# Binary Ant Colony Evolutionary Algorithm

Weiqing Xiong [1] Liuyi Wang [1] Chenyang Yan [2]

[1]School of Information Science and Engineering Ningbo University, Ningbo 315211 China
Weiqing ,xwqdds@tom.com, Liuyi,jameswang@hotmail.com
[2]School Information and Electrical Engineering Lanzhou Jiaotong University Lanzhou 730070 China
chengyang,Yan_chengyan@hotmail.com

**Abstract**

Form the view of the biological viewpoint; each ant of social insects is regarded as neuron. They compose a neural network by stochastic and loose connections with each other; Similar to the artificial neural network simulating the ant colony, the Ant colony algorithm of binary network is presented. As the binary coding is adapted, lower aptitude behave of single ant is requested, less storage is needed, the algorithm efficiency is enhanced largely. The test function and the multi 0/1 Knapsack problem show that the algorithm has better convergence speed and stability, the result is very excellent

**Keywords：** Swarm intelligence, Simulated evolution computation, Binary network, Ant colony algorithm, Genetic algorithm

## 1 Introduction

Ant colony optimization algorithm is a new simulative evolutionary algorithm named ant colony system and was proposed in the 1990s by Italian scholar M. Dorigo. It has been applied to TSP, allocation problem, JSP, and got excellent results. Hence, more attention has arisen to the ant colony system, and the model has been applied to many practical problems [1,2].

Holland's theory of complexity self-adaptation system considers: human brain, immune system, life system, cell, ant colony and the parties and organizations in human society are all parallel and interactive networks composed by agent [3,4].

Form the biological evolution viewpoint; each ant of social insects is regarded as a single neuron. They compose a neural network by random and loose connections with each other; then as artificial neural networks simulate human brain, this loose neural network, in other words, can simulate the colony behavior of the insect to construct a neural network model with a loose brain-colony intelligence stochastic structure. Social insects always behave intelligent such as bee's nest building, ant's foraging. It seems that all this behavior is a cooperative process designed preliminarily and monitored by some general directors, and the whole colony looks like an intelligent human being. Although many biologists pay much attention to this phenomenon, the inside mystery is still unknown. Recently, researchers in artificial intelligence field become interested in the colony behavior of insect as in human intelligence, and want to simulate by computer. That is what is called swarm intelligence[5].

Withal, it can be thought that ant colony algorithm should be a multi-agent

cooperative evolutionary algorithm based on lattice space. Generally speaking, binary representation is the most efficient and economical data representation, and two-value logic is the easiest way to simulate and implement, so we consider that this lattice space is binary network.

Experimental result indicates: Ant Colony Algorithm has very strong ability to find better solutions to combination optimization, and gets good performance on parallel computing, easy combination with other algorithm and nice robustness. However, it still has some shortcomings, for example: at initial stages it lack pheromone, search slowly, and easily to trap into the local optimal solution if the quantity is large, which is to say premature convergence. Regarded as efficiency global parallel optimization search tool, the Genetic Algorithm has some disadvantage: early to convergence to local optimization of target function, thereby is difficult to find global optimization along with has bad local ability, sometimes convergences around the optimization.

In order to overcome these drawbacks, based on the binary ant colony network, combines the Genetic Algorithm with the Ant Colony Algorithm, the new algorithm is called Binary Ant Algorithm-evolution algorithm.

## 2 Binary Ant Colony Evolutionary Design

### 2.1 Binary Ant colony network Construction

Reference [5] indicates that in the view of organic evolution, each ant of social insects is regarded as neuron. They compose a neural network by stochastic, loose connections with each other; Similar to the artificial neural network simulating the ant colony, a loose neural network, which has a stochastic construction of loose brain-colony intelligence, is used to simulate the colony behavior of insect.

Hence, some complicated collective tasks can be cooperatively finished by many simple individuals, and through some simple communication. Meanwhile, this method has strong robustness. In this way, a mathematical model of colony intelligence is setup. Suppose there is a colony, containing n individual (the ability of each individual is the same), and every individual has an ability of *f*, which is a functional computation that every individual can complete. Secondly, every individual can communicate with another one nearby, in other words, every individual can transfer information to another one. Each individual acts stochastically and concurrently. After receiving a terminal instruction, it stops working, then, the whole task completes.

