Transportation and Traffic Theory

Proceedings of the Eleventh International Symposium on Transportation and Traffic Theory, held July 18-20, 1990, in Yokohama, Japan

Editor
Masaki Koshi
Institute of Industrial Science
University of Tokyo
Minato-ku, Tokyo, Japan

Elsevier
New York • Amsterdam • London

Parallel Distributed Processing on Neural Network for Some Transportation Equilibrium Assignment Problems

Takashi Akamatsu [1], Yuji Tsuchiya [2], and Toshikazu Shimazaki [3]
[1][2][3] Department of Civil Enginnering, University of Tokyo, Tokyo, Japan

ABSTRACT

This paper presents a *parallel processing* method for the transportation equilibrium assignment problem. The method based on the *neural network theory* would substantially reduce the computation time, using a *neural computer* which will be available in near future. The neural network consists of mutually connected "*neurons*", which change the states in parallel to the descent direction of the Lyapunov function. In the method proposed in this paper, the link flows are represented by the neural states, and we can obtain the equilibrium flows by the parallel change of the neural states according to the equations of neural dynamics. Using the correspondence of the Lagrangian of the transportation equilibrium problem to the Lyapunov function of the neural network, the neural interconnections and the equations of neural dynamics are derived for the deterministic multi--commodity equilibrium assignment problem. Furthermore, we extend the method so as to be applicable to the stochastic equilibrium assignment. The method are implemented in an actual network and the numerical properties are investigated.

1. INTRODUCTION

The transportation network equilibrium assignment model, which originates from one step in conventional transportation demand forecasting, has grown to one of the most important model in various area, such as transportation planning, land use planning, traffic management and so on. Whenever we apply this model to a real transportation network, the model often includes the numerous number of variables and requires the vast calculation costs. For the reduction of the computational burden, various methods, such as, the Frank-Wolfe method(1)(2), the projection gradient method(3), the sub-gradient method(4), have been proposed. Even these methods, however, are not sufficient to deal with the large scale network in practice. Accordingly, we often find the situations that these transportation models which have been developed to obtain useful information for planners are not fully utilized.

So far as we premise to use the conventional serial processing computer, drastic improvement of this problem may be difficult. However, if there are some parallel processing methods based on a proper principle, it may be possible to solve the problem very rapidly by the parallel calculation, since all the variables used in the network assignment model have similar expressions and structures.

In this research, we pay attention to the neural network theory which has high possibility of realizing the parallel calculation, and aim to develop the method for applying the theory to the transportation equilibrium assignment problem.

In the next chapter, we briefly summarize the neural network theory concerning this research. The third chapter shows the method of applying the theory to the user equilibrium assignment problem. After the illustration of the method in a simple example, the method for a general network is presented. In the fourth chapter, we extend the method to the stochastic user equilibrium assignment problem. In this case, it is difficult to directly apply the neural

Published 1990 by Elsevier Science Publishing Co., Inc.
Transportation and Traffic Theory
M. Koshi, Editor

network method because of the path variables included in the conventional equivalent optimization problem. Alternatively, we first develop an equivalent program represented by only link variables, and then we apply the neural network method to this program. In the fifth chapter, we analyse conditions of the stable convergence in this method. Next, we implement the method in an actual network, and investigate the numerical properties of this method. Finally, the features of the method are discussed comprehensively and the directions of the future research are mentioned.

2. NEURAL NETWORK THEORY

In this chapter, we briefly summarize the neural network theory, concentrating on the themes concerning with this research. General and detailed explanations of the theory can be seen, for example, in references(5) ~ (11).

2.1 Model of Neuron

The brain of a living thing consists of many "neurons" mutually interconnecting and forming the network. The schematical model is shown in Figure 1.

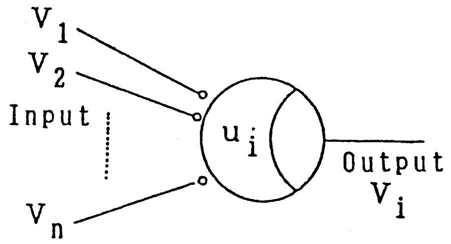


Figure 1 Model of Neural Network.

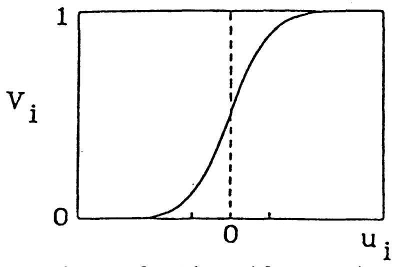


Figure 2 Sigmoid Function.

This figure means that the multiple input signals for each neuron are the output from other many neurons, and these input signals are converted in the neuron, and becomes the output. Hopfield model(9) is one of the most popular models for the conversion forms in a neuron. This model can be expressed as

$$d u_i / d t = - u_i + \sum_j T_{ij} V_j + I_i \tag{1}$$

$$V_i [t] = g (u_i [t]) \tag{2}$$

where, u i, V i, I i are the inner state, the output state, and the inner bias of the *i* th neuron respectively. T is a matrix which represents the strength of interconnections between neurons. The function *g* is a proper sigmoid type function, for example, binary logit function as follows:

$$g (u) = \{ 1 + \tanh(u / \rho) \} / 2 \tag{3}$$

where ρ is the parameter which represents the sensitivity of neuron, and is often called "temperature" from the analogy with statistical mechanics. We call Eqs.(1)-(3) the equations of neural dynamics below.

2.2 Lyapunov Function of Neural Network

When the equations of neural dynamics and the interconnection matrix T are given, we can calculate how the neurons change the states. It is difficult, however, to recognize the global dynamics of the neural network system from the direct calculation of each neural equation. Thus, we consider the "energy" of the neural network by Lyapunov function, which gives the information on the global characteristics of dynamical system. Hopfield(9) showed that the function:

$$E (V) = - \frac{1}{2} \sum_i \sum_j T_{ij} V_i V_j - \sum_i I_i V_i + \sum_i \int_0^{Vi} g^{-1}(v)\, dv , \tag{4}$$

is the Lyapunov function corresponding to the equations of neural dynamics (1)-(2). That is, Eqs.(1)-(2) change the state to the descent direction of E, and the stable point is the state where E is minimized. In other words, if the neurons change each input/output state asynchronously and in parallel, the neurons have the stable state at the minimum of the Lyapunov function. Note that the third term of Eq.(4) represents randomness arising from the continuity of the value of neural state **V**, and we call it "neural entropy term".

2.3 Application to Parallel Processing of Optimization Problem

According to the theory above, the neurons change their states to the direction minimizing E (**V**) spontaneously. Therefore, an optimization problem can be solved by changing the state of each neuron according to Eq.(1) and (2) in parallel, if the variables are represented by the states of neurons and the objective function can be expressed in the form of Eq.(4).

Based on this idea, Hopfield and Tank(10), and Takeda and Goodman(11) showed that a class of combinatorial optimization problem can be solved on neural network efficiently. However, the applications to the optimization problem of other calsses are quite sparse , since their methods for representing the constraints in optimization problem on neural Lyapunov function are ad hoc and they can not be used in general.

