Fifth IFSA World Congress (1993), 972-974

CELL STATE SPACE ALGORITHM AND NEURAL NETWORK BASED FUZZY LOGIC CONTROLLER DESIGN

BaoSheng Hu(IEEE Senior Member) GenYa Ding

The system enginerring institute of Xian Jiao University, Xian 710049 China

Abstract—This paper presents a new method to automatically design fuzzy logic controller(FLC). The main problems of designing FLC are how to optimally and automatically select the control rules and the parameters of membership function (MF). Cell state space algorithms (CSS), differential competitive learning (DCL) and multilayer neural network are combined in this paper to solve the problems. When the dynamical model of a control process is known. CSS can be used to generate a group of optimal input output pairs(X,Y) used by a controller. The(X,Y)then can be used to determine the FLC rules by DCL and to determine the optimal parameters of MF by multilayer neural network trained by BP algorithm.

Keywords: Cell state space algorithm, Neural networks, Fuzzy logic controller design

INTRODUCTION

Fuzzy logic has been widely used in control systems. Nonlinear, time—varying, ill—defined systems can be efficiently controlled by FLC. When the dynamic model of process is known, we can develop FLC for this process either by the expert experience to the process or by the known dynamic model[9]. For very complex nonlinear system it is difficult to use traditional techniques to control it. Developing FLC to deal such complex system is an alternative. CSS[6][7][8] introduced by C. S. Hsu is an effective algorithm to globally analyse nonlinear systems. The optimal control of nonlinear system has been realized by CSS[8]. The optimal control tabel generated by CSS is used to realize optimal discrete control. Optimal continuous control can be realized by FLC after the input output data pairs are extracted from the optimal control table and are used to generate the FLC rules and to fine tune the parameters of MF.

When a group of data pairs of a control pro cess are known, DCL can be used to generate the FLC rules[1][2]. A multilayer neural network is used to realize the optimal seclection of the parameters of MF.

The paper in [3] proposes using CSS to generate the optimal control inputs—outputs, and uses the inputs—outputs to determine the consequent part of the fuzzy rules. The conseqent is not fuzzy set, but is a linear function of the fuzzy rules' input variable (the form is a0 + a1 * x1 + a2 * x2...). The a0,a1,a2... are determined by gradient decent algorithm. The paper in [4] pretends the inputs—outputs of a process are known. Then the paper trys to automatically generate FLC rules by a multilayer neural network. This paper integrates the ideas in [1][2][3][4][9]and forms a new method to generate FLC rules.

CELL STATE SPACE ALGORITHM

CSS is an algorithm to generate the optimal control table. CSS is developed by cell to cell mapping which is originated from point to point mapping represented by x(n)=G(x(n—1),u(n),t(n)), x(n) is state vector, t(n) is the time interval during control u(n). For every control process only a finite region to which the state vectors belong is needed to be considered. So we can represent the finite region by a number of cells. Each N—dimensional vector (x1,x2,...,xn) correspond to N—dimensional cell (z1, z2,..., zn), z1, z2, ..., zn are integers, (zi—1/2) * hi=<xi<=(zi+1/2) * hi, hi is the interval size for state xi. After the cells (z1, z2,..., zn) are generated. we have a cell to cell mapping equation: z(n)=C(z(n—1), u(n), t(n)), u(n) belongs to U, t(n) belongs to J. There are Nu elements in U and Nt elements in J. z(n)= C(z(n—1), u(n), t(n)) represents Nu * Nt possible mappings for each state z(n—1). When z(n—1) is mapped to z(n), there is a cost increment w(n)=Q(z(n—1), u(n), t(n)). If a dynamic process moves from z(0) through z(1), z(2),..., to arrive at z(ne), under a sequence of controls{u(j), t(j)}, j= 1,2,...,ne, then the value of the cost function V for this dynamic process is given by V(p)=V(z(0) ,z(ne),u,t)=Q(z(0),u(1),t(1))+Q(z(0),u(1),t(1))+...+Q(z(ne—1),u(ne),t(ne)). To realize the CSS, we must first specify the possible cost increments w1<w2<w3<.... From the possible increments we can form all the possible cost levels of the processv1<v2<v3<.... vi=k1 * w1+k2 * w2+...+kp * wp, ki>=0. If the target state is z(nt), the aim of the CSS is to find the optimal control table for each state z(j). The optimal control table includes: U(j): identifies the control vector u to state z(j).

