ACO-iRBA: A Hybrid Approach to TSPN with Overlapping Neighborhoods

Yuanlong Qin and Bo Yuan(✉)

Intelligent Computing Lab, Division of Informatics,
Graduate School at Shenzhen, Tsinghua University,
Shenzhen 518055, People's Republic of China
956441594@qq.com, yuanb@sz.tsinghua.edu.cn

Abstract. The traveling salesman problem with neighborhoods (TSPN) is a generalization of TSP and can be regarded as a combination of TSP and TPP (Touring Polygons Problem). In this paper, we propose a hybrid TSPN solution named ACO-iRBA in which the TSP and TPP tasks are tackled simultaneously by ACO (Ant Colony Optimization) and iRBA, an improved version of RBA (Rubber Band Algorithm), respectively. A major feature of ACO-iRBA is that it can properly handle situations where the neighborhoods are heavily overlapped. Experiment results on benchmark problems composed of random ellipses show that ACO-iRBA can solve TSPN instances with up to 70 regions effectively and generally produce higher quality solutions than a recent heuristic method CIH.

Keywords: TSP · TPP · TSPN · Hybrid · iRBA

1 Introduction

TSP with neighborhoods (TSPN), introduced by Arkin and Hassin [1], is a generalization of TSP. The scenario of TSPN can be explained as follows: a salesman wants to meet a group of potential buyers; each buyer specifies a connected region in the plane (neighborhood) within which he/she is willing to meet the salesman; the salesman needs to schedule a tour with the shortest length that visits all buyers and finally returns to the initial departure point [2, 3].

Since TSPN is a generalization of TSP, TSPN is also an NP-hard problem. Traditionally, there are mainly two ways to solve TSPN problems. The first one is to convert TSPN to group TSP (GTSP) by sampling some points in each region and replace neighborhoods with the corresponding points [4–6]. However, it has two drawbacks: (i) the search space can increase significantly if too many points are sampled from each region; (ii) using GTSP to approximate TSPN will inevitably bring in errors and may not obtain satisfactory results.

Another approach relies on the divide-and-conquer strategy. For example, each region can be reduced to a single point as the representative and TSPN is transformed into two sub-problems: TSP and touring polygons problem (TPP) [7]. In practice, a TSP solver is applied on the set of representatives to acquire a visiting sequence. Then, this sequence of regions is used as the input of the TPP solver, which searches for an optimal visiting point for each region. However, it is usually not known how to choose

Y. Shi et al. (Eds.): SEAL 2017, LNCS 10593, pp. 87–96, 2017.
https://doi.org/10.1007/978-3-319-68759-9_8

the proper representatives and consequently the sequence obtained by the TSP solver may not be a good sequence for TPP/TSPN, leading to sub-optimal solutions.

Recently, Gentilini et al. [8] formulated TSPN as a non-convex mixed-integer nonlinear programming (MINLP) problem, which is efficient only for TSPN with a small number of regions. Alatartsev et al. [9] proposed a heuristic TSPN method named Constricting Insertion Heuristic (CIH), which splits TSPN into TSP and TPP and solves them simultaneously. Their experiments show that high quality solutions can be obtained within a short period of time for problems with more than 100 regions.

As to TPP, the rubber band algorithm (RBA) by Bülow and Klette [10] is aimed at finding minimum-length polygonal curves in cube-curves in 3D spaces. Variations of this algorithm have been used to solve various Euclidean shortest path (ESP) problems, such as touring polygons, parts cutting, safari and the watchman route.

In this paper, we first modify RBA to make it suitable for cases with overlapping regions and then propose a hybrid algorithm ACO-iRBA to solve TSPN with overlapping neighborhoods effectively. The proposed algorithm splits TSPN into TSP and TPP, which are solved in parallel rather than step by step. Section 2 introduces the related techniques and the details of ACO-iRBA are presented in Sect. 3. The experiments investigating the effectiveness of iRBA compared with RBA and the performance of ACO-iRBA compared with CIH on 18 TSPN benchmark instances are given in Sect. 4. This paper is concluded in Sect. 5 with some directions for future work.

