Journal of Harbin Institute of Technology (New Series), Vol. 11, No. 5, 2004

Parallel ant colony algorithm and its application in the capacitated lot sizing problem for an agile supply chain

LI Shu-gang, WU Zhi-ming, PANG Xiao-hong
李树刚，　吴智铭，　庞小红
(Dept. of Automation, Shanghai Jiaotong University, Shanghai 200030, China)

Abstract: In order to study the capacitated lot sizing problem for a supply chain of corporate multi-location factories to minimize the total costs of production, inventory and transportation under the system capacity restriction and product due date, while at the same time considering the menu distributed balance, the mathematical programming models are decomposed and reduced from the 3 levels into 2 levels according to the idea of just-in-time production. In order to overcome the premature convergence of ACA (ant colony algorithms), the idea of mute operation is adopted in genetic algorithms and a PACA (parallel ant colony algorithms) is proposed for supply chain optimization. Finally, an illustrative example is given, and a comparison is made with standard BAB (Branch and Bound) and PACA approach. The result shows that the latter is more effective and promising.

Key words: multi-location factories; supply chain; capacitated lot sizing; ant colony algorithm
CLC number: TP273　**Document code**: A　**Article ID**: 1005-9113(2004)05-0573-06

With economics developing globally, the competition between companies becomes more and more volent. Most large companies build more and more factories in different places (even different countries) to balance materials flow and service quality. How to optimize the material flow among factories, retailers and clients is the key problem in the supply chain management for these large enterprises. Usually the research focuses on three fields: product plan, retailer plan and inventory plan[1-3].

The optimizing of material flow in the supply chain includes the production-level, the inventory-level and the transport-level. Each level has great influence on the next level. To a large-scale problem, it is a NP hard problem. Many of the methods, such as the BAB, cannot obtain a satisfactory result because of spending huge computational efforts; therefore, the ant colony algorithm is proposed. At the same time, the model is decomposed according to the idea of just-in-time production in the supply chain. Factories will not provide the overdue products and can start production with 0 inventory at every period. As a result, the objective function is predigested. By coding reasonably, the three level optimal problem is reduced to two levels and efficiency is improved.

The ant colony algorithm was proposed by M. Dorigo[4-6]. It has been applied widely in many areas because of its robustness and fitness to distributed computing and ease of association with other methods. For example, it has been applied in TSP[7] and job shop scheduling problems[5] successfully. But ant colony algorithms also have some shortcomings. One of the obvious problems is that it can stop in one area (stagnation behavior) and cannot reach the better solution. In order to overcome this shortcoming, we propose the parallel ant colony algorithms, that is, split the ant population into two sub-populations.

The ants in one population select the ways according to the information accumulated in the form of a pheromone trail deposited on paths; the ants in another population search paths randomly. Update the pheromone according to the result of the two populations and keep the good individuals. This can improve the searching efficiency and avoid searching in a limited area.

Finally a simulation to the supply chain problem is made with the parallel ant colony algorithm. The result shows that the parallel ant colony algorithm is a high efficiency approach.

1 Describe of Lot Size Problem in the Supply Chain with the Distribute Factory

Now we study the supply chain with $J(j = 1,2,\cdots,J)$ factories, $L(l = 1,2,\cdots,L)$ retailers and M $(m = 1,2,\cdots,M)$ transportation ways about $I(i = 1,2,$

Received 2002-09-28.
Sponsored by the National Natural Science Foundation of China (Grant No. 70071017 and 59889505) and Open Foundation of State Key Laboratory for Manufacturing Systems Engineering (Xi'An).

$3\cdots,I$) types of product. Every factory lies in a different place and every factory has the ability to produce any type of the product at the same time. The producing cycle includes $T(t = 1,2\cdots T)$ short periods. At the beginning of the cycle, the inventory of each factory is 0, supposing the demand of each retailer must be met at every period. Then the main task of the supply chain management is to optimize the product cost, the inventory cost and the transportation cost under the factory capacity constraint. The research problem does not include the tariff interest. Some definitions are given Tab. 1.

