Available online at www.sciencedirect.com

ScienceDirect

Expert Systems with Applications 36 (2009) 481–488

Expert Systems with Applications

www.elsevier.com/locate/eswa

New robust and efficient ant colony algorithms: Using new interpretation of local updating process

Hossein Miar Naimi *, Nima Taherinejad

Electrical Engineering Faculty-University of Mazandaran, Dr. Shariati Street, Babol, Iran

Abstract

Two new efficient and robust ant colony algorithms are proposed. These algorithms contain two new and reasonable local updating rules that make them more efficient and robust. While going forward from start point to end point of a tour, the ants' freedom to make local changes on links is gradually restricted. This idea is implemented in two different forms, leaving two new algorithms, KCC-Ants and ELU-Ants. To evaluate the new algorithms, we run them along with the old one on the standard TSP library, where in almost all of the cases the proposed algorithms had better solutions and even for some problem samples found the optimal solution.

Keywords: Ant colony algorithm; Local updating; Optimizing; Heuristic algorithms

1. Introduction

Many optimization problems were found to be non-polynomial. Using intelligent search algorithms is one strategy to tackle these type problems. Genetic algorithm (GA) (Nagata et al., 1997), Tabu search (Glover & Laguna, 1997), ant colony algorithm (Blum et al., 2005; ANTS, 2000) and … are the main intelligent search algorithms that can be mentioned.

Ant colony algorithm is a mathematical model of ants behavior in finding the shortest path between nest and food. The search capability of ants, using no visual sign, is the most attractive aspect of their behavior. Passing through the paths, each ant leaves pheromone on its path. The amounts of pheromone on different paths make ants able to improve the paths between nest and food they pass through (Beckers, Deneubourg, & Goss, 1992; Goss, Aron, Deneubourg, & Pasteels, 1989; Holldobler & Wilson, 1990). They can also find the new shortest path when the former one is destroyed, i.e. because of an obstacle (Beckers, Aron, Deneubourg, & Pasteels, 1993). Ants' capability in improving and shortening the paths inspired scientist to develop algorithms for solving optimization problems. The best and most successful ones were developed by Deneubourg and his assistants (Goss et al., 1989).

The common problem to evaluate search algorithms especially ant colony algorithm is traveling sale man problem (TSP).

Consider cities C_1, C_2, C_3, …, C_n with $\mathrm{d}(C_i, C_j)$ as the distance between C_i and C_j. we have a complete graph of cities with connecting lines between each pair of cities. If $\mathrm{d}(C_i, C_j) = \mathrm{d}(C_j, C_i)$ then the problem is symmetric and otherwise is asymmetric. Here the problem is finding the shortest Tour, where a Tour is a route which passes through each city once and only once! On the other words a tour is a collection of links that connect cities in a closed and connective ring graph. The direct distance between two cites is named link, and when an ant completes its tour it is said that it end a cycle and to try more may start another cycle. Fig. 1 shows and example of such a problem.

Shown in Fig. 1, there are many different tours. For example we can consider two tours $T_1 = ABCDEA$ and $T_2 = ADEBCA$. The length of each tour can be calculated as below: $L_{T_1} = \mathrm{d}(A,B) + \mathrm{d}(B,C) + \mathrm{d}(C,D) + \mathrm{d}(D,E) +$

* Corresponding author. Tel./fax: +98 111 32 39 214.
E-mail address: h_miare@nit.ac.ir (H.M. Naimi).

0957-4174/$ - see front matter © 2007 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2007.09.048

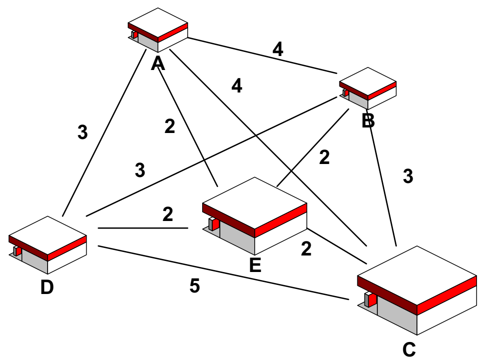

Fig. 1. Symmetric TSP problem with five cities.

$d(E,A) = 4+3+5+2+2 = 16$, $L_{T_2} = d(A,D) + d(D,E) + d(E,B) + d(B,C) + d(C,A) = 3+2+2+3+4 = 14$.

Above calculations show that T_2 is shorter and so better than T_1, but the question is whether another shorter tour exists or T_1 is the shortest one? How can we systematically find the shortest tour? Answering these questions equals to solve the TSP. In different papers it's proved that TSP is a NP-Hard Problem (Lawler, Lenstra, Rinnooy Kan, & Shmoys, 1985).

1.1. The ant colony algorithm

As a top view, the algorithm can be described as below. A number of agents (ants) move through the different paths and leave pheromone on their passed path. Thus they affect other ants while selecting links (or the order of cities in the path) to establish their path. In fact in each step of establishing a tour, an ant selects the links with more pheromones with more probability. Fig. 2 shows the overall view of the algorithm structure.

Different versions of ant algorithms differ in each section of structure shown in Fig. 2.

