Ant Colony Optimization with Immigrants Schemes for the Dynamic Vehicle Routing Problem

Michalis Mavrovouniotis[1] and Shengxiang Yang[2]

[1] Department of Computer Science, University of Leicester
University Road, Leicester LE1 7RH, United Kingdom
mm251@mcs.le.ac.uk
[2] Department of Information Systems and Computing, Brunel University
Uxbridge, Middlesex UB8 3PH, United Kingdom
shengxiang.yang@brunel.ac.uk

Abstract. Ant colony optimization (ACO) algorithms have proved to be able to adapt to dynamic optimization problems (DOPs) when they are enhanced to maintain diversity and transfer knowledge. Several approaches have been integrated with ACO to improve its performance for DOPs. Among these integrations, the ACO algorithm with immigrants schemes has shown good results on the dynamic travelling salesman problem. In this paper, we investigate ACO algorithms to solve a more realistic DOP, the dynamic vehicle routing problem (DVRP) with traffic factors. Random immigrants and elitism-based immigrants are applied to ACO algorithms, which are then investigated on different DVRP test cases. The results show that the proposed ACO algorithms achieve promising results, especially when elitism-based immigrants are used.

1 Introduction

In the vehicle routing problem (VRP), a number of vehicles with limited capacity are routed in order to satisfy the demand of all customers at a minimum cost (usually the total travel time). Ant colony optimization (ACO) algorithms have shown good performance for the VRP, where a population of ants cooperate and construct vehicle routes [5]. The cooperation mechanism of ants is achieved via their pheromone trails, where each ant deposits pheromone to its trails and the remaining ants can exploit it [2].

The dynamic VRP (DVRP) is closer to a real-world application since the traffic jams in the road system are considered. As a result, the travel time between customers may change depending on the time of the day. In dynamic optimization problems (DOPs) the moving optimum needs to be tracked over time. ACO algorithms can adapt to dynamic changes since they are inspired from nature, which is a continuous adaptation process [9]. In practice, they can adapt by transferring knowledge from past environments [1]. The challenge of such algorithms is how quickly they can react to dynamic changes in order to maintain the high quality of output instead of premature convergence.

Developing strategies for ACO algorithms to deal with premature convergence and address DOPs has attracted a lot of attention, which includes local and global restart strategies [7], memory-based approaches [6], pheromone manipulation schemes to maintain diversity [4], and immigrants schemes to increase diversity [11, 12]. These approaches have been applied to the dynamic travelling salesman problem (DTSP), which is the simplest case of a DVRP, i.e., only one vehicle is used. The ACO algorithms that are integrated with immigrants schemes have shown promising results on the DTSP where immigrant ants replace the worst ants in the population every iteration [11].

In this paper, we integrate two immigrants schemes, i.e., random immigrants and elitism-based immigrants, to ACO algorithms and apply them to the DVRP with traffic factor. The aim of random immigrants ACO (RIACO) is to increase the diversity in order to adapt well in DOPs, and the aim of elitism-based immigrants ACO (EIACO) is to generate guided diversity to avoid randomization.

The rest of the paper is organized as follows. Section 2 describes the problem we try to solve, i.e., the DVRP with traffic factors. Section 3 describes the ant colony system (ACS), which is one of the best performing algorithms for the VRP. Section 4 describes our proposed approaches where we incorporate immigrants schemes with ACO. Section 5 describes the experiments carried out by comparing RIACO and EIACO with ACS. Finally, Section 6 concludes this paper with directions for future work.

2 The DVRP with Traffic Jams

The VRP has become one of the most popular combinatorial optimization problems, due to its similarities with many real-world applications. The VRP is classified as *NP*-hard [10]. The basic VRP can be described as follows: a number of vehicles with a fixed capacity need to satisfy the demand of all the customers, starting from and returning to the depot.