2 Related Work

2.1 RBA

Algorithm 1 RBA for a sequence of pairwise disjoint simple polygons

Input: Sequence of n pairwise disjoint areas $A = (A_1, \ldots, A_n)$, accuracy ε
Output: Tour $T = (p_1, \ldots, p_n)$

1. Construct a feasible sequence $T = (p_1, \ldots, p_n)$ so that $p_i \in A_i$
2. Let $L_0 = \infty$. Calculate $L_1 = \sum_{i=1}^{n} d(p_i, p_{i+1})$ where $p_{n+1} = p_1$.
3. **while** $L_0 - L_1 \geq \varepsilon$ **do**
4. **for** $i = 1,2,\cdots,n$ **do**
5. Find $p_i' \in \partial A_i$, such that $d(p_{i-1}, p_i')+d(p_i', p_{i+1})= min\ (d(p_{i-1}, p_i) + d(p_i, p_{i+1}))$, $p_i \in \partial A_i$
6. $p_i \leftarrow p_i'$
7. **end**
8. Let $L_0 = L_1$ and calculate $L_1 = \sum_{i=1}^{n} d(p_i, p_{i+1})$
9. **end**
10. **return** T;

The basic version of TPP is to find a shortest path, which starts at p and then visits the polygons according to the given order, and finally ends at q (p, q are two points that

do not belong to any polygons). There are many algorithms for solving TPP or its variants [11–14] and it is a NP-hard problem except in some special cases [11].

Pan et al. [12] proposed an effective TPP algorithm based on RBA. RBA features linear time complexity and can handle large scale problems efficiently. The general structure of RBA is shown as Algorithm 1 where ∂A_i is the frontier of polygon A_i and $d(p_i, p_{i+1})$ is the distance between p_i and p_{i+1}.

The basic idea of RBA is to construct a feasible tour $T = (p_1, \ldots, p_n)$ by randomly allocating one point inside each region with $p_i \in A_i$ and iteratively improving it. In each iteration, in order to find a better point $p_i' \in \partial A_i$, the adjacent points p_{i-1} and p_{i+1} of p_i are fixed, so that the distance $d(p_{i-1}, p_i') + d(p_i', p_{i+1})$ can be minimized (Fig. 1). For example, if the region is a disk, we can find p_i' by performing a binary search on the radian values within $[0, 2\pi]$. RBA stops when a maximum number of iterations are performed or the desired accuracy threshold ε is reached.

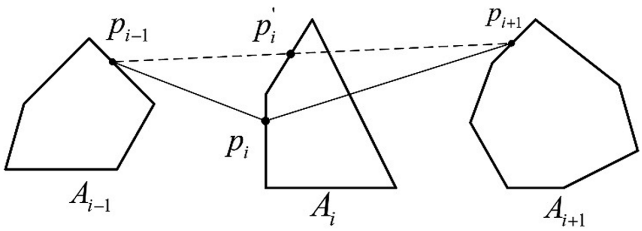

Fig. 1. An illustration of Step 5 in Algorithm 1 where point p_i is replaced by a new point p_i'

2.2 TSP-TPP

The TSP-TPP strategy solves TSPN in two steps: (i) construct a TSP Tour based on representative points $(p_1, \ldots, p_n)$; (ii) search for a set of n meeting/visiting points based on the sequence produced in the previous step. Suppose π^{TSP} is the permutation of representative points in the optimal TSP tour, whereas π^{TSPN} is the permutation of regions in the optimal TSPN tour. The key idea is to use π^{TSP} in place of the unknown π^{TSPN} in the search process for the optimal meeting points.

The validity of the TSP-TPP strategy is established on the assumption that when the sizes of regions are small enough with respect to the distances among them, π^{TSP} is likely to be identical to π^{TSPN}. Yuan et al. [7] investigated TSPN with pairwise disjoint disks and the results show that the TSP-TPP strategy works well in practice. However, Fig. 2 gives a counterexample with $\pi^{TSP} = (1, 2, 3, 4, 5)$ and $\pi^{TSPN} = (1, 3, 2, 4, 5)$, which is more likely to happen when regions are close to each other or even overlapped.

