Towards Solving TSPN with Arbitrary Neighborhoods: A Hybrid Solution

Bo Yuan[1,2(✉)] and Tiantian Zhang[1,2]

[1] Intelligent Computing Lab, Division of Informatics,
Graduate School at Shenzhen, Tsinghua University,
Shenzhen 518055, People's Republic of China
yuanb@sz.tsinghua.edu.cn, 2573546543@qq.com
[2] Shenzhen Engineering Laboratory of Geometry Measurement Technology,
Graduate School at Shenzhen, Tsinghua University,
Shenzhen 518055, People's Republic of China

Abstract. As the generalization of TSP (Travelling Salesman Problem), TSPN (TSP with Neighborhoods) is closely related to several important real-world applications. However, TSPN is significantly more challenging than TSP as it is inherently a mixed optimization task containing both combinatorial and continuous components. Different from previous studies where TSPN is either tackled by approximation algorithms or formulated as a mixed integer problem, we present a hybrid framework in which metaheuristics and classical TSP solvers are combined strategically to produce high quality solutions for TSPN with arbitrary neighborhoods. The most distinctive feature of our solution is that it imposes no explicit restriction on the shape and size of neighborhoods, while many existing TSPN solutions require the neighborhoods to be disks or ellipses. Furthermore, various continuous optimization algorithms and TSP solvers can be conveniently adopted as necessary. Experiment results show that, using two off-the-shelf routines and without any specific performance tuning efforts, our method can efficiently solve TSPN instances with up to 25 regions, which are represented by both convex and concave random polygons.

Keywords: TSP · TSPN · Neighborhood · Hybrid · Metaheuristic

1 Introduction

TSP (Travelling Salesman Problem) is a well-known combinatorial optimization problem, which has been extensively studied in the past decades [1]. Given a set of n cities and their locations, an optimal (shortest) cyclic tour is required that visits each city once and only once. Although it is possible to work out the optimal solution via brute force search for small n values, TSP is an NP-hard problem and the size of the search space (possible permutations of cities) grows quickly as n increases, making exact methods computationally prohibitive. Since TSP has found wide applications in robot motion planning, logistics and manufacturing, there are already a number of techniques that can effectively tackle TSP instances with hundreds of cities using approximation, heuristic or metaheuristic algorithms [2–4]. Furthermore, the classical

M. Wagner et al. (Eds.): ACALCI 2017, LNAI 10142, pp. 204–215, 2017.
DOI: 10.1007/978-3-319-51691-2_18

TSP can be also extended to non-Euclidean spaces as well as a variety of interesting problems, such as Asymmetric TSP and Generalized TSP (One-of-a-Set TSP) [5].

TSPN (TSP with Neighborhoods) is an extension of TSP in which each city is represented by a continuous region called neighborhood and the optimal solution is the shortest path that visits/connects all regions [6]. It is easy to see that when the size of the region reduces to zero, TSPN is identical to TSP. There are several scenarios in practice that can be formulated as TSPN. For example, given a set of geographically distributed wireless sensors, it may be necessary to use a mobile robot to collect the data from each sensor. Since each sensor has an effective communication range, typically represented by a disk, the mobile robot can download the data once it reaches the boundary of the disk, instead of the exact location of the sensor itself. Similarly, a postman delivering parcels to villages does not necessarily need to visit each household in person. Instead, one resident in each village can serve as the agent to distribute parcels to other residents in the same village. In both cases, the objective of route planning is to find the shortest cyclic path that intersects with each region.

A formal definition of TSPN is as follows:

$$\text{minimize:} \sum_{i=1}^{n-1} d\left(p_{\tau(i)}, p_{\tau(i+1)}\right) + d\left(p_{\tau(n)}, p_{\tau(1)}\right) \tag{1}$$

subject to:

$$p_i \in Q_i \subset \mathbb{R}^m, \quad i \in [1, n] \tag{2}$$

$$\tau(i) \in [1, n], \tau(i) \neq \tau(j), \qquad \forall i \neq j \tag{3}$$

According to Eq. 1, a TSPN tour contains n path segments, which sequentially connect n regions. Equation 2 requires that each access point p must be within its corresponding region Q. The fundamental property of TSP is ensured in Eq. 3 so that each access point is visited once and only once. In our work, we assume that the distances between any two points are symmetric.