Usually, the VRP is represented by a complete weighted graph $G = (V, E)$, with $n + 1$ nodes, where $V = \{u_0, \ldots, u_n\}$ is a set of vertices corresponding to the customers (or delivery points) u_i $(i = 1, \cdots, n)$ and the depot u_0 and $E = \{(u_i, u_j) : i \neq j\}$ is a set of edges. Each edge (u_i, u_j) is associated with a non-negative d_{ij} which represents the distance (or travel time) between u_i and u_j. For each customer u_i, a non-negative demand D_i is given. For the depot u_0, a zero demand is associated, i.e., $D_0 = 0$.

The aim of the VRP is to find the route (or a set of routes) with the lowest cost without violating the following constraints: (1) every customer is visited exactly once by only one vehicle; (2) every vehicle starts and finishes at the depot; and (3) the total demand of every vehicle route must not exceed the vehicle capacity Q. The number of routes identifies the corresponding number of vehicles used to generate one VRP solution, which is not fixed but chosen by the algorithm.

The VRP becomes more challenging if it is subject to a dynamic environment. There are many variations of the DVRP, such as the DVRP with dynamic

demand [14]. In this paper, we generate a DVRP with traffic factors, where each edge (u_i, u_j) is associated with a traffic factor t_{ij}. Therefore, the cost to travel from u_i to u_j is $c_{ij} = d_{ij} \times t_{ij}$. Furthermore, the cost to travel from u_j to u_i may differ due to different traffic factor. For example, one road may have more traffic in one direction and less traffic in the opposite direction.

Every f iterations a random number $R \in [F_L, F_U]$ is generated to represent potential traffic jams, where F_L and F_U are the lower and upper bounds of the traffic factor, respectively. Each edge has a probability m to have a traffic factor, by generating a different R to represent high and low traffic jams on different roads, i.e., $t_{ij} = 1 + R$, where the traffic factor of the remaining edges is set to 1 (indicates no traffic). Note that f and m represent the frequency and magnitude of changes in the DVRP, respectively.

3 ACO for the DVRP

The ACO metaheuristic consists of a population of μ ants where they construct solutions and share their information with the others via their pheromone trails. The first ACO algorithm developed is the Ant System (AS) [2]. Many variations of the AS have been developed over the years and applied to difficult optimization problems [3].

The best performing ACO algorithm for the DVRP is the ACS [13]. There is a multi-colony variation of this algorithm applied to the VRP with time windows [5]. However, in this paper we consider the single colony which has been applied to the DVRP [13]. Initially, all the ants are placed on the depot and all pheromone trails are initialized with an equal amount. With a probability $1-q_0$, where $0 \leq q_0 \leq 1$ is a parameter of the *pseudo-random* proportional decision rule (usually 0.9 for ACS), an ant k chooses the next customer j from customer i, as follows:

$$p_{ij}^k = \begin{cases} \frac{[\tau_{ij}]^\alpha [\eta_{ij}]^\beta}{\sum_{l \in N_i^k} [\tau_{il}]^\alpha [\eta_{il}]^\beta}, \text{if } j \in N_i^k, \\ 0, \qquad\qquad \text{otherwise}, \end{cases} \tag{1}$$

where τ_{ij} is the existing pheromone trail between customers i and j, η_{ij} is the heuristic information available a priori, which is defined as $1/c_{ij}$, where c_{ij} is the distance travelled (as calculated in Section 2) between customers i and j, N_i^k denotes the neighbourhood of unvisited customers of ant k when its current customer is i, and α and β are the two parameters that determine the relative influence of pheromone trail and heuristic information, respectively. With the probability q_0, the ant k chooses the next customer with the maximum probability, i.e., $[\tau]^\alpha[\eta]^\beta$, and not probabilistically as in Eq. (1). However, if the choice of the next customer leads to an infeasible solution, i.e., exceed the maximum capacity Q of the vehicle, the depot is chosen and a new vehicle route starts.

When all ants construct their solutions, the best ant retraces the solution and deposits pheromone globally according to its solution quality on the corresponding trails, as follows:

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij} + \rho \Delta \tau_{ij}^{best}, \forall (i,j) \in T^{best}, \tag{2}$$

where $0 < \rho \leq 1$ is the pheromone evaporation rate and $\Delta\tau_{ij}^{best} = 1/C^{best}$, where C^{best} is the total cost of the T^{best} tour. Moreover, a local pheromone update is performed every time an ant chooses another customer j from customer i as follows:

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij} + \rho\tau_0, \tag{3}$$

where ρ is defined as in Eq. (2) and τ_0 is the initial pheromone value.