3. APPLYING NEURAL NETWORK THEORY TO EQUILIBRIUM ASSIGNMENT PART I : DETERMINISTIC EQUILIBRIUM PROBLEM

It is widely known that the transportation network user equilibrium problem can be represented as the optimization problem when the Jacobian of the link performance functions is symmetric. Therefore, if we can find out the proper "number representation" and the Lyapunov function corresponding to the assignment problem on neural network, we must be able to apply the theory. In this research, we adopt the number representation that the link flows are represented by the sum of neural output states, and consider the correspondence between the augmented Lagrangian of equivalent optimization problem for the transportation equilibrium model and the Lyapunov function of neural network.

3.1 Simple Example

First, we explain the application method of the neural network theory to the equilibrium assignment problem on a simple example network as shown in Figure 3. The user equilibrium assignment problem in this network is equivalent to the following optimization problem,

[P 1]

$$\min.\ Z(\mathbf{x}) = \sum_a \int_0^{x_a} c_a(\omega)\,d\omega$$

subject to

$$h(\mathbf{x}) = q - \sum_a x_a = 0$$

$$x_a \geqq 0$$

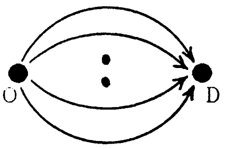


Figure 3 Example Network

where, q is the OD flow, x_a is the flow on link a, and c_a is the link performance function of link *a*, which is continuous and monotonically increasing function.

To solve this problem by the neural network theory, we must determine how to represent the link flows $\mathbf{x}$ on neural network. For this "number representation", we assume that the link flows are not continuous but discrete variables, that is , the link flows take integer value measured by certain unit which represents number of vehicles. Next, we consider many neurons corresponding to these unit flows, and regard the output state of each neuron as the unit flow.

According to this "number representation" method, the link flows $\mathbf{x}$ are represented by the sum of output states of the neurons assigned to each link; that is,

$$x_a = \sum_n V_{an} \tag{5}$$

where V_{an} denotes the output state of the *n* th neuron assigned on link *a*. Also, to represent the objective function of the equilibrium assignment problem by the discrete unit flow, we assign each neuron the value of link performance function, c_{an}, corresponding to the value of the link flow represented by the neuron as shown in Figure 4. As seen from Figure 4, program [P1] can be replaced by

[P 1']

$$\min.\ Z(V) = \sum_a \sum_n c_{an} V_{an}$$

subject to

$$h(V) = q - \sum_a \sum_n V_{an} = 0$$

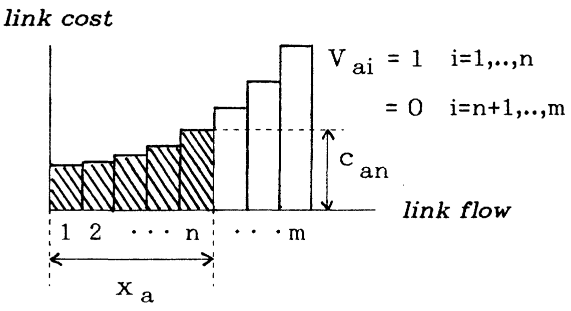


Figure 4 Rrepresentation of Link Flow.

Since problem [P1'] is a constrained optimization problem, we can not yet directly apply the idea of neural network theory. To take the constraints into consideration, we utilize the augmented Lagrangian which converts the constrained problem into unconstrained one in multiplier method(12):

$$E = Z(V) + \mu h(V) + R\{h(V)\}^2 / 2 \tag{6}$$

where μ is the Lagrange multiplier and R is a parameter of appropriate value.

According to the theory of multiplier method, revising the $\mathbf{V}$ to the decent direction of E by proper method, and revising μ by

$$\mu[t+1] = \mu[t] + R \cdot h(V[t]) \tag{7}$$

where *t* in bracket [] denotes the number of iteration,

the neural state $\mathbf{V}$ converges to the optimal solution of original problem.

On the other hand, the Lyapunov function of this problem corresponding to Eq.(4) is represented by

$$\tilde{E}(V) = -\frac{1}{2} \sum_{an} \sum_{bm} T_{anbm} V_{an} V_{bm} - \sum_{an} I_{an} V_{an}\ . \tag{8}$$

Letting E (V) – constant equals $\tilde{E}(V)$, comparing the expansion of Eq.(8) and the coefficient of Eq.(8), we obtain

$$T_{anbm} = -R \qquad \forall a,n,b,m \tag{9}$$

$$I_{an} = Rq + \mu - c_{an} \qquad \forall a,n \tag{10}$$

The derived matrix T means that all the neurons are mutually connected with same "strength" R, and the neurons show "inhibitory" property. Also, the equations of dynamics for the *n* th neuron of link *a* becomes

$$du_{an} / dt = -u_{an} + \sum_{bm} T_{anbm} V_{bm} + I_{an} \tag{11}$$

$$= -u_{an} + R\,(q - \sum_{bm} V_{bm}) + \mu - c_{an}\,, \tag{12}$$

$$V_{an}[\,t\,] = \{\,1 + \tanh(\,u_{an}[\,t\,]/\,\rho\,)\}/\,2 \quad . \tag{13}$$

Thus, if we connect the neurons mutually according to the interconnection matrix T on a neural network computer, the neurons change the states according to the equations of dynamics (12)(13) spontaneously, and we can obtain the user equilibrium flow pattern as the equilibrium state of the neurons.

To check this fact numerically in conventional digital computer, the equations of dynamics (12) should be replaced with the following discrete-time difference equations.

$$u_{an}[\,t\,] = R\{\,q - \sum_{bm} V_{bm}[\,t - \Delta_{bm}]\} + \mu - c_{an} \tag{14}$$

where Δ_{bm} is the time-interval of changing the state of the *bm* th neuron. Note that the temperature parameter ρ at equilibrium should be taken as very small value for the consistency of the analysis, since we derived the equations of dynamics assuming that the neural entropy term of the Lyapunov function can be dropped because it is negligibly small. For this reason, we adopt the method analogous to the simulated annealing method (13), that is, ρ is first set high value and then ρ is "cooled" down gradually and finally decreased to the level that the entropy term can be regarded as zero. Detailed discussion on the relation between the value of this temperature parameter ρ, the penalty parameter R and the convergence condition will be shown in the fifth chapter. Here, we can demonstrate the simple numerical experiment of the method on digital computer, revising ρ by

$$\rho = \rho\max\,/\,\log(\,t+1\,) \qquad (\,t \geqq 1\,), \tag{15}$$

where ρ max : a constant of given positive value; t : number of iteration,

and changing the state of neurons according to Eq.(7),(14), and (12). Some conditions of the numerical experiment are as follows:

① Number of links : 2 ;
② OD flow : 20.0;
③ Link performance function : $c_1 = 200 + 0.02\ x_1^4$, $c_2 = 300 + 0.15\ x_2^4$
④ Value of temperature parameter : ρ max=100;
⑤ Value of penalty parameter : $R = \rho\,/\,2$
⑥ Initial state of the neurons : $V_{an}[0] = 0.5 \quad \forall an$

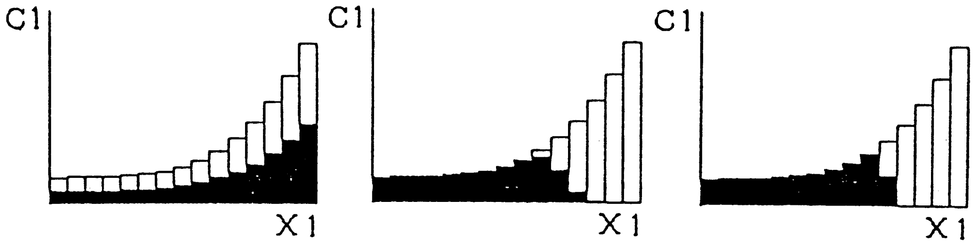


Figure 5 (a) t=1 (b) t=3 (c) t=10

Figures 5 (a)-(c) show the changes of neural states in link 1 to the convergence. The width of each bar means the unit link flow represented by one neuron, and the ratio of the hight of shaded area against the bar represents the value of output state (from 0 to 1) of each neuron. We can see that initially random valued neurons gradually converge to the ordered state of user equilibrium.

