

# Ant Colony Optimization with Direct Communication for the Traveling Salesman Problem

Michalis Mavrovouniotis and Shengxiang Yang

**Abstract**—Ants in conventional ant colony optimization (ACO) algorithms use pheromone to communicate. Usually, this indirect communication leads the algorithm to a stagnation behaviour, where the ants follow the same path from early stages. This occurs because high levels of pheromone are developed, which force the ants to follow the same corresponding trails. As a result, the population gets trapped into a local optimum solution which is difficult to escape from it. In this paper, a direct communication (DC) scheme is proposed where ants are able to exchange cities with other ants that belong to their communication range. Experiments show that the DC scheme delays convergence and improves the solution quality of conventional ACO algorithms regarding the traveling salesman problem, since it guides the population towards the global optimum solution. The ACO algorithm with the proposed DC scheme has better performance, especially on large problem instances, even though it increases the computational time in comparison with a conventional ACO algorithm.

## I. INTRODUCTION

Ant colony optimization (ACO) algorithms emulate the simple behaviour of real ants when they search food from their nest to food sources [4]. Ants cooperate to perform this food searching task as efficiently as possible using an indirect communication mechanism via pheromone. Pheromone is a chemical substance produced by ants and applied to their trails. The more pheromone on a specific trail, the higher the possibility of that trail to be followed by the ants.

The distributed optimization behaviour of ants inspired researchers to develop the first ACO algorithm, called ant system (AS) [3]. This algorithm was first applied to the traveling salesman problem (TSP), which is one of the most widely studied *NP*-hard combinatorial optimization problems. The TSP is the problem of finding the shortest cyclic tour among a topology of cities, by visiting each city once. In general, ACO algorithms have a high risk to get trapped on local optima, since high levels of pheromones on specific trails may force ants to converge on a local optimum solution.

In nature, ants do not only communicate indirectly by pheromone trails, but also directly with other ants and gather important information [10]. In this paper, a direct communication (DC) scheme is proposed for ACO algorithms to address the TSP, where an ant is able to communicate with other ants within its neighbourhood. The neighbourhood is

considered as the communication range of each ant and is based on the common edges between two ants. Ants within their communication range are allowed to exchange cities with each other, only if it is advantageous regarding their tour length. Additionally, a small amount of pheromone is added to the exchanged cities in order to influence ants towards new promising paths generated from DC.

In order to investigate the performance of the proposed scheme for ACO algorithms, referring to the TSP, experiments are carried out to compare a conventional ACO algorithm and an ACO algorithm with the proposed scheme on a set of benchmark TSP instances. Experimental results show that the proposed scheme improves the solution quality of conventional ACO algorithms since it enables ants to avoid local optima and lead the population towards the global optimum. On the other hand, it increases the computational time because of the extra tasks it needs to perform in order to carry out communications between ants.

The rest of the paper is organized as follows. Section II, defines the framework of the TSP. In Section III, we first describe the AS algorithm proposed for the TSP. Then, we describe the best performing ACO, i.e., the max-min AS (MMAS), which will be used for the experiments later on. In Section IV, we describe the proposed DC scheme, giving details on how it can be applied with ACO algorithms. In addition, we discuss possible advantages and disadvantages of using this scheme. Section V presents the experimental results of the MMAS with the proposed DC in comparison with a conventional MMAS on different TSP problem instances. We analyze the effect of the communication range used in the DC scheme, the solution quality, convergence speed and computational runtime. Moreover, we represent statistical tests which show that the MMAS with DC delays convergence and improves the solution quality significantly on most problem instances. Finally, Section VI indicates concluding remarks and several directions of future work.

## II. THE TRAVELING SALESMAN PROBLEM

The TSP is one of the most popular combinatorial optimization problems, which is classified as *NP*-hard. Usually, the problem is represented by weighted graphs. Consider a graph  $G = (V, E)$ , where  $V$  is a set of vertices and  $E$  is a set of edges. The collection of cities is represented by the  $V$  set and the collection of edges between them by the  $E$  set. In addition, the  $E$  set is associated with  $C$ , which is denoted as  $C = (l_{ij})$ , where  $l_{ij}$  is the metric of the edge between cities  $i$  and  $j$ . The objective of the TSP is to determine the

M. Mavrovouniotis is with the Department of Computer Science, University of Leicester, University Road, Leicester LE1 7RH, United Kingdom (email: mm251@mcs.le.ac.uk).