Different parts of the ant colony algorithm affect on its efficiency. As will be discussed in next section, different strategy for each part of the algorithm has been examined to improve the algorithm's efficiency. The efficiency mainly is the quality of the found solutions. Whatever a solution is closer to the optimal solution it has more quality. One of the most important parts of the algorithm is the local updating rule which is emphasized in this paper. We introduce new interpretation of the concept and role of local updating rule and consequently design two new rules based on the interpretation. The new local updating rules are used in the ant algorithm, resulting two new ant algorithms. Our experiments show they are more robust and efficient than the last ones (Fig. 3).

The rest of the paper is as follows: In Section 2 the ant algorithm history, implementations and modifications is studied. Section 3 is dedicated to introduce the proposed algorithms. The results of some experiments on standard TSP problem samples and some random problems along

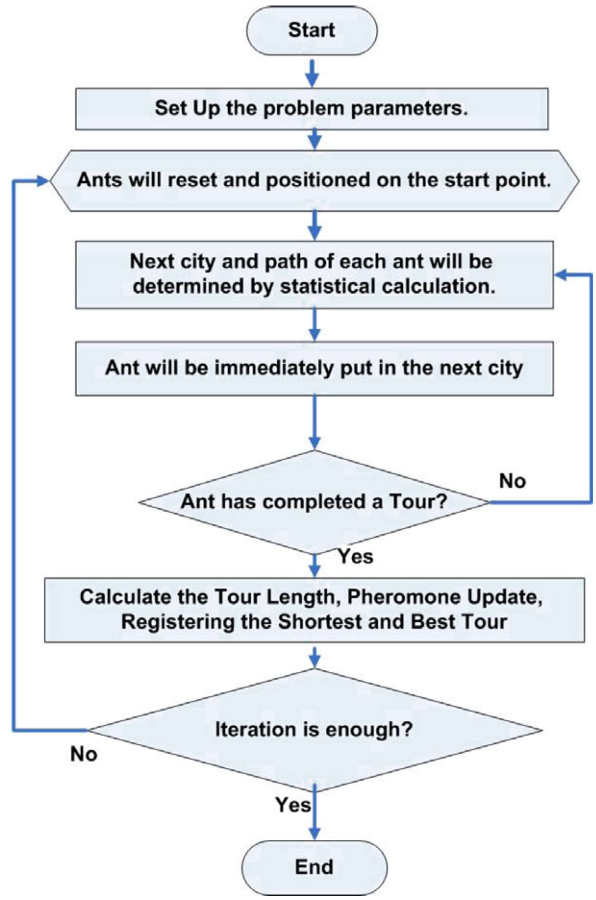

Fig. 2. Ant colony algorithm: a overall view.

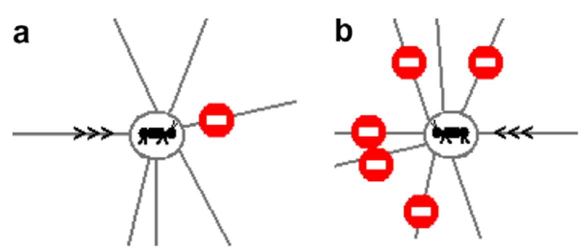

Fig. 3. An agent in primary parts of its tour (a) freer to selects the more desirable link, but in final parts (b) has less possibility of selecting more desirable path.

with comparison with other algorithms are presented in Section 4. Finally Section 5 contains conclusion remarks and some offers for more researches.

2. The literature review

The complete discussion about historical aspects of the ant algorithm is too long to be covered in this paper. So we have selected some of the papers which include important modifications.

For each paper we focus on most important parts of the ant algorithm as below:

- Ant's primary set up.
- The rule of selecting next cities by ants.
- Pheromone updating.

The paper (Colorni, Dorigo, & Maniezzo, 1992) is an introductory paper on ant colony algorithm. In this paper the Ant-Cycle algorithm is introduced and some primitive ideas, like pheromone updating and probability of selecting a link, are described. Then the parameter tuning was investigated and finally the results were reported. Here we pay to different parts of the proposed algorithm.

Ant's primary set up: in this paper, all three possible strategies have been examined.

1. All ants start the tour from the same city.
2. Equal numbers of ants start from each city.
3. Ants are distributed on cities, randomly.

Their investigation proved that the second strategy is better than the first one. The third strategy has a little difference with the second one, but has slightly better results.

Selecting next city (Link): the rule used is a statistical equation where the link with more pheromone and less length has more chance to be selected (if the ant has not passed the link before). Now consider an ant, say number K, be in the city i and intend to select other unselected city to go (to add a link to its uncompleted tour). Here a city, say j, (If ant has not passed it before) would be selected with the following probability:

$$P_{ij}^{k}(t) = \begin{cases} \frac{[\tau_{ij}(t)]^{\alpha}\cdot[\eta_{ij}(t)]^{\beta}}{\sum_{k\notin \text{tabulist}}[\tau_{ik}(t)]^{\alpha}\cdot[\eta_{ik}(t)]^{\beta}} & \text{if } j \notin \text{tabu list} \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

Where τ_{ij} is the pheromone amount of link (edge) between C_i and C_j, η_{ij} is the inverse of d(C_i, C_j), α and β are adjustable parameters to weight the significance of pheromone and length in the selection of next city.