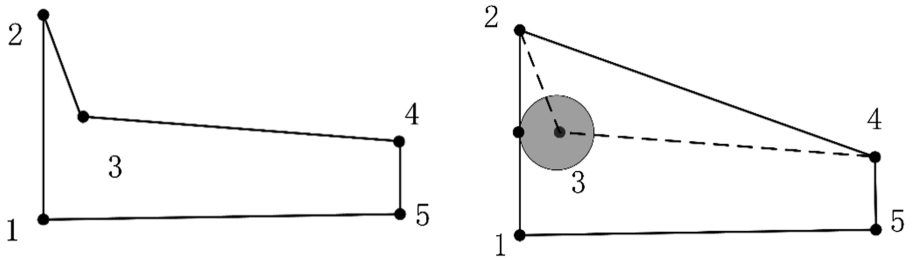

Fig. 2. A case where π^{TSP}(left) is different from π^{TSPN}(right) when point 3 is replaced by a disk. The TSPN tour (right) is also shorter than the TSP tour (left).

3 Methodology

3.1 iRBA

Algorithm 2 iRBA for a sequence of simple polygons

Input: Sequence of n areas $\mathrm{A} = (A_1, \ldots, A_n)$, accuracy ε
Output: Tour $T = (p_1, \ldots, p_n)$
1. Construct a feasible sequence $T = (p_1, \ldots, p_n)$ so that $p_i \in A_i$
2. Let $L_0 = \infty$. Calculate $L_1 = \sum_{i=1}^{n} d(p_i, p_{i+1})$ where $p_{n+1} = p_1$.
3. **while** $L_0 - L_1 \geq \varepsilon$ **do**
4. **for** $i = 1,2,\cdots,n$ **do**
5. **if** $is\_in(A_i, p_{i-1})$ && $is\_in(A_i, p_{i+1})$==True
6. $p_i' = (p_{i-1} + p_{i+1})/2$
7. **else**
8. Find $p_i' \in \partial A_i$, such that $d(p_{i-1}, p_i')+d(p_i', p_{i+1})=$ $min\ (d(p_{i-1}, p_i) + d(p_i, p_{i+1}))$, $p_i \in \partial A_i$
9. **end**
10. $p_i \leftarrow p_i'$
11. **end**
12. Let $L_0 = L_1$ and calculate $L_1 = \sum_{i=1}^{n} d(p_i, p_{i+1})$
13. **end**
14. **return** T;

[#]The function $is\_in(A_i, p_j)$ is to judge whether point p_j is within region A_i.

RBA works well for TPP in non-overlapping cases. However, the regions in TSPN may be significantly overlapped, resulting in unsatisfactory performance. In Fig. 3, p_{i-1} and p_{i+1}, the two adjacent points of p_i, are both located within region A_i. Since RBA only checks the candidate points on the frontier of region A_i, it cannot correctly identify the optimal point located inside A_i.

So, we propose iRBA by modifying the original RBA to solve the above issue. iRBA works by checking whether the two adjacent points of p_i are both inside region A_i before searching for the new point p_i'. If so, iRBA simply takes the midpoint between

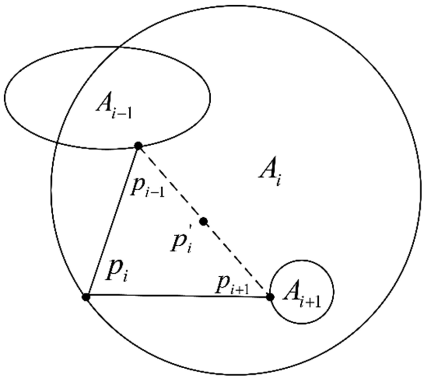

Fig. 3. A case where the optimal point p_i' is within A_i rather than on the frontier of A_i

p_{i-1} and p_{i+1} as p_i', that is, $p_i' = (p_{i-1} + p_{i+1})/2$. The implementation details of iRBA are shown in Algorithm 2.