TSPN is significantly more challenging than TSP as it contains both combinatorial and continuous components. In fact, it is necessary to identify a proper access point for each region as well as simultaneously find the optimal permutation of these access points, which is itself a TSP task. These two objectives are also correlated. Given a set of access points, its quality depends on the specific permutation while the quality of a permutation depends on the access points selected. Since the objective function contains different variable types, most optimization techniques cannot be applied in a straightforward manner. After all, due to the presence of the continuous component, the search space of TSPN is infinite, making it impossible to guarantee an optimal solution, unless additional constraints are imposed on the regions.

In the literature, TSPN is largely attempted by approximation algorithms, which aim at finding a PTAS (Polynomial-Time Approximation Scheme) for TSPN [7–10]. The major issue is that strict assumptions on neighborhoods (e.g., disks or *fat* objects of comparable sizes) are essential for producing relatively compact approximation factors, which are still often very large along with high time complexity. Since the

implementations of these algorithms can be complicated and very few if any experimental studies have been reported, their practicability remains unclear. TSPN can be also formulated as a non-convex Mixed-Integer Nonlinear Program (MINLP), which has the attractive feature that fixing all the integer variables can yield a convex nonlinear problem [11]. However, although the technique is claimed to be highly efficient, only TSPN instances with up to 16 regions were tested. Furthermore, since each region is specified by a set of inequalities, all regions are effectively restricted to convex polygons. There are also a few studies on using metaheuristics for solving TSPN where the regions are represented by disks of varying sizes [12, 13].

In practice, the shapes of regions can be complex. For example, a large number of wireless sensors can be deployed in several areas that are distant from each other. However, within each area, sensors may be densely distributed (sufficiently close to each other) so that it is possible to transfer data from all sensors to a specific sensor using multi-hop communication. As a result, instead of requiring the mobile robot to visit each sensor, it only needs to visit a single sensor located in each area. Since the communication range of each sensor is a disk, the neighborhood can be viewed as the overlapping of many disks, creating an arbitrarily complex region (Fig. 1).

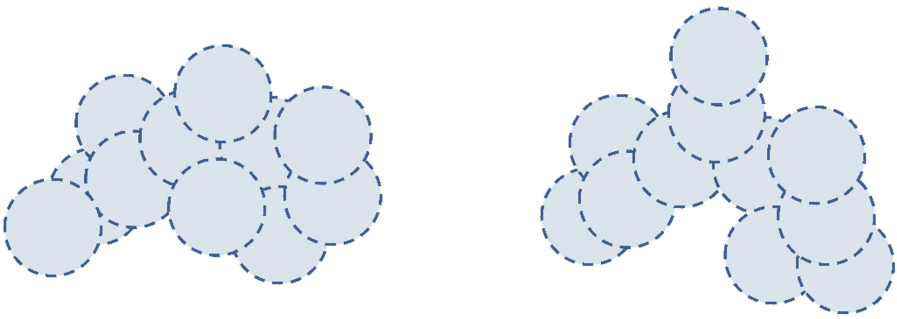

Fig. 1. An example of two clusters of wireless sensors. The effective range of each sensor is shown as a disk and the overall shape of each cluster is very complex.

In this paper, we present a hybrid framework for solving TSPN with arbitrary neighborhoods. The key feature is that there is little if any assumption on the shape of regions, other than being able to sequentially represent each possible access point along the region boundary. Meanwhile, the continuous component (optimization of access points) is handled by a competent metaheuristic while the combinatorial component (optimization of the order of access points) is handled by an efficient TSP solver. Actually, the TSP solver is used as the fitness function in the metaheuristic to evaluate the quality of candidate access points and is otherwise independent from the metaheuristic. By doing so, our method can benefit from state-of-the-art techniques in both communities and is easy to apply by using off-the-shelf implementations.

Section 2 gives the representation of neighborhoods and access points used in our work. It also shows the extra challenge due to non-convex regions. Section 3 presents

the details of the proposed hybrid framework while experiment results are shown in Sect. 4 to demonstrate the performance of our method. This paper is concluded in Sect. 5 with some discussions on the direction of future work.

2 Methodology

For the convenience of computing, each region (neighborhood) is specified by a random simple polygon in the 2D Euclidean space. The number of edges can be manually controlled and each polygon can be either convex or concave (Fig. 2), reflecting the most general situation. Note that any neighborhoods can be reasonably approximated by polygons with a large number of edges.