Aim at the binary network showed as Figure 1, we divide the work of ants, and different kinds of ants search the same routine, and pheromone is left on each edge. As an intelligent body, each ant just chooses one edge of the two. The intelligent behavior of ant is very simple, and the incidence matrix traversed by each kind of ant needs only $2\times n$ 's space, which to some extent solves the descriptive difficulty generated from long coding and the reduction of solution quality. Hence, in the next experiment, the efficiency of algorithm increases greatly.

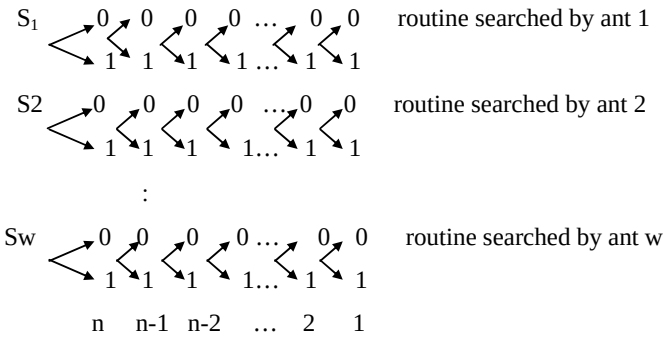


**Fig. 1.** Binary ant colony network

Like the definition of deterministic state automaton, we can design finite probabilistic state automaton and define it as a 5-tuple: $NFSA = \{\sum, Q, Z, \delta, F\}$

Where Q is a state set containing starting state $q_0$, $F \subseteq Q$ is the subset of accepting state, $\sum$ is a set called alphabet, here is {0,1}, λ represents output relation (so called because for every state, one input corresponds to several outputs). δ is transition function, but it contacts state/input, subsequent state and happening probability.

$\lambda : Q \times \sum \to Q \times [0,1]$ , $(q_t, \delta_t)\alpha(q_{t+1}, p(q_{t+1})) \mid\to (q_t, \sigma_t) \in Q \times \sum, p(q_{t+1}) \in [0,1]$

Hence, one condition is needed to satisfy any state $q_t$, for every possible input, the sum of the probability of the transitional output is 1. If it is not 1, match all the probabilities once again: $\mid\to q_t : \sum_{\sigma_t} p(q_{t+1}) = 1$.

We can regard probabilistic finite automaton as Markov chain. Markov chain is a stochastic sequence of events and has anomalous property. That is to say, it is a stochastic process: the probability of next event completely depends on a sequence. 1 degree in Markov chain represents that what will happen next just depends on the former events, which is called state-decidable Markov chain. Likewise, high-order Markov chain is called history-decidable Markov chain, which depends on several former events to decide the probability of next events. Here, we just research 1 degree Markov chain, for it can be treated as finite state automaton, where every transition has some relationship with the generate probability.

Define a directed graph G=(C, L), where the node set C is:

$\{c_0(v_s), c_1(v_N^0), c_2(v_N^1), c_3(v_{N-1}^0), c_4(v_{N-1}^1), ..., c_{2N-3}(v_2^0), c_{2N-2}(v_2^1),$

$c_{2N-1}(v_1^0), c_{2N}(v_1^1)\}$

$v_s$ is the starting node, $v_j^0$ and $v_j^1$ are separately used to represent the value 0 and 1 of position $b_j$ in the binary string. To every node ( $j = 2,3,...N,$ ), there are

two directed arcs pointing to $v_{j-1}^0$ and $v_{j-1}^1$.

### 2.2 Basic Ant Colony Algorithm

Initially, the quantity of information in each routine is equal, suppose $\tau_{ij}(0) = C$ (C is constant), $\Delta\tau_{ij}=0$ (i, j =0, 1, …, n-1). During the movement, ant k shifts its direction according to the quantity of information of the routine. The probability of movement is:

$$p_{ij}^k = \begin{cases} \dfrac{\tau^{\alpha}{}_{ij}(t)\cdot\eta^{\beta}{}_{ij}(t)}{\sum_{s\in allowed_k}\tau^{\alpha}{}_{is}(t)\cdot\eta^{\beta}{}_{is}(t)}, & j\in allowed_k \\ 0 \qquad otherwise \end{cases} \tag{1}$$

