. INTRODUCTION

This paper proposes a Reinforcement Learning (RL) based solution to one of the optimization problems in the power generation sector: Unit Commitment problem (UCP) [1]. Reinforcement Learning based solutions have been proposed to several control and optimization tasks like playing Backgammon [2], robotics and control [3 -5], medical imaging[6] etc.

In the field of power system also a few applications of RL has been proposed [7 - 10]. UCP is one constrained optimization problem in power system scheduling. It involves scheduling the ON / OFF status of a set of units to meet the forecasted load demand over a time horizon under different operational constraints so that the total generation cost is minimized. Since an improved Unit commitment schedule may bring forth savings of large amount for an electric utility, Unit Commitment is an important optimization task in the daily operation planning of power system today. RL has not yet been explored in the solution of this optimization problem.

Priority list methods [11], Dynamic Programming[12], Lagrange Relaxation [13] etc. have been explored by various researchers. Priority List method is simple but the solution obtained is not optimum always. Dynamic Programming provides optimum solution to problems with small number of units.Several soft computing strategies including Genetic Algorithm [15], Simulated Annealing [16] are also being proposed. But these methods are also limited in computational efficiency when a large number of units are to be considered.

The two different exploration strategies used in Reinforcement Learning solutions are ε greedy and pursuit. Even though ε greedy is simple one, it necessitates a proper cooling schedule to provide balancing between exploration and exploitation of the action space. In this paper we propose the pursuit method of action selection and reinforcement Learning solution for the Unit Commitment problem.

The organization of the rest of the paper is as follows. Mathematical formulation of Unit Commitment Problem (UCP) is given in section 2. To make the paper self explanatory a brief description on Reinforcement Learning is given in section 3. In section 4, UCP is formulated as a Multi stage decision making task and solution through RL approach is given. Performance of the developed algorithms is evaluated in section 5. Concluding remarks are given in section 6.

II. UNIT COMMITMENT PROBLEM

If a Power system is having N generating units and a load schedule for T hours is given, the problem is to find out which all units are to be committed in the next T hours so that cost of generation is minimum. Let L = (l_0, l_1,……..l_{T-1}) be the load schedule and a = (a_0, a_1,…..a_{T-1}) denote the variable denoting the schedule or status of units where $a_k = a_k^0, a_k^1, \ldots\ldots\ldots a_k^{N-1}$. That is, action set $\mathcal{A}= \{a_k^i, i = 0, \ldots N-1; k = 0, \ldots T-1\}$. $a_k^i = 1$ indicates that i^{th} unit is ON during k^{th} hour while $a_k^i = 0$ indicates OFF condition. We can denote the state of the system at hour k as x_k where $x_k = \{x_k^0, x_k^1, \ldots\ldots\ldots x_k^{N-1}\}$, x_k^i= 1 indicates ON condition of i^{th} unit and x_k^i= 0 indicate OFF status. Also we can denote the power generation by each of these units as p = { p_k^i, i = 0, 1, …..N - 1}, p_k^i, being the power generated by i^{th} unit at k^{th} hour. Now we can denote the net cost of generation as,

$$F_T = \sum_{t=1}^{T}\sum_{i=1}^{N}[C_i\left(p_k^i\right)a_k^i + ST_i(a_k^i)(1 - a_{k-1}^i)] \quad (1)$$

The aim of the optimization task is to find the commitment status of the generating units such that the net cost of generation given by equation (1) is

The constraints to be satisfied are given by

$$l_k <= \sum_{i=0}^{N-1} p_k^i, \quad k = 0,\ldots..T\text{-}1. \quad (2)$$

978-0-7695-3915-7/09 $26.00 © 2009 IEEE
DOI 10.1109/ACT.2009.87

$$\text{Pmin}_i <= p_k^i <= \text{Pmax}_i \quad (3)$$

III. REINFORCEMENT LEARNING

Reinforcement Learning is a learning strategy which discovers a best policy, mapping of situations to actions [18, 19]. By continued interaction with the environment, learning agent discovers the best action suitable for each situation. Learning agent gets the state of the system and chooses a suitable action from the available action set. On performing this action a_k in state x_k, the agent receives a reward from the environment and the system proceeds to the next state x_{k+1}. Reward is a numerical measure of the goodness of the action and depends on the state transition.

That is, reward $r_k = g(x_k, a_k, x_{k+1})$.