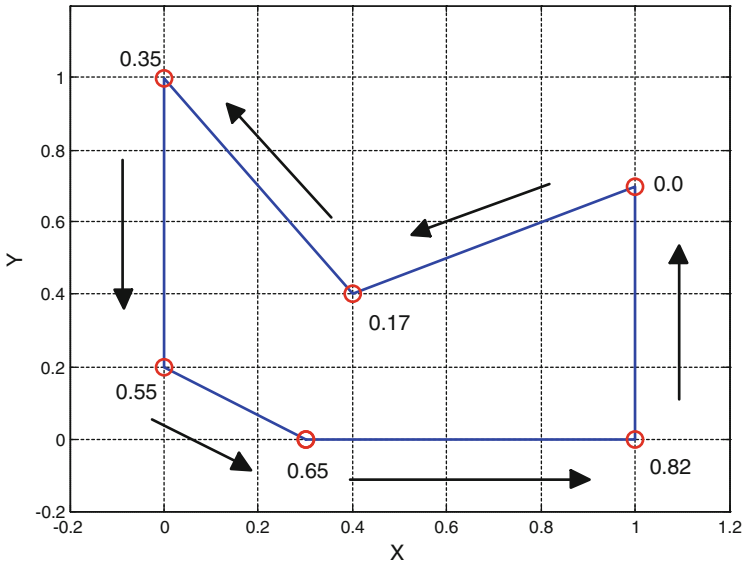

Fig. 2. A concave polygon with 6 vertices. The top-right vertex is assumed to be the starting point. Each vertex is encoded as a continuous value along the anti-clockwise direction.

Meanwhile, creating random simple polygons is not a trivial task. A simple polygon means a polygon without any intersecting sides. In this paper, we use an open source routine based on the Delaunay triangulation of a set of random points [14]. Depending on the desired complexity of polygons, a number of random points are generated within an area and a polygon is then created based on them. Alternatively, given a specific region (e.g., the map of a city), we can sequentially sample data points along the boundary to form the polygon.

Once the parameters (coordinates of vertices) of a polygon are determined, a key factor is how to represent each access point. In the 2D space, each access point can be naïvely represented by its X and Y coordinates. However, it is not convenient as two

values are needed (with possibly different ranges) and special constraint handling is required to make sure that the access point is valid (located on the boundary). More specifically, metaheuristics often apply operators such as crossover and mutation on candidate solutions, altering their values randomly. As a result, a simple box-bounded search space is preferred as metaheuristics often do not come with advanced constraint handling strategies.

For regular shapes such disks, it is possible to use the polar coordinate system according to which each access point is encoded as an angle value. Since there is no specific assumption on the shape of neighborhoods, we cannot rely on any parametric representation of the boundary. Instead, we propose to use the following coding scheme, which is applicable to any type of polygons:

$$E(\mathrm{x}) = \frac{\sum_{\mathrm{i}=0}^{\mathrm{k}-1} \mathrm{d}(\mathrm{v_i}, \mathrm{v_{i+1}}) + \mathrm{d}(\mathrm{v_k}, \mathrm{x})}{L} \tag{4}$$

Given a polygon and its ordered list of vertices v_0, …, v_{n-1}, assume v_k is the vertex directly preceding access point x and L is the length of the entire boundary. Access point x is encoded as the ratio between its distance from v_0 along the boundary and the perimeter of the polygon, as shown in Eq. 4. By doing so, each access point is represented by a single variable within [0, 1) and any value within [0, 1) corresponds to a unique access point on the boundary. Note that any vertex can by chosen as the starting point v_0 and the direction (clockwise vs. anti-clockwise) is not critical. Figure 2 shows the encoded values of the 6 vertices following the anti-clockwise direction where the top-right vertex is regarded as v_0.

Finally, non-convex regions present significant challenges to traditional TSPN techniques. For example, for disk regions, it is easy to predict the distribution of optimal access points given the order of disks, reducing the search space dramatically. Furthermore, convex optimization methods are no longer valid as the search area cannot be represented by a set of inequalities. Also, the structure of the search space is likely to be more complex with non-convex regions. For example, given a disk and an external point x, assume that the nearest point on the disk boundary is x'. As the access point moves away from x', the distance between x and the access point is expected to increase monotonically, creating a smooth landscape. However, for non-convex regions, this is not necessarily the truth. Instead, the resulting landscape may be highly multimodal with several peaks (local optima), which creates much higher-level difficulty for optimization techniques.