T(j): identifies the control time t during U(j). Image (j): identifies the image state of z(j) under this optimal interval control. W(j): cost increment for z(j). V(j): total cost moving to target state z(nt) for z(j).

S(j): the number of steps moving to target state z(nt) for z(j).

The optimal control table is generated by dynamic programming approach. Starting with target state z(ne) and the smallest cost level v1, we find all the states that map to z(nt) at cost v1. We place the states and the z(nt) in set Y. Thenstarting with cost level v2 and each state in the Y, we find all the states that map to z(nt) at cost level v2. This process continues until every state is processed.

After the optimal control table is generated, the array U (j) denotes the control input for each state z(j), z(j) is the cell of state x(j). So we can use (x(j), U(j)), j=1,2,..., N. (N is the total number of cells that can reach to the target state) to design a FLC by DCL and multilayer neural network.

DIFFERENTIAL COMPETITIVE LEARNING

For a controlller, the input is x(j), the output of the controller is U(j). We pretend x(j) is two dimensional, represented by E(j) and CE(j). U(j) is one dimensional. We divide E(j), CE(j), U(j) into d1, d2 and d3 intervals, respectively. The intervals are not overlapped, each interval denotes a term of a linguistic variable. So it is possible to have D=d1 * d2 * d3 fuzzy rules. We then use a two-layered network (Fig. 1) to train (E(j), CE(j), U(j)), j=1,2,..., N.

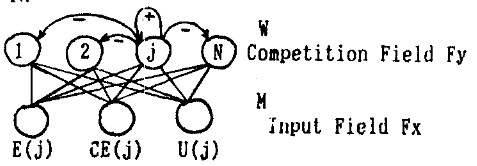

Fig. 1. Topology of the laterally inhibitive DCL network

$$M = \begin{vmatrix} m11 & m21 & \dots & mn1 \\ m12 & m22 & \dots & mn2 \\ m13 & m23 & \dots & mn3 \end{vmatrix}$$

$$M = \begin{vmatrix} 2 & -1 & -1 & \dots & -1 \\ -1 & 2 & -1 & \dots & -1 \\ & & \cdot & & \\ & & \cdot & & \\ -1 & -1 & -1 & \dots & 2 \end{vmatrix}$$

mij is the weight between neuron i in input field Fx and neuron j in competition field Fy. The DCL algorithm that trains the network (Fig. 1) is as follows: We represents (E(i), CE (i), U(i)) by x(i).

(1). initialize mi(0)=x(t). i=1,2,...,D.

(2). Find winning mj(t): ||mj(t)−x(t)||=min||mi (t)−x(t)||. i=1,2,...,D.

(3) Update winning mj(t):

mj(t)=mj(t)+Ct * ISj(yj(t))[S(x(t))−mj (t)] if the jth neuron wins, mi(t+1)=mi(t) if the ith neuron loses. ISj(yj(t)) denotes the time change of the jth neuron's competition signal Sj(yj) in the competition layer Fy:

ISj(yj(t))=sgn[Sj(yj(t+1))−Sj(yj(t))]. sgn (x)=1 if x>0, sgn(x)=0 if x=0, sgn(x)=−1 if x<0. yj(t+1)=yj(t)+ $\sum$ Sj(xi(t)) * mij(t)+ $\sum$ Sk(yk(t)) * Wkj.

After all the data pairs E(i), Ce(i), U(i) i=1,2,..., N, have been trained, the N columns of matrix M indicate the distribution of E, CE and U in the three dimensioanl space. Each column (mi1, mi2, mi3) represents a fuzzy rule whose interval is (It1, It2, It3), mi1, mi2, mi3 are in interval It1, It2, It3, respectively. The corresponding fuzzy rule is If E is L(it1) and CE is L(it2) then U is L(it3). L(iti) corresponds to the term whose interval is iti.

The intervals of MF in fuzzy rules produced by DCL is not overlapped, so the output of the FLC is not continuous and smooth. We can use multilayer neural network to fine tune the MF parameters.

MULTILAYER NEURAL NETWORK

A neural network which realizes a FLC is illustrated at Fig. 2. Two inputs, one output fuzzy controller is represented by Fig. 2. The linguistic variable E, CE and the output linguistic variable have three terms only.