Where m is the number of ants in the colony, $p_{ij}^k$ is the probability of ant k's shift from position i to position j at k time, $\alpha$ is the relative importance of track, $\beta$ is the relative importance of visibility ($\beta>=0$), $\tau_{ij}$ is the remaining information on link i,j at t time, $\eta_{ij}$ is the visibility of arc (i,j), $allowed_k = \{0,1\}$ represents the states that ant k is allowed to choose. Thanks to binary encoding, memory function is not needed; the ant can choose one edge according to the remaining quantity of pheromone on the edge in front. Besides, as time elapses, the former remaining information will disappear, $\rho$ represents the durability of the track ($0\le\rho<1$), parameter $1-\rho$ represents the degree of information evaporation. After time period's n, ant completes one circle, and information on every routine will adjust as follows:

$$\tau_{ij}(t+n) = \rho\cdot\tau_{ij}(t) + (1-\rho)\cdot\Delta\tau_{ij}, \tag{2}$$

$$\Delta\tau_{ij} = \sum_{k=1}^{m}\Delta\tau_{ij}^k, \tag{3}$$

Where $\Delta\tau_{ij}^k$ represents the remaining information left by ant k in routine ij during this circle, $\Delta\tau_{ij}$ represents the increment of information in routine ij.

$$\Delta\tau_{ij}^k = \begin{cases} \dfrac{Q}{L_k} & as\ \ if\ \ ant\ \ k\ \ ever\ \ passed\ \ through\ \ ij, \\ 0 \qquad otherwise \end{cases} \tag{4}$$

Where Q is a constant representing the number of tracks the ant left, $L_k$ represents the length of the routine that ant k passes during this circle.

The global correction strategy proposed in reference [7] is as follows:

$$\tau_{ij}(t+n) = \rho\cdot\tau_{ij}(t) + (1-\rho)\cdot\Delta\tau_{ij}, \tag{5}$$

$\Delta\tau_{ij} = \Delta\tau_{ij}^k$, The k'th ant is the one that find the shortest routine in this circle. $\Delta\tau_{ij}^k$

comes at formula (4). Global correction strategy is to let the ant that best travels around the network release information pheromone. The combination between global correction strategy and modified state transition strategy guarantees that the latter ants can search in the domain that traveled by the excellent elder, which greatly enhances the solving speech.

### 2.3 Binary Ant colony Evolution Algorithm

The basic algorithm is consist of four operator: select operator、cross operator、mutate operator and pheromone renovate operator.

Stopping Criteria: required solution appears and after evolving several generations. Otherwise keeping elite GA converge in accordance with probability 1[6]

Basic steps of the algorithm is as follows:

Step 1: generate the initial solution random

Step 2: genetic algorithm evolves n times, and gains the solution, which is used to initiate the phenomenon of the net.

Step 3: judge whether the stopping condition of the qualification is met, if qualified, ends; otherwise, carries on.

Step 4: ants search

Step 5: use genetic algorithm to compute the solution generated by ant colony algorithm, and reserve the optimal

Step 6: upgrade phenomenon of the net by optimal solution

## 3 Design of Algorithm used to optimize function solution

### 3.1 Mathematic models of multi-dimension function optimization

Suppose that the general mathematic models of multi-dimension function optimization is as follows:

$$\begin{aligned} &min\ f(x_1, x_2, \cdots x_n) \\ &st.\ g_i(x_1, x_2, \cdots x_n) \geq 0,\ i=1, 2, \cdots, m; \\ &\ \ h_i(x_1, x_2, \cdots x_n) = 0,\ i=1, 2, \cdots, l; \end{aligned} \tag{6}$$

where f(x) is object function, $g_i(x_1, x_2, \cdots x_n)$ and $h_i(x_1, x_2, \cdots x_n)$ is restriction function. Vector $\vec{x}=(x_1, x_2, \cdots x_n)$ is decision variable, $g_i(x_1, x_2, \cdots x_n) \geq 0$ is in equation restriction. $h_i(x_1, x_2, \cdots x_n) = 0$ equation restriction.

Punish function is involved, and in some proper condition, the optimization problems containing restriction functions can be transformed to the field restriction optimization problems as follows:

$$\begin{aligned} &min\ f(x_1, x_2, \cdots x_n) \\ &st.\ a_i < x_i < b_i,\ (i=1,2,\cdots,n) \end{aligned} \tag{7}$$

Later, we will discuss field restriction optimization problems.