3 Framework

The motivation of proposing a hybrid framework is largely due to the fact that solving TSPN involves two sub-tasks: the optimization of the locations of access points (continuous) and the optimization of the order of access points (combinatorial). The classical optimization community has produced many efficient TSP solvers, which can reliably find high quality tours for problems with hundreds of cities within a fairly small amount of time. Meanwhile, the metaheuristic community has come up with

competent stochastic algorithms that can effectively handle multimodal problems with little assumption on their structure (e.g., being convex or differentiable). After all, although metaheuristics can be used to solve TSP, their performance is typically not comparable to state-of-the-art TSP solvers based on well-studied heuristics.

The objective is to introduce a general TSPN solution, which builds upon existing research outcomes and can hide most of the unnecessary details from practitioners. Unlike many existing studies that present specifically tailored methods, which are often sophisticated and difficult to deploy, our framework is easy to implement by incorporating off-the-shelf routines. Although it is possible to formulate TSPN as a mixed optimization problem where the two types of variables are optimized simultaneously, the two sub-tasks can be accomplished separately. In Fig. 3, the metaheuristic is dedicated to optimizing the locations of access points while the quality (fitness) of each set of candidate access points is evaluated by a TSP solver, which returns the length of the TSP tour (expected to be identical or sufficiently close to the optimal tour for small scale problems). By doing so, the two sub-tasks are solved alternately, which reduces the complexity of the original problem and makes different algorithms work on their most suitable problems. In fact, users only need to specify the coding scheme and select the desired optimization routines (Algorithm 1).

Algorithm 1: Hybrid TSPN Solution

```
Input:  Coordinates of vertices of n polygons
Output: p(access points), τ(permutation)
P ← a population of random n-D vectors
Repeat until stopping criteria met
    Repeat evaluate each vector p_i in P
        C ← Decode(p_i)
        [L_i, τ_i] ← TSP_Solver(C)
        Return tour length L_i as the fitness
    End
    P ← Metaheuristic(P, L)
End
Return best p* and τ* found
```

Given n regions, the original optimization problem is as follows where x is an $n-$D real vector (locations) and τ is the permutation of n regions:

$$\min_x \min_\tau f(x, \tau) \tag{5}$$

In Eq. 5, $f(x, \tau)$ returns the length of the tour defined by x and τ. In the proposed framework, the objective function used by the metaheuristic is:

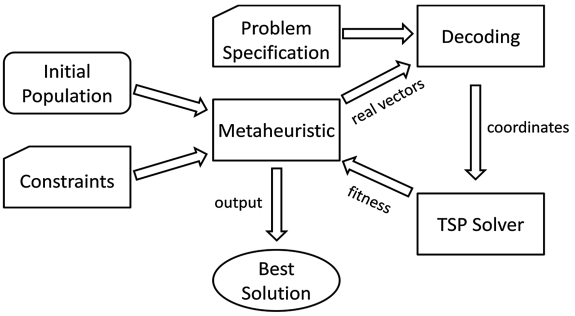

Fig. 3. The proposed hybrid framework for solving TSPN with arbitrary neighborhoods

$$g(\mathrm{x}) = f(\mathrm{x}, \tau^*) \quad \text{where } \tau^* : \min_{\tau} f(\mathrm{x}, \tau) \tag{6}$$

According to Eq. 6, $g(x)$ works by finding the optimal τ^* for the given x and returning the corresponding $f(x, \tau^*)$ value as the quality of the candidate solution. It is clear that, if $f(x, \tau)$ takes its minimum value at $[x^g, \tau^g]$, it is always the truth that $g(x^g)$ is also the minimum value of $g(x)$, which proves the correctness of our method.

4 Experiments

The objective of the experiments is to demonstrate the simplicity and effectiveness of the proposed framework. For this purpose, we selected an open source metaheuristic routine CMA-ES [15, 16] due to its featured capability of working with small populations and handling complex dependences among variables and an open source TSP solver [17]. These two methods were not meant to be the optimal choices and no specific performance tuning was conducted.

4.1 Case Studies

Each polygon was created randomly within a 1-by-1 area with up to 6 edges, which can be either convex or concave. The diversity of generated polygons can be observed intuitively in Fig. 4. All polygons were distributed randomly within a 5-by-5 area without overlapping. Most of the parameter settings in CMA-ES were as the default, except the search boundary, which was bounded between 0 and 1 in each dimension, in accordance with the encoding scheme. Note that the dimension of the continuous search space is equal to the number of regions. As to the TSP solver, it works by randomly selecting a starting node and building an initial tour using the nearest neighbor method, which is then gradually improved by the 2-opt algorithm.