S. Yang is with the Department of Information Systems and Computing, Brunel University, Uxbridge, Middlesex UB8 3PH, United Kingdom (email: shengxiang.yang@brunel.ac.uk).

permutation of cities in the  $V$  set that minimizes the length of a cyclic tour which contains each city once and only once.

A lot of algorithms have been proposed to solve the TSP, either exact or approximation algorithms (also known as heuristics) [9], [11]. Although exact algorithms guarantee to provide the global optimum solution, due to the  $NP$ -hard property, TSPs need, in the worst case scenario, an exponential time to find the global optimum solution. On the other hand, approximation algorithms can provide a good solution efficiently but cannot guarantee the global one [8].

ACO algorithms are able to provide the optimum or near-optimum solution in a sufficient amount of time, since they sacrifice their solution quality for the sake of efficiency (time) [12]. Moreover, it has been shown that their solution quality can be significantly improved with the use of a local search operator. However, such methods may increase the computation time significantly especially on large problem instances [13].

### III. CONVENTIONAL ANT COLONY ALGORITHMS

#### A. Ant System

The AS [3] is the first ACO algorithm applied for the TSP. It has a population of ants which are initially placed on randomly selected cities. Ants read and write pheromones in order to construct their solutions. Each ant  $k$  uses a probabilistic rule to choose the next city to visit. The decision rule an ant  $k$  uses to move from city  $i$  to city  $j$  is defined as follows:

$$p_{ij}^k = \frac{[\tau_{ij}]^\alpha [\eta_{ij}]^\beta}{\sum_{l \in N_i^k} [\tau_{il}]^\alpha [\eta_{il}]^\beta}, \text{ if } j \in N_i^k, \quad (1)$$

where  $\tau_{ij}$  is the existing pheromone trail between cities  $i$  and  $j$ ,  $\eta_{ij}$  is the heuristic information available a priori, which is defined as  $1/d_{ij}$ , where  $d_{ij}$  is the distance between cities  $i$  and  $j$ .  $N_i^k$  denotes the neighbourhood of cities for ant  $k$ , when being on city  $i$ .  $\alpha$  and  $\beta$  are the two parameters which determine the relative influence of  $\tau$  and  $\eta$ , respectively. They have a significant impact on ACO algorithms to achieve a robust behaviour.

After all ants have visited all cities and constructed tours, they update their pheromone trails. Initially, all trails contain an equal amount of pheromone. Each ant retraces its solution and according to the quality of the solution, it deposits pheromone to the corresponding trails. This process is defined as follows:

$$\tau_{ij} \leftarrow \tau_{ij} + \Delta\tau_{ij}^k, \forall (i, j) \in T^k, \quad (2)$$

where  $\Delta\tau_{ij}^k = 1/C^k$  is the amount of pheromone that ant  $k$  deposits and  $C^k$  is the cost of the tour  $T^k$ .

Furthermore, a constant amount of pheromone is deduced from all trails due to the evaporation of pheromone. This process enables the population to eliminate bad decisions from previous tours. This process is defined as follows:

$$\tau_{ij} \leftarrow (1 - \rho) \tau_{ij}, \forall (i, j), \quad (3)$$

---

#### Algorithm 1 Conventional ACO

---

- 1: initialize data
  - 2: **while** termination-condition not satisfied **do**
  - 3:   construct solutions
  - 4:   update best ants
  - 5:   global pheromone update (evaporation + deposit)
  - 6: **end while**
- 

where  $\rho$  is the evaporation rate which satisfies  $0 < \rho \leq 1$ , and  $\tau_{ij}$  is the existing pheromone value. A general framework of a conventional ACO algorithm is represented in Algorithm 1.

Ants in the AS algorithm generate a high intensity of pheromones from early stages, which leads the algorithm into a stagnation behaviour, where all ants follow the same path. This eliminates the exploration of paths since the maximum and minimum amounts of pheromone have a significant difference. As a result, AS is more likely to get trapped in a local optimum solution, which may degrade the solution quality.