3.2 Analysis in General Network with Many to Many OD pairs

We extend the previous analysis for the one OD pair problem to the general network with many to many OD pair. The equivalent optimization problem of user equilibrium assignment in this case can be represented by

$$\min.\ Z(\mathbf{x}) = \sum_{ij} \int_{o}^{x_{ij}} c_{ij}(\omega)\, d\omega \tag{16}$$

subject to

$$\tilde{h}_k^s(\mathbf{x}^s) = q_{ks} + \sum_i x_{ik}^s - \sum_j x_{kj}^s = 0 \qquad \forall k \neq s \tag{17}$$

$$\tilde{g}_{ij}(\mathbf{x}, \mathbf{x}^s) = x_{ij} - \sum_s x_{ij}^s = 0 \qquad \forall (ij) \tag{18}$$

$$x_{ij}^s \geqq 0 \qquad \forall (ij), s \tag{19}$$

where q_{ks} is the OD flow from node k to node s, x_{ij} and c_{ij} denotes the flow and the performance function of link $i \rightarrow j$, respectively, and x_{ij}^s denotes the flow with destination s on link $i \rightarrow j$. As in previous section, we represent the link flows by the sum of the activated (output) state of neurons. Also, in order to express the flow conservation constraints on the neural network, we consider two groups of neurons, V^o, V^s, corresponding to the link flows, $\mathbf{x}$, and the link flows with destination s, $\mathbf{x}^s$, respectively. The link flows without distinguishing destinations are represented by

$$x_{ij} = \sum_n V_{ijn}^o \qquad \forall (ij) \tag{20}$$

and then the objective function Z can be represented as the function of $\mathbf{V}$.

$$Z(\mathbf{x}) = Z(\mathbf{V}) = \sum_{ij} \sum_n c_{ijn} V_{ijn}^o \tag{21}$$

where c_{ijn} denotes the value of the link cost function at $x_{ij} = n$ (unit flow).

Also, the link flows with destination s, $\mathbf{x}^s$, is represented by

$$x_{ij}^s = \sum_n V_{ijn}^s \qquad \forall s, (ij) \tag{22}$$

and then the constraints $\tilde{h}$ can be transformed into

$$h_k^s(\mathbf{V}) = q_{ks} + \sum_i \sum_n V_{ikn}^s - \sum_j \sum_n V_{kjn}^s = 0 \qquad \forall s, k \tag{23}$$

Furthermore, to satisfy the constraints $\tilde{g}$, we introduce next constraints for $\mathbf{V}^o$ and $\mathbf{V}^s$.

$$g_{ijn}(\mathbf{V}) = V_{ijn}^o - \sum_{s \neq 0} V_{ijn}^s = 0 \qquad \forall (ij), n \tag{24}$$

Consequently, the augmented Lagrangian for this problem can be represented with respect to $\mathbf{V} = (\mathbf{V}^o, .., \mathbf{V}^s, ..)$ as follows:

$$E = Z(\mathbf{V}) + \sum_s \sum_k \mu_k^s h_k^s(\mathbf{V}) + \sum_{ij} \sum_n \lambda_{ijn} g_{ijn}(\mathbf{V})$$
$$+ \frac{R_1}{2} \sum_s \sum_k \{h_k^s(\mathbf{V})\}^2 + \frac{R_2}{2} \sum_{ij} \sum_n \{g_{ijn}(\mathbf{V})\}^2 \tag{25}$$

where μ and λ are the dual variables corresponding to the constraints h and g, respectively. On the other hand, the Lyapunov function of this problem corresponding to Eq.(4) becomes

$$\tilde{E} = -\frac{1}{2} \sum_{ij} \sum_n \sum_s \sum_{i'j'} \sum_{n'} \sum_{s'} T_{ijni'j'n'}^{ss'} V_{ijn}^s V_{i'j'n'}^{s'} - \sum_{ij} \sum_n \sum_s I_{ijn}^s V_{ijn}^s \tag{26}$$

where $\tilde{E}(\mathbf{V})$ is $E(\mathbf{V})$ − constant. Comparing the expansion of Eq.(25) and the coefficient of Eq.(26), we obtain the interconnection matrix, T, and the vector of neural threshold value, I.

(i) Case for $s = 0$

$$T^{ss'}_{ijn,i'j'n'} = R_2 (1 - 2\,\delta_{s'0})\, \delta_{ii'}\, \delta_{jj'}\, \delta_{nn'} \tag{27}$$

$$I^{s}_{ijn} = - c_{ijn} - \lambda_{ijn} \tag{28}$$

(ii) Case for $s \neq 0$

$$T^{ss'}_{ijn,i'j'n'} = R_1 \delta_{ss'} (\delta_{ij'} + \delta_{ji'} - \delta_{ii'} - \delta_{jj'}) - R_2 (1 - 2\,\delta_{s'0})\, \delta_{ii'}\, \delta_{jj'}\, \delta_{nn'} \tag{29}$$

$$I^{s}_{ijn} = R_1 (q_{is} - q_{js}) + \lambda_{ijn} + (\mu^{s}_{i} - \mu^{s}_{j}) \tag{30}$$

where δ denote Kronecker's delta; that is,

$$\delta_{xy} \begin{array}{ll} = 1 & \text{if } x = y \\ = 0 & \text{otherwise.} \end{array}$$

We can see from the derived matrix T that the *n* th neuron assignned for x_{ij} has the interconnections between the *n* th neurons for the link flows with destinations *s*, x^{s}_{ij}, corresponding to the constraints g in the case of (i). In the case of (ii), the interconnections have the structure similar to the dual graph of the original transportation network, corresponding to the flow conservation constraints h.