Figure 4 shows an example of a TSPN tour among 10 polygons. The access point of each region is shown in circle. For TSPN, it is generally not feasible to have the prior knowledge about the optimal tour. Meanwhile, existing TSPN techniques are often not directly applicable to complex neighborhoods and cannot be used for comparison

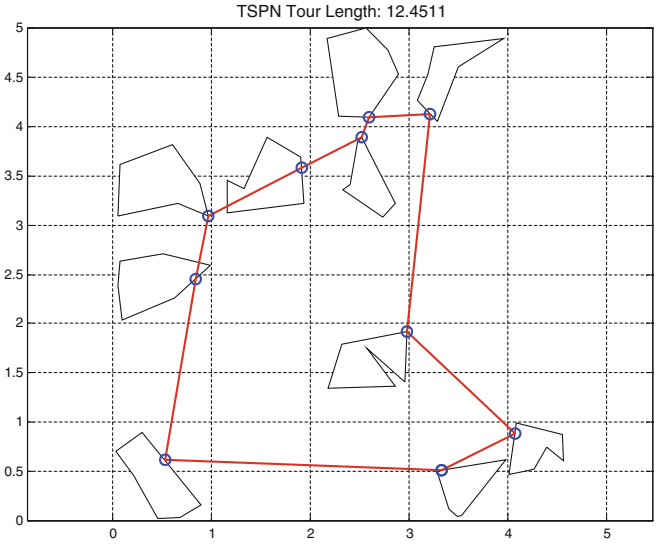

Fig. 4. A 10-region TSPN tour, showing the selected access point of each region

purpose. Nevertheless, it is still possible to verify the effectiveness of our method by examining the performance curve in Fig. 5. At the beginning of iteration, the length of the best tour was around 15.5. In fact, each tour in the initial population was created by randomly selecting a set of access points and applying the TSP solver to find the corresponding shortest path. As a result, these tours were partially optimized TSPN solutions and can be used as the base line. After only 50 iterations, the length of the best tour was already reduced to 12.5 and kept improving slightly till 12.45, which was a common pattern across different trials. After all, for this relatively small scale instance, the quality of the resulting tour can be also inspected visually.

Figure 6 shows an example of a TSPN tour among 25 polygons. It is clear that, for problems at this scale, it is already very difficult to manually work out a near-optimal solution. However, our method constructed a TSPN tour with length under 17 and the quality of the tour can again be observed: there were several cases where a single line segment connected multiple regions, a desirable feature for producing short tours.

The efficiency of our method is also evident. On an entry level desktop computer with Intel i5-3470S at 2.9 GHz CPU, 10-region TSPN instances were typically solved to a reasonable level in less than 5 s while 25-region TSPN instances required less than 15 s. Note that metaheuristics such as CMA-ES often feature good potential of parallelism and one to two orders of magnitude speedup can be expected using advanced parallel computing techniques such as GPU computing [18].

4.2 Problem Analysis

Finally, it is equally important to have some insights into the structure of the problem to understand the challenges confronted by optimization algorithms. We focus on the

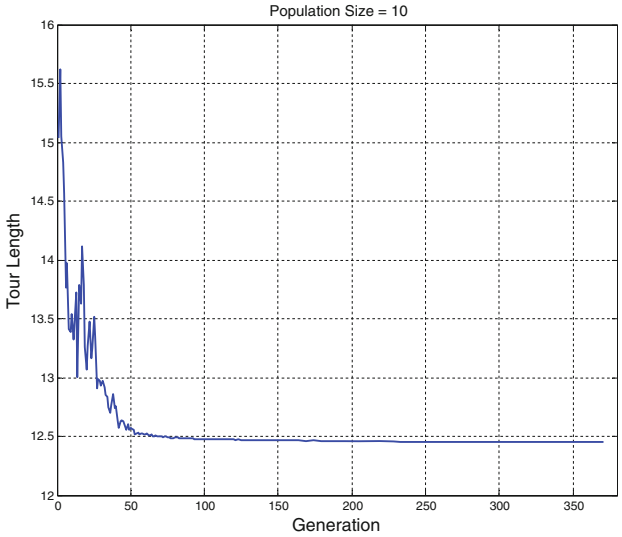

Fig. 5. The current best solution during iteration, showing the fast convergence of CMA-ES