Tab. 1 Definition for the problem

Notation	Definition	Notation	Definition
Pc_{ij}	Unit product cost of product i in factory j	$D_i(t)$	The delivery deadline for product i at period t
Pt_{ij}	Unit product time of product i in factory j	$Dm_{il}(t)$	The order of retailer l for product i at period t
Pi_{ij}	Unit inventory cost of product i in factory j	$PS_{ij}(t)$	The inventory of product i in factory j at period t
Tv_{jml}	Transportation time from factory j to retailer l by transportation way m	$TP^m_{ijl}(t)$	The number of product i transported from factory j to retailer l at period t with the transportation way m
C_{im}	Unit transportation cost of product i by transportation way m at unit time	$P_{ij}(t)$	The output of product i in factory j at period t
$PA_f(t)$	The product capability of factory f at period t	β_i	The Penalty coefficient of product i for overdue delivery
$A_i(t)$	The beginning delivery time of product i at time t	$O_{ij}(t)$	The number of overdue delivery product i in factory j at period t
$Ta_{ij}(t)$	The beginning transportation time of product i in factory j	$Dp_{ij}(t)$	Allowed max producing time of product i in factory j at period t
Menu(t)	The total number of product at period t	$TP_{ijl}(t)$	The number of product i transported from factory j to retailer l at period t

In practice, when the menu is released down to the factory, the management center should not only cut down the cost and improve the efficiency, but also make the resource configuration reasonable to balance the production. Thus, some constraints should be added to the model. Supposing that the production absolute value of the difference of any two factories would not be larger than a quarter of the total output, a multi-factory, multi-retailer and multi-transportable way supply chain model (P_0) can be described as follows:

$$P_0 = \min\Big(\sum_t\sum_j\sum_i Pc_{ij}P_{ij}(t) + \sum_t\sum_j\sum_i Pi_{ij}PS_{ij}(t) + \sum_t\sum_j\sum_i\sum_m\sum_l TP^m_{ijl}(t)\cdot C_{im}Tv_{jlm} + \sum_t\sum_j\sum_i Q_{ij}(t)\beta_i\Big) \quad (1)$$

$$\text{S.t.} \sum_i P_{ij}(t) \leqslant PA_j(t) \quad \forall i,j,t \quad (2)$$

$$Dm_{il}(t) \geqslant \sum_j TP_{ijl}(t) \quad \forall i,l,t \quad (3)$$

$$TP_{ijl}(t) = \sum_m TP^m_{ijl}(t) \quad (4)$$

$$PS_{ij}(t) = PS_{ij}(t-1) + P_{ij}(t) - \sum_l TP_{ijl}(t) \quad \forall i,j,l,t \quad (5)$$

$$\Big|\sum_i P_{ij}(t) - \sum_i P_{if}(t)\Big| \leqslant \frac{1}{4}Menu(t) \quad \forall i,t,j \in J, f \in J \quad (6)$$

P_0 includes the production cost, inventory cost, transportation cost and penalty cost because of overdue. Constraint (2) represents the number of product, which is put into every factory, below its ability. Constraint (3) represents the number of product transported to every retailer should be no larger than the demand of every retailer. Constraint (4) represents the transportation balance. Constraint (5) represents the inventory balance. Constraint (6) represents the menu split balance.

2 Decomposing of the Multi-Factory, Multi-Retailer and Multi-Transport Way Supply Chain Model

2.1 Decomposing Based on Just-In-Time Idea

In the agile supply chain every part product adopts a just-in-time producing idea and the retailers will not accept the overdue product. If the factory can finish producing before the allowed max producing time, then they can provide the product on time. Because of no overdue product the penalty cost in the objective function can be omitted, and each factory can begin its production with 0 inventory at each period.

Supposing the allowed max producing time is defined as the difference between the deadline for delivery product and the time consumed by the lowest transportation way that can be defined as follow::

$$Dp_{ij}(t) = D_i(t) - \max\{\max\{Tv_{j11}, Tv_{j12}, \cdots Tv_{j1L}\}, \cdots, \max\{Tv_{jM1}, Tv_{jM2} \cdots Tv_{jML}\}\}$$

It can be proved that if the number of product which is assigned to the factory is less than the number which the factory produces in the max producing time, the factories can avoid providing overdue production and can begin producing in every period with 0 inventory.

The ways of transportation have great influence on the transportation cost and the inventory cost of every factory when the retailer is selected. The beginning transportation time of the each product is decided by the transportation way (let the start producing time of each period be 0), that is:

$$Ta_{ij}(t) = A_i(t) - Tv_{jml},$$

where transportation way m should make the sum of inventory cost and transportation cost be the least. If the transportation ability is infinite, the inventory is equal to the output within $Ta_{ij}(t)$.

