Proceedings of the 6th World Congress on Intelligent Control
and Automation, June 21 - 23, 2006, Dalian, China

Parallel Ant Colony Optimization Algorithm

Hong Liu, Ping Li and Yu Wen
National Laboratory of Industrial Control Technology
Research Center of Intelligent Transportation System
University of Zhejiang
Hangzhou, Zhejiang Province, 310027,China
{HLiu, PLi & YWen}@iipc.zju.edu.cn

Abstract **–Ant colony optimization algorithm is a good way to solve complex multi-stage decision problems. But the construction graph and computation steps for an ant to construct a solution in the construction graph will be exponentially increased, if the stage number and decision variable dimension of complex multi-stage decision problem are increasing. It will cause the ant colony optimization algorithm can not be computed by a single PC. Therefore, a parallel ant colony optimization algorithm based on the construction graph decomposition is presented to solve this problem. The parallel ant colony optimization algorithm decomposes the construction graph into some parts and each part is placed on different PC. The whole computation task is accomplished by mutual cooperation in the PCs which join in the computation. Experiment has verified that it can solve this problem and improve the computation efficiency.**

Index Terms -Parallel Computing, Ant colony optimization algorithm, Multi-stage decision, Construction Graph Decomposition.

I. INTRODUCTION

Multi-stage decision problems can be represented by mathematics formulas as:

$$MIN\{J(N) = F[x(0), s_{0,N}]\}. \quad (1)$$

$$s.t.\ x(t+1) = f[x(0), s_{0,t+1}]. \quad (2)$$

$$s_{0,t+1} = \{u(0), u(1), \cdots, u(t)\}. \quad (3)$$

$$x(0) = x_0. \quad (4)$$

$$x(t) \in X(t) \subseteq R^n. \quad (5)$$

$$u(t) \in U(t) \subseteq R^r. \quad (6)$$

$$t = 0,1,\cdots,N-1. \quad (7)$$

Where $J(N)$ is the value of objective function, F is the objective function, $x(t)$ is the state variable, $X(t)$ is the feasible state set, $u(t)$ is the decision variable, $U(t)$ is the feasible decision set, $s_{O,N}$ is a strategy of the problem, and $s_{o,t+1}$ is a sub-strategy. (2) is the state transition equation, in which the state variable $x(t+1)$ depends on the initial state x_0 and the sub-strategy $s_{o,t+1}$. When F or f are strongly nonlinear or nonanalytic functions and $x(t)$ or $U(t)$ are high dimensional and complex topological space, the problems represented by (1)-(6) is called complex multi-stage dynamic decision problems.

Many combinational optimization problems and optimal control problems can be represented by the above equations, such as the traveling salesman problem, the job schedule problem, and the signal coordinated control problem of urban traffic system. Reference [2] indicates that dynamic programming algorithm and genetic algorithm could not solve complex multi-stage decision problems very well because these problems are so complex. Based on [3] and [4], the ant colony optimization (ACO) algorithm is adopted to solve the complex multi-stage decision problems with an additive and monotonic objective function. It has a very well application in solving the signal coordinated control problem of urban traffic system in [5].

But the construction graph and computation steps for an ant to construct a solution in the construction graph will be exponentially increased, if the stage number and decision variable dimension of complex multi-stage decision problem are increasing. It will cause ACO algorithm can not be computed by a single PC. Parallel computing through a cluster of standard PC which is coupled via standard 100 Mbit Ethernet LAN (Local Area Network) is an available and feasible method to solve this problem. In [6], the parallel ant colony optimization (PACO) algorithm based on the ants (PACO-A) is presented. The PACO-A algorithm establishes the same construction graph in each PC attended the parallel computing first. Then, some ants begin exploring solutions in each PC at the same time. After the exploration, the pheromones in each PC are updated by a special algorithm insuring that the pheromones in each PC are the same. The final solutions are accomplished through ants' continuous exploring. Although the PACO-A algorithm can improve the computation speed, it can still not solve the problem mentioned above. In PACO-A algorithm, the construction graph and computation steps in each PC are the same as the ACO algorithm. The problem is still remained in PACO-A algorithm.

In order to solve the problem mentioned above, a PACO algorithm based on the construction graph decomposition (PACO-CGD) is presented in this paper. The PACO-CGD algorithm decomposes the construction graph into some parts and each part is placed on different PC. Therefore a large construction graph in one PC is decomposed into some small construction graphs scattered at different PC. According to this, the construction graph and computation steps in each PC are reduced and each PC can execute corresponding ACO algorithm. The whole computation task is accomplished by mutual cooperation in the PCs which join in the computation. As the ant exploring can be synchronously executed and the explor-

ing speed is increased in each PC as the construction graphs and the computation steps in each PC are reduced, the computation speed will be increased.