#### B. MAX-MIN Ant System

The MMAS [13] is an improvement of the AS. It is one of the most studied and best performing ACO algorithms. The motivation of the MMAS is to address the stagnation behaviour of AS and explore a wider region of the search space. MMAS differs from AS in the way pheromone trails are manipulated.

Explicitly, MMAS differs from AS in three main aspects: 1) Only either the best-so-far ant, i.e., the best ant from all iterations, or the iteration-best ant, i.e., the best ant from the current iteration is allowed to update the pheromone trails; 2) the pheromone trail values are bounded into an interval of  $[\tau_{min}, \tau_{max}]$ ; and 3) the pheromone trails are re-initialized to  $\tau_{max}$  once the algorithm approaches stagnation behaviour. The pheromones in MMAS are updated by applying evaporation, as in Eq. (3), with a much smaller evaporation rate and deposit as follows:

$$\tau_{ij} \leftarrow \tau_{ij} + \Delta\tau_{ij}^{best}, \forall (i, j) \in T^{best}, \quad (4)$$

where  $\Delta\tau_{ij}^{best} = 1/C^{best}$  is the amount of pheromone that the best-so-far ant deposits and  $C^{best}$  defines the solution quality of tour  $T^{best}$ . In the case where the iteration-best ant is allowed to deposit pheromone,  $\Delta\tau_{ij}^{best} = 1/C^{ib}$ , where  $C^{ib}$  defines the solution quality of the iteration-best ant. Both update rules are used in an alternative way under a pre-defined criteria (for more details see [14]). Since only the best ant deposits pheromone, the algorithm will quickly converge towards the best solution of the first iteration. Therefore, the pheromone trail limits are imposed in order to avoid this behaviour.

MMAS is more explorative than AS since highly crowded areas of pheromones are eliminated and the difference between  $\tau_{min}$  and  $\tau_{max}$  values is smoother. This gives the less desired paths more chances to be considered during the construction of solutions. MMAS has a longer explorative phase

---

**Algorithm 2** ACO with Direct Communication

---

```
1: initialize data
2: while termination-condition not satisfied do
3:   construct solutions
4:   perform direct communication
5:   update best ants
6:   global pheromone update (evaporation + deposit)
7:   local pheromone update
8: end while
```

---

and converges more slowly than AS does. However, there is a possibility of getting trapped in a local optimum, especially on large problem instances, since the best ant always deposits pheromone. As a result, the tour corresponding to the best ant always rises up to the maximum  $\tau_{max}$ .

MMAS has been improved with the integration of a local search. The 2-opt operator has been applied to the best ant before the pheromone update [13]. This method has also improved the solution quality of other ACO algorithms [2], [5], [6], since a stronger emphasis has been given to the best solution. However, such operators increase the computation time, especially on large problem instances.

#### IV. DIRECT COMMUNICATION OF ANTS

In nature, ant colonies do not only communicate indirectly with their pheromone trails, but also directly by exchanging important information [10]. We can integrate direct communication into ACO algorithms by allowing ants to exchange cities after they construct their solutions, as shown in Algorithm 2.

Each ant  $ant_i$  communicates with another ant within its communication range as follows:

- 1) A city  $c_i$  is randomly selected from ant  $ant_i$ .
- 2) The successor and predecessor of  $c_i$ , i.e., cities  $c_{i-1}$  and  $c_{i+1}$ , respectively, are selected from  $ant_i$ .
- 3) Another ant is selected, i.e.,  $ant_j$ , from the communication range of  $ant_i$  and city  $c_i$  is located in  $ant_j$ .
- 4) The successor and predecessor of  $c_i$ , i.e., cities  $c'_{i-1}$  and  $c'_{i+1}$ , respectively, are selected from  $ant_j$  and located in  $ant_i$ .
- 5) Swaps are performed in  $ant_i$  between cities  $c_{i-1}$  and  $c'_{i-1}$  and between cities  $c_{i+1}$  and  $c'_{i+1}$ .
- 6) A small extra amount of pheromone is deposited to the resulting edges between  $c_i$  and its successor and  $c_i$  and its predecessor in  $ant_i$ .