The pheromone evaporation is the mechanism that eliminates the areas with high intensity of pheromones that are generate by ants, due to stagnation behaviour[3], in order to adapt well to the new environment. The recovery time depends on the size of the problem and magnitude of change.

4 ACO with Immigrants Schemes for the DVRP

4.1 Framework

The framework of the proposed algorithms is based on the ACO algorithms that were used for the DTSP [11, 12]. It will be interesting to observe if the framework based on immigrants schemes is beneficial for more realistic problems, such as the DVRP with traffic factors, as described in Section 2.

The initial phase of the algorithm and the solution construction of the ants are the same with the ACS; see Eq. (1). The difference of the proposed framework is that it uses a short-term memory every iteration t, denoted as $k_{short}(t)$, of limited size, i.e., K_s, which is associated with the pheromone matrix. Initially, $k_{short}(0)$ is empty where at the end of the iteration the K_s best ants will be added to $k_{short}(t)$. Each ant k that enters $k_{short}(t)$ deposits a constant amount of pheromone to the corresponding trails, as follows:

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

where $\Delta\tau_{ij}^{k} = (\tau_{max} - \tau_0)/K_s$ and T^k is the tour of ant k. Here, τ_{max} and τ_0 are the maximum and initial pheromone value, respectively.

Every iteration the ants from $k_{short}(t-1)$ are replaced with the K_s best ants from iteration t, a negative update is performed to their pheromone trails, as follows:

$$\tau_{ij} \leftarrow \tau_{ij} - \Delta\tau_{ij}^{k}, \forall\ (i,j) \in T^k, \tag{5}$$

where $\Delta\tau_{ij}$ and T^k are defined as in Eq. (4). This is because no ants can survive in more than one iteration because of the dynamic environment.

In addition, immigrant ants replace the worst ants in $k_{short}(t)$ every iteration and further adjustments are performed to the pheromone trails since $k_{short}(t)$ changes. The main concern when dealing with immigrants schemes is how to generate immigrant ants, that represent feasible solutions.

[3] A term used when all ants follow the same path and construct the same solution

4.2 Random Immigrants ACO (RIACO)

Traditionally, the immigrants are randomly generated and replace other ants in the population to increase the diversity. A random immigrant ant for the DVRP is generated as follows. First, the depot is added as the starting point; then, an unvisited customer is randomly selected as the next point. This process is repeated until the first segment (starting from the most recent visit to the depot) of customers do not violate the capacity constraint. When the capacity constraint is violated the depot is added and another segment of customers starts. When all customers are visited the solution will represent one feasible VRP solution.

Considering the proposed framework described above, before the pheromone trails are updated, a set S_{ri} of $r \times K_s$ immigrants are generated to replace the worst ants in $k_{short}(t)$, where r is the replacement rate.

RIACO has been found to perform better in fast and significantly changing environments for the DTSP [11]. This is because when the changing environments are not similar it is better to randomly increase the diversity instead of knowledge transfer. Moreover, when the environmental changes are fast the time is not enough to gain useful knowledge in order to transfer it. However, there is a high risk of randomization with RIACO that may disturb the optimization process. A similar behaviour is expected for the DVRP.

4.3 Elitism-based Immigrants ACO (EIACO)

Differently from RIACO, which generates diversity randomly with the immigrants, EIACO generates guided diversity by the knowledge transferred from the best ant of the previous environment. An elitism-based immigrant ant for the DVRP is generated as follows. The best ant of the previous environment is selected in order to use it as the base to generate elitism-based immigrants. The depots of the best ant are removed and adaptive inversion is performed based on the inver-over operator [8]. When the inversion operator finishes, the depots are added so that the capacity constraint is satisfied in order to represent one feasible VRP solution.

