An $n \log n$ Heuristic for the TSP

Éric D. Taillard

Univ. of Applied Sciences of Western Switzerland
HEIG-VD
Rte de Cheseaux 1, CP 521, CH-1401 Yverdon-les-Bains, Suisse
eric.taillard@heig-vd.ch

Abstract

An $n \log n$ randomized method based on POPMUSIC metaheuristic is proposed for generating reasonably good solutions to the travelling salesman problem. The method improves a previous work which algorithmic complexity was in $n^{1.6}$. The method has been tested on instances with billions of cities. Few dozens of runs are able to generate a very high proportion of the edges of the best solutions known. This characteristic is exploited in a new release of the Helsgaun’s implementation of Lin-Kernighan heuristic (LKH) that is also able to produce rapidly extremely good solutions for non Euclidean instances.

1 Introduction

The travelling salesman problem (TSP) is certainly the most studied NP-hard combinatorial optimisation problem [4, 9, 13]. Now, we are able to exactly solve instances up to several thousands of cities and to find solutions for instances with millions of cities at a fraction of percent above the optimum [1, 3, 5, 6, 8, 10].

A key point for implementing a fast and efficient local search is to use a neighbourhood of limited size containing all the pertinent moves. For the travelling salesman problem, the most efficient neighbourhoods are based on Lin-Kernighan moves. In order to speed-up the computation, only a subset of moves are evaluated. This article propose a method with a reduced algorithmic complexity for generating a limited subset of pertinent edges that must be used in the solution tour. This technique for reducing the complexity of the local search was called tour merging by [2].

Previous works [11, 12] proposed a 2-levels method for generating good candidate edges with a complexity in $n^{1.6}$. The method is based on 2 main steps: first, an initial tour is built with a recursive randomized procedure, with an algorithmic complexity in $n \log n$. Then this initial tour is improved in linear time with a fast POPMUSIC metaheuristics.

The algorithm is discussed in Section 2. Section 3 presents numerical results showing firstly that the empirical complexity of the method is quasi-linear, as expected, and secondly that it generates a very large proportion of edges belonging to the best solution tours.

2 The $n \log n$ Heuristic for the TSP

2.1 Building an initial tour

For generating an initial tour that can be further improved with a fast POPMUSIC algorithm, an arbitrary city is duplicated. Let us call $c_0$ and $c_n$ this duplicated city and let $C = \{c_1 \ldots c_{n-1}\}$ be the other cities of an instance. The path $P = c_0, c_1, \ldots, c_{n-1}, c_n$ defines a tour. Let $t$ be the only parameter of the method (in the numerical results presented in Section 3, $t = 15$). If $n \leq t^2$, then a very good path passing once through all cities of $P$, starting at city $c_0$ and ending at city $c_n$ can be found, for instance with a local search using Lin-Kernighan moves. A good tour is built and the method stops.

Otherwise, if $n > t^2$, a sample $S$ of $t$ cities is randomly chosen from $C$. Let $s_1 \in S$ be the city the closest to $c_0$ and $s_t \in S \setminus \{s_1\}$ the city the closest to $c_n$. A good path $P_S$ through all the cities of sample $S$, starting at city $s_1$ and ending at city $s_t$ can be found with a local search. Let us rename the cities of $S$ so that $P_S = s_1, s_2, \ldots, s_{t-1}, s_t$.
Let us decompose all the cities of \( C \) into \( t \) clusters \( C_1 \ldots C_t \), so that cluster \( C_i \) contains all the cities that are closer to \( s_i \) than to the other cities of \( S \) (\( i = 1 \ldots t \)). The initial path \( P \) can be reordered so that it starts with city \( c_0 \), then it has all the cities of cluster \( C_1 \), then those of \( C_2 \), then \ldots , then those of \( C_t \), and ends with \( c_n \).

