Pergamon

0305-0548(95)00066-6

Computers Ops Res. Vol. 23, No. 7, pp. 681–694, 1996

0305-0548/96 $15.00 + 0.00

DIVIDE AND CONQUER STRATEGIES FOR PARALLEL TSP HEURISTICS

Giovanni Cesari†

Institute for Theoretical Computer Science, Swiss Federal Institute of Technology ETH, CH-8092 Zurich, Switzerland

(Received September 1994; in revised form August 1995)

Scope and Purpose—The Traveling Salesman Problem (TSP) is a classic combinatorial problem that has been shown to be NP-hard [E. L. Lawler *et al.*, *The Travelling Salesman Problem*, John Wiley, New York (1985); G. Laporte, *Eur. J. Ops Res.* **59**, 231–247 (1992)]. In many applications, a suboptimal, but reasonably good, solution is quite acceptable. Considerable effort has been devoted in developing efficient heuristic algorithms that, on average, produce good solutions for large problem-instances [D. S. Johnson, In *Proc. 11th Colloquim Automata, Languages and Programming*, pp. 446–461. Springer, Berlin (1990); J. L. Bentley, *ORSA J. Computing* **4**, 387–411 (1990)]. Massively parallel computers may then represent an attractive tool for further improvements in efficiency. In this paper we show how to exploit parallelism by distributing the cities among several processors. The processors simultaneously compute partial solutions, which are then merged to obtain the global solution to the problem. Our approach minimizes the overhead of inter-processor communication, but also induces some loss in the quality of the global solution.

Abstract—The aim of this paper is to study experimentally divide and conquer strategies, used to implement parallel heuristics for the geometric Traveling Salesman Problem (TSP). Starting from Karp's results [R. M. Karp, *Math. Ops Res.* **2**, 209–224 (1977)] we subdivide the set of cities into smaller sets, and we compute an optimal subtour for each subset. These subtours are then combined to obtain the tour for the entire problem. The analysis is restricted to problem instances where points are uniformly distributed in the unit square. For relatively small sets of cities we study the quality of the solution by comparing it with the optimal solution. In large problem-instances, statistical estimates of the optimal solutions are used for asymptotical analysis. We apply the same dissection strategy also to classical heuristics: the final solution, obtained by combining the approximate subsolutions, is compared with the average quality of the heuristic. Our main result is the estimate of the rate of convergence of the approximate solution to the optimal solution as a function of the number of dissection steps, of the criterion used for the plane division and of the quality of the subtours.

We have implemented our programs on MUSIC (MUlti Signal processor system with Intelligent Communication), a Single-Program-Multiple-Data (SPMD) parallel computer with distributed memory, which has been developed at the ETH Zurich.

1. INTRODUCTION

The Traveling Salesman Problem (TSP) is one of the classic problems in combinatorial optimization. It can be stated as follows: let $G = (V, A)$ be a graph where V is a set of N vertices and A is a set of edges. Let $C = (c_{ij})$ be a given distance matrix associated with A. The TSP consists of determining a minimum length Hamiltonian tour, that is a minimum distance circuit passing through each vertex once and only once. In the most common interpretation, the nodes are cities and the problem consists in minimizing the length of the tour for a traveling salesman who wants to visit all the cities. The literature on the TSP is extremely vast; see for example [1] for a comprehensive survey and [23] for more recent work.

In this paper we consider the possibility of calculating approximate solutions of large TSP instances using dissection algorithms implemented on a parallel computer. Our main goals are the

†Giovanni Cesari holds the PhD Degree in Electrical Engineering from the University of Trieste, Italy and is now a Research and Teaching Assistant in the Computer Science Department at the University of Trieste, Italy. Since 1994 he is on leave of absence at the ETH, Swiss Federal Institute of Technology Zurich, Switzerland. His main fields of interest are high performance and parallel computing in various fields such as physics, chemistry, optimization processes, and symbolic computation.

analysis of the performances of our implementations and the experimental study of the quality of the solutions compared with theoretical results given by Karp [4]. We concentrate on the geometric TSP, that is with symmetric and Euclidean distances between cities ($c_{ij} = c_{ji} \forall i,j \in V; c_{ij} + c_{jk} \geqslant c_{ik} \forall i,j,k \in V$). It is well known that the TSP (also in the geometric version) is an *NP–hard* problem [5]. The symmetric TSP has many applications, from VLSI chip fabrication [6] which involves problems with up to 1.2 million cities, to X-ray crystallography [7] (up to 14,000 cities) and circuit board drilling applications [8] (up to 17,000 cities).

The organization of this paper is as follows:

- in Section 2 we give a brief review of the literature about TSP heuristics. One subsection is devoted in particular to describe some parallel implementations and one to present Karp's dissection algorithms;
- in Section 3 we present the MUSIC architecture, describe our parallel implementation of Karp's algorithms and discuss the efficiency of this implementation;
- in Section 4 we analyse experimentally the average quality of the solution, when optimal subtours are computed and compare it with the worst case error;
- in Section 5 we study the asymptotic expected performances of dissection algorithms, when optimal as well as approximate subtours are calculated;
- in Section 6 we give some conclusions.

2. LITERATURE REVIEW

Since the TSP is an *NP–hard* problem, it is natural to try to find non optimal, but reasonably good solutions with heuristic algorithms instead of searching an optimal solution. Research in the field of TSP heuristics has followed three main approaches, one consisting of developing heuristics with a guaranteed worst case performance [9], one which performs a probabilistic analysis of the quality of the solution [10] and one which tries to design heuristics with good empirical performances [11]. Another measure to analyse the quality of a heuristic solution, called the domination analysis is proposed in [12, 13].

In this section we briefly review the most effective empirical heuristics proposed in the literature and indicate the directions of the most recent research work in this area. As we are interested in parallel implementation of dissection algorithms, one subsection is dedicated in particular to Karp's [4] partitioning algorithms and another to review some parallel implementations of TSP heuristics.