Pheromone updating: the paper only uses a global updating. When an ant completes a tour it updates and changes the pheromone amount of its passed path (all links on the passed path). The rule of this update is as shown in Eq. (2).

$$\begin{aligned} \tau_{ij}(t+n) &= \rho\cdot\tau_{ij}(t) + \Delta\tau_{ij} \\ \Delta\tau_{ij} &= \sum_{k=1}^{m}\Delta\tau_{ij}^{k} \end{aligned} \quad (2)$$

Where τ_{ij} is the pheromone of link between cities i and j. n is the time required to complete a tour and ρ is a coefficient such that $(1-\rho)$ represents the evaporation of pheromone in each tour completion time. $\Delta\tau_{ij}^{k}$is computed as Eq. (3).

$$\Delta\tau_{ij}^{k} = \begin{cases} \frac{Q}{L_k} & \text{if}(i,j) \in k\text{-th ant's tour tabu list} \\ 0 & \text{Otherwise} \end{cases} \quad (3)$$

Q is a constant and L^k is the tour length of k-th ant.

Others parts: the paper has a good discussion on parameter tuning. Presenting some curves of the results and values of parameters, it offers optimum values as $\alpha = 1$, $\beta = 2$ or 5, $\rho = 0.5$. It also contains a table about problem solution, search space dimensions, average cycles and time required to find optimum.

The paper (Colorni, Dorigo, & Maniezzo, 1996), first presented ants' behavior and the artificial algorithm (Ant-Cycle), and then proposed two new algorithms, 1-Ant-Density and 2-Ant-Quantity. The main parts of the algorithms are as follows:

Ant's primary set up: the ants are distributed but it has not been mentioned whether uniformly or randomly (it may be cause of their vicinity in results).

Selecting next city: the selecting rule is the same as Eq. (1) and no changes were made in this part.

Pheromone updating: Ant-Cycle uses the previous pheromone updating rule but in the two new algorithms the updating model has been modified so that instead of updating all link of a tour after completing it, each ant updates the pheromone of the recent passed link locally. Thus Local Updating was presented and used in this paper for the first time.

The mentioned local updating rule of Ant-density and Ant-Quantity are as Eqs. (4) and (5), respectively.

$$\Delta\tau_{ij}^{k} = \begin{cases} Q & \text{if the } k\text{-th ant goes from city } i \text{ to } j \\ & \quad \text{between } t \text{ and } t+1 \\ 0 & \text{Otherwise} \end{cases} \quad (4)$$

$$\Delta\tau_{ij}^{k} = \begin{cases} \frac{Q}{d_{ij}} & \text{if the } k\text{-th ant goes from city } i \text{ to } j \\ & \quad \text{between } t \text{ and } t+1 \\ 0 & \text{Otherwise} \end{cases} \quad (5)$$

Where, Q is a constant in both algorithms. d_{ij} is the distance between city i and city j.

Other parts: this paper has also a discussion on parameter tuning. The best results of Ant-Cycle was achieved using previous values of parameters but for two other algorithms the best results was obtained by $\rho = 0.99$. With optimum parameters the performance of the Ant-Cycle was better than the two other. The Ant-Density was better than Ant-Quantity.

The paper includes many other charts and discussions, like stagnation and algorithm behavior for different parameter values. They also showed that the optimal number of ants to achieve better or optimum solutions in minimum cycles is approximately equal to the number of cities.

They also studied the effect of elitist ants (some ants with better result affect the tour pheromone more than others) on the performance of the algorithm. Finally they have examined the algorithm on some other problems too and made some comparisons with other heuristic algorithms.

The paper (Dorigo et al., 1993) is another derivation of ant algorithm, called ant colony system (ACS). They implemented Ant-Q to solve TSP (Dorigo et al., 1996).

Ant's primary set up: Ants are located randomly on selected cities.

Selecting next city: This paper uses a new selection rule named ''State Transition Rule''. In this rule the selection is done using Eq. (6).

$$s = \begin{cases} \arg_{(ij)\notin \text{tabu}} \max\left\{\tau_{ij}^{\alpha} \cdot \eta_{ij}^{\beta}\right\} & \text{if } q \leqslant q_0 \\ S & \text{Otherwise} \end{cases} \tag{6}$$

In this equation q_0 is a parameter and q is a random number (both in $[0,1]$) and s is the selected city. S is a city which will be determined by Eq. (1). So with probability of q_0 the most attractive link will be selected and with $(1-q_0)$ a link will be selected by Eq. (1).