The PACO-CGD algorithm is verified by an experiment in which a multi-stage decision problem with multi-input continuous nonlinear time-variant model is adopted. The experiment also indicates that the PACO-CGD algorithm can improve the computation efficiency.

II. Application of ACO Algorithm To Multi-stage Dynamic Decision Problems

The ACO algorithm uses an ant colony to search the feasible routes with minimum cost over a construction graph $G(V,E)$, where V is a finite set of nodes and E is a finite set of directed links that connect nodes. In the construction graph, nodes represent constitution element of solutions. The node sequences, i.e. routes, correspond to a solution of the problem. When exploring a route in the construction graph, an ant sets out from original node and chooses the next link repetitively in a stochastic way. Therefore, the ant obtains the route through a multi-stage decision process and in fact the route is a solution of the problem.

For example, to apply the ACO algorithm to the multi-stage decision problems with additive and monotonic objective functions, whose objective function is as (8):

$$J(N)=\sum_{t=0}^{N-1} L[x(t),u(t),t], L[x(t),u(t),t]\geq 0 . \quad (8)$$

The decision variable $u(t)$ at each stage is discretized to a series of discrete values $u_l(t)\in U(t)$, $l=1,2,...,q^r$ (suppose that each dimension of the r-dimension decision variable $u(t)$ is discretized to q values, so the number of total discrete values is q^r). $u_l(t)$ is the discrete decision variable, and the set of $u_l(t)$ is denoted as $U_d(t)\subset U(t)$ .Then, the one-to-one mapping between $U_d(t)$ and the nodes of the construction graph can get (9) as:

$$\psi : u_l(t)\in U_d(t)\leftrightarrow v_l^t\in V^t . \quad (9)$$

The mapping ψ results in the layered construction graph in Fig.1. The horizontal layers, which are called the decision layers, correspond to $U_d(t)$ at different stages. For the N-stage decision problems, there are N layers in the layered construction graph. V^t is the tth layer in which v_l^t is the lth node. In the layered graph, ants start from an initial node v_0, choose a node in the next layer repetitively and finally construct a route:

$$\{v_0, v_{n_0}^0, \cdots, v_{n_t}^t, \cdots, v_{n_{N-1}}^{N-1}\} . \quad (10)$$

Where n_t is the index of node selected in the tth layer. To apply the ACO algorithm to the other multi-stage decision problems with more complex objective functions can also use the same way and the detail description can be found in [2]. The algorithm mentioned above is called serial ACO (SACO) algorithm in this paper because it is achieved by a serial way in one PC.

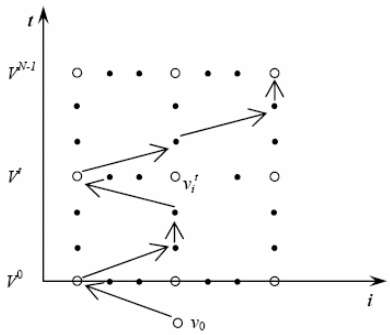

Fig. 1 Layered construction graph

III. PACO Algorithm Based on Construction Graph Decomposition

The total computation complexity of SACO algorithm is researched in [2]. It is $O(CM\sum_{t=0}^{N-1} n_t)$, where M is the number of ants, C is the iterated times for each ant, N is stage number of the multi-stage problem, and n_t is the number of the nodes in the tth layer. When the number of total discrete values is q^r, the total computation complexity is $O(CMNq^r)$. The computation steps for an ant to construct a solution in the construction graph are exponentially increased as the r increases. And C, M and N are the linear factors of it. Furthermore, the memory sizes for storing the layered construction graph also increase exponentially with r. If r, C, M and N are bigger, many memory sizes are needed and the total computation complexity is higher, which needs much time to explore the solution. The memory in a PC is finite and it can not afford very complexity computation task. The parallel computing is a very well way to solve this problem.

A. Construction graph decomposition

The construction graph of SACO algorithm which is adopted in this paper is layered. Based on this idea, the construction graph is decomposed into some parts, which is illustrated in Fig.2. The solid nodes in Fig.2 are called boundary nodes. These nodes decompose the original construction graph into two construction graphs (which can be called partial construction graphs, illustrated in Fig.2 (b) and Fig.2 (c)). If they are placed on two PCs, it is obvious that these PCs can execute the corresponding SACO algorithm at the same time. By this way, the SACO algorithm can be changed into PACO-CGD algorithm and it will be described in detail later. Because the original layered construction graph has been decomposed into two layered construction graphs which are scattered on two PCs, the layered construction graph in each PC is reduced. According to this, the computation complexity and the memory sizes in each PC are reduced.