Substituting the matrix T and the vector I into the basic equations of dynamics, we can derive the following equations of dynamics.

$$V^{s}_{ijn} [t] = \{ 1 + \tanh(u^{s}_{ijn}[t] / \rho) \} / 2 \tag{31}$$

(i) case for $s = 0$ (neurons for link flows)

$$d u^{s}_{ijn} / d t = - u^{s}_{ijn} - \{ c_{ijn} + \lambda_{ijn} + R_2\, g_{ijn}(V) \} \tag{32}$$

(ii) case for $s \neq 0$ (neurons for link flows with destination s)

$$d u^{s}_{ijn} / d t = - u^{s}_{ijn} + (\mu^{s}_{i} - \mu^{s}_{j}) + \lambda_{ijn} + R_1 \{ h^{s}_{i} (V) - h^{s}_{j} (V) \} + R_2\, g_{ijn} (V) \tag{33}$$

In order to see the interconnection between neurons, we derived the matrix T by comparing the coefficients, and then obtained the equations of dynamics by substituting the matrix T into the basic equations of neural dynamics. Note that the equations of the dynamics can be also obtained by merely differentiating Eq.(25) with respect to V, since the neural equations represent the dynamics to the descent direction of the Lyapunov function.

4. APPLYING NEURAL NETWORK THEORY TO EQUILIBRIUM ASSIGNMENT PART II : STOCHASTIC EQUILIBRIUM PROBLEM

4.1 Equivalent Optimization Problem

In this chapter, we develop the neural network method for solving stochastic user equilibrium (SUE) model which is known as a generalization of deterministic user equilibrium (e.g.(14)-(17)). From the practical view point, we concentrate on the case for logit type path choice model below. Conventionally, the minimization problem developed by Fisk(15) has been known as an equivalent program for the logit based SUE assignment model. The program, however, includes the path flow variables which are troublesome to deal with in general network, and it is difficult to directly apply our neural network method by the similar manner in earlier deterministic equilibrium case. Thus, alternatively, we derives the equivalent optimization problem represented by only link flows, and apply the neural network method to the link based program.

To show the essence of "decomposition into link based program" without notational complexity, we first consider the case for flow independent stochastic assignment problem with one OD pair and unit OD flow.

In the logit type stochastic assignment model, the probabilities of choosing path p, P_p, is given by

$$P_p = \frac{\exp[-\theta C_p]}{\sum_{p'} \exp[-\theta C_{p'}]}, \tag{34}$$

where θ is the dispersion parameter for path choice, and C_p is the travel cost of path p, which is the sum of the link cost t on the path. The probability (flow) of choosing link $i \rightarrow j$, p_{ij}, is determined by the flow conservation equations represented by the link and path flows:

$$p_{ij} = \sum_p P_p \delta_{ij,p}, \tag{35}$$

where $\delta_{ij,p} \begin{cases} =1\text{: if link } i \rightarrow j \text{ is on path } p, \\ =0\text{: otherwise.} \end{cases}$

The flow pattern obtained by this assignment model has the property of "Marcov chain"; that is,

$$\prod_{ij} \left[\frac{\text{Prob}[i,j]}{\text{Prob}[j]} \right]^{\delta_{ij,p}} = P_p \tag{36}$$

where $\text{Prob}[j] = \sum_m p_{mj}$, $\text{Prob}[i,j] = p_{ij}$.

In other words, if a certain link flow pattern satisfies

$$\prod_{ij} \left[\frac{p_{ij}}{\sum_m p_{mj}} \right]^{\delta_{ij,p}} = \frac{\exp[-\theta C_p]}{\sum_{p'} \exp[-\theta C_{p'}]}, \tag{37}$$

then the flow pattern is consistent with the logit type assignment model.

Keeping in mind the above property of logit based stochastic assignment model, let's consider the following program:

[SA]

$$\min. Z(p) = \frac{1}{\theta} \{ \sum_{ij} p_{ij} \ln p_{ij} - \sum_j (\sum_i p_{ij}) \ln (\sum_i p_{ij}) \} + \sum_{ij} p_{ij} c_{ij}$$

subject to

$$\sum_i p_{ik} - \sum_j p_{kj} + (\delta_{rk} - \delta_{sk}) = 0 \qquad \forall k$$

$$p_{ij} \geqq 0 \qquad \forall ij$$

where r is the origin and s is the destination node.

The solution of this problem has the property of Marcov chain, and the resulting *link flow* pattern is identical to those obtained by logit type assignment model (Eq.(34)(35)). In other words, this program is equivalent to the logit type assignment model in terms of the *link flow* pattern. The equivalency can be shown by comparing the optimality conditions of this program and the properties of Marcov chain mentioned above (Eq.(37)). Using the Lagrangian, which is given by

$$L(p, \mu) = Z(p) + \sum_k \mu_k \{ \sum_i p_{ik} - \sum_j p_{kj} + (\delta_{rk} - \delta_{sk}) \}, \tag{38}$$

we can express the optimality conditions for program [SA] as follows:

$$\frac{\partial L}{\partial p_{ij}} \geqq 0 \ , \qquad p_{ij}\frac{\partial L}{\partial p_{ij}} = 0 \ , \qquad \forall ij \tag{39}$$

and the flow conservation constraints.

After some algebraic calculation, the optimality conditions Eq.(39) yield

$$\frac{p_{ij}}{\sum_m p_{mj}} = \exp\,[\,-\theta\,\{\,c_{ij} - (\,\mu_i - \mu_j)\}\,]\,. \qquad \forall ij \tag{40}$$

Multiplying the both side of Eq.(40) on appropriate path p,

$$\prod_{ij}\left|\frac{p_{ij}}{\sum_m p_{mj}}\right|^{\delta_{ij,p}} = \prod_{ij} \exp\,[\,-\theta\,\{\,c_{ij} - (\,\mu_i - \mu_j)\}\,]^{\delta_{ij,p}} \tag{41}$$

$$= \exp\,[\,-\theta\,\{\,C_p - (\,\mu_r - \mu_s)\}\,] \tag{42}$$

Furthermore, considering the flow conservative in Eq.(40), we have

$$\exp[\,\theta\,\mu_j\,] = \sum_i\,\{\exp[\,-\theta\,c_{ij}\,]\exp[\,\theta\,\mu_i\,]\}\,. \tag{43}$$

Eq.(43) shows that the value of μ at node j is determined by the value of μ at node i which has links entering to node i (in other words, nodes i are just preceeding to node j). Thus, sorting all the node from destination s to origin r by appropriate order (s,s−1,s−2,· · ·,r+2,r+1,r) and evaluating the value of μ on this order of node reversely,

$$\exp[\,\theta\,\mu_s\,] = \sum_{s-1}\{\exp[\,-\theta\,c_{s-1,s}\,]\exp[\,\theta\,\mu_{s-1}\,]\}$$

$$= \sum_{s-1}\{\exp[\,-\theta\,c_{s-1,s}\,]\sum_{s-2}\exp[\,-\theta\,c_{s-2,s-1}\,]\cdot\cdot\cdot\cdot\cdot \times \exp[\,\theta\,\mu_r\,]\}$$

$$= \sum_{s-1}\sum_{s-2}\cdot\cdot\sum_{r+1}\exp[\,-\theta\,(\,c_{s-1,s} + c_{s-2,s-1} + \cdot\cdot + c_{r+1,r}] \times \exp[\,\theta\,\mu_r\,]$$

$$= \sum_p \exp[\,-\theta\,C_p] \times \exp[\,\theta\,\mu_r.] \tag{44}$$

Hence,

$$\mu_r - \mu_s = \ -(\,1\,/\,\theta\,)\ln \sum_p[\,-\theta\,C_p]\,. \tag{45}$$

Substituting Eq.(45) into Eq.(42), we have Eq.(37) as the optimality condition for the program [SA]. This means that we can obtain the link flow pattern consistent with logit model by solving the program [SA].

