Improving Ant Colony Optimization with Chaos
MOZHGAN MOMBEINI
MOHAMMAD ALI NEKOUI

Institute for Research in Fundamental Sciences
Electrical engineering - control
K.N.Toosi University of Technology
IRAN
Mzh.mombeini@ipm.ira, manekoui@eeetd.kntu.ac.ir

Abstract: - Ant colony is one of the most important methods for solving discrete optimization problems. There are some attempts to improve this method of optimization. Because of the specific properties of the chaos, it has been used as a specific method of optimization called Chaos Optimization Algorithm (COA). To improve the ACO method some tried to combine chaos sequences with ACO algorithm in some ways. In this paper these different methods are categorized and then a new method for improvement of the ant system (AS) with chaos is proposed.

Key-Words: - Ant Colony Optimization, Ant System, Chaos, Chaos Optimization Algorithm (COA), TSP;

1 Introduction
Ant colony optimization (ACO) algorithm is a meta-heuristic method for solving combinatorial optimization problems [1-4]. Chaos sequences are known with the sensitivity to initial condition and ergodic property. These special properties made them a choice as deterministic random generators and more developments resulted in introducing Chaos Search Algorithm [5]. These specific application and properties, in advance, made them a candidate for improvement of evolutionary algorithms [6]. Here, different attempts of using chaos for improving ACO are categorized. Then, a new procedure for using chaos to improve AS is introduced and finally this new method implemented on TSP. This paper is organized as follows. In part 2 properties of chaos sequences and Chaos Search algorithm illustrated. In part 3 a summary of ACO algorithm with pseudo-code illustrated. In part 4 different Chaotic ACO algorithms categorized and a new method introduced and implemented on TSP. And part 5 is conclusion.

2 Chaos Properties
Chaos sequences are generated by simple deterministic systems. They have some properties that make them proper candidate as random generator systems.

Property 1: Sensitivity to initial conditions.

Property 2: Ergodicity property. Chaos movement meets through all states of the attractor space without any repeat. These properties of chaotic systems together made the idea of Chaos Search for optimization problems. Simplicity of the implementation is also an important factor. In literature most used function is Logistic map which is one of the simplest chaos maps. Logistic equation is:

$Z_{n+1} = \mu Z_n (1 - Z_n)$

For nominal value, $\mu = 3.8$, logistic map (1) shows chaotic behavior. The interval of this chaotic sequence is between 0 and 1.

Chaos Optimization Algorithm (COA)
According to above special properties and simplicity of the implementation, Chaos Optimization Algorithm (COA) proposed by Li and Wang [5]. The steps of COA implementation are as following:

Step (1) Make chaotic sequences by chaotic map (1).

Step (2) Scale chaotic sequences to variable interval $(\min Z, \max Z)$ according to the particular resolved problem:

$Z'_n = \min Z + (\max Z - \min Z)Z_n$ (2)

Step (3) If value of the function remain constant choose the best function value of each chaotic sequence as output.
3 ACO Algorithms for the TSP
There are different versions of ant colony algorithms that are applicable for TSP problem. [1-4]
Pseudo-code for these algorithms is shown in fig.1. Local search is optional and main phases are tour construction or "ConstructAntsSolutions" and Update of Pheromone Trails or "UpdatePheromones". Difference between different ACO versions it related to these two main phases.

Procedure
ACOMetaheuristic
Set parameters, initialize pheromone trails while (termination condition not met) do
ConstructAntsSolutions
ApplyLocalSearch % optional
UpdatePheromones
end-while
end-procedure

Fig.1 The ACO meta-heuristic in pseudo-code.

4 Categories of different chaotic ant colonies
There are many studies in literature to combine chaos and ACO to improve the efficiency of the ACO algorithm. In this paper, these different methods are categorized in 5 groups which 5th one is a novel method that introduced here for ACO. This categorization include the studies that use chaos for improvement of the ACO, and the cases that are application of the ACO for chaotic systems, for example control and synchronization or parameter estimation of the chaotic systems are not matters for this study.

4.1 Chaos as random generator for initializing the swarm
The question “does chaos work better than noise?” is still an open question. But investigation on using deterministic chaotic signals instead of random signals in experiment showed excellence of the chaos in biologically computing methods [7]. In ACO algorithm in the first iteration inserting randomly initial value of pheromone is common way to have more exploration. Replacing this random generator with a logistic map (1) resulted in more efficient algorithm for example in [8].