2.1. Tour construction heuristics and local optimization

Considerable effort has been devoted to find algorithms which on average produce good tours for large problem instances. The best heuristics of this research stream proposed in the literature are composite heuristics in which the final tour is obtained in two steps: first a tour is constructed growing a fragment of a tour or a smaller tour by adding points according to an appropriate insertion procedure; then the solution is improved using some local optimization method. The most commonly used such modifications are 2- and 3-changes, which affect 2 or 3 edges respectively. By themselves, they give rise to the well-known 2-Opt and 3-Opt algorithms. In more complicated combinations, they give rise to the famous Lin–Kernighan algorithm [14] and provide the basic engines for most applications of other algorithmic techniques, like Simulated Annealing [15, 16], Genetic Algorithms [17] and Tabu Search [18, 19]. Two other effective insertion and post-optimization heuristics have recently been proposed: the CCAO [11] and the Genius [20] algorithm. The first exploits the property that in any optimal solution, vertices located on the convex hull of all vertices are visited in the order in which they appear on the convex hull boundary. The second has the particular feature of having the possibility of removing nodes during the post-optimization procedure and of reinserting them in a better position.

Most of these heuristics are very time consuming: considerable effort has recently been put to solve large problems using efficient data structures [2, 21–24].

2.2. Heuristic algorithms for the TSP: a parallel approach

Parallel algorithms may represent another way to solve large problem instances either in less CPU

time or improving the quality of the solution by using more "expensive" heuristics. Two general methods can be used to parallelize TSP heuristics: the first consists in the parallelization of the search of the next move to perform in the optimization process. This requires the partition of the set of feasible moves in subsets. Each of these subsets is assigned to one process which computes the best move on it. The best of these moves is then determined and applied to the current solution. This technique requires a lot of communication since synchronization is necessary at each step. It is therefore only worth applying it to problems in which the search of the best move is relatively complex and time consuming. The second parallelization technique can be used to perform several moves concurrently. This can be done by partitioning the problem into several subproblems. The global solution is then obtained by merging the sub-solutions. This method needs no communication and synchronization except for initialization and for merging the sub-solutions at the end. We can therefore expect a real efficiency of the parallel algorithm even when moves are simple. The problem with this kind of parallelism is that it strongly limits the movement possibilities: moves across different subproblems cannot be contemplated. This generally induces a loss of quality of the global solution.

To our knowledge only few attempts to parallelize TSP heuristics have been performed so far. Theoretical work to analyse the complexity of some TSP heuristics in the PRAM model of parallel computation have been done in [25]. Parallel techniques to implement TSP heuristics are presented in [26]. For the farthest insertion, the Christofides heuristic [27] and the 2-Opt and 3-Opt exchange heuristics it is shown how the quality of the solution is improved making several independent runs simultaneously and taking the best solution. It is also shown how 2-Opt and 3-Opt can be parallelized dividing pairs or, respectively, triplets of edges between several processors. A parallel tabu search algorithm and its implementation on a transputer network is proposed in [18]. In this paper it is shown how a tabu search framework can be used successfully to implement the two parallel techniques described above.

2.3. A heuristic proposed by Karp

As anticipated in the previous section, divide and conquer algorithms are a class of algorithms which have a natural parallel implementation. Karp [4] proposed partitioning algorithms for the approximate solution of the TSP. These algorithms divide the set of cities into small groups and construct an optimal tour through each group. These subtours are then patched to form a tour through all the cities. Karp has presented two partitioning schemes: an adaptive dissection scheme in which the location of the cities determine the dissection and a fixed dissection method in which all cells are congruent squares of the same size. For the first dissection algorithm, Karp has given an upper bound of the worst case error. In addition, for both algorithms, he has performed a probabilistic analysis of the quality of the solution, obtaining asymptotical bounds, based on the assumption that the points were drawn independently over the unit square. These bounds are calculated considering a larger set of feasible solutions, i.e. not only tours which visit all cities once and only once, but also spanning walks. A spanning walk is a connected graph with each node of even degree. Using spanning walks instead of tours does not make a real change in the problem, because it is always possible to transform a spanning walk in a tour with shorter length. In the following paragraph we describe in detail some results given by Karp for the adaptive dissection algorithm, which are the basis of our work.

2.3.1. Adaptive dissection method. This partitioning scheme is used to construct a spanning walk through N given cities. The algorithm divides the rectangle in which the N cities are distributed into $B = 2^k$ subrectangles (buckets) each containing at most t cities, where $k = \log_2\lceil N-1)/(t-1)\rceil$. An optimum tour is constructed through the cities in each subrectangle. The union of the 2^k optimal subtours forms a spanning walk through all N cities. The subdivision is done recursively in the following way: let Y be a rectangle containing m cities: the subdivision in two subrectangles is done in correspondence of the $\lceil m/2\rceil$th farthest city from the shorter side of the rectangle. This city belongs to the common boundary of the 2 subrectangles. For this partitioning scheme Karp gives the following results.

Let N be the number of cities in a rectangle X and t the maximum number of cities in each subrectangle in which X is divided. Then the worst case error defined as the difference between the

length of the walk W and the length of the optimal tour L_{opt} through the N cities is

$$W - L_{\text{opt}} \leqslant \frac{3}{2} \sum_{i=1}^{2k} \text{Per}(Y_i) \tag{1}$$

where $\text{Per}(Y_i)$ is the perimeter of the ith subrectangle.

Let a, b be the dimensions of the rectangle X; using the result of the *cutting game* given by Karp, we can give an upper bound for $\sum_{i=1}^{2k} \text{Per}(Y_i)$

$$\sum_{i=1}^{2k} \text{Per}(Y_i) \leqslant 2\, 2^{\frac{\alpha+\beta}{2}} \left(2^{\lceil \frac{k}{2} \rceil + \lfloor \frac{k}{2} \rfloor}\right) \tag{2}$$

Therefore we have

$$W - L_{\text{opt}} \leqslant \frac{3}{2} \left[2\, 2^{\frac{\alpha+\beta}{2}} \left(2^{\lceil \frac{k}{2} \rceil + \lfloor \frac{k}{2} \rfloor}\right)\right] \tag{3}$$

where $2^{\alpha} = a$ and $2^{\beta} = b$.
If $a = b$ then

$$W - L_{\text{opt}} \leqslant \frac{3}{2} \left[2\, a \left(2^{\lceil \frac{k}{2} \rceil + \lfloor \frac{k}{2} \rfloor}\right)\right] \tag{4}$$

