Features of Easy and Hard Instances for Approximation Algorithms and the Traveling Salesperson Problem

Samadhi Nallaperuma[1], Markus Wagner[1], Frank Neumann[1], Bernd Bischl[2], Olaf Mersmann[2], and Heike Trautmann[2]

[1] School of Computer Science, University of Adelaide, Adelaide, SA 5005, Australia
[2] Statistics Faculty, TU Dortmund University, 44221 Dortmund, Germany

1 Introduction

Understanding the performance of algorithms for hard optimization problems such as the Travelling Salesperson Problem (TSP) is still a difficult task. Classical approaches taking a worst-case or an average-case perspective hardly capture what is happening for real instances. For a given instance I of a combinatorial optimization problem, it is often hard to predict the performance of an algorithm A without running A on I. Hyper heuristics in the optimisation domain and meta-learning in the machine learning domain focus on finding the conditions which determine algorithm performance in advance. Smith-Miles and Lopes [5] classify the research on problem hardness analysis into two different directions. The first direction is to consider the problem as a learning problem, where automatic algorithm selection [2] is done based on learned knowledge from previous algorithm performance. The second direction is to analyse the algorithms and problems theoretically [3] and experimentally [5] to understand the reasons for performance on different problem instances. This understanding is the key to future algorithm design for more complex real world problems. The study consideres both approaches, where we investigate the performance of several approximation algorithms for the TSP on different instances. We evolve instances that are hard or easy for these algorithms and characterize features of such instances. Our approach aligns well with the investigations of Mersmann et al. [4] for the 2-opt algorithm. The insights can be used to improve feature based performance prediction in order to support automatic algorithm selection.

1.1 The Travelling Salesperson Problem

The TSP is one of the most famous NP-hard combinatorial optimization problems problems. Given a set of n cities $\{1, \ldots, n\}$ and a distance matrix $d = (d_{ij})$, $1 \leq i, j \leq n$, the goal is to compute a tour of minimal length which visits each city exactly once and returns to the origin.

In general, the TSP is not only NP-hard but also hard to approximate. Therefore we consider the still NP-hard Euclidean TSP where cities are given by points in the plane and distances are given by the Euclidean intances between these points. The Euclidean TSP is a special case of the Metric TSP where the distances between the cities have to fullfill the triangle inequality, i. e. $d_{ik} \leq d_{ij} + d_{jk}$ holds for $i, j, k \in \{1, \ldots, n\}$. For the Metric TSP different approximation algorithms are known and we investigate two of the most common ones.

Our goal is to evolve easy and hard instances for a well-known 2-approximation algorithm and the 3/2-approximation algorithm by Christofides. A detailed description and analysis of these algorithms can be found in the textbook of Vazirani (Section 3.2) [6].

2 EA based hard and easy instance generation

We use an evolutionary algorithm introduced by Mersmann et al. [4] to evolve easy and hard instances for the two approximation algorithms. Evolutionary algorithms are based to a large extend on random decisions. We use several runs of an algorithm to create a diverse set of hard and easy instances.

We measure the hardness of an instance I for a given algorithm A by the approximation ratio $\alpha_A(I)$. For short we write $\alpha(I)$ if it is clear which algorithm A is under investigation. The approximation ratio of an algorithm A for a given instance I is defined as

$$\alpha(I) = A(I)/OPT(I)$$

where $A(I)$ is the tour length produced by algorithm A for the given instance I and OPT(I) is the value of an optimal solution of I. An algorithm A is an r-approximation algorithm if for any input I, $\alpha(I) \leq r$ holds, i. e. the hardest instance can have an approximation ratio of at most r.

Given an algorithm A, $\alpha(I)$ is chosen as the fitness function that assigns to an instance I the approximation ratio. We only consider deterministic approximation algorithms in this paper which implies that we obtain $A(I)$ by a single run of algorithm A and a given instance I. $OPT(I)$is obtained by using the exact TSP solver Concorde [1]. The search is guided by the approximation ratio of an instances which is used as the fitness function in the evolutionary algorithm. We maximize $\alpha(I)$ in order to generate hard instances and minimize $\alpha(I)$ in order to generate easy instances for a given fixed algorithm A.

3 2-Approximation

We study features that lead to easy and hard instances in a similar way as Mersmann et al. [4], including statistics based on: the distance matrix, clusters, nearest-neighbour distances between cities, the minimum spanning tree, angles between cities and the convex hull of the cities in the plane. Different TSP instance sizes are considered for the analysis. Cities are generated in $[0, 1]^2$ and placed on a discretized grid enabling cross comparison of features. Instances with various difficulty levels in between easy and hard are generated using a point matching strategy. These instances provide an understanding of the correlation between instance features and problem difficulty.