The above communication scheme has a high risk to degrade the solution quality of the tours constructed by the ants and disturb the optimization process. Therefore, only the swaps which are beneficial are allowed in order to limit the risk. For example, if the distance between the current successor city of city  $c_i$  in ant  $ant_i$  has less distance from the successor city obtained from the neighbour ant  $ant_j$ , then  $ant_i$  remains unchanged. The same happens with the predecessor city of city  $c_i$  in ant  $ant_i$ . A similar swap method based on the position of the cities has been used on a discrete

version of particle swarm optimization on the TSP [15]. However, the positions of the swapped cities are predefined and they are not obtained from other individuals as in the proposed DC scheme.

Apart from the swaps, a small amount of pheromone is deposited to the edges affected by the swaps in order to determine the influence of DC and explore possible improvements. This process is defined as follows:

$$\tau_{ij} \leftarrow \tau_{ij} + \frac{(\tau_{max} - \tau_{min})(1 - w)}{n}, \quad (5)$$

where  $w$  is a parameter which indicates the degree of influence for the DC scheme and  $n$  is the number of cities.

The communication range of an ant  $ant_i$  with other ants is based on the similarities of ants and is defined as follows:

$$R_i = \{ant_j \in P | 1 - \frac{CE_{ij}}{n} \leq T_r\}, \quad (6)$$

where  $P$  is the population of ants,  $T_r$  is a predefined threshold which determines the size of the communication range of  $ant_i$ ,  $n$  is the number of cities, and  $CE_{ij}$  is the similarity metric between two ants, i.e.,  $ant_i$  and  $ant_j$ , which is defined as the number of common edges between them. If an edge  $E_{kl}$  or  $E_{lk}$  that appears in the solution of  $ant_i$  also appears in the solution of  $ant_j$ , then it is counted as a common edge between  $ant_i$  and  $ant_j$ . A larger value of  $T_r$  indicates a larger communication neighbourhood of dissimilar ants, whereas a value closer to 0 indicates a smaller communication neighbourhood of similar or identical ants.

The DC scheme can be applied to any ACO algorithm right after all ants construct their solutions and before update their pheromone trails globally. Note that the exchange of cities is performed locally and cities are exchanged when there is an improvement on the distance of the edge. In addition, the edge receives an extra amount of pheromone locally, as in Eq. (5), to attract ants to perform more exploration on possible promising areas in the search space. Therefore, the newly discovered areas on the search space will be considered by the ants in the next iteration.

The effect of a more complicated DC scheme has also been studied in AS [1]. Their experiments on one TSP problem instance indicate that DC provides a good balance between centralization and diversification of the search if the communication range defines a small neighbourhood. However, the range of the neighbourhood is defined according to the tour length of ants and DC is enabled with a virtual individual pheromone trail which is only recognized by themselves. The probability of selecting the next city is calculated by using both shared and individual pheromone trails.

The aim of our proposed DC scheme is to exchange cities and take advantage of the different solutions constructed by ants on each iteration. It is possible that the solution of an ant may be worse than the best ant, but a subtour may belong to the global optimum. It is also possible that a subtour in the best tour may belong to a local optimum. It is difficult for an ACO algorithm to escape from local optimum because the pheromone trails always lead the ants into the same path.

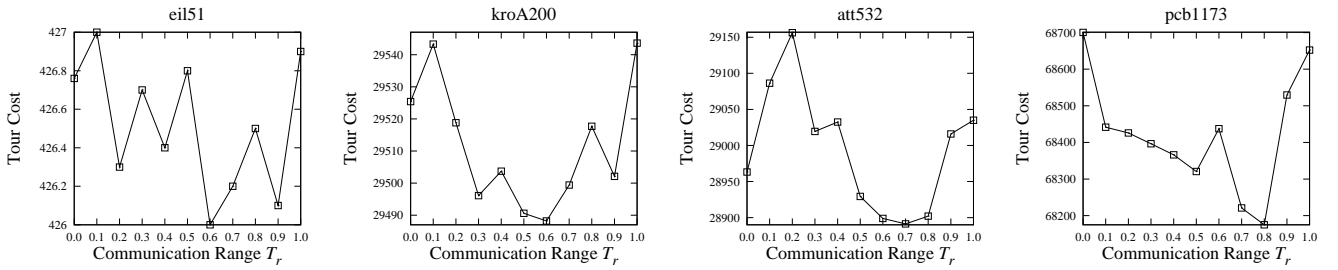


Fig. 1. The effect of the communication range parameter  $T_r$  used in the DC scheme for different problem instances

DC may help eliminate such behaviour and possibly enhance the solution quality of conventional ACO algorithms.

