A Chaotic Neural Network Algorithm for Task Scheduling in Overlay Grid

CAO Huai-hu, YU Zhen-wei, WANG Yin-yan
(Dept. Computer of China University of Mining and Technology Beijing, Beijing 100083, china)
chhu@cumtb.edu.cn

Abstract: Task scheduling is one key issue in computational overlay grid. For finding solution to this problem, we propose a Chaotic Neural Network Algorithm for Task Scheduling, the goal of this algorithm is to minimize the total execution time of task by appropriately mapping the task to processor in the parallel system. We formulate the task scheduling problem that considers heterogeneousness. To evaluate the effectiveness of this algorithm, we compare this algorithm with the simulated annealing, and the genetic algorithm etc. The average experimental result of the algorithm is obviously superior to other conventional algorithms.
Key words: Hopfield neural network, chaotic neural network, combinatorial optimization, task scheduling algorithm

1. Introduction

Task scheduling is one key issue in computational overlay grid, promising in utilizing geographically separated heterogeneous computational resources to solve large-scale complex scientific problems. Given a parallel program to be executed on a multiprocessor system, the scheduling problem consists of finding a task schedule that minimizes the entire execution time. The execution time yielded by a schedule is usually called makespan. A solution to a scheduling problem is an assignment for each task of a starting time and a processor. Optimizing allocation under time and precedence constraints in a multiprocessor system is an NP-hard problem in general [1, 2, 3].

Task scheduling methods can be divided into list heuristics and meta-heuristics [4]. List heuristics assign each task a priority and sort them in decreasing order. As processors become available, the task with the highest priority is selected and allocated to the most suited processor. Most of them are efficient but often can't obtain reasonable solutions in all situations. Meta-heuristics, known as Genetic Algorithms, is a guided random search method which mimics the principles of evolution and natural genetics [9]. Because genetic algorithms search optimal solutions from entire solution space, they often can obtain reasonable solutions in all situations. Nevertheless, their main drawback is to spend much time doing scheduling. Hence, we propose a new algorithm to overcome these drawbacks.

In this paper, first, we formulate the task scheduling problem on an overlay grid environment. Then we construct a framework of the task scheduling problem that uses the chaotic neural network, and propose a Chaotic Neural Network Algorithm for Task Scheduling (CNNTS), one that utilized for controlling heuristic methods applicable to very large problems. The experimental result shows that this method has an ability to find best optimum solutions. Finally, to evaluate the effectiveness of this method, we compare the performance of the proposed algorithm with the simulated annealing, and the genetic algorithm, etc. The average experimental result of the algorithm is obviously superior to other conventional algorithms.

2. Scheduling problem description and model

To formulate the problem, we consider T_n independent user task agent n={1,2,….N} on R_m heterogeneous resources m={1,2,….,M} with an objective of minimizing the completion time and utilizing the resources effectively. The speed of each resource is expressed in number of cycles per unit time, and the length of each task in number of cycles. Each task T_n has processing requirement P_j cycles and resource R_m has speed of S_i cycles/second. Any task T_n has to be processed in resource R_m, until completion[4].

More formally, the generalized assignment problem can be stated as follows.

Let I = {1,…, m} be the set of processors and J = {1,…n} the set of task agents. For each processor i we are given a resource capacity $b_i > 0$. For each $i \in I$ and each $j \in J$ we are given costs $c_{i,j} > 0$ and resource requirements $r_{i,j} > 0$ for assigning task agent j to processor i.