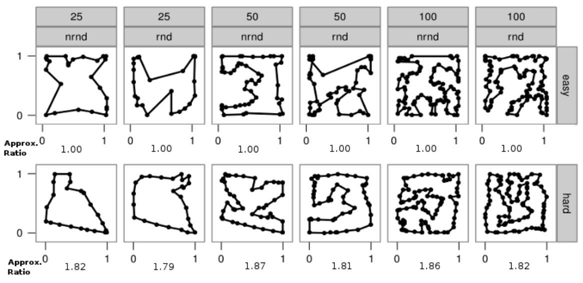

Fig. 1. Examples of the evolved instances of both types (easy, hard) for 2 Approximation including the optimal tours computed by Concorde for different instance sizes and rounding strategies (round before normal mutation (rnd), round after normal mutation (nrnd)).

Our experimental results for the 2-approximation algorithm show the following. The distances between cities on the optimal tour are more uniform in the hard instances than in the easy ones. Examples of the hard and the easy instances are shown in Figure 1. The approximation ratio is very close to 1 for all the generated easy instances where as for the hard instances it is significantly higher ranging from 1.79 to 1.87. Standard deviations of the distances on the optimal tour of the easy instances are roughly twice as high than for the hard instances when considering small instance sizes. This gap gets decreased to 1.5 for larger instances. It is observable that the easy instances consist of small clusters of cities opposed to a more uniform distribution in the hard instances. Visually, optimal tours for the easy instances lead to higher angles than in optimal tours of the hard instances. The mean angles of the easy instances are significantly smaller than the values of the hard instances. These mean angle values for both the hard and the easy instances slightly decrease with the instance size. Instance shapes for small instances structurally differ from the respective shapes of larger instances. Consequently, the area covered by the convex hull is higher for larger instances.

4 Christofides 3/2-Approximation

We conduct a similar analysis for the Christofides algorithm. Example instances are shown in Figure 2. Instance shapes do not exhibit a significant difference between easy and hard instances. However, statistics provide more evidence on differences. Approximation ratio is close to 1 for all the easy instances and roughly 1.4 for the hard instances. Standard deviations of the distances on the optimal tour of the easy instances are considerably higher than for the hard instances. This gap stays stable with increasing instance size. The mean angles of the easy instances are higher than for the hard instances when considering small instances, and lower for larger instance sizes. The opposite is true for the standard

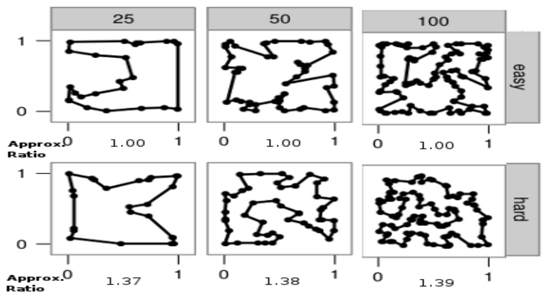

Fig. 2. Examples of the evolved instances of both types (easy, hard) for Christofides including the optimal tours computed by Concorde for different instance sizes and rounding strategies (round before normal mutation (rnd), round after normal mutation (nrnd)).

deviation of angles. This is a hint that the hard instances have higher angles than the easy ones for larger instance sizes.

5 Future Work

We have carried out an evolutionary algorithm approach to generate easy and hard instances for two classical approximation algorithms and the traveling salesperson problem. Future work will concentrate on the derivation of rules to classify easy and hard instances based on feature values, comparing these approximation algorithms to heuristic methods such as local search, and using our insights for algorithm selection.

References

1. Applegate, D., Cook, W.J., Dash, S., Rohe, A.: Solution of a min-max vehicle routing problem. INFORMS Journal on Computing 14(2), 132–143 (2002)
2. Bischl, B., Mersmann, O., Trautmann, H., Preuß, M.: Algorithm selection based on exploratory landscape analysis and cost-sensitive learning. In: Proceedings of the fourteenth international conference on Genetic and evolutionary computation conference. GECCO '12, ACM (2012)
3. Kötzing, T., Neumann, F., Röglin, H., Witt, C.: Theoretical properties of two ACO approaches for the traveling salesman problem. In: Proc. of ANTS 2010. LNCS, vol. 6234, pp. 324–335 (2010)
4. Mersmann, O., Bischl, B., Bossek, J., Trautmann, H., Wagner, M., Neumann, F.: Local search and the traveling salesman problem: A feature-based characterization of problem hardness. In: Proceedings of the Learning and Intelligent Optimization Conference (LION). Lecture Notes in Computer Science, Springer (2012)
5. Smith-Miles, K., Lopes, L.: Measuring instance difficulty for combinatorial optimization problems. Computers & OR 39(5), 875–889 (2012)
6. Vazirani, V.V.: Approximation algorithms. Springer (2001)