Then the model (P_0) can be decomposed as follow:

$$P_0 = \min\Big(\sum_t\sum_j\sum_i Pc_{ij}P_{ij}(t) + \sum_t\sum_j\sum_i Pi_{ij}PS_{ij}(t) + \sum_t\sum_j\sum_i\sum_l\sum_m TP^m_{ijl}(t)C_{im}Tv_{jlm}\Big), \quad (7)$$

$$\text{S. t.} \quad P_{ij}(t) \leqslant \frac{Dp_{ij}(t)}{Pt_{ij}} \quad \forall i,j,t, \quad (8)$$

$$D_{il}(t) = \sum_j TP_{ijl}(t) \quad \forall i,l,t, \quad (9)$$

$$PS_{ij}(t) = P_{ij}(Ta_{ij}(t)) \quad \forall i,j,l,t, \quad (10)$$

$$\Big|\sum_i P_{ij}(t) - \sum_i P_{if}(t)\Big| \leqslant \frac{1}{4}Menu(t) \quad \forall i,t,j \in J, f \in J, \quad (11)$$

$$Pt_{ij} \geqslant 0, Pi_{ij} \geqslant 0, Pc_{ij} \geqslant 0, Tv_{ij} \geqslant 0, c_{ij} \geqslant 0. \quad (12)$$

P_0 includes the product cost, inventory cost and transportation cost. Constraint (8) represents the output of every product in every factory is below its production ability. Constraint (9) represents the demand of every retailer should be met with. Constraint (10) represents the inventory equal the output within $Ta_{ij}(t)$. Constraint (11) represents the menu splitting balance. Constraint (12) represents every parameter should have its physical meaning.

2.2 Optimal Tactics

There is a three level optimizing problem in the supply chain system: product level, inventory level and transportation level optimizing. If the ant colony algorithm were used in each level the computation consumption would be very huge; so we have to reduce the optimizing level.

We assign an ID to each retailer in every menu, split them reasonably and assign them to the factories. When the factories finish producing, products are transported to the retailers according to the ID. Since the number of transportation ways is limited, by traditional dynamic programming method the transportation way can be selected easily.

So the optimal tactics of the supply chain is to split the menu and assign them to the factories by the parallel ant colony algorithm. Then select the transportation path by the dynamic programming to transport the product to the retailers.

3 Problem Solved by the Parallel Ant Colony Algorithm

Just as described above, the managing problem of the multi-factory, multi-retailer and multi-transport way supply chain is a NP hard problem. With the increase of problem scale, the computation will increase exponentially. By the traditional method, it is impossible to get a satisfactory result. In this paper, the parallel ant colony algorithm is proposed.

3.1 Parallel Ant Colony Algorithm

The main idea of the ant colony algorithm is that it simulates the ants finding the food in nature. According to the information accumulated in the form of a pheromone trail deposited on the paths, ants can select the tracks. Ant colony algorithm has been applied in many ways and obtains a good result. But the ant colony algorithm also has some shortcomings. For example, it may converge to a local optimal solution. In order to overcome this shortcoming, a parallel ant colony algorithm is proposed, that is, split the ant population in two subpopulations. The ants in one population select ways according to the pheromones. The ants in another population search paths randomly. According to the results of the two populations, update the pheromones and keep the good individuals. Thus it can improve the searching efficiency and keep from searching in an area.

Some definitions of the notation are given as Tab.2.

3.2 Code

According to the rules that try best to keep the menu as a whole, one may assign as many as possible of the same type of production to one factory. The code is designed as follows:

Take the total demand number of every product at every period as a main menu. So there are I main menus. Then split the main menus into the submenu according the number of demands by every retailer. Totally there are L submenus. The problem is to assign the submenu to the factories under the constraint. For example: $P_i = \{p_{i1l}, p_{i2l}, \cdots, p_{iJl}\}_{l=1}^{L}$ where p_{ikl} represents the number of product i produced in factory k and demanded by the retailer l, so the total code at every period is: $\{P_i\}_{i=1}^{I}$.