Reinforcement learning agent keeps track of the rewards received at different system states which are used in action selection when the same situation arises in future. For the same, Q learning is a widely used method in Reinforcement Learning. Here Q values associated with each state – action pair, $Q(x,a)$ is updated based on the reward value on performing an action a at state x. These Q values of the different actions can then be compared for selecting an action when the same state x is encountered in future.

In Q learning algorithm we will first initialize all Q values with some initial value, $Q^0(x_k, a_k)$. At each iteration n, on reaching x_k an action a_k is taken based on the current estimate of $Q^*(x_k, a_k)$ ie, $Q^n(x_k, a_k)$. Once action is taken at state x_k, it makes transition to x_{k+1} and the reward $g(x_k, a_k, x_{k+1})$ can be found from simulation. We update the Q value using the equation,

$$Q^{n+1}(x_k, a_k) = Q^n(x_k, a_k) + \alpha \left[g(x_k, a_k, x_{k+1}) + \gamma \min_{a' \in Ak+1} Qnxk+1, a' - Qnxk, ak \right]$$

$0 < \gamma < 1$ is a constant and is called step size of learning. As the learning proceeds, $Q^{n+1}(x_k, a_k)$ will be converging to the optimum value $Q^*(x_k, a_k)$ so that best action can be selected by just finding the greedy action (action having minimum Q value (a_g) or in other words best action found from previous experience)

That is, $a_g = \text{argmin}_{a' \in A} Q(x_k, a')$

During the learning phase, the agent should explore the action space at the same time exploit the previous acquired knowledge on good actions found so far. Therefore one of the tasks to be resolved in Reinforcement Learning strategy is, to find a balance between exploration and exploitation. That is, while selecting an action during the learning phase, a reinforcement learning agent must consider the actions which it has tried in the past and found to be good. At the same time it should accommodate the fact that there may be actions in the action space which may be better or as good as the one it had already experienced to be good.

To provide sufficient exploration and exploitation in the action selection, different methods are employed in Reinforcement Learning.

One method is ε - greedy method in which exploration rate is decided by the parameter ε. The action which has already found as good in previous attempts is termed as greedy action. In ε - greedy strategy, on reaching at any state the greedy action is chosen with a probability of (1- ε) while one among the remaining actions from the action space in random is performed with a probability of ε. For providing better exploration in the initial phases of learning while exploiting the goodness of greedy action during the later phases, value of exploration parameter ε should be large initially and to be reduced gradually. That is, a proper cooling schedule is to be designed which gradually updates the value of ε as the learning proceeds so that proper convergence and correctness of the result are assured. The length of learning phase mainly depends on this cooling schedule and therefore it is one significant part of ε- greedy method. Also it is a difficult task to develop a good cooling schedule so as to ensure that time for convergence is minimum.

Pursuit method provides an alternate method of learning through Reinforcement. In this method along with maintaining estimates of Q values as measure of goodness of actions, also some preference is associated with actions. Each action a_k at any state x_k is having a probability $p_{xk}(a_k)$ of being chosen. These probability values will be same for all actions at any state initially assuring sufficient exploration of the action space. Then on performing an action a_k at any state x_k during learning, the numerical reward is used to update the estimate of Q value associated with the state – action pair. Along with that, based on the current estimates of Q values, probability values associated with actions are also modified as.

That is,

$p^{n+1}(a_k) = a_k) + \beta [1 - p^n(a_k)]$, when $a_k = a_g$

$p^{n+1}(a_k) = a_k) - \beta [p^n(a_k)]$, when $a_k \neq a_g$

where $0 < \beta < 1$ is a constant. Thus at each iteration n of the learning phase, algorithm will slightly increase the probability of choosing the greedy action a_g in state x_k and proportionally decrease the probability associated with all other possible actions. Initially since all probabilities are made equal, sufficient exploration of action space is assured. As the algorithm proceeds, since the probability of greedy action increases it provides us with the needed exploitation of previous experience. When the parameter β is properly chosen, after sufficient number of iterations, probability of greedy action will reach nearer to unity while all others to zero. This indicates a convergence condition since greedy action is chosen at each system state.

IV. Solution of Unit Commitment Problem Using Pursuit Method

Consider a Power system having N generating units and let load schedule for T hours be given. At each slot time there will be 2^N possible combinations of ON-OFF schedule from which to select. The problem is to find out the units to be committed (ON – OFF schedule) in each slot of time. We can denote the state of the system at time slot k as x_k. We can represent this as a tuple (k, s) where s being a binary string of length N indicating the ON – OFF schedule of the N generating units. For eg., (5,0001) or equivalent decimal notation (5, 1) can denote that only one among the four units is ON during 5^{th} time slot (hour). We can then denote a_k as action during k^{th} hour which is also represented by a similar tuple (k, a), a being the binary string indicating ON – OFF status or equivalent integer. The action set A_{xk} depends basically on the load demand during k^{th} hour. It consists of all the actions or combinations of units which give out power equal to or greater than load demand during k^{th} hour.