4.2 Chaos energy for trapping from local minimum
In this category, chaos energy is inserted in algorithm motions to avoid trapping in local minimum [8-17]. This energy insertion is done in pheromone update phase by adding a chaos map multiplied to a factor to the new pheromone value. For instance in for pheromone update rules

\[ \tau_{ij} \leftarrow (1 - \rho)\tau_{ij} + \sum_{k=1}^{m} \Delta \tau_{ij}^{k} \]  

Formula (3) changes to (4)

\[ \tau_{ij} \leftarrow (1 - \rho)\tau_{ij} + \sum_{k=1}^{m} \Delta \tau_{ij}^{k} + \varepsilon Z_{ij} \]

Where \( \Delta \tau_{ij}^{k} \) is the amount of pheromone level added to \( \tau_{ij} \) by the artificial ant \( k \) and \( Z_{ij} \) is the chaos perturbation using (1). The parameter \( \rho (0 \leq \rho \leq 1) \) is the pheromone evaporating rate, and \( \varepsilon \) is constant coefficient.

4.3 COA and ACO
In this category local search as shown in fig.1, which is an optional in phase in ACO, is implemented with Chaos Optimization Algorithm (COA). COA and ACO combined together to avoid from trapping into local optima and also improve the accuracy of the solution. This strategy proposed under the title "Scaleable Chaotic Ant Colony Optimization"[18]. Fig.2 shows pseudo-code for this algorithm.

Procedure
SCALABLE-ACO
Set parameters, initialize pheromone trails while (termination condition not met) do
ConstructAntsSolutions. (Select the best ant in current iteration and if it is better than the global best ant, replace it).
Work out COA around the best ant in the range of the parameters. (If COA solution is better than the global best ant, replace it).
UpdatePheromones
end-while
end-procedure

Fig.2 The Scaleable Chaotic Ant Colony in pseudo-code.
In literature this method implemented for generating SHEPWM switching patterns and applied to three-level NPC inverter [19] and reactive power optimization in power system [20] and for preliminary ship design [21]. This algorithm also implemented for "optimization of tool motion trajectories for pocket milling"; but in this work COA is applied just in first iteration [22].

### 4.4 Chaos ants

All the methods in previous sections were combination of chaotic maps with classical methods of ACO. In this section another type of chaotic ant colony introduced with this difference that ant behavior is modeled with chaotic maps. It has been shown by Cloe that the movement activity of single special type of ants in nature has characteristics of a low-dimensional chaos [23]. Sole et al. modeled chaotic behavior of individual by a chaotic map \( x_{n+1} = x_n e^{i(1 - x_n)} \) [24]. Based this model of single ant and colony behavior in finding optimal solution, an optimization method called CAS introduced with following equations [25]

\[
y_i(n) = (y_i(n-1) + V_i \frac{7}{\psi_d}) \times e^{(1 - a y_i(n))}(3 - \psi_d z_{id}(n-1) + \frac{7.5}{\psi_d} y_i) + (p_{id}(n-1) - z_{id}(n-1)) \times e^{-2 a y_i(n)+b} - \frac{7.5}{\psi_d} V_i
\]

where

1. \( n \) means the current time step, and \( n - 1 \) is the previous step;
2. \( z_{id}(n) \) is the current state of the \( d \)th dimension of the individual ant \( i \), where \( d = 1, 2, \ldots, l \);
3. \( p_{id}(n-1) \) is the best position found by \( i \)th ant and its neighbors within \( n - 1 \) steps;
4. \( y_i(n) \) is the current state of the organization variable;
5. \( a \) is a sufficiently large positive constants and can be selected as \( a = 200 \);
6. \( b \) is a constant and \( 0 \leq b \leq \frac{2}{3} \);
7. \( \psi_d \) determines the selection of the search range of \( d \)th element of variable in search space,
8. \( r_i > 0 \) is a positive constant less than one and is termed by us as the organization factor of ant \( i \), \( y_i(0) = 0.999 \), and \( 0 \leq V_i \leq 1 \) determines the search region of ant \( i \) and offers the advantage that ants could search diverse regions of the problem space.

### 4.5 New Chaotic Ant Colony to overcome stiffness of the parameters

In this part tried to use chaotic maps to overcome stiffness of the parameters. This idea implemented on Ant System proposed by Dorigo, 1992[4]. Formulation of two main phases in AS are as bellow.

#### Tour Construction

In AS, \( m \) ants build a tour of the TSP concurrently. At first run, ants randomly choose cities. Then, at each iteration ant \( k \) decides which \( j \) city to visit next using random proportional rule. The probability which ant \( k \), at city \( i \), chooses to go to next city \( j \) is

\[
p_{ij}^k = \frac{[\tau_{ij}]^\alpha [\eta_{ij}]^\beta}{\sum_{i \in N_i^k} [\tau_{ij}]^\alpha [\eta_{ij}]^\beta} \quad \text{if } j \in N_i^k
\]

Where, \( \eta_{ij} = \frac{1}{d_{ij}} \), \( d_{ij} \) is a the length of the path, \( \tau_{ij} \) is the pheromone between \( i \) and \( j \) at current iteration, \( \alpha \) is the stimulating factor of pheromone, \( \beta \) is the stimulating factor of expectations, and \( N_i^k \) is the feasible neighborhood of ant \( k \) at city \( i \), it contains cities that ant \( k \) has not visited yet.