If there are P PCs attending parallel computing and each of them is encoded from 1 to P, the construction graph decomposition algorithm will be as (11) which has been considered computation load balance. Where g_l is the layer number of construction graph on different PC, *Mod* is to take the remainder function and *Floor* is to take $-\infty$ whole function. Each PC can establish its layered construction graph directly based on the (11). This will be the same as that the construction graph is decomposed into some parts first, and then

decomposed into some parts first, and then placed on different PC. But the layered construction graph can be built at the same time, which can reduce the computation time.

$$Let: Z = Mod((N+P-1)/P).$$
$$If(Z=0): g_i = (N+P-1)/P, 1 \le i \le P.$$
$$If(Z \ne 0): g_i = Floor((N+P-1)/P)+1, 1 \le i \le Z.$$
$$g_i = Floor((N+P-1)/P), Z < i \le P. \quad (11)$$

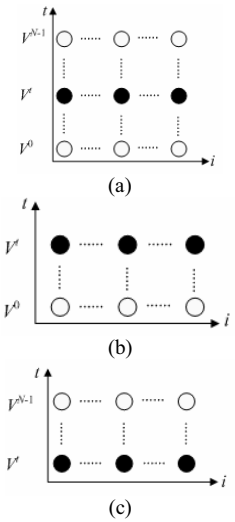

Fig. 2 Sketch map of construction graph decomposition

B. PACO-CGD algorithm

There are two important steps in the SACO algorithm. One is that an ant needs exploring the whole layered construction graph to obtain a feasible solution. The other is that the pheromones of the SACO algorithm need updated before another ant starts exploring. Assuming that there are *P* PCs attending parallel computing and each of them is encoded from 1 to *P*, the PACO-CGD algorithm accomplishes these two steps as follows. First, each PC establishes its layered construction graph based on the construction graph decomposition. Second, an ant starts exploring in PC 1. When the ant finishes its exploring, it will be transferred into the next PC, i.e. PC 2, with messages which are needed in computation. Third, the ant starts exploring in PC 2. After it finishes exploring, it will be transferred into PC 3. The ant repeats the above process until it finishes exploring in PC *P*. By this way, the ant explores the whole layered construction graph and obtains a feasible solution. Fourth, the ant in PC *P* sets up pheromone updating messages. The messages are broadcasted to each PC and the pheromones in each PC are updated according to them. Therefore, the PACO-CGD algorithm updates all pheromones. We describe the PACO-CGD algorithm in detail by using two PCs, *m* ants iterating once (Fig.3).

Step 1. PC *A* and *B* establish their layered construction graph respectively based on the construction graph decomposition algorithm. The layered construction graphs are shown in Fig.2(b) and Fig.2(c).

Step 2. The ant 1 begins its exploring between 0th and *t*th layers in PC A. The PC *B* is idleness at the same time. After the ant 1 finishes its exploring, it will be transferred into PC B by LAN with the messages which are needed in the computation. The messages include the explored route, local heuristic information and so on.

Step 3. The ant 1 begins its exploring between *t*th and *N*-1th layers in PC *B*. The new ant 2 can begin its exploring between 0th and *t*th layers in PC A.

Step 4. Repeat step 3 until *m* ants have iterated once.

Step 5.The PC *B* will set up pheromone updating messages based on the exploring result of *m* ants. The messages are broadcasted to PC *A* and *B* and the pheromones in each PC are updated according to them.

Step 6. After updating pheromones, the PC *A* and *B* will stop exploring. The exploring result can be obtained in PC *B*.

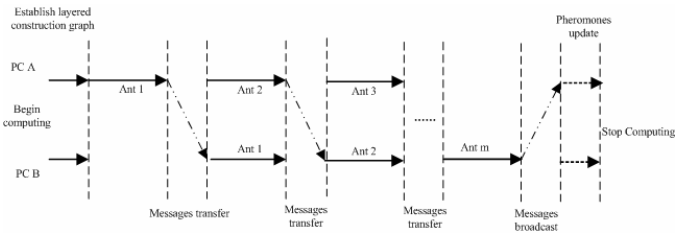

Fig. 3 Sketch map of PACO algorithm

The parallelism of the PACO-CGD algorithm can be represented by two aspects. First, two ants can start their exploring in two PCs at the same time. Second, the layered construction graphs in two PCs are established at the same time and the pheromones in two PCs are updated at the same time. The computation speed will be increased because the exploring speed is increased in each PC for the layer number of construction graph is reduced in each PC and the ant exploring can be synchronously executed. PACO-CGD algorithm can be extended from using two PCs to some PCs.