At this step, the order of visit of the cities of a cluster is arbitrary (as it was for \( P \) at the beginning of the procedure). This order can be improved by calling recursively the procedure for each sub-path \( C_i \) (\( i = 1 \ldots t \)), the last city of \( C_{i-1} \) playing the role of \( c_0 \) (or being actually \( c_0 \) for \( C_1 \)) and the first city of \( C_{i+1} \) playing the role of \( c_n \) (or being actually \( c_n \) for \( C_t \)).

When all recursions stop, the cities are ordered in such a way that it can be successfully improved with a POPMUSIC-based heuristic. If \( t \) is considered as a fixed parameter (not depending on the problem size \( n \)), the algorithmic complexity of this procedure is \( n \log n \).

### 2.2 Improving the initial tour with a fast POPMUSIC

In [11], POPMUSIC metaheuristic is adapted for the TSP by optimizing sub-paths containing \( R \) consecutive cities of the tour, where \( R \) is a parameter. The optimization procedure is a local search based on Lin-Kernighan moves. The optimizations are repeated until there is no subset of \( R \) consecutive cities in the tour that can be improved with the Lin-Kernighan neighbourhood.

For getting good candidate edges by the tour merging technique, it is not required to run POPMUSIC until all sub-paths of \( R \) consecutive cities have been optimized. Instead, we propose to speed-up the method by optimizing the tour in 2 scans. A first scan optimizes non-overlapping sub-paths of \( R \) cities starting with the first city. Then, the tour is shifted by \( R/2 \) cities and a second scan optimizes sub-paths involving \( R/2 \) cities for each of two adjacent sub-paths in the first scan.

If \( R \) does not depend on \( n \) (we have chosen \( R = t^2 \) to have a method with a unique parameter), then the algorithmic complexity of the improvement with POPMUSIC is linear.

### 3 Computational results

#### 3.1 Computational times

The main goal of this work was to produce moderately good solutions to TSP instances with a short computational time. In figure 1, we provide the computational time of our method as a function of the problem size. The problem instances were randomly generated in the unit square and toroidal distances are considered (as if the square was folded so that opposite border are contiguous). We provide the time for generating the initial solution and for improving it with the fast POPMUSIC presented above. In this figure, we have included the computational time for generating a tour with the previous POPMUSIC implementation of [11].

The main limitation of our method is due to the memory required for storing the problem data and the solution. A personal computer with 64Gb of main memory was able to deal with instances with 2G cities.

#### 3.2 Solution quality

The optimal solution of randomly generated solutions in the unit square with toroidal distances was estimated by [7] as \( (0.7124 \pm 0.0002) \sqrt{n} \). The quality of the solutions produced for such instances is given in Figure 2. The method proposed is able to produce solution at about 10% above optimum for all instance size. Let us mention that the nearest neighbour heuristic produces solution about 22% above optimum.

In Figure 3, we give, for few instances of the literature, the proportion of missing edges of the best solution known as a function of the number of runs of our method. We see in this figure that the union of 50 solutions produced by the method contains more than 99.9% of the edges of the best solution known.

Cartagena, July 28-31, 2019
Figure 1: Computational time for producing one solution as a function of problem size.

Figure 2: Quality of solutions (expressed in % above expected optimum) as a function of problem size. Uniformly distributed cities with toroidal distances in 2D.

Figure 3: Proportion of missing edges as a function of the number of POPMUSIC solutions generated.
4 Conclusion

This work propose a method for generating moderately good TSP solutions in $n \log n$. The method does not make assumption about the problem structure. For the first time, to our knowledge, instances with more than a billion of cities have been tackled with a metaheuristic. The LKH 2.0.9 release includes the proposed method for generating candidate edge sets.

References


[13] Voigt, editor. Der Handlungsreisende wie er sein soll und was er zu thun hat, um Aufträge zu erhalten und eines glücklichen Erfolgs in seinen Geschäften gewiß zu sein ; Mit einem Titelkupf. Voigt, Ilmenau, Germany, 1832.