Considering the proposed framework above, on iteration t, the elite ant from $k_{short}(t-1)$ is used as the base to generate a set S_{ei} of $r \times K_s$ immigrants, where r is the replacement rate. The elitism-based immigrants replace the worst ants in $k_{short}(t)$ before the pheromone trails are updated.

The EIACO has been found to perform better in slowly and slightly changing environments for the DTSP [11]. This is because the knowledge transferred when the changing environments are similar will be more useful. However, there is a risk to transfer too much knowledge and start the optimization process from a local optimum and get stuck there. A similar behaviour is expected for the DVRP.

5 Simulation Experiments

5.1 Experimental Setup

In the experiments, we compare the proposed RIACO and EIACO with the existing ACS, described in Section 3. All the algorithms have been applied to the `vrp45`, `vrp72`, and `vrp135` problem instances[4].

To achieve a good balance between exploration and exploitation, most of the parameters have been obtained from our preliminary experiments where others have been inspired from literature [11]. For all algorithms, $\mu = 50$ ants are used, $\alpha = 1$, $\beta = 5$, and $\tau_0 = 1/n$. For ACS, $q_0 = 0.9$, and $\rho = 0.7$. Note that a lower evaporation rate has been used for ACS, i.e. $\rho = 0.1$, with similar or worse results. For the proposed algorithms, $q_0 = 0.0$, $K_s = 10$, $\tau_{max} = 1.0$ and $r = 0.4$.

For each algorithm on a DVRP instance, $N = 30$ independent runs were executed on the same environmental changes. The algorithms were executed for $G = 1000$ iterations and the overall offline performance is calculated as follows:

$$\bar{P}_{offline} = \frac{1}{G}\sum_{i=1}^{G}\left(\frac{1}{N}\sum_{j=1}^{N}P_{ij}^{*}\right) \tag{6}$$

where P_{ij}^{*} defines the tour cost of the best ant since the last dynamic change of iteration i of run j [9].

The value of f was set to 10 and 100, which indicate fast and slowly changing environments, respectively. The value of m was set to 0.1, 0.25, 0.5, and 0.75, which indicate the degree of environmental changes from small, to medium, to large, respectively. The bounds of the traffic factor are set as $F_L = 0$ and $F_U = 5$. As a result, eight dynamic environments, i.e., 2 values of $f \times 4$ values of m, were generated from each stationary VRP instance, as described in Section 2, to systematically analyze the adaptation and searching capability of each algorithm on the DVRP.

5.2 Experimental Results and Analysis

The experimental results regarding the offline performance of the algorithms are presented in Table 1 and the corresponding statistical results of Wilcoxon rank-sum test, at the 0.05 level of significance are presented in Table 2. Moreover, to better understand the dynamic behaviour of the algorithms, the results of the largest problem instance, i.e., `vrp135`, are plotted in Fig. 1 with $f = 10$, $m = 0.1$ and $m = 0.75$, and $f = 100$, $m = 0.1$ and $m = 0.75$, for the first 500 iterations. From the experimental results, several observations can be made by comparing the behaviour of the algorithms.

First, RIACO outperforms ACS in all the dynamic test cases; see the results of RIACO $\Leftrightarrow$ ACS in Table 2. This validates our expectation that ACS need

[4] Taken from the Fisher benchmark instances available at http://neo.lcc.uma.es/radi-aeb/WebVRP/

Table 1. Comparison of algorithms regarding the results of the offline performance

	$f = 10$				$f = 100$			
$m \Rightarrow$	0.1	0.25	0.5	0.75	0.1	0.25	0.5	0.75
Alg. & Inst.	vrp45							
ACS	897.5	972.5	1205.6	1648.0	883.4	929.1	1120.2	1536.9
RIACO	841.2	902.4	1089.5	1482.9	834.9	867.5	1016.1	1375.1
EIACO	840.1	899.8	1083.8	1473.5	839.8	860.6	1009.1	1355.5
Alg. & Inst.	vrp72							
ACS	305.3	338.6	426.2	596.2	297.3	324.6	412.7	547.9
RIACO	294.4	322.8	401.7	562.5	280.6	303.5	375.2	489.6
EIACO	289.9	319.4	397.8	557.0	276.2	298.5	366.7	476.5
Alg. & Inst.	vrp135							
ACS	1427.7	1567.3	1967.4	2745.7	1383.7	1519.4	1820.5	2536.2
RIACO	1417.8	1554.2	1922.1	2676.0	1353.1	1457.2	1698.6	2358.4
EIACO	1401.3	1542.1	1907.6	2663.1	1329.1	1444.3	1668.5	2293.8