The objective is to find an assignment matrix x = ($x_{i,j}$), with $x_{i,j}$ = 1 if processor i performs task j and 0 otherwise, which minimizes the total costs.

$$c(x) = \sum_{i=1}^{m} \sum_{j=1}^{n} c_{i,j} \cdot x_{i,j} \qquad (1)$$

Subject to

$$\sum_{j=1}^{n} r_{i,j} \cdot x_{i,j} \le b_i , \forall i \in I , \qquad (2)$$

$$\sum_{i=1}^{m} x_{i,j} = 1, \forall j \in J , \qquad (3)$$

$$x_{i,j} \in \{0,1\}, \forall i \in I , \forall j \in J \qquad (4)$$

The constraint of capacity (2) ensure that the total resource requirement of the jobs assigned to each agent do not exceed its capacity. The assignment constraints (3) guarantee that each job is assigned to exactly one agent. The next section will provide a new task scheduling strategy for grid.

3. Need new task scheduling strategy for grid

As mentioned before, most scheduling algorithms consider about homogeneous platform and performance of each platform is same. But such network computing like Grid, heterogeneous systems are gathered to make collaborative system and they differ at performance. More over these systems are large-scale and the number of processors and task agent are great. So, existing scheduling strategies are not suitable for that kind of system.

3.1 Formulation of task scheduling in overlay grid using Chaotic Neural Network

First, we explain the chaotic neural network and parameters used to express task scheduling problem. Second, we describe the correspondence of a quasi-energy function of the chaotic neural network and the cost function of task scheduling in overlay grid to be minimized.

3.1.1 Chaotic Neuron Model

Chaotic dynamics is complicated dynamics generated by deterministic nonlinear dynamical systems. The chaotic neural network model is based upon the Caianiello neuronic equation (Caianiello, 1961) and the Nagumo-Sato neuron model (Nagumo and Sato, 1972) [5]. By introducing an analog sigmoidal function for the output of such neurons with the refractory effect, the chaotic neuron model can be derived.

The chaotic neural network in this approach is utilized for controlling heuristic methods applicable to very large problems. The performance of this method is higher than those of the conventional searching methods, such as the conventional simulated annealing and tabu searches, even on very large combinatorial optimization problems (Hasegawa et al., 2000, 2002)[6].

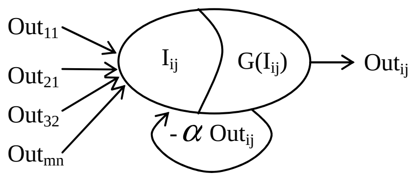

Fig.1. Chaotic Neuron Model

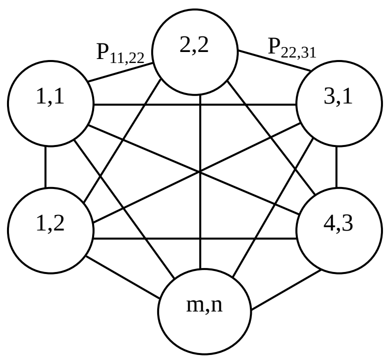

Fig.2. Chaotic Neural Network

The chaotic neural network consists of multiple chaotic neurons (Fig.1.). The chaotic neurons are mutually connected to each other (Fig.2.). The chaotic neuron model(Aihara et al., 1990) is extension of the Caianiello neuronic equation (Caianiello, 1961) and the Nagumo-Sato neuron model (Nagumo and Sato, 1972) which includes a refractory effect. In the proposed method, each) (i,j) th neuron is labeled as follows. The i identifies a task agent. If the number of such task agent is N , the range of i is from 0 to 1 − N . The j is the processor identifier. Each chaotic neuron has an internal state I_{ij}, and receives the output signals from the other connected chaotic neurons. The output value of the chaotic neuron is obtained by using a logistic function G. The chaotic neuron has inhibitory self-feedback (refractory effect) (Fig.1.). Because of this negative self-feedback, the behavior of the chaotic neural network is completely different from that of the Hopfield neural network composed of neurons, which have no self-feedback.