Furthermore, there is a trade-off between the solution quality and the computation time. Usually, when the solution quality is improved, the computation time is increased. Similarly, the DC scheme may improve the solution quality but increase the computation time because of the extra tasks. A similar case occurred when the 2-opt or 3-opt operator has been applied in MMAS [13], where the computation time increases as the problem size enlarges. However, we believe that the computation time of DC scheme will not increase significantly on large problem instances as it is not considered a pure local search operator.

## V. EXPERIMENTAL STUDY

### A. Experimental Setup

All problem instances studied as test cases in this paper are symmetric TSPs, taken from TSPLIB<sup>1</sup> and our implementation is based on the source code of ACOTSP<sup>2</sup>. For each experiment on each problem instance, 25 independent runs of each algorithm were performed for statistical purposes. For each run, 5000 iterations are performed in order to have the same number of evaluations. Since MMAS is considered as the state-of-the-art ACO algorithm for TSP [14], we use it as a peer ACO algorithm for the experiments and we apply the proposed DC scheme with MMAS, which is denoted as MMAS+DC in this paper.

Most of the parameters used in the algorithms are inspired from the literature since they have been found effective [4, p. 71]. For both MMAS and MMAS+DC the parameters  $\alpha$  and  $\beta$  used in Eq. (1), were set to 1 and 5, respectively. The evaporation constant  $\rho$  used in Eq. (3) was set to 0.2. The population size was set to  $m = 25$  for both algorithms. A good value for the  $w$  parameter used in Eq. (5) was found to be 0.5 from our preliminary experiments.

In Tables I, II and III the mean results and standard deviations are presented with their corresponding statistical results of two-tailed  $t$ -test with 48 degree of freedom at a 0.05 level of significance. The “+” or “-” indicates either the algorithm in the first column or in the second one is slightly better, respectively, and “s+” or “s-” indicates either the

algorithm in the first column or in the second is significantly better, respectively.

### B. Analysis Regarding the Effect of Communication Range

The communication range, i.e.,  $T_r$ , is an important parameter for the proposed DC scheme in order to achieve a robust behaviour for the ACO algorithm. In Fig. 1, we illustrate different values of the  $T_r$  parameter and show the effect they have on the solution quality for different problem instances. We have selected four instances, i.e., eil51, kroA200, att532 and pcb1173, indicating small, to medium, and large problem instances, respectively.

When the range parameter is 0, it means that the ants do not communicate with each other and, thus, we have the conventional MMAS algorithm. On the other hand, if the range value is 1, it means that the ants are allowed to communicate with all other ants.

The parameter is problem dependent and hence depends on the size of the problem instance. As the problem size increases the communication range should increase. This can be observed from Fig. 1 where a smaller range performs better on the smallest problem instance, i.e., eil51, and a larger range performs better on the largest problem instance, i.e., pcb1173.

In general, a good range is  $0.6 \leq T_r \leq 0.8$  which means that it would be good for the ants to communicate with dissimilar ants but to generate a reasonable size of neighbourhood. A larger value of the range may lead the population into a higher level of diversity, which may result to randomization, whereas a smaller value may be ineffective since the ants selected have more chances to be similar. An appropriate communication range influences ants to concentrate on the paths found from the ACO searching, and not disturb the optimization process.

In the following experiments, we set  $T_r = 0.6$  for eil51, eil76, eil101, kroA100, kroA150 and kroA200,  $T_r = 0.8$  for pcb1173 and pr2392, and  $T_r = 0.7$  for the remaining problem instances.

### C. Analysis Regarding the Solution Quality

In Table I, experimental results of the best solution are presented and it can be observed that the solution quality of MMAS is significantly improved for most problem instances when the DC scheme is used.