At each hour (time slot) the possible states are found out by considering the respective the load demand. Initially the probability associated with each action a_k in the action set A_k corresponding to x_k are initialized with equal values as

$Px_k(a_k) = 1/n_{Ak,}$

n_{Ak} - Total number of permissible actions in state x_k.

The Q values are also initialized to zero.

That is,

$Q^0(x,a) = 0 \qquad \forall x \in X_k \text{ and } \forall a \in A_k$

$0 <= k <= T\text{-}1$

Then at each iteration step, an action a_k is selected based on the probability distribution. On Performing action a_k, state transition occurs as

$x_{k+1} = (k+1, a_k)$

The cost incurred in each step of learning is calculated as the sum of cost of producing power l_k with the generating units given by the binary string s in a_k and the start up cost associated with s.

That is the reward function,

$g(x_k, a_k, x_{k+1}) = C(a_k) + S(x_k, a_k)$, $S(x_k, a_k)$ being the start up cost.

Q values are then updated using the equation,

$$Q^{n+1}(x_k, a_k) = Q^n(x_k, a_k) + \alpha [g(x_k, a_k, x_{k+1}) + \gamma \min a' \in Ak+1 Qnxk+1, a' - Qnxk, ak \quad (4)$$

During the last stage, since there is no more future pay off to be accounted, the update equation is modified as,

$$Q^{n+1}(x_k, a_k) = Q^n(x_k, a_k) + \alpha [g(x_k, a_k, x_{k+1}) - Qnxk, ak \quad (5)$$

Correspondingly probabilities of actions in the action set are also updated as

$$p^{n+1}(a_k) = a_k) + \beta [1 - p^n(a_k)], \text{ when } a_k = a_g$$
$$p^{n+1}(a_k) = a_k) - \beta [p^n(a_k)], \text{ when } a_k \neq a_g \quad (6)$$

Th algorithm proceeds through several number of iterations when ultimately the probability of best action in each hour is increased sufficiently which indicate convergence of the algorithm. The entire algorithm is given in Table I

Table I

Read the Unit data and Load data for T hours
Find out set of possible states (χ) and actions($\mathcal{A}$)
Read the learning parameters
Read the initial status of units x_0
Initialise $Q^0(x,a) = 0$, $\forall x \in \chi$ and $\forall a \in \mathcal{A}$
Identify the feasible action set in each hour k as $\mathcal{A}_k$
Initialize $p^0_{x_k}(a_k)$ to $1/n_k$, n_k, the number of actions in $\mathcal{A}_k$
For n = 0 to max_iteration
Begin
  For k = 0 to T-1
    Do
      Choose action a_k based on the current probability distribution $p_{x_k}( )$
      Find the next state x_{k+1}
      Calculate $g(x_k, a_k, x_{k+1})$
      Update Q^n to Q^{n+1} using equations (4) and (5)
      Update probability $p^n_{x_k}(a_k)$ to $p^{n+1}_{x_k}(a_k)$ using equation(6)
    End do
End.

V. Simulation Results

To test the efficacy of the we consider a simple power system with four thermal units [1]. The Unit characteristics and Load pattern for eight hours are given in Table II and III respectively. For efficient learning, we have to choose suitable values for the learning parameters α and β. The parameter α decides the extent to which a training sample modifies the Q value and its value affects the speed of convergence and accuracy of result. By trial and error we choose a value of 0.1. The parameter β decides the exploration and exploitation during action selection. It is also initialized with a value of 0.01 in order to ensure convergence along with assuring sufficient exploration of action space.

The RL algorithm using pursuit method given in Table I is now used to find out an optimum policy for the Unit Commitment problem. The algorithm converges in 2000 iterations and the results obtained are tabulated in Table IV. The solution was consistent with the result given in [1].

Table III– Unit Characteristics

Unit	Pmin (MW)	Pmax (MW)	Inc. cost	No Load Cost	Start up Cost
1	75	300	17.46	684.74	1100
2	60	250	18	585.62	400
3	25	80	20.88	213	350
4	20	60	23.8	252	0.02

Load profile for 8 hours is considered and given in table IV.