3.2 ACO-iRBA

Although it is tempting to identify the sequence of regions by solving a TSP problem in the first place and fix it during the subsequent TPP phase, there is no guarantee that this TSP sequence is optimal for the corresponding TSPN, as shown in Sect. 2.2.

In this paper, we propose a hybrid algorithm that combines evolutionary algorithms (EAs) for the TSP phase and dedicated heuristic methods for the TPP phase. The main idea is to evolve a population of individuals representing candidate sequences for TSPN and apply a TPP solver on each individual. The tour length obtained by the TPP solver is returned as the fitness value of the specific individual, which is the key information that drives the evolution. In this way, the TSP phase and the TPP phase are optimized simultaneously, making it possible to achieve better solutions.

There are many TSP algorithms in the domain of EAs [15–18]. We adopt ant colony optimization (ACO) largely due to two concerns:

(i) ACO is purposefully designed to solve path planning problems;
(ii) ACO is a metaheuristic algorithm with great flexibility and can be potentially applied to different problems.

The basic version of ACO [17] was used in our work, as our current focus is to demonstrate the feasibility of the proposed hybrid strategy while leaving in-depth refinement as future work.

In ACO, the migration probability (i.e., ant k moves from city i to city j) when constructing a route is defined as follows:

$$p_{ij}^k = \frac{\tau_{ij}^{\alpha}\eta_{ij}^{\beta}}{\sum_s \tau_{is}^{\alpha}\eta_{is}^{\beta}} \quad \text{s} \in allowed_k \tag{1}$$

where $allowed_k = C - tabu_k$ is the collection of alternative cities for ant k ($tabu_k$ represents the cities that ant k has visited and C is the collection of total cities). η_{ij} is a heuristic function, which is the inverse of the distance between cities $i$and j ($\eta_{ij} = 1/d_{ij}$). β is a parameter that weights the importance of heuristic information, and the larger β is, the more greedy the rule is. τ_{ij} is the amount of pheromone trail on edge e_{ij}, and α is a parameter that weights the importance of the accumulated pheromone: the larger its value, the tighter the cooperation among the population.

Once the travel of ants is finished, τ_{ij} is updated according to the following rules:

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

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

where $\rho(0 \le \rho \le 1)$ is the pheromone volatilization coefficient and $\Delta\tau_{ij}^{k}$ is the pheromone that ant k leaves on the edge e_{ij}. Dorigo and Gambardella [16] presented three rules of $\Delta\tau_{ij}^{k}$ among which the Ant Cycle System is widely used:

$$\Delta\tau_{ij}^{k} = \begin{cases} \frac{Q}{L_k}, & \text{if ant } k \text{ has passed } e_{ij} \text{ in this cycle} \\ 0, & else \end{cases} \tag{4}$$

where Q is a constant that can be simply set to 1 and L_k is the route length of ant k in this cycle. Note that we use the TPP length T_k returned by iRBA as L_k, which means that iRBA is applied to every sequence s_k to get T_k (i.e., $T_k = iRBA(s_k)$) with region centers selected as the initial points.

4 Experiments

The following experimental studies were conducted on a PC with Intel Core i7-3770 CPU at 3.4 GHz with 8 GB RAM and all algorithms were implemented in Matlab 2014a. In the first part of experiments, we compared iRBA with RBA on randomly generated synthetic problems with overlapping regions. In the second part of experiments, we compared ACO-iRBA with CIH on 18 benchmark problems.

4.1 Experiment on iRBA

Each problem consisted of a set of ellipses in a 4×4 rectangle with random radius values and locations. Therefore, the overlap rate is expected to grow with the number of ellipses. We selected the centers as initial points to get the best TSP sequence, and then used this sequence as the input for RBA and iRBA. The results are given in Fig. 4.