Similarly, we can obtain an equivalent program for SUE with many to many OD pairs represented by only link variables as follows:

$$\min.\ Z\,(\,\mathbf{x}\,) = \frac{1}{\theta}\sum_s\{\,-H\,L_s + H\,N_s\} + \sum_{ij}\int_0^{x_{ij}} c_{ij}(\,\omega\,)d\,\omega \tag{46}$$

subject to

Eq.(17), Eq.(18), and inequalities (19),

where $$H\,L_s(\,\mathbf{x}^s\,) \equiv \ -\sum_{ij} x^s_{ij}\ln x^s_{ij} \ ; \tag{47}$$

$$H\,N_s(\,\mathbf{x}^s\,) \equiv \ -\sum_j(\,\sum_i x^s_{ij}\,)\ln(\,\sum_i x^s_{ij}\,)\,. \tag{48}$$

4.2 Deriving the Equations of Neural Dynamics

In order to apply the neural network method, it is rather convenient to convert the entropy term in the objective function Z into the following "conditional entropy" form:

$$\sum_s \{ -HL_s + HN_s \} = \sum_s \{ \sum_j (\sum_i x^s_{ij})(\sum_i p^s_{ij} \ln p^s_{ij}) \}$$

$$= \sum_s \{ \sum_j (\sum_i x^s_{ij}) \sum_i (\int_0^{p^s_{ij}} \ln \omega \, d\omega + p^s_{ij}) \} \quad (49)$$

where $p^s_{ij} \equiv x^s_{ij} / \sum_i x^s_{ij}$.

As in deterministic case, we represent the link flows by the sum of the output state of neurons. In addition to the neural variables $\mathbf{V} = (\mathbf{V}^{\circ},.., \mathbf{V}^{s},..)$ in deterministic equilibrium case, we further introduce a new set of neurons representing the "turn probabilities", $\mathbf{p}$, as follows:

$$\sum_n V^{s1}_{ijn} = p^s_{ij} \quad (50)$$

where the superscript "*1*" denotes this set of neurons, and the other types of neurons representing the link flows are distinguished by superscript "*0*" below. From the definition of $\mathbf{p}$ and the other neuron variables representing the link flows, these variables should satisfy the following equations .

$$f^s_{ij} (V^{s1}_{ij\cdot}, V^{s0}_{ij\cdot}) \equiv x^s_{ij} - p^s_{ij} \sum_i x^s_{ij}$$

$$= \sum_n V^{s0}_{ijn} - (\sum_n V^{s1}_{ijn})(\sum_i \sum_n V^{s0}_{ijn}) = 0 \quad (51)$$

Similar to the discrete representation in the integral term of the link cost function, we can express the integral of the entropy term by

$$\sum_s \sum_j (\sum_i x^s_{ij})(\sum_i \int_0^{p^s_{ij}} \ln \omega \, d\omega) = \sum_s \sum_j (\sum_i \sum_n V^{s0}_{ijn}) \{ \sum_i \sum_n L_n V^{s1}_{ijn} \} , \quad (52)$$

where L_n denotes the value of function *ln*(·) corresponding to the *n* th division of the interval [0,1]. Consequently, we have the following augmented Lagrangian with respect to $\mathbf{V}$, adding the constraints Eq.(51) and the entropy term to those of the deterministic equilibrium case.

$$E = \sum_{ij} \sum_n c_{ijn} V_{ijn} + \sum_s \sum_{ij} (\sum_{i'} \sum_n V^{s0}_{i'jn}) \{ \sum_n (1 + L_n) V^{s1}_{ijn} \} / \theta$$

$$+ \sum_s \sum_k \mu^s_k h^s_k (\mathbf{V}) + \sum_{ij} \sum_n \lambda_{ijn} g_{ijn} (\mathbf{V}) + \sum_s \sum_{ij} \nu^s_{ij} f^s_{ij} (\mathbf{V})$$

$$+ \frac{R_1}{2} \sum_s \sum_k \{ h^s_k (\mathbf{V}) \}^2 + \frac{R_2}{2} \sum_{ij} \sum_n \{ g_{ijn} (\mathbf{V}) \}^2 \quad (53)$$

where μ, λ and ν are the dual variables corresponding to the constraints $\mathbf{h}$, $\mathbf{g}$, and $\mathbf{f}$, respectively. On the other hand, the Lyapunov function of this problem corresponding to Eq.(4) becomes

$$\tilde{E} = -\frac{1}{2} \sum_{ij} \sum_n \sum_s \sum_d \sum_{i'j'} \sum_{n'} \sum_{s'} \sum_{d'} T^{ss'dd'}_{ijni'j'n'} V^{sd}_{ijn} V^{s'd'}_{i'j'n'} - \sum_{ij} \sum_n \sum_s \sum_d I^{sd}_{ijn} V^{sd}_{ijn}$$

$$(54)$$

Similar to the deterministic equilibrium case, we obtain the following interconnection matrix, T, and the vector of threshold value, I.

$$\begin{aligned} T^{ss'dd'}_{ijn,i'j'n'} = {} & \delta_{d0}\,\delta_{d'0}\,[\,\delta_{s0}\,\{R_2(1-2\delta_{s'0})\,\delta_{ii'}\,\delta_{jj'}\,\delta_{nn'}\} \\ & + (1-\delta_{s0})\{-R_2\,(1-2\delta_{s'0})\,\delta_{ii'}\,\delta_{jj'}\,\delta_{nn'} \\ & - R_1\,\delta_{ss'}(\delta_{ij'}+\delta_{ji'}-\delta_{ii'}-\delta_{jj'})\}\,] \\ & - \delta_{d0}\,\delta_{d'1}\,\delta_{jj'}\,\delta_{ss'}(1-\delta_{s0})\,(1+L_{n'}-\nu^{s}_{ij})/\theta \end{aligned} \tag{55}$$

$$\begin{aligned} I^{sd}_{ijn} = {} & \delta_{d0}\,[\,(1-\delta_{s0})\{R_1(q_{is}-q_{js})+\lambda_{ijn}+(\mu^{s}_{i}-\mu^{s}_{j})-\nu^{s}_{ij}\} \\ & - \delta_{so}(C_{ijn}+\lambda_{ijn})\,] \end{aligned} \tag{56}$$

The first line of Eq.(55) means the interconnection of the neurons for s=0 (the neurons representing link flows), the second line is those for s ≠ 0 (the neruons representing link flows with destination s), the third line is common to all the neurons excluding "turn probability neurons", and the last line shows the connection for the "turn probability neurons".

