Finding Good Tours for Huge Euclidean TSP Instances by Iterative Backbone Contraction

Christian Ernst\textsuperscript{1,3}, Changxing Dong\textsuperscript{1}, Gerold Jäger\textsuperscript{1,2}, Dirk Richter\textsuperscript{1}, and Paul Molitor\textsuperscript{1}

\textsuperscript{1} Martin-Luther-University Halle-Wittenberg, D-06120 Halle (Saale), Germany
\textsuperscript{2} Christian-Albrechts-University Kiel, D-24118 Kiel, Germany
\textsuperscript{3} GISA GmbH, D-06112 Halle (Saale), Germany

Abstract. This paper presents an iterative, highly parallelizable approach to find good tours for very large instances of the Euclidian version of the well-known Traveling Salesman Problem (TSP). The basic idea of the approach consists of iteratively transforming the TSP instance to another one with smaller size by contracting pseudo backbone edges. The iteration is stopped, if the new TSP instance is small enough for directly applying an exact algorithm or an efficient TSP heuristic. The pseudo backbone edges of each iteration are computed by a window based technique in which the TSP instance is tiled in non-disjoint sub-instances. For each of these sub-instances a good tour is computed, independently of the other sub-instances. An edge which is contained in the computed tour of every sub-instance (of the current iteration) containing this edge is denoted to be a pseudo backbone edge. Paths of pseudo-backbone edges are contracted to single edges which are fixed during the subsequent process.

1 Introduction

The Traveling Salesman Problem (TSP) is a well known and intensively studied problem \cite{15,10,17} which plays a very important role in combinatorial optimization. It can be simply stated as follows. Given a set of cities and the distances between each pair of them, find a shortest cycle visiting each city exactly once. If the distance between two cities does not depend on the direction, the problem is called symmetric. The size of the problem instance is defined as the number $n$ of cities. Formally, for a complete, undirected and weighted graph with $n$ vertices, the problem consists of finding a minimal Hamiltonian cycle. In this paper we consider Euclidean TSP (ETSP) instances whose cities are embedded in the Euclidean plane\textsuperscript{[16]}. Although TSP is easy to understand, it is hard to solve, namely \textit{NP}-hard. We distinguish two classes of algorithms for the symmetric TSP, namely heuristics

\textsuperscript{1} However, the ideas presented in this paper can be easily extended to the case in which the cities are specified by their latitude and longitude, treating the Earth as a ball (see \cite{19}).
and exact algorithms. For the exact algorithms the program package *Concorde* [1][20], which combines techniques of linear programming and branch-and-cut, is the currently leading code. Concorde has exactly solved many benchmark instances, the largest one has size 85,900 [2]. On the other hand, in the field of symmetric TSP heuristics, Helsgaun’s code [6][7][8][21] (LKH), which is an effective implementation of the Lin-Kernighan heuristic [11], is one of the best packages. Especially for the most yet not exactly solved TSP benchmark instances [14][15][16][18][19], this code found the currently best tours.

An interesting observation [13] is that tours with good quality are likely to share many edges. Dong et al. [4] exploited this observation by first computing a number of good tours of a given TSP instance by using several different heuristical approaches, collecting the edges which are contained in each of these (not necessarily optimal) tours, computing the maximal paths consisting of only these edges, and contracting these maximal paths to single edges which are kept fixed during the following process. By the contraction step, a new TSP instance with smaller size is created which can be attacked more effectively. For some TSP benchmark instances of the VLSI Data Set [18] with sizes up to 47,608, this approach found better tours than the best ones so far reported.

The idea of fixing edges and reducing chains of fixed edges to single edges is not new. It has already been presented by Walshaw in his multilevel version of Helsgaun’s LKH [12]. Walshaw’s process of fixing edges however is rather naive as it only matches vertices with their nearest unmatched neighbours instead of using more sophisticated edge measures.

An alternative to the approach would be fixing without backbone contraction. Thus the search space is considerably cut, although the size of the problem is not reduced. This basic concept of edge fixing was already used by Lin, Kernighan [11] and is implemented in LKH. The main difference between edge fixing without backbone contraction and the approaches presented in [4][12] is the reduction of the size by contracting. This reduction has great influence to the effectiveness of the approach. The reason is that all the edges incident to an inner vertex of the contracted paths do not appear in the new instance anymore. Another idea related to [4] is Cook and Seymour’s tour merging algorithm [3], which merges a given set of starting tours to get an improved tour.

The bottleneck of the approach presented in [4] when applied to huge TSP instances is the computation of several good starting tours, i.e., tours of high quality, by using several different TSP methods. Using different TSP heuristics during the computation of the starting tours hopefully increases the probability that edges contained in each of the starting tours are edges which are also contained in optimal tours.

This paper focuses on TSP instances with very large sizes. Only a tiny part of the search space of such a huge TSP instance can be traversed in reasonable time. To overcome this problem, huge TSP instances are usually partitioned. In our new approach, which handles ETSP, this partitioning is done by moving a window frame across the bounding box of the vertices. The amount of the step-wise shift is chosen as a fraction $1/s$ of the width (height) of the window frame.