The dynamics on the internal state of the (i,j) th chaotic neuron with graded output and relative refractoriness, $I_{ij}(t)$ is described as follows (Aihara et al., 1990)[7].

$$I_{ij}(t+1) = \sum_{m,n} p_{ij,mn} Out_{mn}(t) + (1-k_d) q_{ij} + k_d I_{ij}(t) - \alpha Out_{ij}(t) \qquad (5)$$

where t is the discrete time step (t = 0,1,2,…),) Out_{mn}(t)is the output of the (m ,n) th neuron with a continuous value between 0 and 1 at time step *t* . In this system, if the output of the chaotic neuron is close to 1, the light source corresponding to this neuron is turned on. $P_{ij,mn}$ is the synaptic connection weight from the (m,n) th neuron, k_d is the decay factor between 0 and 1, q_{ij} is the bias parameter and α is the refractory scaling parameter. The output values of neurons can be obtained by using the following logistic function G.

$$Out_{ij}(t+1) = G(I_{ij}(t+1)) = \frac{1}{1+\exp(\frac{-I_{ij}(t+1)}{\varepsilon})} \qquad (6)$$

where ε is a positive steepness parameter, and the smaller the steepness parameter is, the steeper the slope between 0 and 1 is.

3.1.2 Obtaining Connection Weights and Bias Values

A set of patterns (i.e., specific network state configurations) are stored in the network as attractors of the system dynamics, such that, whenever a distorted version of one of the patterns is presented to the network as input, the original is retrieved upon iteration. The distortion has to be small enough so that the input pattern is not outside the basin of attraction of the desired attractor. In networks using convergent dynamics, the stored patterns necessarily have to be time-invariant or at most, periodic. Chaos provides rapid and unbiased access to all attractors, any of which may be selected on presentation of a stimulus, depending upon the network state and external environment. It also acts as a “novelty detector”, classifying a stimulus as being previously unknown, by not converging to any of the existing attractors. This suggests the use of chaotic networks for auto association[8]. The solution to this task scheduling problem can be evaluated using the following cost function ϕ. The optimum solution corresponds to the minimum of ϕ.

$$\phi = \phi_i + \phi_j \qquad (7)$$

$$\phi_i = \sum_{i=1}^{n} w_{ij} X_{ij} \qquad (8)$$

$$\phi_j = \sum_{j=1}^{m} w_{ij} X_{ij} \qquad (9)$$

ϕ_i denotes the computational request of the task i. ϕ_j denotes the computational power of the processor j. X_{ij} denotes the computational power that processor j provides for task agent i.

3.1.3 Correspondence of Quasi Energy Function and Cost Function

The quasi energy function of this chaotic neural network, on the other hand, can be defined by the following equation.

$$E = -\sum_{i=1}^{n}\sum_{j=1}^{m} w_{ij}\phi_j x_{ij} + \frac{a}{2}\sum_{j=1}^{m}(\sum_{i=1}^{n} w_{ij}\phi_j x_{ij} - \phi_j)^2 + \frac{b}{2}\sum_{j=1}^{m}(\sum_{i=1}^{n} w_{ij}\phi_j x_{ij} - \phi_i)^2 \quad (10)$$

The right first factor of the equation is object function, the second factor is computational power constraint factor, the third factor is task agent request constraint factor.

The quasi energy function is utilized to monitor the dynamical behavior of the chaotic neural network, and it decreases monotonously as time step *t* increases in the conventional Hopfield-Tank neural network (Hopfield and Tank, 1985). This property can be utilized in the task scheduling problem. In this paper, the local minimum problem of such a conventional neural network is solved by applying chaotic dynamics for searching better solutions. Where both a and b are scaling parameters, and let

$X_{ij} = \phi_j x_{ij}$,

$$w_{ij} = \begin{cases} 1, \text{if when the processor j executes task i}. \\ 0, \text{the processor j doesnt execute task i}. \end{cases}$$