Substituting the matrix T and the vector I into the basic equations of dynamics, we can derive the following equations of dynamics for three types of neurons.

$$V^{sd}_{ijn}[t] = \{1+\tanh(u^{sd}_{ijn}[t]/\rho)\}/2 \tag{57}$$

<u>(i) case for $s=0$ and $d=0$ (neurons for link flows)</u>

$$du^{sd}_{ijn}/dt = -u^{sd}_{ijn} - \{c_{ijn}+\lambda_{ijn}+R_2\,g_{ijn}(V)\} \tag{58}$$

<u>(ii) case for $s \neq 0$ and $d=0$ (neurons for link flows with destination s)</u>

$$\begin{aligned} du^{sd}_{ijn}/dt = {} & -u^{sd}_{ijn} + (\mu^{s}_{i}-\mu^{s}_{j}) + \lambda_{ijn} - \nu^{s}_{ij} \\ & + R_1\{h^{s}_{i}(V)-h^{s}_{j}(V)\} + R_2\,g_{ijn}(V) \\ & + \{\nu^{s}_{ij} - L_n - 1\}\sum_i\sum_n V^{s0}_{ijn}/\theta \end{aligned} \tag{59}$$

<u>(iii) case for $d=1$ (neurons for "turn probabilities")</u>

$$du^{sd}_{ijn}/dt = -u^{sd}_{ijn} + (\nu^{s}_{ij} - L_n - 1)\sum_i\sum_n V^{s0}_{ijn}/\theta \tag{60}$$

Thus, we know that the stochastic equilibrium flows can be calculated on neural network, by simply connecting the neurons according to the derived matrix T and changing the neural states, as in deterministic case.

5. STABILITY ANALYSIS

Replacement of the continuous-time differential equations (Eq.(12)) with the discrete-time equations (Eq.(14)) sometimes causes the unstable dynamics of neurons, depending on the value of parameter ρ and R. Thus, this chapter analyses the conditions that the equations of dynamics always converge to the equilibrium solution. To avoid the notational complexity, we analyse 1 OD pair problem here, which is expressed by the following equations of dynamics.

$$V_{an}[t+1] = g_{an}[V[t]] \equiv g[u_{an}(V[t])] \tag{61}$$

where

$$g(u_{an}) = [\,1 + \exp(-2\,u_{an} / \rho)\,]^{-1} \tag{62}$$

$$u_{an} = R \cdot (q - \sum_{bm} V_{bm}) + \mu - c_{an} \tag{63}$$

In the case for many to many OD pairs, the basic way of analysis is same as the one OD pair case, though the neural equations corresponding to Eq.(62) becomes more complicated.

Suppose that **V** (t) are at the equilibrium state **V** *, and that the output state of the *n* th neuron of link *a* is changed to $V_{an}^{*} + \varepsilon_{an}$. Since **V** * satisfies the flow conservation, the output state at time t+1 becomes

$$V_{an}[t+1] = [\,1 + M \cdot \exp(-2\,\varepsilon_{an} R / \rho)\,]^{-1}, \tag{64}$$

$$\text{where} \quad M \equiv \exp\{(\mu - c_{an}) / \rho\}. \tag{65}$$

On the other hand, the condition for stable convergence may be expressed as follows:

$$|\,V_{an}[t+1] - V_{an}[t]\,| < |\,\varepsilon_{an}|, \tag{66}$$

Substituting Eq.(64) into Eq.(66) and rearranging (R, ρ, M > 0) it, we have

$$1 - \varepsilon_{an}(1 + 1/M)(1 + M \cdot L) < L \tag{67}$$

$$\text{where} \quad L \equiv \exp(-2\,\varepsilon_{an} R / \rho). \tag{68}$$

Taking the logarithm of both side, expanding by progression, and neglecting the higher order term, we have

$$2R / \rho < (1 + 1/M)(1 + M \cdot L). \tag{69}$$

Since ε_{an} is small enough, we can see L = 1 , and then the right hand side have the minimum value of 2 ρ at M = 1 ($\mu = c_{an}$) . Hence, the most strong condition for convergence is expressed as $\rho > R/2$.

Generalizing the analysis above, we can obtain the following conditions for convergence based on the *contraction mapping theorem.*

$$\|\,J\,g(\mathbf{V})\,\| < 1, \tag{70}$$

where J **g** (**V**) is Jacobian of **g** (**V**). If we assume the changing ratio of μ and the distribution of neural states **V**, we can derive the convergence conditions for general case from Eq.(70). However, this condition is too strong for the practical calculation, since we do not have to consider the extreme case that *all* the neurons *always* converge. Besides, we do not always need the link flows by OD pairs $\mathbf{x}^{s}$, but rather need the link flow without distinguishing OD pairs **x**. Therefore, we may be able to achieve the convergence according to more weak conditions in many cases. Unfortunately, the analytical derivation of the weak convergence condition in general case would be extremely complicated, and the practical assumption on the distribution of neural states is not clear. Thus, we investigate the convergence properties of this method by some numerical experiments in the following chapter.

6. NUMERICAL EXPERIMENTS

We apply the method of neural network (we call it MNN below) to solve the user equilibrium assignment problem in practical size network, and the accuracy of the solution and the convergency of the method are investigated in this numerical experiment.

We implement the method in a road network of Sioux Falls city in South Dakota, which has 76 links, 24 nodes (all the nodes are OD nodes), and 529 OD pairs. The network is same one where Leblanc et al (1) investigated the performance of Frank-Wolfe method for solving the fixed demand user equilibrium assignment problem, and the OD flows and the link performance function parameters are the same as the Leblanc's experiment.

The conditions in MNN are summarized as follows:

① Number of neurons per link: 30
② Revising equation of temperature parameter ρ : Eq.(15) , ρ max = 10.0
③ Revising equations of penalty parameter R i (i=1,2):

$$R_i = R\max - A \times t, \quad R\max = 15.0, \quad A = 0.01$$

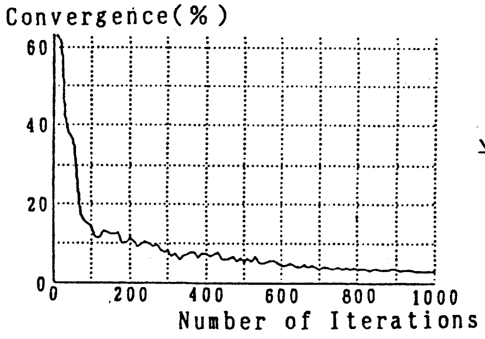
④ Revising equations of Lagrange multipliers:

$$\mu_k^s[t+1] = \mu_k^s[t] + R_1 \cdot h_k(V[t]) \qquad (71)$$

$$\lambda_{ijn}[t+1] = \lambda_{ijn}[t] + R_2 \cdot g_{ijn}(V[t]) \qquad (72)$$

⑤ Equations of neural dynamics: Eq.(31),(32),(33)

Figure 6 shows the convergence pattern of MNN in this experiments. From the theoretical view point, we can not obtain the rigorous solution by MNN, since there exists the neural entropy and we adopted the discrete representation of the link flows . The solution is, however, very close to the rigorous solution, when we set the lower bound of parameter ρ as in this numerical example. Also, the correlation coefficient between the rigorous solution and the neural network solution after the 1000 th change of neural states is 0.996. It is appropriate to note from these fact that MNN can produce the solution which has practical accuracy.



"Convergence (%)" is defined as follows:

$$\frac{\sqrt{\sum_{ij}(x_{ij}[t] - x^*_{ij})^2} \Big/ \sqrt{\text{Number of Links}}}{\sum_{ij} x^*_{ij}} \times 100$$

where x^* is the rigorous solution.