C. Real Time Ratio

The real time ratio is a very important index, which is used to evaluate the performance of parallel algorithm. Its definition is $S_p=T_s/T_p$. Where T_s is the computation time by using one PC to solve a problem, T_p is the computation time by using P PCs to solve the same problem. If the computation load balance in each PC is equal, the theoretic real time ratio of PACO-CGD algorithm will be:

$$\frac{MC(T_{ss}+T_{su})}{C[(P+M-1)T_{ssP}+MT_{suP}+(P+M-2)t_T+t_B]} \quad (12)$$

Where T_{ss} is the time of one ants exploring the construction graphs once, T_{su} is the time of one ants updating the construction graphs once, t_T is the time of messages transferring once, t_B is the time of messages broadcasting once, $T_{ssP}=T_{ss}/P$ is the time of one ants exploring the partial construction graphs once in each PC by using *P* PCs , $T_{suP}=T_{su}/P$ is the time of one ants updating the partial construction graphs once in each PC by using *P* PCs. Equation (12) means that the iterated times *C* don't effect the real time ratio.

IV. Experiment Verification

The experiment is taken in 100M LAN. The configuration of each PC attending the parallel computation is unified. The program language is c and passing of the messages and data are accomplished by MPI [7].

Consider the following multi-stage decision problem with multi-input continuous nonlinear time-variant model:

$$J(N)=\sum_{t=0}^{N-1}\{|\cos[\frac{\pi}{2}x_1(t)x_2(t)]|+\sqrt{u_1^2(t)+u_2^2(t)}\}. \tag{13}$$

$$x_1(t+1)=(t+1)\sin[\frac{\pi}{2}x_1(t)]+x_2(t)u_1(t). \tag{14}$$

$$x_2(t+1)=(t+1)x_1(t)\cos[\frac{\pi}{2}x_2(t)]+u_2(t). \tag{15}$$

$$|u_1(t)|\le 1;\quad |u_2(t)|\le 1;\quad t=0,1,\cdots,N-1. \tag{16}$$

Where N=10, $x_1(0)$=0, $x_2(0)$=0. We solve the problem with PACO-CGD algorithm (P=2) and SACO algorithm separately. The decision variables u_1 and u_2 are discretized uniformly into q=200 segments, i.e. the precision is 0.01. The PACO-CGD algorithm and SACO algorithm calculate the problem separately with 100 times, each time setting M=20, C=150. The other parameters of the two algorithms are the same. The results are shown in Table.1. Tabel.1 verified that the PACO algorithm is correct. In experiment, when the $N\cong 212$ and the other parameters is the same, the PACO-A algorithm can not be executed. The PACO-CGD algorithm can also be carry out by increasing the number of PC.

Table 1. Comparison of 100 times calculations

Algorithm type	$J(N)$ of best result in the 100 results	Mean $J(N)$ of 100 results
SACO	5.78	6.67
PACO	5.79	6.71

In this experiment, the t_T•2 ms, t_B• 2×(P-1) ms, T_{ss}•5×N ms, T_{su}•1×N ms, (12) can be transformed into (17).

$$\frac{1}{\frac{5}{6M}+\frac{1}{P}-\frac{5}{6PM}+\frac{2(P-1)}{3NM}+\frac{M-1}{3NM}}. \tag{17}$$

Based on (17), the theoretic real time ratio will be $3NP/(3N+P)$ when P and N are constant and M takes to $+\infty$. It will be $6PM/(5P+6M-5)$ when P and M are constant and N takes to $+\infty$. And it will be 0 when N and M are constant and P takes to $+\infty$. But this case will not appear because the number of PCs is finite in LAN and NM can be bigger than P very easily.

In order to research the actual real time ratio of PACO-CGD algorithm, we first set C=150, N=11, P=2, M=10,20,…, 100 and the actual real time ratio curve is Fig.4(a). Second we set C=150, M=20, P=2, N=11,21,…,91 and the actual real time ratio curve is Fig.4(b). Third we set C=150, M=20, N=100, P=3, 4,…,10 and the actual real time ration curve is Fig.4(c). From Fig.4, we know that the curvilinear path of actual real time ratios is same as the theoretic real time ratios'. Based on the theory analyse and Fig.4, the real time ratio of PACO-CGD will increase by increasing N and M when P is constant or increasing P when N and M are constant and NM is bigger than P.