Table 2. Statistical tests of comparing algorithms regarding the offline performance, where "+" or "−" means that the first algorithm is significantly better or the second algorithm is significantly better

Alg. & Inst.	vrp45				vrp72				vrp135			
$f = 10, m \Rightarrow$	0.1	0.25	0.5	0.75	0.1	0.25	0.5	0.75	0.1	0.25	0.5	0.75
RIACO ⇔ ACS	+	+	+	+	+	+	+	+	+	+	+	+
EIACO ⇔ ACS	+	+	+	+	+	+	+	+	+	+	+	+
EIACO ⇔ RIACO	+	+	+	+	+	+	+	+	+	+	+	+
$f = 100, m \Rightarrow$	0.1	0.25	0.5	0.75	0.1	0.25	0.5	0.75	0.1	0.25	0.5	0.75
RIACO ⇔ ACS	+	+	+	+	+	+	+	+	+	+	+	+
EIACO ⇔ ACS	+	+	+	+	+	+	+	+	+	+	+	+
EIACO ⇔ RIACO	−	+	+	+	+	+	+	+	+	+	+	+

sufficient time to recover when a dynamic change occurs, which can be also observed from Fig. 1 in the environmental case with $f = 100$. This is because the pheromone evaporation is the only mechanism used to eliminate pheromone trails that are not useful to the new environment, and may bias the population to areas that are not near the new optimum. On the other hand, RIACO uses the proposed framework where the pheromone trails exist only in one iteration.

Second, EIACO outperforms ACS in all the dynamic test cases as the RIACO; see the results EIACO ⇔ ACS in Table 2. This is due to the same reasons RIACO outperforms the traditional ACS. However, EIACO outperforms RIACO in almost all dynamic test cases; see the results of EIACO ⇔ RIACO in Table 2. In slowly and slightly changing environments EIACO has sufficient time to gain knowledge from the previous environment, and the knowledge transferred has more chances to help when the changing environments are similar. However, on the smallest problem instance, i.e., vrp45, with $f = 100$ and $m = 0.1$ RIACO performs better than EIACO. This validates our expectation where too much