<sup>1</sup>Available from <http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/>

<sup>2</sup>Available from <http://www.aco-metaheuristic.org/aco-code>

TABLE I

THE MEAN RESULTS AND STANDARD DEVIATIONS OF THE BEST SOLUTION FOUND AFTER 5000 ITERATIONS OF THE ALGORITHMS WITH THEIR STATISTICAL RESULTS.

| Instance | MMAS      |         | MMAS+DC   |         | <i>t</i> -Test |
|----------|-----------|---------|-----------|---------|----------------|
|          | Mean      | Std Dev | Mean      | Std Dev |                |
| eil51    | 426.64    | 0.76    | 426.36    | 0.49    | –              |
| eil76    | 538.72    | 1.17    | 538.56    | 0.82    | –              |
| eil101   | 632.28    | 2.85    | 631.36    | 3.07    | –              |
| kroA100  | 21358.24  | 52.99   | 21322.76  | 49.96   | <i>s</i> –     |
| kroA150  | 26976.56  | 94.61   | 26910.68  | 88.36   | <i>s</i> –     |
| kroA200  | 29522.32  | 87.21   | 29491.36  | 44.52   | –              |
| lin318   | 42746.48  | 221.74  | 42618.16  | 215.47  | <i>s</i> –     |
| pcb442   | 52425.76  | 610.58  | 52036.80  | 592.49  | <i>s</i> –     |
| att532   | 28298.60  | 178.03  | 28104.60  | 255.85  | <i>s</i> –     |
| rat783   | 9003.68   | 51.90   | 8988.00   | 38.84   | –              |
| pcb1173  | 63583.36  | 687.46  | 63143.48  | 749.26  | <i>s</i> –     |
| pr2392   | 431250.60 | 4287.98 | 428917.20 | 3267.67 | <i>s</i> –     |

On small problem instances, the solution quality is slightly improved whereas on larger problem instances the improvement is significantly better. This is due to the fact that for small problem instances MMAS will converge to a single solution quickly. Therefore, neighbourhoods which consist of dissimilar ants cannot be easily generated in order for the DC scheme to work properly. Recall that once the algorithm converges to a solution the DC scheme becomes ineffective, until the algorithm re-initializes its pheromone trails.

On large problem instances, the conventional MMAS has more chances to get trapped on a local optimum since the search space is larger. Large problem instances require more exploration but MMAS loses its exploration ability quickly because of the stagnation behaviour. Therefore, the DC scheme is effective on such cases since different neighbourhoods of dissimilar ants are generated and provide guided exploration. This can be observed from Table I in which MMAS+DC is significantly better on most large problem instances.

#### D. Analysis Regarding the Runtime

In Table II, the experimental results of the total runtime are presented and it can be observed that the DC scheme increases the runtime significantly on most problem instances because of the extra computations.

On small problem instances the runtime of MMAS+DC scheme is significantly worse. This is normal because of the extra computation the DC scheme performs. However, it was expected that the computation time will be increased as the problem size increases, as in the case of the local search operators [9]. In our case as the problem size increases the runtime results of the two algorithms become similar.

It can be observed from the two largest problem instances, i.e., pcb1173 and pr2392, that the runtime results of MMAS+DC are better than the conventional MMAS. This is due to the pheromone re-initialization mechanism of MMAS,

TABLE II

THE MEAN RESULTS AND STANDARD DEVIATIONS OF THE TOTAL RUNTIME AFTER 5000 ITERATIONS OF THE ALGORITHMS WITH THEIR STATISTICAL RESULTS. THE VALUES ARE INDICATED IN SECONDS

| Instance | MMAS    |         | MMAS+DC |         | <i>t</i> -Test |
|----------|---------|---------|---------|---------|----------------|
|          | Mean    | Std Dev | Mean    | Std Dev |                |
| eil51    | 2.04    | 0.03    | 3.31    | 0.01    | <i>s</i> +     |
| eil76    | 3.77    | 0.04    | 5.45    | 0.03    | <i>s</i> +     |
| eil101   | 6.02    | 0.01    | 8.12    | 0.04    | <i>s</i> +     |
| kroA100  | 6.06    | 0.01    | 8.23    | 0.06    | <i>s</i> +     |
| kroA150  | 12.02   | 0.03    | 14.87   | 0.10    | <i>s</i> +     |
| kroA200  | 19.89   | 0.06    | 23.68   | 0.09    | <i>s</i> +     |
| lin318   | 46.36   | 0.05    | 49.34   | 0.11    | <i>s</i> +     |
| pcb442   | 86.52   | 4.46    | 101.53  | 1.19    | <i>s</i> +     |
| att532   | 137.09  | 1.72    | 138.70  | 2.36    | <i>s</i> +     |
| rat783   | 294.21  | 3.65    | 302.06  | 13.24   | <i>s</i> +     |
| pcb1173  | 448.89  | 183.83  | 448.24  | 150.91  | –              |
| pr2392   | 2838.09 | 7.25    | 2795.93 | 2.60    | <i>s</i> –     |