Pheromone updating: Both *local* and *global* updating are used. The *global updating* rule obeys Eq. (7).

$$\tau(t+1) = (1-\rho)\tau(t) + \rho\Delta\tau \tag{7}$$

$$\Delta\tau = \frac{1}{L_{\text{ShT}}}$$

L_{ShT} is the length of shortest tour founded in a cycle, i.e. this implementation uses the elitist ants' idea with only one elite ant. Here only the best elite ant can update its path and other ants are not allowed affect their tours.The *local updating* rule is something like Eq. (8) with the local updating constant (LUC) of τ_0.

$$\tau(r,s) = (1-\lambda)\cdot\tau(r,s) + \lambda\cdot\tau_0 \tag{8}$$

Here $\tau(r,s)$ is the pheromone of the link connecting cities r and$s$. In this paper λ and ρ are equal (both are the same and model pheromone evaporation) but they may differ in general.

The paper also offers Eq. (9) as a formula for choosing τ_0 adaptively.

$$\tau_0 = \frac{1}{nL_{\text{nearest}}} \tag{9}$$

Where n is the number of cities and L_{nearest} is the tour length produce by the nearest neighbor heuristic.

Other parts: In this paper the experiments were done by 10 ants and $\beta = 2$, $\alpha = 1$ and $q_0 = 0.9$ and the results were compared with other algorithms (such as SA, NNs, GA, EP, a combination of simulated annealing and genetic algorithms (AG) ....) as well.

They also examined ACS for some bigger problems. For these problems they implemented a slightly modified version of ACS which incorporates a more advanced data structure known as *candidate list* (Reinelt, 1994; Johnson et al., in press). A candidate list is a list of preferred cities to be visited as next city; it is a static data structure which contains, for a given city i, the cl closest cities. Here an ant chooses the next city to move to from candidate list. Only if none of the cities in the candidate list can be visited it considers the other cities. They reported better results than the other algorithms, using many charts and tables.

In the paper (Kaegi & White, 2003) the idea of elitist ants was used with another implementation, which is more suitable for parallel processing.

Ant's primary set up: Each ant is placed on a random city.

Selecting next city: Eq. (1) is the selection rule with the same parameters.

Pheromone updating: this paper used the updating rule of Ant-Cycle (Colorni et al., 1992). No *local update* was used and only *global update* was considered. The pheromone updating was reinforced by elitism concept.

In this paper, instead of reinforcing the pheromone update only for some elitist ants after Ant-Cycle update, the Ant-Cycle update is done again for whole ants but with the ant's local best tour, saved in the ant's memory. So after completing a tour, each ant updates its local tour memory and if the current tour is better than former ones, then the new tour will be saved as the best tour of the ant.

Other parts: Experiments were done by parameters of $\alpha = 1$, $\beta = 5$, $\rho = 0.5$ and $\tau_0 = 10^{-6}$. Their results showed that their algorithm is better than the last best algorithms in convergence time.

The document (Maniezzo, Gambardella, & De Luigi, 2004) discusses about some special Ant Algorithm implementation like ant system (AS), ant colony system (ACS) and approximated non-deterministic tree search (ANTS). The two first one were discussed above and the last one which is based on partial solutions will be discussed here.

Ant's primary set up: Random distribution.

Selecting next city: State transition rule was used with a little change: Eq. (1) was replaced with Eq. (10).

$$P_{ij}^{k}(t) = \begin{cases} \dfrac{[\tau_{ij}(t)]^{\alpha} + [\eta_{ij}(t)]^{\beta}}{\sum_{k \notin \text{tabu list}}([\tau_{ik}(t)]^{\alpha} + [\eta_{ik}(t)]^{\beta})} & \text{if} \quad j \notin \text{tabu list} \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

The parameters of the equation are the same as used before.

Pheromone updating: Here link pheromones are updated immediately when ants pass them. This updating obeys Eq. (11).

$$\Delta\tau_{ij} = \tau_0\left(1 - \frac{Z_{\text{curr}} - \text{LB}}{\overline{Z} - \text{LB}}\right) \tag{11}$$

Where Z_{curr} is current solution, $\overline{Z}$is the average of previous solutions and LB is a lower bound on the optimal problem solution.

So the pheromone level of last link is increase if Z_{curr} is lower than $\overline{Z}$, otherwise decreased.

Other parts: After representing ant algorithm different implementations the document investigates non-TSP problems like sequential ordering problem, vehicle routing problem, quadratic assignment problem and some others.

The paper (Meibodi & Noferesty, 2006) is a different approach aimed to improve the algorithm's performance. The main idea of the paper is to modify values of parameters while the algorithm is running. The approach uses learning automata to find the best values of parameters.

Generally this approach could be implemented on each derivatives of ant algorithm. This algorithm is an AS adapted to the Steiner Tree problem. Their report demonstrate their success in improvement the main algorithm results.

3. The proposed algorithms

As said above we propose two new algorithms which contain innovative parts especially local updating rule. In these algorithms different parts are designed more logically. The main novel points are two innovative *local updating* rules as follows:

When a tour starts, all links have the same amount of pheromone and so they have equal chance to be selected by ants. When an ant selects a link and passes through it, the pheromone amount of the link will be increased by the *local update* rule (which is reverse proportional to the length of link). This process makes the edge more desirable for other ants that have this edge as a choice in their path. More amount of pheromone on an edge, more desirable to select.