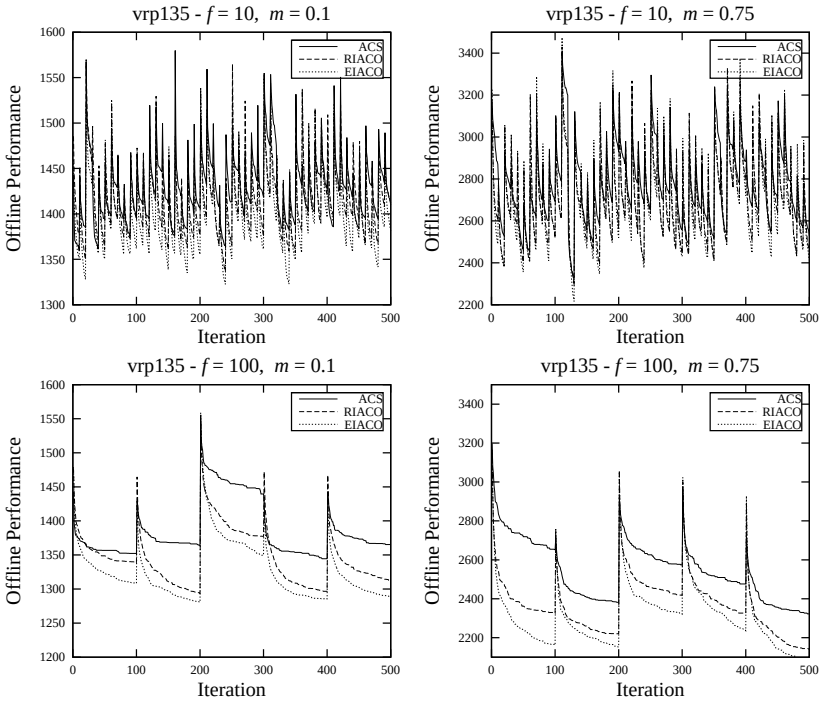

Fig. 1. Offline performance of algorithms for different dynamic test problems

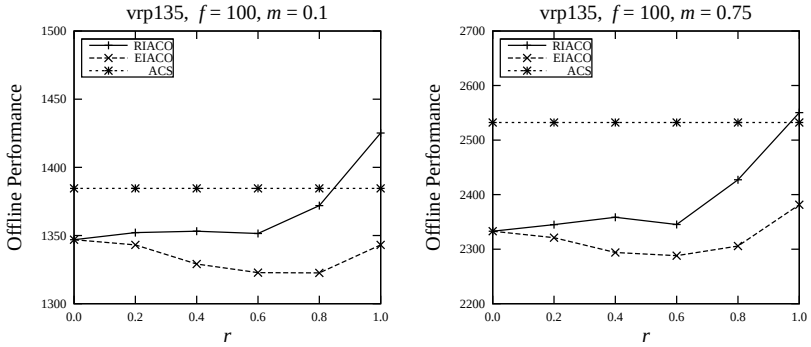

Fig. 2. Offline performance of RIACO and EIACO with different replacement rates against the performance of ACS in slowly changing environments

knowledge transferred does not always mean better results in dynamic environments. On the other hand RIACO, was expected to perform better than EIACO in fast and significantly changing environments, since the random immigrants only increase the diversity, but that it is not the case. This may be possibly

because of too much randomization that may disturb the optimization process and requires further investigation regarding the effect of the immigrant ants.

Third, in order to investigate the effectiveness of the immigrants schemes, further experiments have been performed on the same problem instances with the same parameters used before but with different immigrant replacement rates, i.e., $r \in \{0.0, 0.2, 0.4, 0.6, 0.8, 1.0\}$. In Fig. 2 the offline performance of RIACO and EIACO with the varying replacement rates are presented[5], against the ACS performance, where $r = 0.0$ means that no immigrants are generated to replace ants in the $k_{short}(t)$. The results confirm our expectation above, where the random immigrants in RIACO sometimes may disturb the optimization and degrade the performance. On the other hand, elitism-based immigrants in EIACO improve the performance, especially in slightly changing environments.

Finally, the proposed framework performs better than ACS, even if no immigrants are generated; see Fig. 2. The RIACO with $r = 1.0$ performs worse than the ACS, whereas the EIACO with $r = 1.0$ better than ACS. This is because RIACO destroys all the knowledge transferred to the $k_{short}(t)$ from the ants of the previous iteration with random immigrants, whereas EIACO destroys that knowledge but transfers new knowledge using the best ant from the previous iteration.

6 Conclusions

Different immigrants schemes have been successfully applied to evolutionary algorithms and ACO algorithms to address different DOPs [11, 16]. ACO-based algorithms with immigrants, i.e., RIACO and EIACO, have shown good performance on different variations of the DTSP [11, 12]. In this paper, we modify and apply such algorithms to address the DVRP with traffic factors, which is closer to a real-world application. The immigrant ants are generated either randomly or using the previous best ant as the base and replace the worst ones in the population. The aim is to maintain the diversity of solutions and transfer knowledge from previous environments in order to adapt well in DOPs.

Comparing RIACO and EIACO with ACS, one of the best performing ACO algorithms for VRP, on different test cases of DVRPs, the following concluding remarks can be drawn. First, the proposed framework used to integrate ACO with immigrants schemes, performs better than the traditional framework, even when immigrant ants are not generated. Second, EIACO is significantly better than RIACO and ACS in almost all dynamic test cases. Third, RIACO is significantly better than ACS in all dynamic test cases. Finally, the random immigrants may disturb the optimization process with a result to degrade the performance, whereas elitism-based immigrants transfers knowledge with a result to improves the performance for the DVRP with traffic factor.