Tab. 2 Definition for the optimal algorithm

Notation	Definition	Notation	Definition
p_{ijl}	The number of product i produced in factory j and transported to retailer l	$\tau_{ij}(t)$	The pheromone that make ant select $e(i,j)$
$e(i,j)$	Menu allocating match, meaning that assign the product i to factory j	$Q_{ij}^k(t)$	The number of product i assigned to the factory j when ant k searching t times
$fit^k(t)$	The value of the objective function of ant k when searching t times	Tota l_i	The total number of the product i

3.3 Update of the Pheromone

$\tau_{ij}(t+1) = \rho\tau_{ij}(t) + \Delta\tau_{ij}$, where ρ is a parameter, $\Delta\tau_{ij} = \sum_{k=1}^{K} \Delta\tau_{ij}^k$ where $\Delta\tau_{ij}^k$ is the pheromone on the menu allocating match $e(i,j)$ left by the ant k when searching between t times and $t+1$ times. If the ant k, selects $e(i,j)$ then $\Delta\tau_{ij}^k = \frac{\gamma}{fit^k(t)} \cdot \frac{Q_{ij}^k(t)}{Total_i}$, else is ω. γ is a constant number. K is the total number of the ants.

Try best to assign the sample type of product to a factory under the constraints. The probability that product i is assigned to factory j is: $\theta_{ij}(t) = \frac{[\tau_{ij}(t)]^{\partial} \cdot [\eta_{ij}(t)]^{\beta}}{\sum_{v \in allow(J)} ([\tau_{iv}(t)]^{\alpha} \cdot [\eta_{iv}(t)]^{\beta})} (v \in allow(J))$, where α, β is a parameter to control the pheromone and visible degree, η_{ij} is a tuning parameter, $allow(J)$ is the set of the factories which the ant can select while splitting the menu and its elements change with the time. If no cities are selected with the probability, select them randomly.

3.4 Describe the Parallel Ant Colony Algorithm

1) Select two ant populations.

2) Initial the pheromone.

3) In the first ant population assign the product i to factory j according the $\theta_{ij}(t)$ and in the second population assigned the product i to the factory j randomly.

4) Compute the objective function.

5) Update the pheromone according to the computed result of the two populations and keep the best individual.

6) Repeat step 3 ~ 5 until obtaining max times.

4 Simulation

We make a simulation on the supply chain with the 4 factories, 3 retailers, 2 transport ways on 6 types of the product in 3 periods. Supposing every type of product can be produced in any factory (other data see Tab. 3, supposing the factory produces 24 hours a day). Regarding all the factories in the supply chain as a whole product system, we know there are reasonable solutions.

Tab. 3 Parameter for the supply chain problem

	PT_1	PT_2	PT_3	PT_4	PT_5	PT_6	PC_1	PC_2	PC_3	PC_4	PC_5	PC_6	PI_1	PI_2	PI_3	PI_4	PI_5	PI_6	R_{11}	R_{12}	R_{21}	R_{22}	R_{31}	R_{32}
P_1	1	2	1	2	2	2	20	24	27	27	26	26	7	2	6	6	7	7	4	2	5	4	4	6
P_2	2	2	2	2	1	2	20	25	28	28	22	30	5	3	2	3	6	1	5	4	6	5	5	3
P_3	2	2	2	1	1	2	30	30	28	25	32	32	3	1	5	1	1	7	4	6	5	6	6	3
P_4	1	1	2	2	1	1	30	32	28	20	26	24	2	2	5	5	6	6	4	1	6	5	2	6

(P_i: factory i, T_j: type j, PT_i: unit product time of type i (hour), PC_i: unit product cost in every factory of type i (dollar), PI_i: :unit inventory cost of type i (dollar), R_{ij}: transport time to the retailer i by the transport way j (day))

	TR_1	TR_2	$M_1(1)$	$M_2(1)$	$M_3(1)$	$M_1(2)$	$M_2(2)$	$M_3(2)$	$M_1(3)$	$M_2(3)$	$M_3(3)$	$B(1)$	$D(1)$	$B(2)$	$D(2)$	$B(3)$	$D(3)$
T_1	2	4	300	100	200	200	200	200	100	200	300	6	15	6	16	7	15
T_2	4	2	200	200	200	100	100	200	300	100	200	6	16	7	15	6	16
T_3	2	4	100	200	300	300	200	100	200	100	200	7	16	5	16	6	15
T_4	4	2	300	100	200	200	100	200	100	100	300	7	16	6	16	5	16
T_5	4	2	200	100	300	300	100	100	300	100	200	6	15	7	17	6	16
T_6	2	4	200	300	300	200	100	300	300	200	200	7	23	8	20	6	21