Is it a good strategy to give all agents the same possibility of affecting on the edges' pheromone? Consider an ant in its primary steps, while the ant arrives to a city and wants to choose its next link (city), the number of possible cases is y big, because it has passed just a few cities and consequently just a few links are prohibited to be selected, it can freely choose the most desirable link (with more pheromone and less length) as its next link.

Now consider the same ant in its final steps of tour. Here it has passed most cities and now have few city to choose. Because at this situation it passed most of the cities, the current selected link may not have a significant effect on the quality of the tour, so it seems logical to reduce its ability of changing the last links pheromone.

Consider a bad choice in primary parts of a tour (cause of *selecting path rule* or some obligation of links), it may cause successive bad selections in latter parts and may increase errors and finally cause to bad result. And so a good start may lead in bad result, therefore it seems to be logical to let agents have more effect on pheromone update where they are in their initial steps and less effect when they're going to finish the tour. Based on above discussions two new ant algorithms were designed and a lot of experiments were done to compare new algorithms with former ones. These two algorithms have the same global structure as the standard one. In the first algorithm called Kcc-AntS, Eq. (8) was replaced with Eq. (12).

$$\tau(r,s) = \tau(r,s) + \frac{K \cdot cc \cdot \tau_0}{\mathrm{Cl}^{\frac{cc}{\eta}}} \tag{12}$$

Where "cc" is the current city number (i.e. the number of cities passed till now). Cl is the current length of passed path of each ant and finally K & η are two parameters which determine the significance of the number of passed cities (cc) and length of past paths (Cl) in the updating process. In the second algorithm called ELU, *local updating* rule for a problem with M cities (node) obeys Eq. (13).

$$\tau(r,s) = \tau(r,s) + \tau_0 \cdot \mathrm{e}^{-\frac{5cc}{M}} \tag{13}$$

As it is obvious, increasing cc, the second term of Eq. (13) is exponentially decreasing toward zero and when $cc = M$ the term is approximately zero ($\mathrm{e}^{-5} \approx 0$). So the ants play fewer roles in local pheromone update when they are in their final parts of tour. As it was discussed before, the main idea is that the ants have more ability to change the pheromone of the initial links than the last links. One of the logical choices is using a decreasing exponential function of cc. But as discussed before in the first cycle, when an ant starts, the links have equal pheromone, so the probability has more effect on selecting links than links' pheromone. After some cycles the pheromone would demonstrate its effect. So it could be better to increase ants' effect in local pheromone update when the pheromone has made its effect on edges and after some step decrease ants affect. Also the length of passed paths could show how elite is an ant. Considering all these reasons, Eq. (12) could be satisfying.

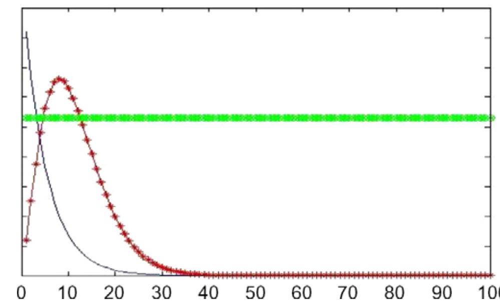

Fig. 4. The effect of old algorithms (Green-Wide), Kcc-AntS (Red-Star) and ELU-AntS (Blue-Narrow) on local pheromone update along their tour.

Fig. 4 clarifies the difference between *local updating* rule used in former algorithms and two one (Kcc-AntS and ELU-AntS). In this virtual problem with 100 cities, ants have a constant effect along the tour in the previous *local updating rule.* In ELU-AntS, ants have less ability to change pheromone (almost zero) when they are in the last part of the tour. In Kcc-AntS the ants have some starting chance for *local update* which increases for a while and then will decreases toward zero.

4. Experiments

To evaluate the two new algorithms, good experiments were done on 17 *standard TSP problems* caught from *TSP Library* and the results were compared with the standard algorithm.[1] In addition, the τ_0 was varied in order to evaluate algorithms' result while the primary parameter set up is changed. We interpret this as the robustness against parameter. The experiments were done in similar situation except in *local update* rule which was different for each algorithm. The two new algorithms were structured as follows:

Primary setup: Random distribution.
Selecting next city: State transition rule, using Eq. (10).
Global pheromone update: It was done using Eq. (7).
Local pheromone update: It was done using

[1] http://elib.zib.de/pub/Packages/mp-testdata/tsp/tsplib/tsplib.html.

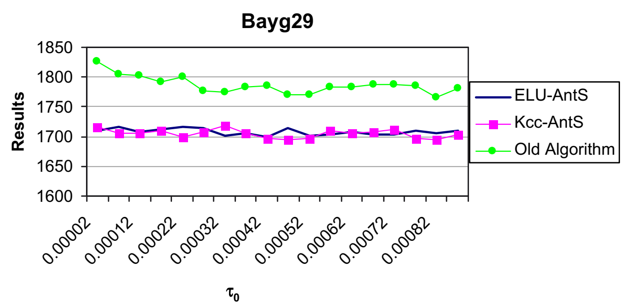

Fig. 5. New algorithms has better results and less variation.