Figure 6 Convergence Pattern of Neural-Network Method

Next, we examine the influence of the change of parameter value on the convergence. Although we implemented the experiments for various cases, it is difficult to show all the results comprehensively in this paper. We concentrate on showing some important facts founded in the experiments.

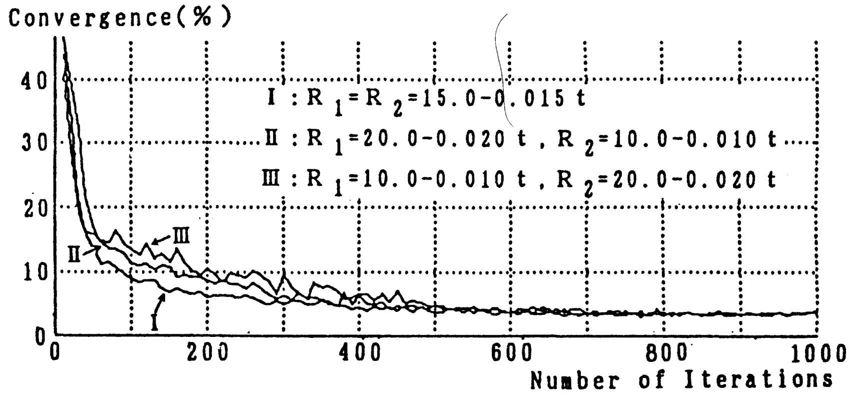


Figure 7-1 Convergence patterns for the various relations between R1 andR2, where ρ is determined by Eq.(15) and ρ max=10.0 for all cases.

Figure 7-1 depicts the comparative experiments of convergence pattern with the change of the relation between R 1 and R 2. Judging from our some experiments (in addition to this example), it is appropriate to make R 1 and R 2 almost same value.

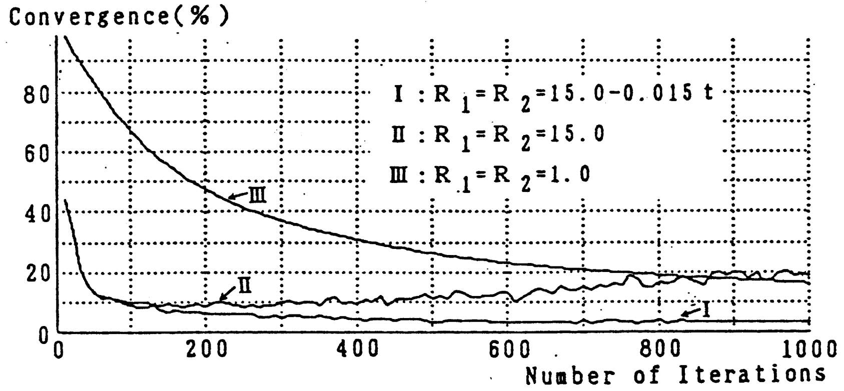


Figure 7-2 Convergence patterns for the various values of R, where ρ is determined by Eq.(15) and ρ max=10.0 for all cases.

Also, we had better decrease R i in accordance with the convergence. Figure 7-2 shows the example suggesting this fact. When R i is large throughout the iteration, the neural state often oscillates , and furthermore, it converge to the point different from rigorous solution. The reason can be infer that if R i has large value, the errors caused by the discrete representation scheme of the link flows are amplified. In the case that R i is too small, the convergence rate is very slow, though the neural states shows stable convergence.

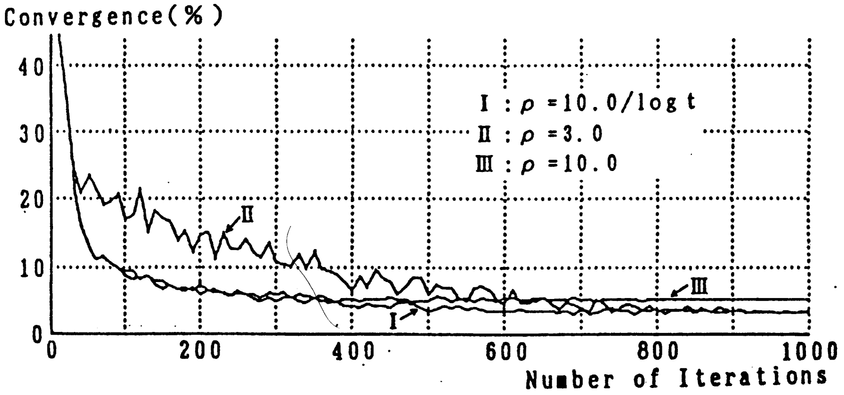


Figure 7-3 Convergence patterns for the various values of ρ, where R is determined by Ri=15.0 - 0.015 × t (i=1,2) for all cases.

As for ρ, we can suggest that ρ should have large value at the first stage of the iteration (the change of neural state), and ρ should be gradually decreased. As seen from Figure 7-3, when ρ is too small from beginning to end, the convergence rate is slow, and the neural states easily oscillate. On the other hand, when we set large value for ρ throughout the iteration, the converged states are different from the rigorous solutions, though the neurons behave the stable convergence. The reason is that "neural entropy" term in Lyapunov function can not be negligible when ρ has large value.

7. DISCUSSIONS

7.1 Features of Neural Network Method

Changing the state of all the neurons in the neural network method corresponds to the calculation of one iteration in conventional algorithm. Therefore, if a neural computer available with parallel processing is realized in future, the computational time required in each iteration may be negligibly small, and the total time to solve the equilibrium problem can be radically reduced, comparing with conventional algorithm.

Besides this features, our method can easily solve the equilibrium assignment problem with capacity constraints. Specifically, the capacity constrained assignment problem can be solved just by adjusting the number of neurons representing the link flows, since the flows above capacities can not be loaded on the links because of the relation between the link flow and the neurons in our number representation method. This can be one of the advantages over the complicated treatment (for example, the method of obtaining initial feasible solution and line search must be devised in Frank-Wolfe method) in conventional algorithm.

Moreover, the simplicity of the method may be the one of advantages. In emulating the method in conventional computer, the required code in a module for the calculation is less than some dozens of lines when we use proper computer language(eg. FORTRAN, C, PASCAL etc), since we only to describe the procedures for the iterative calculation of the equations of dynamics.

We described some advantages of neural network method above. We examine the deficiencies of our method. Firstly, it may be posed a question that the number of neurons required in this method becomes too numerous. Judging from our some numerical experiments, assigning only a few scores of neuron per link is enough for practical accuracy ($2 \sim 3$ digit) of the solution. Besides, we can utilize some hundred thousands of neurons (6) even in the currently available neuro-computer emulation system. Considering these facts, it would not be the serious problem in practical.

Next, the problem of convergency and convergence rate in this method can be pointed out. Since the times required in changing the state of all the neurons are very small because of the parallel dynamics of neurons, the problem of the convergence rate can not be fatal, even if the convergence rate is slow to some extent. However, whether the neural dynamics converges to the equilibrium solution or not is an important problem. For the sure convergence, it is required to make the value of R small to some degree in response to the value of ρ. On the other hand, too small value of R slows down the convergence rate.