For calculating extremum of object function, First, calculate $dE/d\, x_{ij}$, then let $d\, u_{ij}/dt = -dE/d\, x_{ij}$, so get Hopfield-Tank neural network differential form:

$$\frac{d u_{ij}}{dt} = -a\, w_{ij}\phi_j^2 \sum_{k=1}^{n} w_{kj} x_{kj} - b\, w_{ij}\phi_j^2 \sum_{l=1}^{m} w_{il} x_{il} \quad (11)$$
$$w_{ij}(\phi_j + a\phi_j^2 + b\phi_j\phi_i)$$

$$x_{ij} = f(u_{ij}) \quad (12)$$

4. Implement of Chaotic Neural Network Algorithm for Task Scheduling

First, we calculate the value of the difference between the calculated computational power that processors have and the task agent requested computational power at all nodes. Next, we sum up the value and call it the objective function. This objective function is the difference between the calculated solution and the requested condition. Therefore, by minimizing this objective function, we can get the optimum solution. To minimize this objective function, our system uses the chaotic neural network [12].

In order to solve the task scheduling problem in Overlay Grid, we design a Chaotic Neural Network Algorithm for Task Scheduling (CNNTS) as follow:

```
Algorithm1 CNNTS (Ai, Amin, iterate_number, iterate-given)
1.  input  Ai, Amin, iterate_number, iterate-given.
2.  Product a group of vector by chaotic neuron network.
3.  input the group of vector product by step 1 to Hopfield Neuron Network.
4.  loop iterated by Hopfield Neuron Network until convergence to some
     chaotic attractor.
5.  note the energy value Ai of attractor of step 4.
6.  if      Ai < Amin,
     then   Amin= Ai.
7.  if      iterate_number>iterate-given,
     then   putout the Amin as optimization solution,
     else   goto 2.
8.  end
```

5. Experiments and Results

In this section, we compare the performance of the proposed method using the chaotic neural network with the conventional neural network search, the genetic algorithm search, and the simulated annealing search[9,10,11]. We apply these algorithms to different size problems which (processors number, task agent number) are (10,100), (50,200), (500,1000), (1000,2000), (2000,5000). The average percent of finding best optimum solution is shown in Table 1. While the problem is small enough, all algorithms can actually find an optimum solution in 100%. But when the problem becomes large, these results show that only CNNTS can find an optimum solution in 100%. The total execution time of scheduling the tasks to processors is shown in Fig.3. As Fig.3. show, although the run time of the conventional neural network search is smaller than the algorithm CNNTS, the conventional neural network method obviously converged to a local minimum. These results show that the algorithm CNNTS can always find a optimum solution in a reasonable time. The average result of the algorithm CNNTS is obviously superior to the conventional neural network search, the genetic algorithm search, and the simulated annealing search.

Table 1. The average percent of finding an optimum solution

(m,n)	Neural network	Genetic algorithm	Simulated annealing	CNNTS
(10,100)	99.8%	99.9%	100%	100%
(50,200)	95.8%	99.9%	100%	100%
(500,1000)	50.2%	95.2%	96.1%	100%
(1000,2000)	42.3%	93.4%	93.5%	100%
(2000,5000)	30.7%	91.6%	92.6%	100%