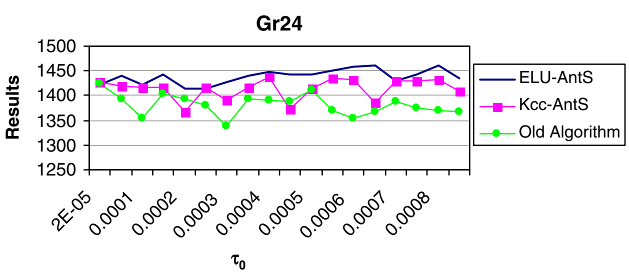

Fig. 6. Even in those few problems which old algorithm had better results, it had worse variance/average.

- *Old algorithm*: Eq. (8).
- *Kcc-AntS*: Eq. (12).
- *ELU-AntS*: Eq. (13).

Parameters setup:
PDS[2] $= 0.2$ $\lambda = 0.1$ $q_0 = 0.9$ $\eta = 9$
$\rho = 0.9$ $\alpha = 0.1$ $\beta = 2$ $K = 0.1$

It should be mentioned that α, β, ρ & q_0 were selected as was advised in Colorni et al. (1992), Colorni et al. (1996) and other parameters were selected optionally. For each value of τ_0, algorithms were run 15 times and the average was assigned as the result of that value of τ_0. Also the algorithms were iterated on each problem for 18 different values of τ_0. The results demonstrate that Old Algorithm has worst results and large amount of variations respect to the two new algorithms. Also the results show that the offered value for τ_0 is not optimum in all situations and all problems. As an example Fig. 5 may be attended.

Although the old algorithm has better results for some problems (only two problems) but as shown in Fig. 6, the variance/average of the results of the old algorithm was bigger than new ones (it was more than two times of ELU-AntS).

By the way when the problem dimension gets larger (number of cities increases) new algorithms give better results.

Table 1 includes the average of 18 algorithms' results for different values of τ_0, assigned as average, minimum of these 18 averages (each one stand for 15 iterations with one value of τ_0) and finally variance/average multiplied by 100.

It can simply be find out from the above tables that new algorithms have better results and variation respect to old algorithms.

Table 1
Average, minimum and variation coefficient of results caught in experiments different algorithms. New ones have obviously better results

Problem	Old	Kcc	ELU
Average			
Gr24	**1381.2**	1414.0	1438.3
Fri26	992.3	**940.5**	941.8
Bayg29	1786.6	**1704.8**	1707.9
Bays29	2296.9	**2137.7**	2224.2
Dantzig42	**813.3**	843.4	843.9
Swiss42	1478.4	**1446.2**	1446.8
Gr48	5952.0	**5718.5**	5759.5
HK48	5990.7	**5722.1**	5772.7
Brazil58	24838.5	23735.2	**23540.1**
Pr76	75408.0	**72519.9**	73117.0
Eil101	255.6	**228.4**	229.5
Bier127	55469.6	**48278.3**	49126.1
KroB150	18795.2	17278.2	**17046.0**
KroB200	20911.7	19626.5	**19527.9**
Tsp225	2056.4	**1882.4**	1892.3
A280	56048.7	**47931.3**	48880.5
Lin318	24927.4	**23059.0**	23085.7
Minimum			
Gr24	**1338**	1366.8	1412.9
Fri26	974.8	937.7	**937.5**
Bayg29	1766.7	**1694.9**	1698.9
Bays29	2235.4	**2134**	2187.6
Dantzig42	**799.8**	820.4	832
Swiss42	1451.5	1436.2	**1426**
Gr48	5869.8	5665.4	**5613.2**
HK48	5898.8	**5653.1**	5662
Brazil58	24123.3	23314.4	**23110.8**
Pr76	73190.8	**71012.4**	72060.6
Eil101	247.6	**221.6**	225.3
Bier127	54150.2	**47286.6**	47778.2
KroB150	18540.3	17069.3	**16888.4**
KroB200	20633.7	19383.1	**19280.9**
Tsp225	2019.3	**1841.8**	1849.5
A280	54601.2	**46696.1**	47297.6
Lin318	24566.5	**22885.7**	22890.0
100 Variance/average*			
Gr24	34.44	32.71	**15.14**
Fri26	9.84	**0.21**	0.54
Bayg29	11.86	2.85	**1.52**
Bays29	28.44	**0.51**	30.15
Dantzig42	6.21	14.14	**5.34**
Swiss42	19.02	**2.14**	9.82
Gr48	**21.28**	22.24	62.56
HK48	59.30	**16.73**	47.61
Brazil58	513.95	441.02	**396.06**
Pr76	1898.6	671.82	**401.90**
Eil101	3.68	3.75	**2.31**
Bier127	1020.16	**391.75**	635.84
KroB150	114.48	71.49	**62.68**
KroB200	105.09	**83.10**	142.02
Tsp225	37.50	37.68	**16.97**
A280	858.74	**616.56**	887.8
Lin318	196.51	**78.66**	96.96

[2] Pheromone density setup.