### 3.2 Algorithm Design

*（1）Encoding Design*

Binary encoding is adopted. The dimension of the optimization function decides the number of the routine that ants traverse in every circle. The first routine that an ant has traversed is the first variable of the corresponding function, and so is the second routine, by analogy.

Every variable $x_i$ in candidate solution $\{x_1, x_2, ...\}$ is expressed by a N bits long binary string $\{b_N b_N ... b_2 b_1\}$, where $b_j \in \{0,1\}$, $j = 1,2...N$ , $b_{N-1}$ is the topmost place, $b_0$ is the lowermost place. The leftmost real number of variable $x_i$ is $x_{i\min}$, the rightmost one is $x_{i\max}$, the corresponding decimal integer of binary string is k, and we can execute parameter decoding according to the formula below:

$$x_i = \frac{(x_{i\max} - x_{i\min})k}{2^N - 1} + x_{i\min} \circ \tag{8}$$

The whole directed network is showed as Figure 2, where m is the dimension number of variable, n is the length of the binary encoding of every corresponding variable, and algorithm traverses from top position to low position

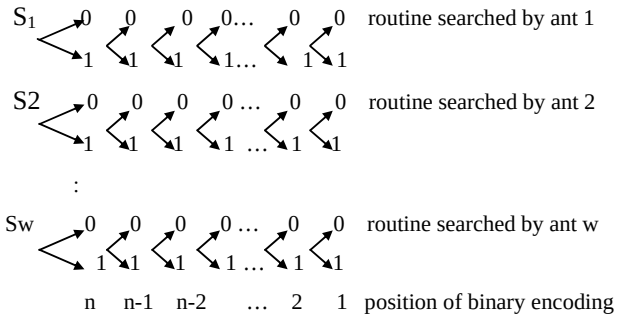


**Fig. 2.** Binary network for function optimization

*(2) Initialization*

Generate m initial solutions randomly, and compute these solution degrees of adaptability. Then candidate group is got by the m initial solutions, and compute their information quantity according to the solution degree of adaptability of the value in candidate group.

### 3.2 Experiment of Function Optimization

Test function is achieved from Reference [3]. In the test, we use α=1,β=2,ρ=0.35, Gen (number of revolutionary generation)=150,m (number of ants)=80.Lenchrom (length of encoding)=40, Q=1.0,Pc=0.95, Pm=0.05

**Testing function 1:** $100*(x_1^2+x_2)^2+(1-x_1)^2 \quad (-2.048 \le x_i \le 2.048)$, where the minimum 0 is in (1,1);

$f$（1，1）=0 can be got after executing certain generations;

**Testing function 2:** $\sum_{i=1}^{5} \text{int}(x_i) \quad (-5.12 \le x_i \le 5.12)$, where the global minimum –30 is in (-5.12, 5.12);

After executing, several solution equal to –30 can be got, including $f$ (-5.11061, -5.08007)=-30.

**Testing function 3:**

$$\frac{1}{1/K+\sum_{j=1}^{25} 1/f_j(x_1,x_2)}, \qquad K=500, f_j(x_1,x_2)=c_j+\sum_{i=1}^{2}(x_i-a_{ij})^6, c_j=j,$$

$$[a_{ij}]=\begin{bmatrix} -32 & -16 & 0 & 16 & 32 & -32 & -16\cdots 0 & 16 & 32 \\ -32 & -32 & -32 & -32 & -32 & -16 & -16\cdots 32 & 32 & 32 \end{bmatrix}$$

$(-65.536 \le x_i \le 65.536)$

Where the minimum 0.998004 is in (-32, -32);

After executing, several solutions equal to 0.998004 can be got, including $f$ (-31.9817,31.9991)=0.998004.

**Testing function 4:** $\sum_{i=1}^{30} i*x_i^4+Gaussian(0.1) \quad (-1.28 \le x_i \le 1.28)$, where minimum 0 is in (0,0, … 0).

After executing, $f$ (0,0…0)=0 can be got.

**Testing function 5:** $0.5+\frac{\sin^2\sqrt{x_1^2+x_2^2}-0.5}{[1.0+0.001(x_1^2+x_2^2)]^2}$, where -100≤$x_i$≤100 （i=1,2） and global minimum 0 is in (0,0);

After executing, $f$ (0,0)=0 can be got.