This equation can be explicitated in the following way

$$\text{for } k \text{ even} \qquad W - L_{\text{opt}} \leqslant 3\, a\, 2^{\frac{k}{2}+1} \tag{5}$$

$$\text{for } k \text{ odd} \qquad W - L_{\text{opt}} \leqslant 3\, a\, \frac{3}{\sqrt{2}} 2^{\frac{k}{2}} \tag{6}$$

If $\log_2(N-1)/(t-1)$ is an integer, this equation can be written as a function of N and t

$$\text{for } k \text{ even} \qquad W - L_{\text{opt}} \leqslant 3\, a\, 2\sqrt{\frac{N-1}{t-1}} \tag{7}$$

$$\text{for } k \text{ odd} \qquad W - L_{\text{opt}} \leqslant 3\, a\, \frac{3}{\sqrt{2}} \sqrt{\frac{N-1}{t-1}} \tag{8}$$

These results are independent from the points distribution.
The results which we consider now are asymptotical results and hold only for uniform distributions. Let N be the number of cities distributed randomly in a rectangle X of area $v(X)$ following a uniform distribution. Let $T_N(X)$ be a random variable which denotes the length of an optimal tour through the N cities in X. A theorem due to [28] shows that there exists a positive constant β (independent from X) such that $\forall \epsilon > 0$

$$\text{Prob}\left\{ \lim_{N \to \infty} \left(\frac{T_N(X)}{\sqrt{Nv(X)} - \beta) > \epsilon} \right\} = 0 \right. \tag{9}$$

Combining this result with equation (1), Karp has shown that the relative error between a spanning walk and an optimum tour can be estimated in the following way

$$\forall \epsilon > 0 \quad \text{Prob}\left\{ \lim_{N \to \infty} \left(\frac{W - T_N(X)}{T_n(X)} - \frac{C}{\sqrt{t}} \right) > \epsilon \right\} = 0 \tag{10}$$

where C is a constant > 0.

The last Karp's result we want to discuss here is the following: Let $T_t(X)$ be the length of an optimal tour through $t (< N)$ cities in the rectangle $X[a, b]$ with $ab = 1$. We want to compare the average length $T_t(X)$ with the average length of the optimal tour for all N cities.

Let $\beta_X(t)$ be defined as

$$\beta_X(t) = \frac{E(T_t(X))}{\sqrt{t}} \tag{11}$$

As a consequence of equation (10) it follows that

$$\lim_{t\to\infty} \beta_X(t) = \beta \tag{12}$$

Karp has given the following bound on the rate of convergence of $\beta_X(t)$ to β

$$\forall\, t \qquad \beta_X(t) - \beta \leqslant \frac{6(a+b)}{\sqrt{t}} \tag{13}$$

Equation (12) says that the length of a tour through t cities tends to the length of the optimal tour when t becomes larger. Equation (13) indicates how fast this is done.

Karp has shown that most of these probabilistic results can be used with some modifications also for other dissection methods.

3. IMPLEMENTATION RESULTS

In this section we first present the architecture on which we have implemented our programs. Then we describe our implementations of dissection algorithms and analyse the performances which we have obtained.

3.1. The MUSIC system

The architecture on which we have implemented our programs is MUSIC (MUlti Signal processor system with Intelligent Communication) a parallel computer developed at the ETH Zurich [29, 30]. MUSIC is a distributed memory array processor system with a fixed connection network, used as Single Program Multiple Data machine (SPMD). In this model of computation the parallelization is done in the data space: that is all processors run the same program, with different input data. However, unlike single instruction multiple data machines, each processor might execute a different part of the code depending on the processed data.

A single processing element is a Digital Signal Processor (DSP) 96002 from Motorola with 60 MFlops peak performance, program and data memory and a fast communication interface. A system with 60 Processors with 3.6 GFlops peak performance is operational. MUSIC is driven by a host-computer; the sequential part of applications runs on the host-side and make function calls to the DSP side. The exchange of data between processors follows a broadcast paradigm: data produced by each processor are broadcast to all processors through the communication network and any processor can consume a specified part of these data. This communication paradigm is implemented using a barrier synchronization. This means that even if communication can start during the processing phase as soon as data are available, all processors must wait until the current communication cycle is finished before beginning a new cycle. A similar scheme is used to communicate data between host and DSPs.

3.2. Parallel implementation of dissection algorithms

We have implemented two dissection algorithms considering the following three distinct phases [31]:

We subdivide the plane recursively till B subregions with a determined number of points t are created. This is a data preprocessing step which is done sequentially on the host computer. It creates a binary tree in which the leaves contain the points of the subrectangles. These leaves are then sent to the DSPs. The way in which the subdivision is done characterize the dissection method

- —Our first implementation corresponds to the adaptive dissection algorithm proposed by Karp. Let N be the number of cities, t the number of cities in each subrectangle and $B = 2^k$ the number of subrectangles, then $N = B(t-1)+1$.
- —Our second implementation is the following: let Y be a rectangle containing m cities and oriented so that its longer sides are horizontal. Then Y is subdivided into two rectangles by a vertical cut equidistant between the $\lfloor m/2 \rfloor$th farthest city from the left and the $(\lceil m/2 \rceil + 1)$th from the left. Let N be the number of cities, t the number of cities in each subrectangle and $B = 2^k$ the number of subrectangles, then $N = Bt$.

We calculate the optimum tours in parallel in each subrectangle. In the first algorithm the tours form a spanning walk; in the second the tours do not have points in common. The subtours are

Table 1. Average time and standard deviation to calculate B optimal subtours of t points. The average time and the standard deviation time are calculated over the B tours

B	t	Average time (s)	Standard deviation (s)
16	4	0.057	0.020
16	8	0.162	0.083
16	16	1.074	0.977
16	32	63.522	88.476

distributed equally between the processors. This distribution is static, i.e. it is defined at compilation time.

We merge the subtours together to obtain the final tour. This is done connecting adjacent tours together at each level of the tree, starting from the leaves and proceeding toward the root. This phase can be done either sequentially on the host computer or in parallel on the DSP side redistributing the tours of each level of the tree between the processors. Our two dissection algorithms have different merging procedures

—In the first algorithm the tours are merged two by two, until a TSP tour is created. The merging is performed considering only the four edges which have the cutting point in common and minimizing the length of the new tour.

—In the second merging procedure the partial tours do not have a common point. The new tour is obtained eliminating one edge of each subtours and reconnecting them with two appropriate new edges. The edge to eliminate and the way in which the connection has to be done is chosen so that the length of the new tour is minimized.