(Continue) (TR_i: Unit transportation cost in unit time by transportation way i (dollar), $M_i(t)$: The order of the retailer i in period t, $B(t)$: the beginning time of the delivery in period t, $D(t)$: the deadline of the delivery in period t)

4.1 Searching Efficiency

Using the optimal tactics proposed in this paper, compute the optimizing problems of supply chain with VISUAL C++6.0 (Celeron300 Windows98 operation system). Select two ant populations: there are 2 ants in one population, which select ways according to the pheromone and 1 ant in another population, which select ways randomly ($\partial = 2, \beta = 1, \omega = -0.001, \gamma =$

1 000, $\eta_{ij}(t) = Pc_{ij}, \rho = 0.9$). Running fifteen minutes (searching 2 000 times) we get the objective function in every period as follows: $J_0(1) = 125\ 569$, $J_0(2) = 104\ 161$, $J_0(3) = 113\ 709$. Total: $J_0 = 343\ 439$ (Tab. 4 provides the solution). On the other hand, with LP PROC (which is an algorithm of branch and band) in SAS/Or model in SAS V8.0 software, running 4 hours, we obtain the objective function in every period as follows: $J_0(1) = 127\ 448$, $J_0(2) = 110\ 482$, $J_0(3) = 117\ 910$. Total: $J_0 = 355\ 840$.

Tab. 4 Solution in every period

	period											
	1				2				3			
	factory											
	1	2	3	4	1	2	3	4	1	2	3	4
a(b,c,d)	1(216,1,1)	1(76,2,1)	1(84,1,1)	1(168,3,1)	1(200,1,1)	1(120,2,1)		1(40,2. 1)	1(84,2,1)	1(108,2,1)	1(100,1,1)	1(168,3,1)
		1(32,3,1)	1(24,2,1)		1(40,2,1)	2(92,2,2)	2(108,3,2),	1(200,3,1)	1(132,3,1)	2(40,2,2)	1(8,2,1)	2(240,1,2)
	2(120,2,2)	2(40,2,2),	2(120,3,2)	2(200,1,2,1)	2(100,1,2)	2(16,3,2)		2(76,3,1)	2(60,1,2)	2(80,3,2)	2(120,3,2)	3(108,1,2)
		2(80,3,2)		2(40,2,2)	2(8,2,2)	3(20,2,1)	3(120,2,1)	3(120,1,2)	2(60,2,2)			
	3(60,2,1)	3(100,1,1),	3(120,2,1)	3(120,3,1)	3(180,1,1)	3(100,3,1)			3(48,2,1)	3(68,3,1)	3(92,1,1)	
	3(180,3,1)	3(20,2,1)			3(60,2,1)	4(20,3,2)	4(60,2,2),	4(120,1,2)	3(132,3,1)		3(16,2,1)	
	4(120,1,2)	4(60,1,2),	4(40,2,2)	4(120,1,2)	4(80,1,2)	5(36,1,2)	4(180,3,2)	5(264,1,2)	4(80,2,2)	4(20,3,2)	4(240,3,2)	4(100,1,2)
		4(60,2,2)	4(200,3,2)	5(200,1,2)	4(40,2,2)	5(100,2,2),			4(40,3,2)			4(20,2,2)
	5(84,2,2),	5(216,3,2),	5(60,3,2)	5(16,2,2)		5(100,3,2)			5(60,1,2)	5(40,2,2)		5(240,1,2)
	5(24,3,2)					6(32,1,1)			5(60,2,2)	5(200,3,2)		
	6(204,2,1),	6(200,1,1),		6(92,2,1)	6(168,1,1)	6(100,2,1),	6(4,3,1)	6(260,3,1)	6(180,3,1)	6(16,3,1)	6(176,2,1)	6(300,1,2)
		6(4,2,1)		6(300,3,1)		6(36,3,1)					6(4,3,1)	6(24,2,1)

(a(b,c,d) represents the solution, where *a* represents product type, *b* product number; *c*: retailer Id and *d* transport way)

From the above data we can see that the result of parallel ant colony algorithm is smaller than SAS's by 3.5% while time of use is far less than SAS. So the parallel ant colony has a higher efficiency. To the different scale and different constrained lot size problem in the supply chain, the parallel ant colony algorithm also has a higher efficiency.