In Fig. 4, ρ is an index used to measure the relative improvement of iRBA: $\rho = (\text{RBA} - \text{iRBA})/\text{RBA}$. The greater ρ is, the better the relative performance of iRBA. The horizontal axis represents the number of regions in each problem, ranging from 5 to 25. To account for randomness, 20 instances were generated for each case and the average ρ values and error bars are plotted. It is clear that there is a positive

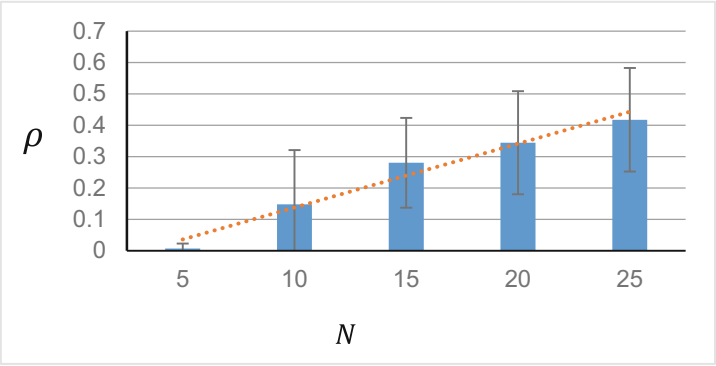

Fig. 4. The effectiveness of path reduction by iRBA compared to RBA

correlation between ρ and N, which means that the benefit of iRBA over RBA is more significant when regions are heavily overlapped. The results on one of the instances are shown in Fig. 5.

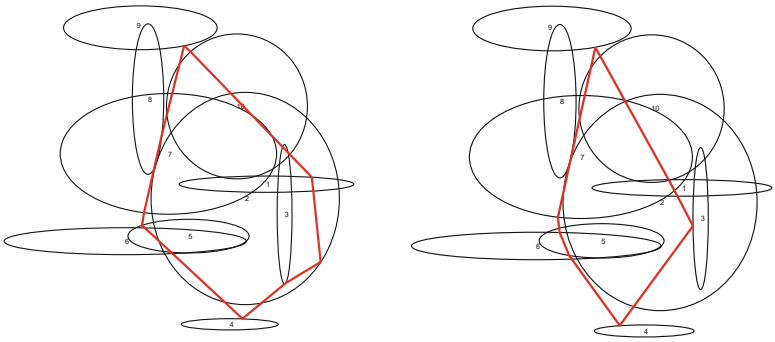

Fig. 5. The comparison of routes found for an instance ($N = 10$) by RBA (left, length = 5.05) and iRBA (right, length = 4.47)

4.2 Experiment on ACO-iRBA

In order to evaluate the performance of ACO-iRBA, we chose the 18 test instances developed by Gentilini et al. [8]. For example, the instance labeled "30_1_5" is an instance with 30 ellipses and the radius of one axis is stretched by one to five times, in comparison to another axis. Alatartsev et al. [9] gives a detailed description on how to construct the instances and the best known values of instances are also available [19].

In our experiment, the number of ants m was 200 and the number of iterations N was 100. Other parameter values were: $\alpha = 1$, $\beta = 5$, $Q = 1$, $\varepsilon = 0.2$. Figure 6 shows a TSPN tour for instance "30_1_10", and all results over 30 trials are given in Table 1.

In Table 1, the performance is measured by the relative distance to the best known result with optimal value 0. For example, 2% means that the length of the TSPN tour is