3.3. Computational results: time analysis

The total time for the computation of a global tour is given by the sum of the times needed for each phase described above plus the time necessary for communication and initialization of the network of DSPs from the host computer. Limitations of the time improvement are given by

(1) Plane division which is done sequentially. This can be implemented in $O(N\log N)$ average time.

(2) Network initialization and communication time. The network initialization depends mostly on the load of the host computer and on how many users try to access the system. It is therefore extremely difficult to have meaningful measurements. We have observed network initialization times between 12 and 17 s.

(3) Merging: the first merging procedure depends only from the number of merging steps $2^k - 1$. The second merging procedure has $O(n^2)$ complexity, where n is the number of points of the tours which are merged.

(4) Unbalanced work division between processors: optimal tours are calculated with a Branch and Bound procedure [32]. The time needed to calculate a tour is extremely data dependent. In Table 1 we present the average time and the standard deviation needed to

Table 2. Time (in seconds) and speed-up (sequential time/parallel time) to calculate B optimal tours of t points. N is total number of points and p is the number of processors

	$p=1$	$p=2$	$p=4$	$p=8$	$p=16$
$B=64$	8517	4886	4030	3810	2842
$t=32$	—	1.7	2.1	2.2	3.0
$N=1985$					
$B=128$	11337	5971	4544	3659	3346
$t=32$	—	1.9	2.5	3.1	3.4
$N=3969$					
$B=256$	9486	4987	2822	1744	1312
$t=32$	—	1.9	3.4	5.4	7.2
$N=7937$					

calculate B subtours of different sizes. Each problem instance is generated randomly and independently. We can note that the standard deviation of the computation time increases with the size of the subtours (relatively to the average time). It is in fact always possible to find a problem instance which takes exponential time to be solved. This means that because the task are allocated statically to the processors, one processor can eventually have much more work to do than the others. In this case, the computation time needed by this processor will be the total time needed for the calculation of all the subtours. This problem does not arise if, instead of calculating optimal subtours, we calculate approximate subtours with a heuristic running in polynomial time (see Section 5).

In Table 2 we present the time improvement which can be obtained calculating several subtours in parallel using an increasing number of processors. We have generated three set of N points randomly. For each of them we give the time (in seconds) needed for the calculation of B optimal subtour and the speed-up (sequential/parallel time). The first two sets are generated in the square [100,100] and the last in the square [1000,1000].

4. AVERAGE QUALITY OF THE SOLUTION OBTAINED WITH DISSECTION ALGORITHMS

In this section we analyse experimentally the average quality of the solution for point sets distributed randomly in the plane and compare our computational results with the theoretical results given by Karp for the worst case error.

Karp does not present experimental results for the TSP; he instead analyses the effectiveness of partitioning algorithms on the minimum spanning tree problem using an algorithm similar to our second implementation scheme. To our knowledge there is no experimental work based on Karp's results. In the following subsection we use our first dissection algorithm, corresponding to Karp's adaptive dissection algorithm.

In all the experiments we have performed the total number of cities and the plane divisions are determined in order to have the same number of cities in each bucket.

4.1. Quality of the solution obtained with the adaptive dissection method

Our first goal is to compare the solution obtained with the adaptive dissection algorithm with the optimal tour.

In Table 3 we present ten problem instances. The points of each problem are distributed randomly in the unit square. We have used the standard random number generator rand( ) available in ANSI-C and in order to have the possibility to reproduce these results we have indicated for each experiment the seed srand( ) of the random number generator. Each set of N points is divided with a number of recursive cuts k ranging from one to five. For each problem we give the length W of the spanning walk and the length L of the tour obtained after merging. We calculate also the length L_{opt} of the optimal tours.

With these data we calculate the average absolute error and the average relative error for the spanning walks and for the tours and we perform a linear regression analysis based on the least square method. According with equation (4) we have found that the average absolute error has the following behavior

$$\overline{E_w} = \overline{W - L_{\text{opt}}} \approx \alpha_w(2^{k/2} - 1) \tag{14}$$

and respectively

$$\overline{E} = \overline{L - L_{\text{opt}}} \approx \alpha(2^{k/2} - 1) \tag{15}$$

This equation holds when each bucket has the same number of points. Therefore equation (15) [and similarly equation (14)] can also be written as

$$\overline{E} \approx \alpha\left(\sqrt{\frac{N-1}{t-1}} - 1\right) \tag{16}$$

We have estimated $\alpha_w \approx 1.271$ (intercept $= -0.197$) and $\alpha \approx 0.547$ (intercept $= 0.130$). This

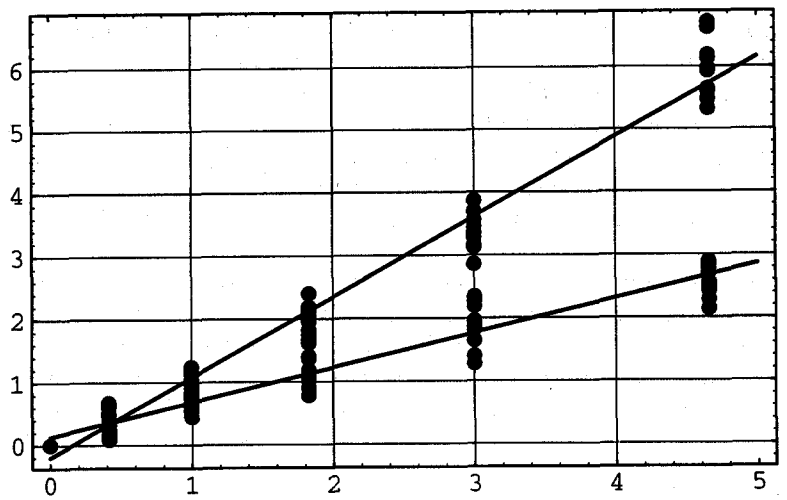

Fig. 1. Walk-length errors $W - L_{opt}$ (top) and tour-length errors $L - L_{opt}$ (bottom) calculated as a function of $2^{k/2} - 1$ using the adaptive dissection algorithm. k is the number of dissection steps (see Table 3).