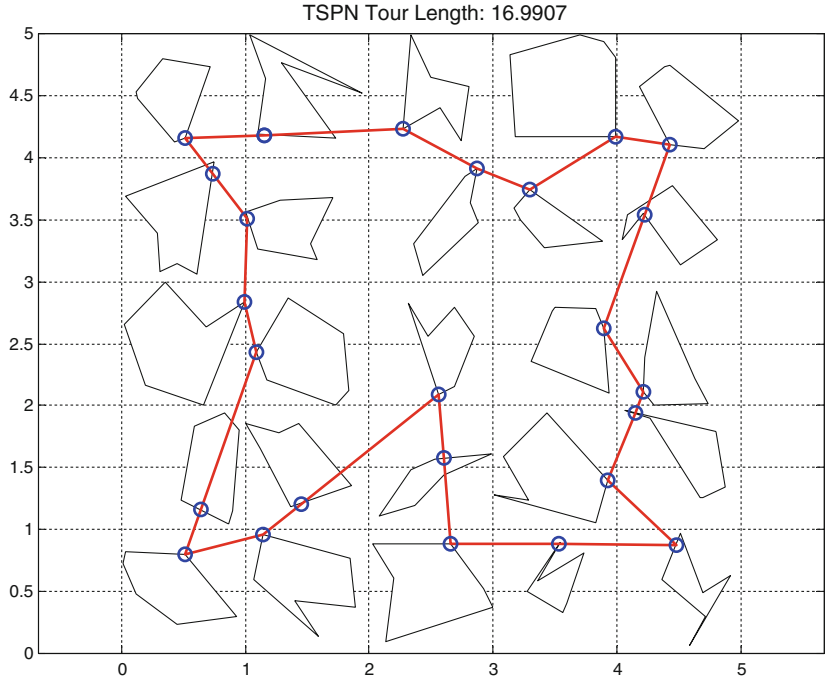

Fig. 6. A 25-region TSPN tour, showing the selected access point of each region

continuous sub-task as the combinatorial part (TSP) has been extensively studied in the literature. Figure 7 (left) shows a TSPN instance with three polygons so that the optimal permutation is trivial. Totally 10,000 random candidates were generated with each one represented by a 3-D vector within the range [0, 1). The fitness of each candidate was evaluated by the length of the corresponding tour (i.e., a triangle). The candidate with the shortest distance was regarded as the optimal solution and its distances to all other candidates were calculated.

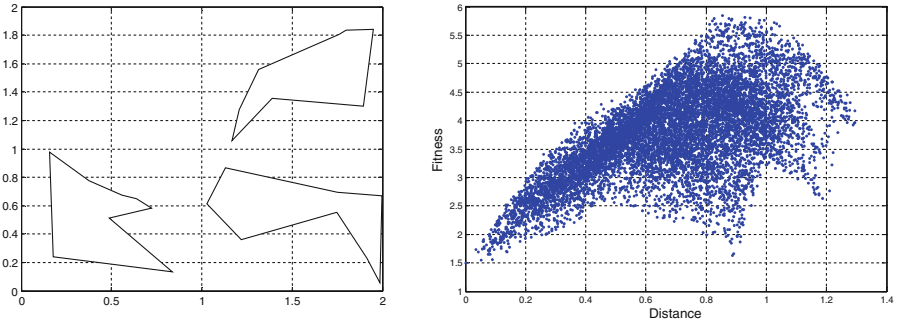

Fig. 7. A TSPN example with three polygons (left) and the fitness-distance plot (right), which shows the relationship between the quality of candidates and their distances to the best solution.

In Fig. 7 (right), each candidate is represented by its Euclidean distance to the best solution (horizontal axis) and its fitness value (vertical axis). This type of plot is often referred to as the fitness-distance plot [19], which can provide intuitive information about the difficulty of problems. It is clear that in the region close to the best solution, there is a positive correlation between fitness and distance: the closer a candidate to the best candidate, the better its quality. This pattern generally indicates optimization friendly problem structure, as it provides an effective guidance on the search direction. However, things are quite different on the far side. Candidates in these regions tend to get worse (longer tours) when moving closer to the best candidate, which may unfortunately create a deceptive search space and mislead the optimization process.

There are a number of factors that can influence the correlation between fitness and distance. For example, the pattern in Fig. 7 (right) may imply that the problem is inherently multimodal and traditional gradient based methods may not work well. Note that the encoding scheme will also have direct impact on the problem structure.