According to transition probability rule (5) the more the pheromone on the path, increases the probability that ants choose that path.

#### Update of Pheromone Trails

Each iteration ends when all the ants passed their tour. After each iteration, in the next phase, by global pheromone update, pheromone on the paths updated and next iteration starts. Cycle of these two phase continue until to reach maximum number of iteration.

With the global pheromone update rule Pheromone of the Path updates;

\[
\tau_{ij} \leftarrow (1 - \rho)\tau_{ij} + \sum_{k=1}^{m} \Delta \tau_{ij} \quad \forall (i, j) \in L
\]

\[
\Delta \tau_{ij} = \begin{cases} 
Q/L_k & \text{if } \text{arc}(i,j) \text{ used by ant } k \\
0 & \text{otherwise}
\end{cases}
\]

where \( Q \) is a constant and \( L_k \) is the tour length of the \( k \)th ant. \( \Delta \tau_{ij} \) is pheromone deposit by ant \( k \) on...
Fig. 3 Qatar map: Evolution of optimal tour length by iteration in left, and the final optimal tour found by the ants in the right. Up figures belong to original AS, bellow belongs to Chaotic AS.

Fig. 4 Uruguay map: Evolution of optimal tour length by iteration in left, and the final optimal tour found by the ants in the right. Up figures belong to original AS, bellow belongs to Chaotic AS.
the arcs it has visited, \( 0 < \rho < 1 \) is the pheromone evaporation rate.

Arcs that are part of short tours (tours with small values of the objective function) receive more pheromone and the probability of being chosen by more ants in next iterations of the algorithm increases.

**Chaos Ant Colony Algorithm**

One of the challenges in running heuristic methods for solving an optimization problem is adjustment of values for the constants. Here tried to replace a chaotic map instead of the \( \rho \) evaporation parameter and let it to explore more in the solution space.

Using Logistic equation (1) with scaling formula (2) for interval \((0,\rho)\) and pheromone update rule (6), results in the following modifications for a new chaos algorithm

\[
\rho_{n+1} = \mu \rho_n (1 - \rho_n) \quad n = 1, 2, 3, ...
\]

\[
\rho'_n = \frac{\rho \rho_n}{n} 
\]

\[
\tau_{ij} \leftarrow (1 - \rho'_n) \tau_{ij} + \sum_{k=1}^{L} \Delta \tau_{ij} \quad \forall (i,j) \in L
\]

Convergence of the AS proofed by Dorigo and Stützle [26]; this proof is extendable to this modified version.

Setting of the parameters and TSP problem types are shown in table 1 and table 2.

Table 1 Setting of the algorithm parameters

| \( m \) | \( 100 \) |
| \( \beta \) | \( 3 \) |
| \( \alpha \) | \( 1 \) |
| \( \rho \) | \( 0.5 \) |
| \( \mu \) | \( 3.8 \) |

<table>
<thead>
<tr>
<th>Problem</th>
<th>Number of Cities</th>
<th>Distance Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Qatar Map</td>
<td>194</td>
<td>Euclidean</td>
</tr>
<tr>
<td>Uruguay Map</td>
<td>734</td>
<td>Euclidean</td>
</tr>
</tbody>
</table>

Figures (3,4,) shows simulation results for running AS and chaotic AS on TSP problems, Qatar and Uruguay maps, which in both cases chaotic AS results in better performance in convergence.

According to Fig.3 shows a sample of running both algorithms on Qatar map. Chaotic AS reaches in less number of iterations to better solutions. For a nominal optimal value for path as 11893; Chaotic AS reaches to optimal value in 600 iterations but AS does not reach to this value even after 1000 iterations.

According to Fig. 4, for running the algorithms on Uruguay map chaotic AS reaches in less iterations to better solution. For a nominal optimal value for path as 102590; Chaotic AS reaches to optimal value in 900 iterations but AS does not reach to this value even after 1500 iterations.

Then inserting chaos map in AS, or new chaotic AS, represents improvement in the algorithm performance for both TSP problems, the Uruguay map and Qatar map.

**5Conclusion**

Attempts in various researches for inserting chaos maps in ACO represent improvement in the algorithm. New chaotic AS proposed here to overcome stiffness of the parameters, showed improvement in convergence to solution on TSP.

**References:**