Table 3. Spanning walks length W and tours length L for different problem instances of N cities and different number of plane cuts. B is the number of buckets and t the number of cities per bucket. The plane division is performed with the adaptive dissection method. In each bucket optimal subsolutions are calculated. The global optimum tour is shown in the first row

1st diss alg	Problem 1 seed = 7 10^2	Problem 2 seed = 17 10^2	Problem 3 seed = 27 10^2	Problem 4 seed = 37 10^2	Problem 5 seed = 47 10^2
$N = 129$ $t = 129$ $B = 1$	$L_{opt} = 869.57$	$L_{opt} = 908.34$	$L_{opt} = 909.18$	$L_{opt} = 882.70$	$L_{opt} = 860.27$
$N = 129$ $t = 65$ $B = 2$	$W = 917.11$ $L = 906.63$	$W = 946.78$ $L = 945.56$	$W = 957.14$ $L = 949.38$	$W = 944.09$ $L = 932.80$	$W = 883.43$ $L = 876.37$
$N = 129$ $t = 33$ $B = 4$	$W = 953.12$ $L = 925.54$	$W = 1013.08$ $L = 992.86$	$W = 985.82$ $L = 969.58$	$W = 1007.00$ $L = 965.90$	$W = 958.47$ $L = 939.53$
$N = 129$ $t = 17$ $B = 8$	$W = 1031.07$ $L = 960.02$	$W = 1104.10$ $L = 1048.73$	$W = 1106.12$ $L = 1027.52$	$W = 1122.98$ $L = 1050.16$	$W = 1079.37$ $L = 1021.89$
$N = 129$ $t = 9$ $B = 16$	$W = 1239.18$ $L = 1104.08$	$W = 1237.14$ $L = 1099.56$	$W = 1243.53$ $L = 1073.85$	$W = 1269.62$ $L = 1111.27$	$W = 1197.16$ $L = 1042.67$
$N = 129$ $t = 5$ $B = 32$	$W = 1417.59$ $L = 1154.38$	$W = 1469.30$ $L = 1154.26$	1506.66 $L = 1137.86$	$W = 1546.16$ $L = 1160.67$	$W = 1453.41$ $L = 1130.39$
	Problem 6 seed = 57 10^2	Problem 7 seed = 67 10^2	Problem 8 seed = 77 10^2	Problem 9 seed = 87 10^2	Problem 10 seed = 97 10^2
$N = 129$ $t = 129$ $B = 1$	$L_{opt} = 869.91$	$L_{opt} = 878.63$	$L_{opt} = 880.34$	$L_{opt} = 877.93$	$L_{opt} = 909.72$
$N = 129$ $t = 65$ $B = 2$	$W = 937.29$ $L = 936.17$	$W = 915.82$ $L = 903.00$	$W = 945.08$ $L = 944.18$	$W = 891.63$ $L = 886.49$	$W = 950.36$ $L = 941.61$
$N = 129$ $t = 33$ $B = 4$	$W = 968.65$ $L = 948.57$	$W = 955.99$ $L = 932.75$	$W = 994.08$ $L = 953.95$	$W = 942.47$ $L = 922.47$	$W = 1001.41$ $L = 978.13$
$N = 129$ $t = 17$ $B = 8$	$W = 1044.57$ $L = 972.37$	$W = 1060.77$ $L = 956.78$	$W = 1093.85$ $L = 1019.60$	$W = 1045.01$ $L = 987.67$	$W = 1114.21$ $L = 1046.79$
$N = 129$ $t = 9$ $B = 16$	$W = 1215.92$ $L = 1065.69$	$W = 1191.48$ $L = 1006.92$	$W = 1196.73$ $L = 1067.85$	$W = 1163.96$ $L = 1018.01$	$W = 1265.71$ $L = 1129.07$
$N = 129$ $t = 5$ $B = 32$	$W = 1434.40$ $L = 1121.65$	$W = 1492.90$ $L = 1167.38$	$W = 1498.98$ $L = 1136.89$	$W = 1410.61$ $L = 1091.83$	$W = 1582.02$ $L = 1152.46$

Table 4. Subtours sum length W and tours length L for different problem instances of N cities and different number of plane cuts. B is the number of buckets and t the number of cities per bucket. The plane division is performed with the second dissection method. In each bucket optimal subsolutions are calculated. The global optimum tour is shown in the first row

2nd diss alg	Problem 1 seed = 7 10^2	Problem 2 seed = 17 10^2	Problem 3 seed = 27 10^2	Problem 4 seed = 37 10^2	Problem 5 seed = 47 10^2
$N = 128$ $t = 128$ $B = 1$	$L_{opt} = 867.09$	$L_{opt} = 907.61$	$L_{opt} = 907.37$	$L_{opt} = 881.37$	$L_{opt} = 855.30$
$N = 128$ $t = 64$ $B = 2$	$W = 912.36$ $L = 904.15$	$W = 951.37$ $L = 944.32$	$W = 945.29$ $L = 939.48$	$W - 941.80$ $L = 922.54$	$W = 890.93$ $L = 887.41$
$N = 128$ $t = 32$ $B = 4$	$W = 937.82$ $L = 919.07$	$W = 989.28$ $L = 978.19$	$W = 969.56$ $L = 953.04$	$W = 995.03$ $L = 958.98.$	$W = 915.45$ $L = 896.34$
$N = 128$ $t = 16$ $B = 8$	$W = 983.73$ $L = 945.73$	$W = 1065.02$ $L = 1021.43$	$W = 1036.97$ $L = 996.66$	$W = 1093.95$ $L = 1032.39$	$W = 995.66$ $L = 958.65$
$N = 128$ $t = 8$ $B = 16$	$W = 1118.46$ $L = 1025.72$	$W = 1144.20$ $L = 1041.03$	$W = 1113.07$ $L = 1013.46$	$W = 1172.19$ $L = 1065.74$	$W = 1035.38$ $L = 983.13$
$N = 128$ $t = 4$ $B = 32$	$W = 1183.90$ $L = 1034.55$	$W = 1208.92$ $L = 1037.62$	1257.22 $L = 1049.51$	$W = 1282.67$ $L = 1083.14$	$W = 1132.27$ $L = 1028.03$
	Problem 6 seed = 57 10^2	Problem 7 seed = 67 10^2	Problem 8 seed = 77 10^2	Problem 9 seed = 87 10^2	Problem 10 seed = 97 10^2
$N = 128$ $t = 128$ $B = 1$	$L_{opt} = 867.89$	$L_{opt} = 878.20$	$L_{opt} = 878.45$	$L_{opt} = 866.79$	$L_{opt} = 989.32$
$N = 128$ $t = 64$ $B = 2$	$W = 932.78$ $L = 928.86$	$W = 901.52$ $L = 900.57$	$W = 911.32$ $L - 902.47$	$W = 878.42$ $L = 867.42$	$W = 935.77$ $L = 912.02$
$N = 128$ $t = 32$ $B = 4$	$W = 953.67$ $L = 939.24$	$W = 927.36$ $L = 906.06$	$W = 959.45$ $L = 926.24$	$W = 917.23$ $L = 912.56$	$W = 979.48$ $L = 956.27$
$N = 128$ $t = 16$ $B = 8$	$W = 1017.78$ $L = 942.68$	$W = 974.74$ $L = 935.45$	$W = 1058.35$ $L = 982.45$	$W = 1031.62$ $L = 979.15$	$W = 1061.26$ $L = 1004.71$
$N = 128$ $t = 8$ $B = 16$	$W = 1149.60$ $L = 1000.35$	$W = 1069.79$ $L = 986.06$	$W = 1151.130$ $L = 1011.13$	$W = 1072.20$ $L = 986.85$	$W = 1172.50$ $L = 1073.69$
$N = 128$ $t = 4$ $B = 32$	$W = 1214.13$ $L = 1063.46$	$W = 1126.78$ $L = 984.50$	$W = 1216.01$ $L = 1019.17$	$W = 1134.90$ $L = 1021.27$	$W = 1290.80$ $L = 1078.52$