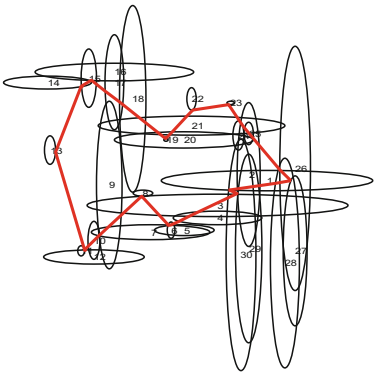

Fig. 6. The TSPN route for instance "30_1_10" with length 307.895

Table 1. The performance of ACO-iRBA on TSPN instances with 20 to 70 ellipses

No.	Instance	Best known value	CIH	ACO-iRBA		
			(%)	Best (%)	Avg. (%)	Std
1	20_1_1	318.904	2.39	**0.00**	0.02	0.338
2	20_1_5	312.915	3.30	**0.00**	0.01	0.607
3	20_1_10	252.350	0.00	0.17	2.57	1.420
4	30_1_1	383.578	1.44	**0.06**	0.14	0.864
5	30_1_5	316.854	0.00	1.47	2.47	2.384
6	30_1_10	306.637	0.00	0.34	0.74	1.017
7	40_1_1	416.556	3.58	**2.06**	2.76	1.327
8	40_1_5	366.637	0.53	1.08	1.89	3.474
9	40_1_10	311.714	0.00	8.44	9.30	4.391
10	50_1_1	438.215	3.10	**1.97**	2.50	2.242
11	50_1_5	435.158	6.97	**2.67**	3.40	0.943
12	50_1_10	391.303	2.44	6.71	8.65	4.147
13	60_1_1	559.042	8.87	**1.66**	1.89	2.431
14	60_1_5	550.121	2.93	**1.64**	2.25	4.379
15	60_1_10	482.289	7.85	**5.31**	5.84	1.540
16	70_1_1	599.819	5.74	**2.93**	3.91	2.224
17	70_1_5	564.303	7.60	**4.13**	5.58	7.580
18	70_1_10	447.452	9.28	**6.56**	9.05	5.788
Avg.			3.67	**2.57**	3.51	

2% longer than the best known shortest tour. On 12 out of the 18 test instances, the best tours found by ACO-iRBA over 30 trials were better than those found by CIH. On average, the best tours found by ACO-iRBA were only 2.57% longer than the best known shortest tours, compared to 3.67% of CIH.

Moreover, there is a clear tendency that when the number of ellipses was larger, the advantage of ACO-iRBA over CIH was also more consistent. Furthermore, Fig. 7 shows the convergence pattern of ACO-iRBA averaged over 30 trials on problems with different numbers of ellipses. The vertical axis shows the ratio between the best tour length at the 100^{th} iteration and that at the current iteration, showing that in most cases our algorithm features quick convergence speed.

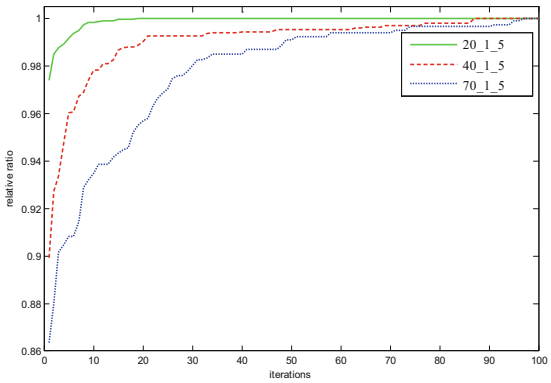

Fig. 7. The convergence of ACO-iRBA with different numbers of eclipses. The vertical axis shows the relative performance compared to the final value at the 100^{th} iteration.

5 Conclusion

This paper presented a hybrid algorithm ACO-iRBA to solve the challenging TSPN problems, which is purposefully targeted at cases with significantly overlapping neighborhoods. In order to solve TSPN, we divided TSPN into two sub-problems: TSP and TPP and solved them simultaneously, avoiding the potential drawbacks of solving them in separate stages. The proposed iRBA is an extension of RBA and suits for touring a sequence of overlapping regions. Note that, as the TSP solver, ACO can be conveniently replaced by other EA-based TSP algorithms that take the tour lengths produced by iRBA as inputs, bringing high flexibility to the proposed method. Experimental studies on 18 standard test instances confirmed that ACO-iRBA worked reasonably well in most cases compared to CIH, which is one of the latest heuristic methods for TSPN, especially when the space was densely populated by ellipses. As to future work, we will generalize our algorithm to situations where the regions are represented by irregular shapes, instead of circles or ellipses.