[5] The experimental results of the remaining problem instances and dynamic test cases are similar for EIACO, whereas for RIACO there is an improvement when $r > 0.0$ on the smallest problem instance

An obvious direction for future work is to hybridize the two immigrants schemes. However, from our preliminary results the performance of the hybrid scheme is better than RIACO but worse than EIACO in all dynamic test cases. Therefore, to find another way to achieve a good balance between the knowledge transferred and the diversity generated would be interesting for future work. Another future work is to integrate memory-based immigrants with ACO, which have also performed well on the DTSP [12], to the DVRP with traffic factors.

References

1. Bonabeau, E., Dorigo, M., Theraulaz, G.: Swarm Intelligence: From Natural to Artificial Systems. Oxford University Press, New York (1999)
2. Dorigo, M., Maniezzo, V., Colorni, A.: Ant system: optimization by a colony of cooperating agents. IEEE Trans. on Syst., Man and Cybern., Part B: Cybern. 26(1), 29–41 (1996)
3. Dorigo, M., Stützle, T.: Ant Colony Optimization. The MIT Press, London (2004)
4. Eyckelhof, C. J., Snoek, M.: Ant Systems for a Dynamic TSP. In: ANTS 2002: Proc. of the 3rd Int. Workshop on Ant Algorithms, pp. 88–99 (2002)
5. Gambardella, L.M., Taillard, E., Agazzi, G.: MACS-VRPTW: A multiple ant colony system for vehicle routing problems with time windows. In Corne D., et al. (eds.), New Ideas in Optimization, pp. 63–76 (1999)
6. Guntsch, M., Middendorf, M.: Applying population based ACO to dynamic optimization problems. In: Proc. of the 3rd Int. Workshop on Ant Algorithms, LNCS, vol. 2463, pp. 111–122 (2002)
7. Guntsch, M., Middendorf, M.: Pheromone modification strategies for ant algorithms applied to dynamic TSP. In: EvoApplications 2001: Appl. of Evol. Comput., LNCS, vol. 2037, pp. 213-222 (2001)
8. Guo, T., Michalewicz, Z.: Inver-over operator for the TSP. In: Proc. of the 5th Int. Conf. on Parallel Problem Solving from Nature, pp. 803–812 (1998)
9. Jin, Y., Branke, J.: Evolutionary optimization in uncertain environments - a survey. IEEE Trans. on Evol. Comput. 9(3), 303–317 (2005)
10. Labbe, M., Laporte, G., Mercure, H.: Capacitated vehicle routing on trees. Operations Research 39(4), 61–622 (1991)
11. Mavrovouniotis, M., Yang, S.: Ant colony optimization with immigrants schemes for dynamic environments. In: Proc. of the 11th Int. Conf. on Parallel Problem Solving from Nature, LNCS, vol. 6238, pp. 371–380 (2010)
12. Mavrovouniotis, M., Yang, S.: Memory-based immigrants for ant colony optimization in changing environments. In: EvoApplications 2011: Appl. of Evol. Comput., LNCS, vol. 6624, pp. 324–333 (2011)
13. Montemanni, R., Gambardella, L., Rizzoli, A., Donati, A.: Ant colony system for a dynamic vehicle routing problem. Journal of Combinatorial Optimization 10(4), 327–343 (2005)
14. Psaraftis, H.: Dynamic vehicle routing: status and prospects. Annals of Operations Research 61, 143–164 (1995)
15. Rizzoli, A.E., Montemanni, R., Lucibello, E., Gambardella, L. M.: Ant colony optimization for real-world vehicle routing problems - from theory to applications. Swarm Intelli. 1(2), 135–151 (2007)
16. Yang, S.: Genetic algorithms with memory and elitism based immigrants in dynamic environments. Evol. Comput. 16(3), 385–416 (2008)