means that the average absolute error is proportional to $\sum_{i=1}^{2k} \mathrm{Per}(Y_i) - \mathrm{Per}(X)$ with a much smaller constant of proportionality than that given in equation (4) which holds for the worst case error.

In Fig. 1 are presented the values of the absolute errors of Table 3 and the corresponding regression lines.

4.2. Quality of the solution obtained with the second dissection method

In this section we perform similar experiments using our second dissection method (see Table 4). We calculate the length W of the sum of the subtours (which do not form a spanning walk) and the length L of the merged tours. We calculate also the regression line for W and L finding respectively $\alpha_w = 0.707$ (intercept = 0.0932) and $\alpha = 0.344$ (intercept = 0.174). In Fig. 2 are compared the average values of the tour-length errors obtained with the two dissection methods.

Using the second dissection algorithm the average quality of the solution is better. There is, however, a stronger second order term due to the additional links between subtours. This can be seen analysing the residuals $R_{ji} = E_{ji} - \hat{E}_j$ for the two dissection methods. $\hat{E}_j$ is the value calculated with the regression equations for j plane-cuts and E_{ji} is the value observed in the ith experiment with j plane-cuts.

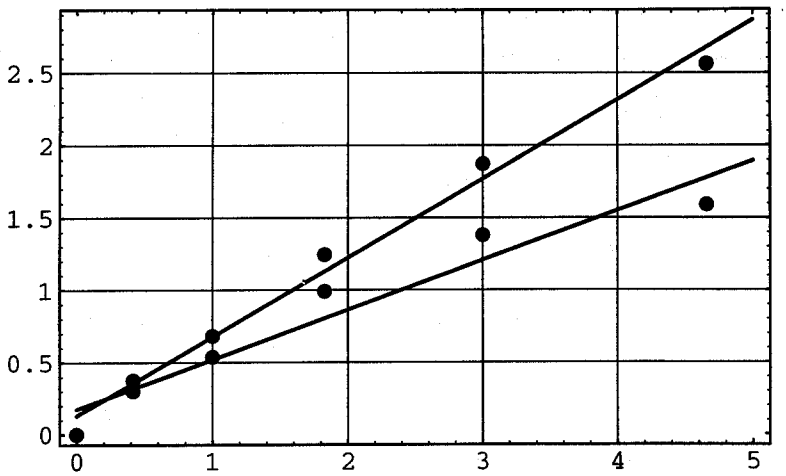

Fig. 2. Average length errors $L - L_{\text{opt}}$ calculated using the first dissection algorithm (top) and the second dissection algorithm (bottom) as a function of $2^{k/2} - 1$.

5. PARALLEL IMPLEMENTATIONS OF DIVIDE AND CONQUER STRATEGIES FOR LARGE TSP INSTANCES: AN EMPIRICAL EVALUATION OF THE ASYMPTOTICAL EXPECTED PERFORMANCES

As we have shown in Section 3, our dissection algorithms have a very natural parallel implementation which may allow us to use them to solve large instances of the TSP. In this case even at the leaf nodes fairly large TSPs have to be solved, and this can only be done using approximation algorithms.

We are interested in the quality of the global solution of large TSP instances, obtained with either optimal or approximate subtours. This analysis can be done statistically: it is based on the fact that optimal as well as approximate tours of points distributed randomly in the plane present a regular behavior for N large enough.

The length of the optimal tour can be estimated by using equation (9). This equation can be interpreted in the following way: for a very high number N of cities distributed uniformly in the unit square X, the optimal length of a tour tends to be proportional to $\sqrt{N}$ with very high probability. The constant of proportionality does not depend from the problem instance. That is

$$\text{for } N \gg 1 \qquad L_{\text{opt}} \approx \beta\sqrt{N} \tag{17}$$

with β independent from X. The value of β has been estimated in [28] to be ≈ 0.749. More recently, other experiments done in [33] have shown that for instances up to 100,000 points $\beta \approx 0.713$.

It has been shown in [34] that a good approximate algorithm for the TSP has a similar behavior as N gets larger. That is, the expected tour length produced by a heuristic can be expressed as

$$\frac{E[L]}{\sqrt{N}} = \beta_e + \frac{\gamma_e(N)}{\sqrt{N}} \tag{18}$$

where $\gamma_e(N)/\sqrt{N} \to 0$ as $N \to \infty$ and the value of β_e depends on the heuristic. This means that the average tour length of a heuristic can be fitted with a linear regression line as a function of $\sqrt{N}$

$$\overline{L_{N_j}} = \beta_e\sqrt{N_j} + \gamma_e + E_j \tag{19}$$

where N_j is the number of points on the jth experiment, $\overline{L}_{N_j}$ is the average length of the tour through N_j cities and E_j is the random error.