From tests, we can see that the results got from cooperative evolutionary algorithm with stochastic binary hierarchical network is really exciting, and it manages to overcome the shortcoming of ant colony algorithm not suitable to the successive space.

## 4 Design of Algorithm used to Multi-Dimension Knapsack Problem

### 4.1 Mathematic models of Multi-Dimension Knapsack Problem

Multi-Dimension Knapsack Problem is a kind of knapsack problems with a group of restriction. It can be described as follows:

There are n goods with weights of W[1]，W[2]，…，W[n] and values of

V[1], V[2], …, V[n] separate, and m knapsacks with capacities of C[1], C[2], …, C[m]. It is required to put as many goods as possible into the knapsack, and maximize the total value of the goods in all the knapsacks with insurance that the whole weight of every knapsack is not overloaded. Suppose that X [i][j] is a binary variable. If goods I is put in the knapsack j, X [i][j]=1; otherwise, X [i][j]=0. Mathematic description of multi-dimension knapsack problem is as follows:

$$\max = \sum_{j=1}^{m}\sum_{i=1}^{n} V[i]X[i][j] \tag{9}$$

$$\text{st.} \quad \sum_{j=1}^{m} X[i][j] \le 1 \qquad j = 1,2,\cdots,m;$$

$$\sum_{i=1}^{n} W[i]X[i][j] \le C[j] \qquad i = 1,2,\cdots,n$$

$$X[i][j] \in \{0,1\}, \quad i = 1,2,\cdots n,\ j = 1,2,\cdots m;$$

Knapsack problem is a special integer layout problem, and is also a NP-hard problem. To multi-dimension knapsack problem, with the premise that every goods can be put into one knapsack most, its time complexity is $O(2^{mn})$ [8].

### 4.2 Design of Algorithm

*(1) Design of Encoding*

Suppose X[i][j] is a binary variable. If goods I is put into knapsack j, X[i][j]=1; otherwise, X[i][j]=0. Corresponding binary net is showed as Figure 3, where m is the number of knapsack, and n is the number of goods.

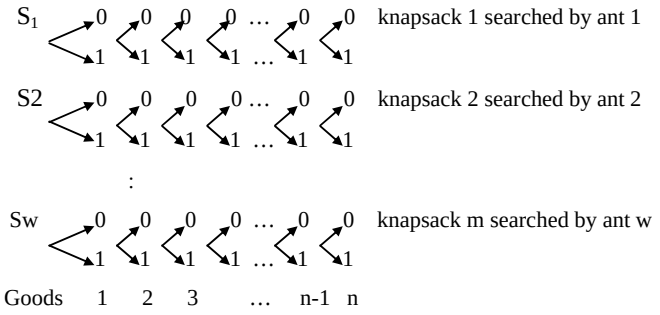


**Fig. 3.** . Binary network for multi-dimension knapsack problem

*(2) Degree function of adaptability*

Individual degree function of adaptability in knapsack problem is determined by the total value of goods in all knapsack decided by individual solution, viz.

$$f = \sum_{j=1}^{m} \sum_{i=1}^{n} V[i]X[i][j]. \quad (10)$$

*(3) Illegal individual modification operator*

In the process of generating new individuals, it is inevitable to produce some illegal individuals. To maintain the validity of individual in the colony, it is required to modify the illegal individuals, and get the reasonable, excellent ones. The best way to modify the illegal individual is proceed real time judging modification, viz. when one genetic position is 1, judge whether it breaks the restriction, if it does, reset the value to 0.

### 4.3 Examples Testing

In this example [9], suppose that there is a knapsack problem with 100 goods and 3 knapsacks. To make the analysis of the result more straightforward, we suppose that the value density of all the goods are the same. Here, integer 0/1 knapsack problem can be simplified as: maximize the total weight of the goods in all the knapsacks with the premise of not overloading the capacities of all the knapsacks, viz. the maximum of the required value is simplified to the maximum of the quality. The qualities of the given 100 goods are showed as Table 1, and the capacities of 3 knapsacks are c[1]=3398,c[2]=1327,c[3]=1873.