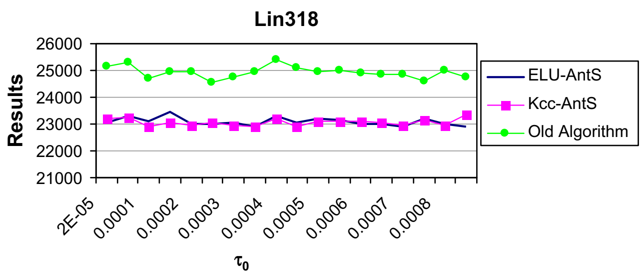

Fig. 7. Two new algorithms have very similar behavior.

Comparing two new algorithms, it can be understood that Kcc-AntS have better Averages respect to ELU-AntS but worse Minimums. These two are equal in variance/average of the results.

As it is shown in Fig. 7, two new algorithms have very similar behavior but Kcc-AntS algorithm is slightly better. Selecting an algorithm is dependent to the problem is to be solved. Since the algorithms are tolerant against changes of τ_0, it could be ignored as a parameter that needs tuning (i.e. using new algorithms there is no critical need to tune τ_0). As it is clear in Eq. (12), Kcc-AntS has two parameters to tune (K & η) but ELU-AntS (Eq. (13)) has no parameter to tune.

Table 2 includes the relative distance respect to the best result found by three algorithms. And as it's shown in the table, old algorithm has usually errors between 4% and 17% and where the new algorithms have usually errors less than 2% and some times they found the optimal solutions.

So based on discussions above, new algorithms are almost always preferred to old algorithms, especially when the problem dimension increases. And between two new algorithms Kcc-AntS is slightly better than the other. This slight superiority may be ignored, regarding to 2% error and NO tuning parameters of ELU-AntS.

Figs. 8 and 9 are graphs of Table 2 which are embedded to have better overview about difference of algorithms.

As it is shown in Figs. 8, increasing the problem dimension, error percent of old algorithm increases but the new ones decrease.

Variance of error decreases by problem dimension increase but new algorithms are faster, as it is shown in Fig. 9.

It's interesting to be mentioned that we found shorter tours and improve results in six problems, respect to what exists in *TSP Library*. The problems which better results were found are:

- Brazil58
- Pr76
- Bier127
- KroB150
- KroB200
- TSP225

Table 2
Although Kcc AntS is better than ELU AntS but it can be ignored somehow because it's less than 2% worse than Kcc AntS

Problem	Old	Kcc	ELU
Old algorithm			
Gr24	0	0	127.5
Fri26	5.5	4	4585.7
Bayg29	4.8	4.2	680.3
Bays29	7.4	4.7	5476.5
Dantzig42	0	0	16.3
Swiss42	2.2	1.8	788.8
Gr48	4.1	4.6	0
HK48	4.7	4.3	254.4
Brazil58	5.5	4.4	29.8
Pr76	4	3.1	372.4
Eil101	11.9	11.7	59.3
Bier127	15.2	14.5	160.4
KroB150	10.3	9.8	82.6
KroB200	7.1	7	26.5
Tsp225	9.2	9.6	121
A280	16.9	16.9	39.3
Lin318	8.1	7.3	149.8
Kcc-AntS			
Gr24	2.4	2.4	2.4
Fri26	0	0	0
Bayg29	0	0	0
Bays29	0	0	0
Dantzig42	3.7	3.7	3.7
Swiss42	0	0	0
Gr48	0	0	0
HK48	0	0	0
Brazil58	0.8	0.8	0.8
Pr76	0	0	0
Eil101	0	0	0
Bier127	0	0	0
KroB150	1.4	1.4	1.4
KroB200	0.5	0.5	0.5
Tsp225	0	0	0
A280	0	0	0
Lin318	0	0	0
ELU-AntS			
Gr24	4.1	4.1	4.1
Fri26	0.1	0.1	0.1
Bayg29	0.2	0.2	0.2
Bays29	4	4	4
Dantzig42	3.8	3.8	3.8
Swiss42	0	0	0
Gr48	0.7	0.7	0.7
HK48	0.9	0.9	0.9
Brazil58	0	0	0
Pr76	0.8	0.8	0.8
Eil101	0.5	0.5	0.5
Bier127	1.8	1.8	1.8
KroB150	0	0	0
KroB200	0	0	0
Tsp225	0.5	0.5	0.5
A280	2	2	2
Lin318	0.1	0.1	0.1

And in one problem (Fri26) the same result with *TSP Library* was caught.

As a final note, it should be mentioned that Kcc-AntS was run for $K = 1$ and $K = 1/\text{cc}$ but generally they could not be better than $0.1cc$-AntS and ELU-AntS.

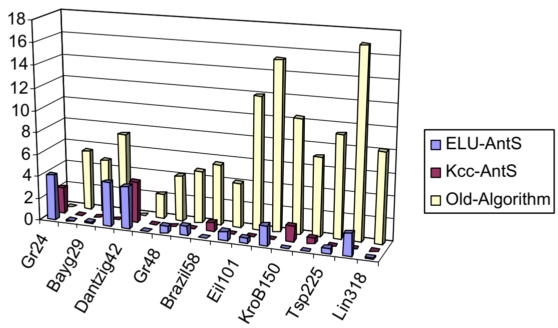

Fig. 8. Graph for average column of Table 2 which compares algorithms relative error respect to best result caught. (Best column has similar graph.)