In order to make a comparison between the searching efficiency of PACA and traditional ACA (3 ants), using the same data (the value of the parameter is as above) we make another simulation. Each algorithm runs 6 times, and each time searching 1 000 times, the PACA cost 360 s and ACA cost 364 s. Because the ACA will stagnate at 200 times while PACA has no such cases, the results of PACA are more optimal than the ACA's. Tab. 5 reports the simulation results.

Tab. 5 Result of the PACA and ACA

NO.	PACA	ACA
1	356 742	387 204
2	356 742	387 204
3	360 610	372 854
4	358 608	387 204
5	360 610	387 204
6	356 742	372 854
Average	358 342	382 420

4.2 Analysis of the Result

From the Tab. 6 (The first row represents the max producing time and the second row represent the practice productime in the factory 1 to 4) we can see that every factory can finish producing before max producing time, so there is no overdue production.

We make Tab. 7 (The first row is the total number of the product demanded from period 1 to period 3, the second row is the number of the product produced in the factory 1 to factory 4) according the factory production in different periods. From the table, we can see that the difference between two factories is less than 1/4 of the total product in every period and the demand of the retailer can be met, so the menu splitting is reasonable.

Tab. 6 The max producing time and the practice product time in every period

Period	max producing time	216	216	216	216	240	240	240	240	240	240	240	240	240	240	240	240	216	216	216	216	408	408	408	408
one	practice product time	216	54	54	168	60	60	60	240	240	60	60	60	60	60	240	60	54	216	60	216	102	102	0	392
Period	max producing time	240	240	240	240	216	216	216	216	240	240	240	240	240	240	240	240	264	264	264	264	336	336	336	336
two	practice product time	240	60	0	240	54	54	54	76	240	60	60	60	60	10	240	60	0	236	0	264	84	84	2	260
Period	max producing time	216	216	216	216	240	240	240	240	216	216	216	216	240	240	240	240	240	240	240	240	360	360	360	360
three	practice product time	216	54	54	168	60	60	60	240	216	34	54	54	60	10	240	60	60	240	0	240	90	8	90	324

Tab. 7 The factory production in different period

3 800				3 200				3 500			
1 008	888	648	1 256	876	772	472	1 080	972	572	756	1 200

5 Conclusion

Supply chain problem is a focused field of research in the production and material flow fields. The lot size problem in a single factory is a NP-hard problem. To the supply chain, it is a harder problem. This paper provides a model for agile supply chain with the multi-factory, multi-retailer and multi-transport way. Moreover, we provide optimal tactics with the parallel ant colony algorithm for the model. On the other hand, we also endow a PACA to improve the searching efficiency of the ACA. Finally simulations are made, and the results show that the optimizing tactic is reasonable and the PACA is of high efficiency. The approach provides a way for big corporations with distributed factories and more retailers to solve the supply chain lot size problem.

References:

[1] THOMAS D J, GRIFFIN P M. Coordinated supply chain management [J]. European Journal of Operational research, 1996, 94(1): 1-15.

[2] ERENGUC S S, SIMPSON N C, VAKHARIA A J. Integrated production/distribution planning in supply chains: An invited review [J]. European Journal of Operational research, 1999, 115(2): 219-236.

[3] MACCARTHY B L, LIU J. Addressing the gap in scheduling research: A review of optimization and heuristic methods in production scheduling [J]. International Journal of Production Research, 1993, 31(1): 59-79.

[4] COLORNI A. An investigation of some properities of an ant algorithm[A]. Proc of Parallel Problems Solving from Nature(PPSN)[C]. Elsvier:[s. n.],1992. 509-520.

[5] COLORNI A. Ant system for job shop scheduling [J]. Belgian Journal of Operations Research Statitics and Computer Science,1994,34(1):39-53.

[6] COLORNI A. Distributed optimazition by ant colonies[A]. Proc of 1st European Conf Artifical Life [C]. Elsevier:[s. n.],1991. 134-142.

[7] DORIGO M, GAM BARDELLA L M. Ant colony system: A coopera-tive learning approaches to the traveling salesman problem [J]. IEEE Transaction on Evolutionary Computation, 1997, 1 (1):53-66.