TABLE III

THE MEAN RESULTS AND STANDARD DEVIATIONS OF THE RUNTIME UNTIL THE BEST SOLUTION IS FOUND BY THE ALGORITHMS WITH THEIR STATISTICAL RESULTS. THE VALUES ARE INDICATED IN SECONDS

| Instance | MMAS    |         | MMAS+DC |         | <i>t</i> -Test |
|----------|---------|---------|---------|---------|----------------|
|          | Mean    | Std Dev | Mean    | Std Dev |                |
| eil51    | 0.79    | 0.52    | 1.39    | 0.91    | <i>s</i> +     |
| eil76    | 1.27    | 1.05    | 1.76    | 1.37    | +              |
| eil101   | 3.20    | 1.60    | 3.50    | 2.43    | +              |
| kroA100  | 2.84    | 1.68    | 4.22    | 2.48    | <i>s</i> +     |
| kroA150  | 5.01    | 3.53    | 6.43    | 4.38    | +              |
| kroA200  | 8.68    | 5.11    | 13.15   | 7.16    | <i>s</i> +     |
| lin318   | 32.05   | 10.30   | 30.69   | 12.20   | –              |
| pcb442   | 47.20   | 26.06   | 64.29   | 28.27   | <i>s</i> +     |
| att532   | 116.30  | 18.96   | 110.12  | 35.42   | –              |
| rat783   | 183.89  | 62.23   | 186.53  | 76.68   | +              |
| pcb1173  | 675.49  | 3.78    | 688.69  | 6.99    | <i>s</i> +     |
| pr2392   | 1592.32 | 803.55  | 1988.50 | 980.18  | <i>s</i> +     |

which acts as a global restart of the algorithm and requires high computational efforts. The proposed scheme decreases the probability of activating this mechanism because it provides more exploration, using the swaps and the local pheromone update, which make it more difficult to reach stagnation behaviour and re-initialize pheromone trails.

#### E. Analysis Regarding the Convergence Speed

In Table III, the experimental results of the runtime until the best solution is reached are presented in order to compare the convergence speed of the two algorithms.

The running time until the best solution is reached of the MMAS+DC is significantly slower than a conventional MMAS. The initial phase of MMAS is very explorative. However, it becomes even more explorative when using DC and, thus, it takes more time to reach an optimum.

On the other hand, sometimes the convergence speed of the proposed scheme is competitive with a conventional MMAS, i.e., on problem instances lin318 and att532. This is because it is less likely to have the pheromone trails to be re-initialized when the ants reach stagnation behaviour, as with the conventional MMAS. This is due to the extra amount of local pheromone the affected edges receive from the swaps in the DC scheme.

Considering the results of the convergence speed with the solution quality of the two algorithms it can be observed that the conventional MMAS gets trapped to a local optimum solution, because of premature convergence and needs to re-initialize pheromone trails to escape from it. The DC scheme delays convergence which helps the population to converge into a better solution, avoiding possible local optima, and can be observed from the fact that the solution quality of MMAS+DC is significantly better.

## VI. CONCLUSIONS

The communication between ants in conventional ACO algorithms is achieved indirectly by pheromones. In this paper a DC scheme is proposed, which enables the ants to communicate both directly and indirectly. The scheme allows ants to communicate and exchange subtours. Although one solution may not be better than another, it may contain a subtour which corresponds to the global optimum. The DC scheme avoids the re-initialization of pheromone trails, which is time consuming, when the algorithm reaches the stagnation behaviour. However, when the stagnation behaviour is reached, the DC scheme becomes ineffective until the pheromone trails are re-initialized.

For the experiments we use the MMAS algorithm, which is one of the best performing ACO algorithms, both with or without DC. The experimental results show that the proposed scheme sacrifices convergence speed to improve the solution quality on almost all problem instances. It takes significantly more computational time because of the extra tasks required, in order to generate the neighbourhood of ants and perform the swaps. However, it is competitive on large problem instances since it avoids to re-initialize pheromone trails regularly.