Table IV – Load profile

Hour	0	1	2	3
Load	450	530	600	540
Hour	4	5	6	7
Load	400	280	290	500

The obtained commitment schedule is given in Table V.

Table V – Commitment status of units

Hour	Status	Hour	Status
1	0011	5	0011
2	0011	6	0001
3	1011	7	0001
4	0011	8	0011

VI. CONCLUSION

In this paper Reinforcement learning is suggested as a good solution strategy for solving one of the major optimization problems in the power system. Several Numerical and stochastic solutions have been proposed for solution of this constrained optimization problem. Reinforcement Learning provides with a good solution strategy which provide optimum scheduling with lesser computation time. The developed solution is also capable of handling the stochastic nature of cost functions and uncertainty associated with the availability of generating units. Thus it provides with a more suitable solution strategy for practical generator scheduling.

In this paper, only thermal generating units are considered As a next step the algorithm can be made to take actual data from a practical system incorporating other generating sources. Also the solution strategy provides with the scope of solving the power system problems in an efficient and faster mode.

REFERENCES

[1] A.J.Wood, B.F.Wollenberg., "Power Generation and Control" John Wiley Sons 2002.

[2] G.J. Tesauro, TD Gammon. "Temporal difference Learning and TD gammon", Communications of ACM, 38 (3) (1995): 58 – 67.

[3] Robert H.Crites, Andrew G.Barto, "Elevator control using multiple Reinforcement Learning Agents" Kulwer Academeic Pulishers, Boston , (1997)

[4] Hirashi Handa, Akira Ninimy, "Adaptive state construction for Reinforcement Learning and its application to Robot Navigation problem", IEEE Transaction on Industrial Electronics, 7(2) 1436: 1441, 2001.

[5] Andrew Y.N., A.Coats, M.Diel, V.Ganpapthi, "Autonomous helicopter flight via Reinforcement Learning" Symposium on Experimental robotics, (2004).

[6] Farhad Sahba, Hamid R.Tizhoosh, "Application of Reinforcement Learning for segmentation of transrectal ultra sound images", BMC Medical Imaging, 2008

[7] T.P.Imthias Ahamed, P.S.nagendra Rao, P.S.sastry "A Reinforcement Learning approach to automatic generation control" Electric Power System research 63 (2002): 9-26

[8] Damien Earnst, Mevludin Glavic, "Power system stability control: A Reinforcement Learning framework.", IEEE Transactions on Power Systems, 19 (1) (2004).

[9] D. Ernst, M.Glavic "Approximate value iteration in the Reinforcement Learning context: Application to Electric Power system control" International journal of Emerging Electric Power Systems, 3(1) (2005).

[10] E.A.Jasmin,T. P. Imthias Ahamed,V.P.Jagathyraj "A Reinforcement Learning algorithm to Economic Dispatch considering transmission losses", Proceedings of TENCON 2008.

[11] F.N.Lee, "Short term Unit Commitment – a new method", *IEEE Transactions on Power Systems* 99 (2) (1988): 691 – 698.

[12] Walter L.Snyder, H.david Powell. "Dynamic Programming approach to Unit Commitment", IEEE Transactions on Power Systems, 2 (2) (1987): 339 – 349.

[13] John A. Muckstadt, Sherri A.Koenig. "An Application of Lagrange Relaxation to scheduling in power generation systems", Operations research, 25 (3) (1977): 387 – 403.

[14] G.S.Lauer, N.R. Sandell, D.P. Bertsekas, T.S.Posbergh, Solution of Large scale optimal Unit Commitment problems, IEEE Transactions on Power Apparatus and Systems,101 (1) (1982): 79 - 86.

[15] Charles W.Richter, Gerald B.Sheble, "A profit based unit commitment GA for the competitive environment", IEEE Transactions on Power systems 15, 2 (2000): 715 – 722.

[16] A F.Zhaung and F.D.Galiana, "Simulated Annealing Approach to Unit Commitment solution", *IEEE Transactions on Power Systems* 5(1) (1990) : 311 – 317.

[17] K.ShantiSwarup, P.V.Simi, "Neural Computation using discrete and continuous hopfield networks for power system economic dispatch and unit commitment" *NeuroComputing* 70 (2006): 119 – 129.

[18] D.P.Bertsekas, J.N. Tsitsikilis. "Neuro Dynamic Programming" Athena Scientific, Belmount MA., 1996.

[19] R.S.Sutton, A.G.Barto. "Reinforcement Learning : An introduction", MIT Press, Cambridge, MA, 1998