5 Conclusion

As a class of optimization problems with significant practical implications, TSPN with arbitrary neighborhoods presents unprecedented challenges to approximation algorithms and convex optimization techniques and no effective solutions are available in the literature. In our work, we present a novel hybrid framework that combines

competent algorithms from both continuous and combinatorial optimization communities to tackle TSPN with complex neighborhoods. Without the need to directly confront a mixed optimization problem, our method features demonstrated simplicity (largely using off-the-shelf routines), flexibility (imposing little assumption on neighborhoods) and efficiency (solving non-trivial TSPN instances in seconds). The current preliminary study can serve as the foundation for more comprehensive research on solving large-scale TSPN instances as well as many interesting related problems such as the safari route problem and the zookeeper route problem. We will also make the source code available online to help researchers and practitioners further extend and investigate this hybrid framework.

Acknowledgement. This work was supported by Natural Science Foundation of Guangdong Province (No. 2014A030310318) and Research Foundation of Shenzhen (No. JCYJ20160301153317415).

References

1. Applegate, D., Bixby, R., Chvátal, V., Cook, W.: The Traveling Salesman Problem: A Computational Study. Princeton University Press, Princeton (2007)
2. Arora, S.: Polynomial time approximation schemes for Euclidean traveling salesman and other geometric problems. J. ACM **45**, 753–782 (1998)
3. Larrañaga, P., Kuijpers, C., Murga, R., Inza, I., Dizdarevic, S.: Genetic algorithms for the travelling salesman problem: a review of representations and operators. Artif. Intell. Rev. **13**, 129–170 (1999)
4. Helsgaun, K.: An effective implementation of the Lin-Kernighan traveling salesman heuristic. Eur. J. Oper. Res. **126**, 106–130 (2000)
5. Alatartsev, S., Stellmacher, S., Ortmeier, F.: Robotic task sequencing problem: a survey. J. Intell. Robot. Syst. **80**, 279–298 (2015)
6. Arkin, E.M., Hassin, R.: Approximation algorithms for the geometric covering salesman problem. Discret. Appl. Math. **55**, 197–218 (1994)
7. Mitchell, J.: A PTAS for TSP with neighborhoods among fat regions in the plane. In: Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 11–18 (2007)
8. Elbassioni, K., Fishkin, A., Sitters, R.: Approximation algorithms for the Euclidean traveling salesman problem with discrete and continuous neighborhoods. Int. J. Comput. Geom. Appl. **19**, 173–193 (2009)
9. Chan, T., Elbassioni, K.: A QPTAS for TSP with fat weakly disjoint neighborhoods in doubling metrics. Discret. Comput. Geom. **46**, 704–723 (2011)
10. Dumitrescu, A., Tóth, C.: Constant-factor approximation for TSP with disks (2016). arXiv: 1506.07903v3 [cs.CG]
11. Gentilini, I., Margot, F., Shimada, K.: The travelling salesman problem with neighborhoods: MINLP solution. Optim. Methods Softw. **28**, 364–378 (2013)
12. Yuan, B., Orlowska, M., Sadiq, S.: On the optimal robot routing problem in wireless sensor networks. IEEE Trans. Knowl. Data Eng. **19**, 1252–1261 (2007)
13. Chang, W., Zeng, D., Chen, R., Guo, S.: An artificial bee colony algorithm for data collection path planning in sparse wireless sensor networks. Int. J. Mach. Learn. Cybern. **6**, 375–383 (2015)

14. Random 2D Polygon Code. http://stackoverflow.com/questions/8997099/algorithm-to-generate-random-2d-polygon
15. CMA-ES Source Code. https://www.lri.fr/∼hansen/cmaes_inmatlab.html
16. Hansen, N., Ostermeier, A.: Completely derandomized self-adaptation in evolution strategies. Evol. Comput. **9**, 159–195 (2001)
17. TSPSEARCH. http://www.mathworks.com/matlabcentral/fileexchange/35178-tspsearch
18. Kirk, D., Hwu, W.: Programming Massively Parallel Processors: A Hands-on Approach. Morgan Kaufmann, San Francisco (2012)
19. Jones, T., Forrest, S.: Fitness distance correlation as a measure of problem difficulty for genetic algorithms. In: Proceedings of 6th International Conference on Genetic Algorithms, pp. 184–192 (1995)