Finally, the accuracy of the solution, which is also relating to the problem above, should be considered. Since we derived the equations of dynamics neglecting the neuron entropy term in Lyapunov function, our method can not necessarily produce the rigorous solution from the theoretical view point. As seen from the numerical experiments in previous chapter, however, satisfactory accuracy may be obtained practically with sufficient number of neurons and low temperature. Nevertheless, the number of neurons and the value of ρ should be determined by the trade off relation between the accuracy and the convergence rate, since too many neurons and too small value of ρ worsen the convergence condition.

Last two problems may depend on the following factors:
① the number representation method on neural network;
② the method of setting the Lyapunov function;
③ the method of discretizing the continuous equations;
④ the value and changing ratio of temperature parameter ρ ;
⑤ the value and changing ratio of penalty parameter R ;
⑥ the unit flow represented by ar neuron;
⑦ the congestion rate in transportation network;
⑧ the form of the link performance functions;

⑨ network scale(number of links, nodes, OD pairs etc.), and
⑩ network structure.

First three factors (① ~ ③) are concerned with the formulation of the problem. The next threes (④ ~ ⑥) are the problem of relation between parameter value and convergency, and the remainings (⑦ ~ ⑩) are the problem of "congeniality" between the properties of network and the method which has been also discussed in conventional algorithm.

7.2 Correspondence between Neural Network Method and Other Theories

The features of MNN considered in this paper are summarized as follows:
① link flows are treated as the set of the discrete unit trip by the neurons;
② the discrete neuron variables are processed in parallel;
③ Lyapunov function of neural network coincides with the augmented Lagrangian of assignment problem;
④ Lagrange multipliers are revised according to the theory of multiplier method,

In examining these features, we can see the characteristics common to the theory for other problem. In relation to ① and ②, We might think of the relation with the "cost – efficiency theory" by T.E.Smith(18). He has derived the user equilibrium assignment from the consideration in the case that network flows are represented as discrete variables. Since our MNN represents the link flows as the discrete variables by many neurons and both theory of the neural network and cost-efficiency have common methodology similar to statistical mechanics, we may infer that our MNN has certain relation with his theory.

Concerning ③ and ④, we can see that, in the method by Hopfield(10), the parameters corresponding to the Lagrange multiplier are determined empirically by trial-and-error, and the value of neural threshold (I) and the interconnection strength (T) are constant. On the other hand, our method is generalized to be able to treat with general case, since the Lagrange multiplier (μ, λ) and the value of neural threshold (I) are not constant but are revised with the change of neural state. Note that cost variables (μ, λ) also can be represented on the neural networks, though we represent, for simplicity, only the flow variables on neural network in this paper. The reason and the way can be shown below. Our method in this paper is equivalent to solve the primary problem by the certain gradient method, since the equations of dynamics of neural state **V** representing the flow variables move to the gradint direction (with respect to **x**) of the augmented Lagrangian. On the other hand, the method of revising (μ, λ) in this paper is equivalent to solving the dual problem by a gradient method. Therefore, revision of (μ, λ) can be described by the equations of dynamics of neural state, as in primary problem. In other words, though our MNN in this paper iterates to solve the primary problem on neural network and to solve the dual problem by the conventional gradient method, the latter also can be calculated on the neural network. Thus, if we make two stratified neural network, that is, the network for the flow variables and for the cost variables, we can obtain the solution only on the neural networks by changing the state of neurons in each network in turn, giving and taking the mutual information.

8. CONCLUSIONS

This paper considered a method of parallel processing on neural network for solving some equilibrium assignment problems. The major remarks are summarized below.
(1) Parallel processing method based on the neural network theory is proposed for solving the transportation equilibrium assignment problem. More specifically, the method is based on the correspondence between the augmented Lagrangian of equivalent optimization problem and the Lyapunov function of neural network.
(2) The interconnection matrix and the equations of dynamics of neurons for the parallel calculation on neural networks are derived for the user equilibrium assignment problem with many to many OD pairs.
(3) Applying this method to practical network, the convergence properties and

the accuracy of the solution are investigated and the results are encouraging for the practical use.
(4) An equivalent optimization problem for logit based stochastic equilibrium assignment model represented by only link variables is formulated. Using this program, it is shown that neural network method can be applied to the stochastic equilibrium assignment problem as in deterministic case.
(5) It is suggested that the neural network method proposed in this paper has some correspondences between T.E.Smith's cost-efficiency principle.

REFERENCES

(1) LeBlanc,L.J., Morlok,E.K. and Pierskalla,W.P., "*An efficient approach to solving the road network equilibrium traffic assignment problem,*" Trans.Res.9(5), pp.309-318, 1974.
(2) LeBlanc,L.J., Helgason,R.V. and Boyce,D.E., "*Improved efficiency of the Frank-Wolfe algorithm for convex network programs,*" Trans.Sci.19(4), pp.445-462, 1985.
(3) Inouye,H., "*A computational method for equilibrium traffic assignment,*" Proc. of JSCE.313, pp.125-133, 1981.(in Japanese)
(4) Fukushima,M., "*On the dual approach to the traffic assignment problem,*" Trans.Res.18B, pp. 235-245,1984
(5) Amari,S. , "*Mathematical Theory of Neural Network,*" Sangyotosho, Tokyo, 1978.(in Japanese)
(6) Asou,H. , "*Information Processing on Neural Network,*" Sangyotosho, Tokyo, 1988.(in Japanese)
(7) Edited by Amari,S. , "*Neural Network,*" Mathematical Science, No.289, Science-sha, Tokyo, 1987. (in Japanese)
(8) Kohnen,T. "*Self-Organization and Associative Memory,*" Springer-Verlag, New York, 1984.
(9) Hopfield, J.J., "*Neurons with graded response have collective computational properties like those of two-state neurons,*" Proc.Natl.Acad.Sci. USA.81, pp.3088-3092, 1984.
(10) Hopfield, J.J. and Tank,D.W., "*Neural computation of decisions in optimization problems,*" Biological Cybernetics.52, pp.141-152, 1985.
(11) Takeda,M. and Goodman,J.W., "*Neural networks for computation: number representations and programming complexity,*" Applied Optics.25(18), pp.3033-3046, 1986.
(12) Bertsekas,D.P., "*Constrained Optimization and Lagrange Multiplier Methods,*" Academic Press,1982.
(13) Kirkpatrick,S., Glatt,C.D., and Vecchi, M.P., "*Optimization by simulated annealing,*" Science 220, pp.671-680, May 1983.
(14) Sheffi,Y., "*Urban Transportation Networks,*" Prentice-Hall,INC.,1985.
(15) Fisk, C.S., "*Some developments in equilibrium traffic assignment,*" Trans.Res.14B(3), pp.243-255, 1980.
(16) Daganzo, C.F., "*Unconstrained extremal formulation of some transportation equilibrium problems,*" Trans.Sci.16(3), pp.332-360,1982.
(17) Akamatsu,T. and Matsumoto,Y., "*A stochastic network equilibrium model with elastic demand and its solution method,*" Proc. of JSCE.396, pp.109-118, 1989.(in Japanese)
(18) Smith,T.E., "*A cost-efficiency theory of dispersed network equilibria,*" Environ.Plan.A20, pp.231-266,1988.