In Hong and Huang [34] four heuristics have been tested. To check the adequacy of the fitted regression model $Y = BX$ the coefficient of determination $R^2 = \sum(\hat{Y}_i - \bar{Y})/\sum(Y_i - \bar{Y})$ has been used, where Y_i is an actual value of Y and $\hat{Y}_i$ is the estimate of Y_i obtained from the regression equation. The best results have been obtained using the 3-Opt algorithm. The values of β_e and γ_e which have been calculated are: $\hat{\beta}_e = 0.7425$ $\hat{\gamma}_e = 0.5501$ with $R^2 = 0.9997$ using problems ranging from 25 to 200 nodes ($\hat{\beta}_e$ is the estimate of β_e; in the following we will not distinguish explicitly

between the two values). Using the Nearest Neighbor heuristic on problem instances ranging from 25 to 1000 points β_e has been estimated = 0.8952 and $\gamma_e = 0.7429$ with $R^2 = 0.9995$.

In Fiechter [18], using a Tabu Search algorithm for problem instances from 500 to 100,000 nodes, β_e has been estimated = 0.7298 and $\gamma_e = 0.43$ with $R^2 = 0.9999$.

Using dissection algorithms the ratio $L/\sqrt{N}$ depends from the quality and from the number of points t used to construct the partial solutions. From equation (15) and (17), we suppose that the expected value $E[L]/\sqrt{N}$ follows the following formula

$$\frac{E[L]}{\sqrt{N}} = \beta_k + \frac{\alpha_k}{\sqrt{t}} + \frac{\gamma_k(N,t)}{\sqrt{N}} \tag{20}$$

where α_k and β_k are two constants independent from X and $\gamma_k(N,t)/\sqrt{N} \to 0$ as $N, t \to \infty$. β_k and α_k depend from the quality of the subtours and from the criterion used to cut the plane. If optimal subtours are calculated β_k corresponds to the value of β in equation (17). If non optimal subtours are calculated it is the asymptotical value β_e of equation (18). α_k gives the rate of convergence to the global solution [see equation (13)].

The values of α_k, β_k, γ_k can be estimated performing a multiple regression

$$\overline{L_{N_j,t_i}} = \beta_k\sqrt{N_j} + \alpha_k\frac{\sqrt{N_j}}{\sqrt{t_i}} + \gamma_k + E_{ij} \tag{21}$$

where $\overline{L_{N_j,t_i}}$ is the average length of the tour through N_j cities obtained merging subtours of t_i cities. To limit the number of experiments, however, we have performed a simple regression maintaining fixed $\sqrt{t}$ or $\sqrt{N}/t$ respectively.

5.1. Quality of the solution obtained with optimal subtours

In this subsection we evaluate the quality of the solution by calculating optimum subtours and by using our two different dissection methods. All the experiments have been done using the standard drand48() C random number generator under UNIX which is based on the congruential method [35].

5.1.1. First dissection method. To check equation (20) we perform two series of experiments.

First we maintain fixed the number of cities t per bucket and vary the total number of cities N. As expected the quality of the solution remains constant for N large enough. This result means that, once we have fixed t, after a certain number of merging steps the quality of the solution does not depend from the number of merging steps which we perform.

Secondly we analyze the quality of the solution varying the number t of points per bucket (see Table 5).

Table 5. Quality of the solution for different set of N cities with constant number of B buckets. The first dissection method is used and optimal subtours are calculated. W is the length of the spanning walk and L is the length of the global tour. The average values and the standard deviations are calculated over 10 runs

N	B	t	$\overline{W/\sqrt{N}}$	$\sigma(W/\sqrt{N})$ 10^3	$\overline{L/\sqrt{N}}$	$\sigma(L/\sqrt{N})$ 10^3
1025	256	5	1.303	4.683	1.012	3.475
1537	256	7	1.169	7.919	0.977	7.703
2049	256	9	1.086	7.783	0.946	6.246
2561	256	11	1.030	5.909	0.913	7.068
3073	256	13	0.990	5.139	0.898	5.054
3585	256	15	0.962	3.534	0.885	5.355
4097	256	17	0.944	2.900	0.875	3.300
4609	256	19	0.927	4.174	0.867	4.500
5121	256	21	0.915	3.174	0.861	2.379
5633	256	23	0.901	5.201	0.852	6.322
6145	256	25	0.893	3.256	0.848	2.629
6657	256	27	0.884	3.786	0.841	3.000
N	B	t	$\overline{W/\sqrt{N}}$	$\sigma(W/\sqrt{N})$ 10^3	$\overline{L/\sqrt{N}}$	$\sigma(L/\sqrt{N})$ 10^3
4097	1024	5	1.301	5.242	1.013	6.340
6145	1024	7	1.166	4.140	0.974	3.282
8193	1024	9	1.084	2.958	0.938	4.610
10241	1024	11	1.030	1.680	0.915	2.410
12289	1024	13	0.994	3.312	0.899	3.675
14337	1024	15	0.966	2.480	0.886	2.488
16385	1024	17	0.945	1.736	0.876	2.272

Table 6. Quality of the solution for different set of N cities with constant number of B bucket. The second dissection method is used and optimal subtours of t cities are calculated. W is the sum of the length of the partial solutions and L is the length of the global tour. The average values and the standard deviations are calculated over 10 runs

N	B	t	$\overline{W/\sqrt{N}}$	$\frac{\sigma(W/\sqrt{N})}{10^3}$	$\overline{L/\sqrt{N}}$	$\frac{\sigma(L/\sqrt{N})}{10^3}$
1024	256	4	1.051	13.509	0.902	6.946
1536	256	6	1.025	13.159	0.898	5.800
2048	256	8	0.986	7.670	0.883	6.389
2560	256	10	0.952	6.455	0.867	5.047
3072	256	12	0.927	5.379	0.855	5.351
3584	256	14	0.911	4.360	0.848	3.593
4096	256	16	0.892	3.875	0.838	4.792
4608	256	18	0.883	4.645	0.834	3.484
5120	256	20	0.873	2.783	0.829	3.112
5632	256	22	0.867	3.222	0.827	2.360
6144	256	24	0.859	3.348	0.822	3.542
6656	256	26	0.853	3.268	0.818	2.939

Making a linear regression we have found that the length of the tour is given by

$$\frac{\overline{L}}{\sqrt{N}} \approx \frac{0.588}{\sqrt{t-1}} + 0.729 \tag{22}$$

that is $\alpha_k \approx 0.588$ and $\beta_k + \gamma_k/\sqrt{N} \approx 0.729$.