Generally, this paper shows clearly that the use of DC scheme with an appropriate communication range between ants improves the overall performance of ACO algorithms for the TSP. The improvement regarding solution quality is not significant on small problem instances. However, as the problem size increases the improvement is significant on the majority of large problem instances. It is difficult to develop an algorithm to outperform significantly other algorithms on all problem instances. However, large problem instances share more similarities with real-world problems and, thus, DC scheme can be more useful on such applications.

For further work, it will be interesting to apply the DC scheme with other ACO algorithms. The DC scheme is sensitive to its parameters, e.g., the communication range, which has been selected with experiments. A possible future study is to adapt the specific parameter. Another future study, is to investigate the effect of DC when an ACO algorithm is applied with a local search operator. Usually, on larger problem instances the solution quality of algorithms is more likely to be degraded, whereas a local search may improve it but increase the computation time. Therefore, DC may be able to guide the local search operator for better solution quality and computation time. Finally, the proposed approach may be effective for dynamic TSPs since it delays convergence and provides valuable diversity to the population of ACO algorithms. Such characteristics are suitable to help the population to adapt well on environmental changes [7].

## REFERENCES

- [1] H. Akira, I. Takumi, T. Tetsuyuki, I. Yoshinori, and S. Motoki, "Effect of direct communication in ant system," in *Knowledge-Based Intelligent Information and Engineering Systems*, pp. 925–931, 2005.
- [2] B. Bullnheimer, R. F. Hartl, and C. Strauss, "A new rank based version of the ant system - a computational study," Technical Report, *Institute of Management Science, University of Vienna*, Austria, 1999.
- [3] A. Colomi, M. Dorigo, and V. Maniezzo, "Distributed optimization by ant colonies," in *Proc. of the 1st European Conf. on Artificial Life*, pp. 134–142, 1992.
- [4] M. Dorigo and T. Stützle. *Ant Colony Optimization*. The MIT Press, London, England, 2004.
- [5] M. Dorigo, V. Maniezzo, and A. Colomi, "Ant system: optimization by a colony of cooperating agents," *IEEE Trans. on Systems, Man and Cybernetics - Part B*, vol. 26, no. 1, pp. 29–41, 1996.
- [6] M. Dorigo and L. M. Gambardella, "Ant colony system: a cooperative learning approach to the traveling salesman problem," *IEEE Trans. on Evolutionary Computation*, vol. 1, no. 1, pp. 53–66, 1997.
- [7] Y. Jin and J. Branke, "Evolutionary optimization in uncertain environments - a survey," *IEEE Trans. on Evolutionary Computation*, vol. 9, no. 3, pp. 303–317, 2005.
- [8] J. He and X. Yao, "From an individual to a population: An analysis of the first hitting time of population-based evolutionary algorithms," *IEEE Trans. on Evolutionary Computation*, vol. 6, no. 5, pp. 495–511, 2002.
- [9] S. Lin and B. W. Kernighan, "An effective heuristic algorithm for the traveling salesman problem," *Operations Research*, vol. 21, no. 2, pp. 498–516, 1973.
- [10] Y. Li and S. Gong, "Dynamic ant colony optimization for TSP," *International Journal of Advanced Manufacturing Technology*, vol. 22, no. 7-8, pp. 528–533, 2003.
- [11] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*. Springer-Verlag, Berlin, third edition, 1999.
- [12] F. Neumann and C. Witt, "Runtime analysis of a simple ant colony optimization algorithm," *Algorithmica*, vol. 54, no. 2, pp. 243–255, 2009.
- [13] T. Stützle and H. Hoos, "The MAX-MIN Ant System and local search for the traveling salesman problem," in T. Bäck, Z. Michalewicz, and X. Yao, editors, *Proc. of the 1997 IEEE Int. Conf. on Evolutionary Computation*, pp. 309–314, 1997.
- [14] T. Stützle and H. Hoos, "MAX-MIN Ant System," *Future Generation Computer Systems*, vol. 8, no. 16, pp. 889–914, 2000.
- [15] K. P. Wang, L. Huang, C. G. Zhou and W. Pang, "Particle swarm optimization for traveling salesman problem," in *Proc. of the 2nd Int. Conf. on Machine Learning and Cybernetics*, vol. 3, pp. 1583–1585, 1993.