**Table 1.** . Goods Quality Table

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 253 | 245 | 243 | 239 | 239 | 239 | 238 | 238 | 237 | 232 |
| 231 | 231 | 230 | 229 | 228 | 227 | 224 | 217 | 213 | 207 |
| 203 | 201 | 195 | 194 | 191 | 187 | 187 | 177 | 175 | 171 |
| 169 | 168 | 166 | 164 | 161 | 160 | 158 | 150 | 149 | 147 |
| 141 | 140 | 139 | 136 | 135 | 132 | 128 | 126 | 122 | 120 |
| 119 | 115 | 116 | 114 | 111 | 110 | 105 | 105 | 104 | 103 |
| 93 | 92 | 90 | 79 | 78 | 77 | 76 | 76 | 75 | 73 |
| 62 | 62 | 61 | 60 | 60 | 59 | 57 | 56 | 53 | 53 |
| 51 | 50 | 44 | 44 | 42 | 42 | 38 | 36 | 34 | 28 |
| 27 | 24 | 22 | 18 | 12 | 10 | 7 | 4 | 4 | 1 |

In this example, we use $\alpha$=1,$\beta$=2,$\rho$=0.35, Q=1.0,Pc=0.95, Pm=0.1.

Two-dimension knapsack problem test: Choose two knapsacks of c[1]=3398，c[2]=1327 respectively, and C[1]+C[2]=4725. Here, m（number of ants）=40, division number of ants is 2, each dimension is traversed by 20 ants, and number of generation is 200. Results are 4724 after every execution.

Three-dimension knapsack problem test: we use three knapsack in the emulation example where c[1]=3398，c[2]=1327，c[3]=1873, and C[1]+C[2]+C[3]=6598. here, m（number of ants）=60, division number of ants is 3, each dimension is traversed by 20 ants, and number of generation is 500. Results are 6596 after every execution.

After the testing above, we can see that the result of the ant colony algorithm designed to deal with multi-dimension knapsack problem is quite satisfactory, the

speed of computing is really quick and the solution is stable. So it can be concluded that hierarchical cooperative evolutionary algorithm with stochastic binary network still has good performance on combination optimization.

## 5 Conclusion

In this thesis, we simulate the colony behavior of insect by loose neural network, viz. design a Binary ant colony evolutionary algorithm by the illumination of the construction of the neural network model of loose brain-colony intelligence stochastic structure. The algorithm overcomes the shortcoming that ant colony algorithm is not suitable to solve successive parameter optimization, and it is also applicable to combination optimization (e.g. knapsack problem, schedule problem). Moreover, it better solves the shortages of ant colony algorithm and the bad locate searching ability of genetic algorithm. In the process of function optimization and knapsack problem, the results show that Binary ant colony evolutionary algorithm is practical and efficient, and the solutions are satisfactory.

## Reference

[1] Dorigo M, Caro GD. Ant colony optimization: A new meta-heuristic. In: Proc. of the 1999 Congress on Evolutionary Computation, Washington, DC, USA: IEEE Press, 1999: 1470~1477.
[2] Dorigo M, Gambardella LM. Ant colony system: A cooperative learning approach to the traveling salesman problem [J]. IEEE Trans on Evolutionary Computation, 1997,1(l) : 53~66.
[3] Zbigniew Michalewicz. Genetic Algorithms +Data Structures=Evolution Programs [M] Science Press, Beijing 2000
[4] Li Xia, Dai Ruwei. System Science and Complexity. Automatization Transaction 1998,24(4): 476-483.
[5] Zhang Ling, Cheng Junsheng. Mathematic Model of Loose Brain-Colony intelligence Pattern Recognition and Artificial Intelligence 2003（1）1－5
[6] Stutzle T, Hoos H. MAX-MIN Ant System. Future Gener. Comput Syst, 2001,16(8): 889-914
[7] Zhang Wenxiu, Liang Yi. Mathematic Basis of Genetic Algorithm. Xi'an Jiaotong University Press Xi'an 2000
[8] Li Qinghua，Li Kenli，Jiang Shengyi，Zhang Wei. Optimal Parallel Algorithm of Knapsack Problem. Software Transaction, 2003, Vol 14(5): 891-896
[9] http://nrich.maths.org/public

**Weiqing Xiong** now is a associate professor in the School of Information Science and Engineering, Ningbo University, his current research interests include evolutionary computation, artificial intelligence, etc.

**Liuyi Wang** now is a graduate student in the School of Information Science and Engineering, Ningbo University, his current research interests include evolutionary computation, artificial intelligence, etc.