References

1. Arkin, E.M., Hassin, R.: Approximation algorithms for the geometric covering salesman problem. Discrete Appl. Math. **55**(3), 197–218 (1994)

2. Dumitrescu, A., Mitchell, J. S.: Approximation algorithms for TSP with neighborhoods in the plane. In: Proceedings of the Twelfth Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 38–46. Society for Industrial and Applied Mathematics (2001)
3. de Berg, M., Gudmundsson, J., Katz, M.J., Levcopoulos, C., Overmars, M.H., van der Stappen, A.F.: TSP with neighborhoods of varying size. J. Algorithms **57**(1), 22–36 (2005)
4. Jang, D.S., Chae, H.J., Choi, H.L.: Optimal control-based UAV path planning with dynamically-constrained TSP with neighborhoods. arXiv preprint arXiv:1612.06008 (2016)
5. Isaacs, J.T., Klein, D.J., Hespanha, J.P.: Algorithms for the traveling salesman problem with neighborhoods involving a Dubins vehicle. In: American Control Conference, pp. 1704–1709 (2011)
6. Wang, W., Shi, H.S., Wu, D.J., Huang, P.Y., Gao, B.J., Wu, F.P., Xu, D., Chen, X.J.: VD-PSO: an efficient mobile sink routing algorithm in wireless sensor networks. Peer-to-Peer Netw. Appl. **10**, 1–10 (2016)
7. Yuan, B., Orlowska, M., Sadiq, S.: On the optimal robot routing problem in wireless sensor networks. IEEE Trans. Knowl. Data Eng. **19**(9), 1252–1261 (2007)
8. Gentilini, I., Margot, F., Shimada, K.: The travelling salesman problem with neighbourhoods: MINLP solution. Optim. Methods Softw. **28**(2), 364–378 (2013)
9. Alatartsev, S., Augustine, M., Ortmeier, F.: Constricting insertion heuristic for traveling salesman problem with neighborhoods. In: ICAPS (2013)
10. Klette, R., Bülow, T.: Critical edges in simple cube-curves. In: 9th International Conference on Discrete Geometry for Computer Imagery, pp. 467–478 (2000)
11. Dror, M., Efrat, A., Lubiw, A., Mitchell, J.S.: Touring a sequence of polygons. In: Proceedings of the Thirty-Fifth Annual ACM Symposium on Theory of Computing, pp. 473–482 (2003)
12. Pan, X., Li, F., Klette, R.: Approximate shortest path algorithms for sequences of pairwise disjoint simple polygons. Department of Computer Science, University of Auckland, pp. 175–178 (2010)
13. Li, F., Klette, R.: Rubberband algorithms for solving various 2D or 3D shortest path problems. In: Computing: Theory and Applications, pp. 9–19. IEEE Press (2007)
14. Ahadi, A., Mozafari, A., Zarei, A.: Touring a sequence of disjoint polygons: complexity and extension. Theoret. Comput. Sci. **556**, 45–54 (2014)
15. Grefenstette, J., Gopal, R., Rosmaita, B., Van Gucht, D.: Genetic algorithms for the traveling salesman problem. In: Proceedings of the First International Conference on Genetic Algorithms and Their Applications, pp. 160–165 (1986)
16. Dorigo, M., Gambardella, L.M.: Ant colonies for the travelling salesman problem. Biosystems **43**(2), 73–81 (1997)
17. Wang, K.P., Huang, L., Zhou, C.G., Pang, W.: Particle swarm optimization for traveling salesman problem. In: 2003 International Conference on Machine Learning and Cybernetics, pp. 1583–1585 (2003)
18. Wong, L.P., Low, M.Y. H., Chong, C.S.: A bee colony optimization algorithm for traveling salesman problem. In: AICMS 2008 Second Asia International Conference on Modeling and Simulation, pp. 818–823 (2008)
19. Alatartsev, S., Mersheeva, V., Augustine, M., Ortmeier, F.: On optimizing a sequence of robotic tasks. In: 2013 IEEE/RSJ International Conference Intelligent Robots and Systems (IROS), pp. 217–223 (2013)