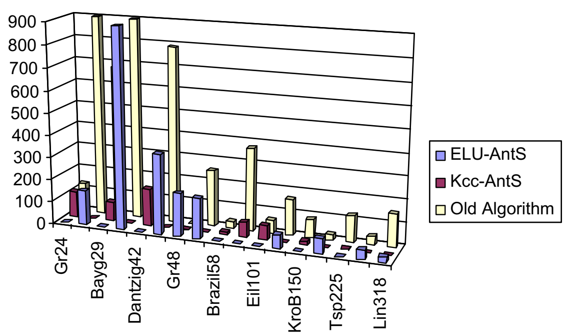

Fig. 9. An overview of variance/average of the results. (Huge values are normalized to 900 in order to have a better view from other points.)

5. Conclusion and further researches

Two new ant algorithms were presented in this paper and were compared with the best former algorithm and found better results.

It was shown that offered τ_0 in former algorithms was not adequate but in new algorithms the error percent against τ_0 variation is usually less than 2% which should be ignored and there is no need to tune this parameter in these algorithms.

Comparing two new algorithms, both have very similar behavior but Kcc-AntS (with $K = 0.1$) is slightly better but it had two parameters to tune, while ELU-AntS had NO tuning parameter. So while we face with combinational algorithm which tune themselves (Meibodi & Noferesty, 2006) or we have tuning possibility the Kcc-AntS is offered and otherwise ELU-AntS.

Further researches:

Further researches should be done on:

- $K\&\eta$ tuning in Kcc-AntS and tuning other parameters of algorithms.
- Studying algorithm behavior against problem specifications and find an exact relation to explain it.
- Decreasing tuning parameters and make algorithm robust against parameter tunes (as it was done in this paper for τ_0).

References

An introduction to ant algorithms and to ant colony optimization (2000). In *ANTS'2000 – From ant colonies to artificial ants: Second international workshop on ant algorithms*, Brussels, Belgium, September 7, 2000.

Beckers, R., Aron, S., Deneubourg, J. L., & Pasteels, J. M. (1993). Memory and chemical communication in the orientation of two mass-recruiting ant species. *Insectes Sociaux, 40*, 369–380.

Beckers, R., Deneubourg, J. L., & Goss, S. (1992). Trails and U-turns in the selection of the shortestpath by the ant Lasius Niger. *Journal of Theoretical Biology, 159*, 397–415.

Blum, C. (2005). Ant colony optimization: Introduction and recent advances. In *Fifth international conference on hybrid intelligent systems (HIS'05)*, Rio de Janeiro – Brazil, 6–9 November.

Colorni, A., Dorigo, M., Maniezzo, V., et al. (1992). *An investigation of some properties of ant algorithm*. Brussels, Belgium: Elsevier Publishing (pp. 509–520), PPSN.

Colorni, A., Dorigo, M., Maniezzo, V., et al. (1996). *The ant system optimization by a colony of cooperating agent*, IEEE.

Dorigo, M., Gambardella, L. M., et al. *Ant colonies for travelling salesman problem*, TR/IRIDIA/1993-6, Universite' Libre de Bruxelles, Belgium.

Dorigo, M., Gambardella, L. M. *A study of some properties of ant-q*, TR/IRIDIA/1996-4 Université Libre de Bruxelle,Belgium, published in PPSN IV–fourth international conference.

Glover, F., & Laguna, M. (1997). *Tabu search*. Boston: Kluwer Academic Publishers.

Goss, S., Aron, S., Deneubourg, J. L., & Pasteels, J. M. (1989). Self-organized shortcuts in the argentine ant. *Naturwissenschaften, 76*, 579–581.

Holldobler, B., & Wilson, E. O. (1990). *The Ants*. Berlin: Springer-Verlag.

Johnson, D.S. and McGeoch, L.A., in press. The travelling salesman problem: A case study in local optimization. In *Local search in combinatorial optimization*, E. H. L., Aarts, & J. K. Lenstra (Eds.), Wiley, New York.

Kaegi, Simon, & White, Tony (2003). *Using local information to guide ant based search, school of computer science*. Ottawa, Ontario, Canada: Carleton University.

Lawler, E. L., Lenstra, J. K., Rinnooy Kan, A. H. G., & Shmoys, D. B. (Eds.). (1985). *The traveling salesman problem*. Wiley.

Maniezzo, V., Gambardella, L. M., F., De Luigi, et al. (2004). *Ant colony optimization*, Aco 2004.

Meibodi, M., Noferesty, S., (2006). A combinational algorithm (Ant colony + Learning automata) to resolve steiner static tree problem. In *ICEE* 2006.

Nagata, Y., & Kobayashi, S. (1997). Edge assembly crossover: A high-power genetic algorithm for the traveling salesman problem. In *Proceedings of seventh international conference on genetic algorithms (ICGA'97)*, East Lansing, Michigan (pp. 450–457).

Reinelt, G. (1994). *The traveling salesman: Computational solutions for TSP applications*. Berlin: Springer-Verlag.