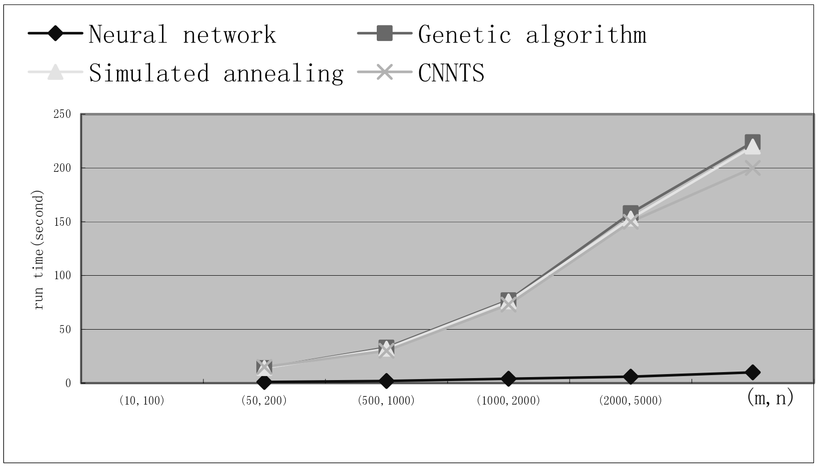

Fig.3. Results compared of CNNTS with other algorthms

6. Conclusion

In this paper, we have proposed the method based on the chaotic neural network to solve that ask scheduling problem. We have compared the proposed method with other methods based on the conventional neural network, the genetic algorithm, and the simulated annealing. First, we have shown that our method and other methods have an ability to find an optimum solution of a small size problem. Then, we have applied these methods to larger problems. Compared with other methods, the proposed method using the chaotic neural network has the property that does not require very much time to find a optimum solution. These experiments imply that as the size of the problem increases, the proposed method shows better performance compared with other methods. It should be noted, however, that the methods with the conventional neural network, the genetic algorithm, and the simulated annealing can be also improved by adding further contrivances. The results show that the chaotic neural network can effectively solve the task scheduling problem.

REFERENCES

1. Buyya R, Abramson D, Giddy J:Grid Resource Management, Scheduling, and Computational Economy,

International Workshop on Global and Cluster Computing, Japan, (2000)
2. Ricardo C. Correa, Afonso Ferreira, and Pascal Rebreyend.:Scheduling Multiprocessor Tasks with Genetic Algorithms. IEEE Transactions on Parallel and Distributed Systems, Vol. 10, No. 8, (Aug. 1999): 825-837
3. Sih GC, Lee EA. A compile-time scheduling heuristic for interconnection-constrained heterogeneous processor architectures. IEEE Trans. on Parallel and Distributed Systems.(1993):175~186
4. Topcuoglu H, Hariri S, Wu MY. :Performance-Effective and low-complexity task scheduling for heterogeneous computing. IEEE Trans. on Parallel and Distributed Systems, (2002):260~274
5. Andreyev Y V , Belsky YL , Dmittriev A S , et al. :neural networks implementation[J] Information processing using dynamical chaos. IEEE Trans Neural Networks , 7.(1996):2902299.
6. W. J. Freeman.:Tutorial on neurobiology: from single neurons to brain chaos. Int. J. Bif. and Chaos. 2. (1992) 451–482.
7. A.Y. Zomaya, C. Ward, and B. Macey.:Genetic Scheduling for Parallel Processor Systems: Comparative Studies and Performance Issues", IEEE Transactions on Parallel and Distributed Systems, Vol. 10, No. 8, (Aug. 1999) :795-812,
8. Y. V. Andreyev, Y. L. Belsky, A. S. Dmitriev and D. A. Kuminov.:Information pro-cessing using dynamical chaos: neural networks implementation. IEEE Trans. on Neural Networks. 7. (1996) 290–299.
9. M. Lin and L.T. Yang.:Hybrid Genetic Algorithms for Scheduling Partially Ordered Tasks in a Multiprocessor Environment.Proc. of 6th International Conference on Real-time Computing Systems and Applications. (1999): 382-387
10. R. Kozma, W.J. Freeman.:Chaotic Resonance – Methods and Applications of Noisy and Variable Patterns. Int. J.Bifurcation & Chaos, 11(6).(2001)2307-2322
11. Barrie J.M., Freeman W.J., Lenhart M.D. :Spatiotemporal analysis of prepyriform, visual, auditory, and somesthetic surface EEGs in trained rabbits, J. of Neurophysiology, 76(1996): 520-539
12. Yamaguch i Y, Ish imura K,W ada M. :Synch ronized oscillation and dynam ical clustering in chaotic PCNN [J]. S ICE, (2002): 730—733