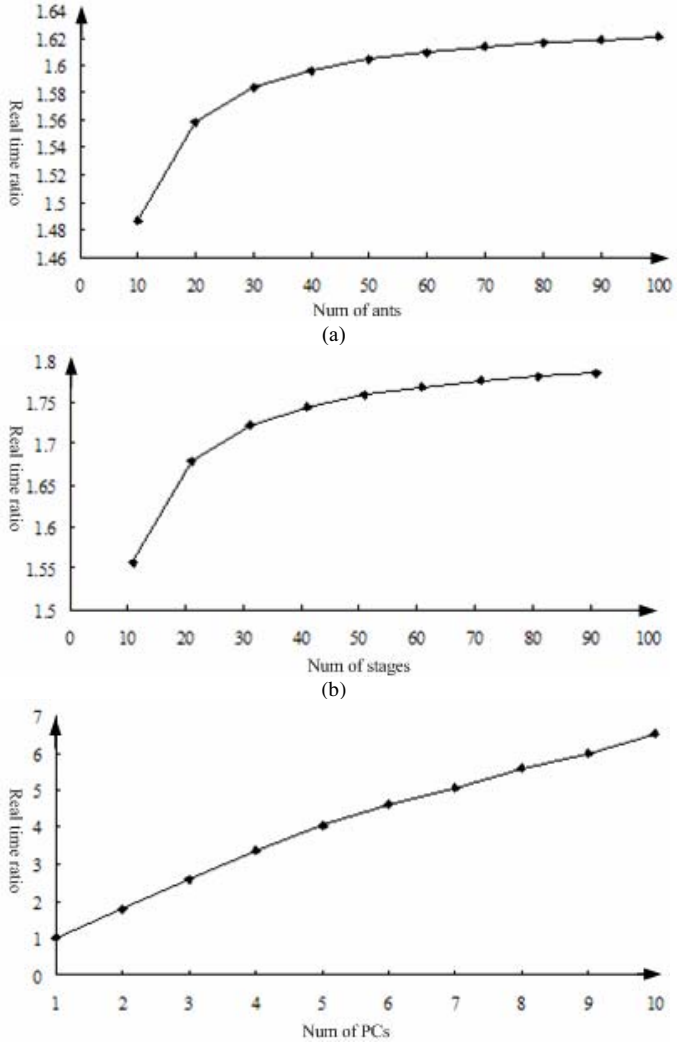

Fig. 4. Actual real time ratio curve

V. Conclusion

In this paper, a parallel ant colony optimization algorithm based on the construction graph decomposition (PACO-CGD) is presented. It can distribute the concentrated computation and big memory requirement into different PC attended parallel computation. According to this, the computation complexity and the memory sizes are reduced and the ant colony optimization can be executed in each PC even if the stage number and decision variable dimension of complex multi-stage decision problem are increasing. The construction graph decomposition algorithm is presented to achieve the construction graph

decomposition. The real time ratio also is discussed in it. Furthermore, the validity of PACO-CGD algorithm is verified by an experiment. The experiment also indicates that it can improve the computation efficiency and reality real time ratio is reasonable.

REFERENCES

[1] Y.Q. Hu, "Tutorial of Operational Research," Peking, Tsinghua University Press, 1998.

[2] Y. Wen and T.J. Wu, "Dynamic-window-search of ant colony optimization for complex multi-stage decision problems," Acta Automatica Sinica, 2004, vol.30, pp. 872-879.

[3] M. Dorigo and L.M. Gambardela, "The ant system: optimization by a colony of cooperating agents," IEEE Trans. System Man Cybernetics, 1996, vol.26, pp. 29-41.

[4] M. Dorigo, E. Bonabeau, and G. Theraulaz, "Ant algorithms and stigmergy," Future Generation Computer Systems, 2000, vol.16, pp. 851-871.

[5] Y. Wen and T.J. Wu, "Real-time rolling horizon optimization of urban traffic control based on ant algorithm," Control and Decision. 2004, vol.19, pp. 1057-1059.

[6] R. Marcus, "A parallel implementation of ant colony optimization," Parallel and distributed computing, 2002, vol.62, pp. 1421-1432.

[7] M. Snir, S. Otto, S. Huss-Lederman and et al. "MPI: The Complete Reference," Cambridge, MA. USA：MIT Press, 1998.