The neural network is composed of three parts: the first part is fuzzify that turns linguistic variable into MF values. The second part is the inference part. Every link in this part representsa rule. So there are nine rules at Fig. 2. The lower nodes in this part execute the precondition of a rule and the higher nodes execute the consequence of a rule. The third part is the defuzzify which is used to get the nonfuzzy control values.

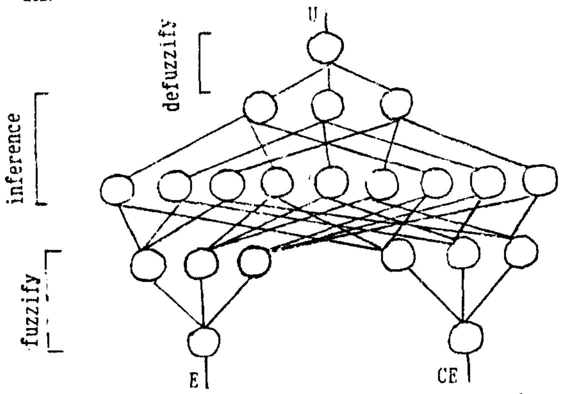

Fig. 2 A Multilayer neural network used to fine tune the parameters of the membership functions.

All MF are represented by bell-shaped function exp(−(ui−mij)/oij), ui is the input, mij is the center of the MF, oij is the width of the MF. The weights in the fuzzify part is mij. The nodes function in this part is exp(−(ui−mij)/oij). The weights between the fuzzify part and the inference part are 1. the weights in the inference part are 1. The lower nodes function in this part is min(u1,u2,. . ,up). The higher nodes function is min(1,u1+u2+. . . +up). The weights in the defuzzify part is mij * oij. The nodes function in this part is (mij * oij) * ui/ oij * ui. We have used DCL to determine the FLC rules. The intervals of MF have been fixed by hand. So we can map rules and the MF on this neural network. For a linguistic variable E, it has three terms corresponding to three intervals[I1,I2],[I2,I3] and [I3,I4].

The first term width o11 is I2−I1, its center m11 is (I1+I2)/2. So o12=I3−I2,m12=(I2+I3)/2,o13=I4−I3, m13=(I3+I4)/2.

The parameters oij,mij can be fine tuned by this neural network using BP—algorithm and the training datas(x(j),U(j)) j=1,2,. . . ,N, which generated by the CSS. The training time is substantially reduced for the oij and mij have been determined coarsely.

CONCLUSION

Using the modern computer speed and storage to solve problems will get great benefits. The method proposed in this paper is computational — oriented. It is interesting to get a group of rules automatically from some complex linear or nonlinear differential equations. The CSS produces the optimal input U(j) for each state z(j). If all the optimal control data pairs (z(j),U(j)) are trained on the multilayer neural network successfully, then the parameters oij and mij is optimal. If the training does not converge, then it is necessary to redivide the state z(j) intervals needed by DCL and to use DCL to get FLC rules once more, and to map the FLC rules to the multilayer neural network, and to train this network again until the training finally converges. The final FLC produced in this way will be near optimal.

REFERENCES

[1] B. Kosko, Neural Network and Fuzzy systems: A Dynamical systems approach to machine intelligence, Englehood Cliffs, NJ: Prentice—Hall, 1992

[2] Seong — gon Kong and Bart KosKo, Adaptive Fuzzy systems for backing up a truck — and — trailer, IEEE Tran. on Neural networks, vol. 3, no. 2, march 1992

[3] Samuel M. Smith and David J. Comer, Autoamted Calibration of a fuzzy logic controller using a cell state space algorithm, IEEE control systems, august 1991

[4] Chin — Teng Lin and C. S. George Lee, Neural — network — based fuzzy logic control and decision system, IEEE Tran. on Computer, vol. 40, no. 12, dec. 1991

[5] C. S. Hsu, A theory of cell—to cell mapping dynamical systems, J. Appl. Mechan. , vol. 47, pp. 931 — 939, Dec. 1980

[6] C. S. Hsu and R. S. Guttalu, An unravelling algorithm for global analysis of dynamical systems: an application to cell to cell mappiing, J. Appl. Mechan. vol. 47, pp. 940—948, dec. 1980

[7] C. S. Hsu, A discrete method of optimal control based upon the cell state space concept, J. optimization theory and Appl. vol. 46, no. 4, pp. 547—569, aug, 1985

[8] Peng Xian—Tu, Generating rules for fuzzy logic controller by functions, Fuzzy sets and systems 36(1990) 83—89