It should be noted that the behavior is not linear on the whole range of t. Our value of β is close to the values given in the literature. The small value of α tells us that the rate of convergence to the optimum tour is much faster than that foreseen in the upper bound of equation (13).

5.1.2. Second dissection algorithm. In this section we analyse the quality of the solution using our second dissection method. We have done similar experiments as in the previous section (see Table 6). We have found: $\alpha \approx 0.309$ and $\beta + \gamma/\sqrt{N} \approx 0.763$. It is interesting to note that a larger number of cities per bucket is necessary to obtain a stable behavior. As expected the rate of convergence to the optimum is faster with this dissection method, but the asymptotical values of the tour lengths are similar and tend to the values given by equation (17). In Fig. 3 are compared the results obtained with the two dissection methods.

5.2 Quality of the solution obtained with approximate subtours

In this subsection instead of calculating optimal subsolutions, we calculate approximate tours in each subregion using the Nearest Neighbor and the 2-Opt heuristics. To divide the plane we use the adaptive algorithm. We assume that the average value of $L/\sqrt{N}$ can be fitted with a regression line as in equation (21), but with different values of the constants.

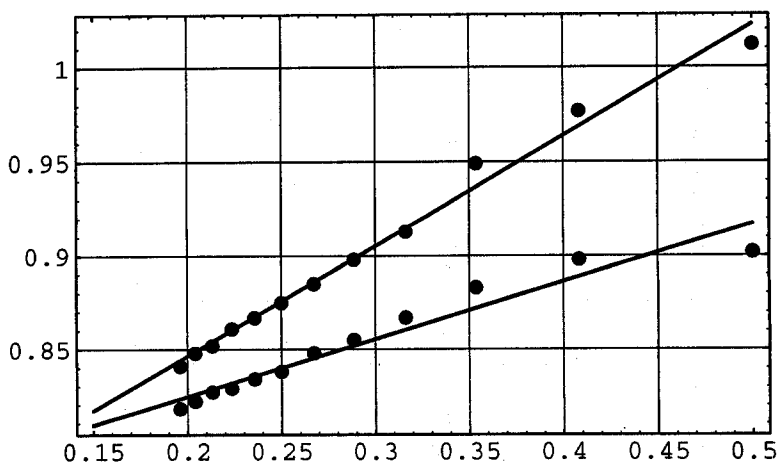

Fig. 3. Quality of the solution L/sqrt(N) as a function of 1/sqrt(t) obtained by calculating optimal subtours and by using the first (top) and second dissection algorithm (bottom).

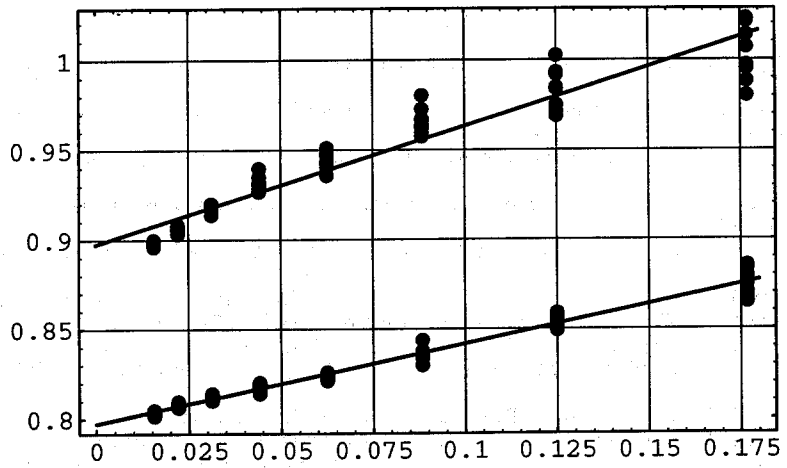

Fig. 4. Quality of the solution $L/\text{sqrt}(N)$ as a function of $1/\text{sqrt}(t-1)$ obtained by calculating Nearest Neighbor (top) and 2-Opt (bottom) subtours and by using the first dissection algorithm.

First we have evaluated the asymptotic value of β_e of these heuristics fitting the values of Nearest Neighbor and 2-Opt tours for different problem instances [see equation (19)].

- For the Nearest Neighbor we have found $\beta_e = 0.878$ and $\gamma_e = 1.185$. The number of points N ranges between 513 and 7681: $N = 128 * (m * 4) + 1$ with $m = 1..15$.
  We have then evaluated the values of β_k and α_k [equation (21)] using the first dissection algorithm. We have found $\beta_k + \gamma_k/\sqrt{N} = 0.897$ and $\alpha_k = 0.658$.
- For the 2-Opt we have found $\beta_e = 0.795$, $\gamma_e = 0.527$. The constants which characterize the dissection algorithm are $\beta_k + \gamma_k/\sqrt{N} = 0.797$ and $\alpha_k = 0.443$.

In Fig. 4 are presented the results for the two heuristics. We can note that the asymptotical values β_e of the heuristics are close to the values β_k obtained with the corresponding dissection algorithms and that the rate of convergence α_k have similar behaviors.

6. CONCLUSIONS

In this paper we have analysed experimentally the quality of the solution which can be obtained using dissection algorithms in a divide a conquer approach suitable for parallel computation. Our analysis has followed two main directions

(1) We have checked the average quality of the solution trying to extend Karp's results holding for the worst case analysis. In particular we have found that, as in the worst case, the average error between the tour lengths and the optimal tour is proportional to the sum of the perimeter of the regions in which the plane is divided, but with a much smaller constant of proportionality.
(2) We have performed an asymptotical analysis showing that the length of the tours obtained with dissection algorithms can be estimated with a model linear in $1/\sqrt{t}$. We have also seen that the global tour tends to the asymptotical quality of the subtours and that the rate of convergence depends from the dissection algorithm.

When using dissection algorithms, a tradeoff has therefore to be achieved between the global quality of the solution and the computational time, playing with the number of dissection steps, the quality of the subtours and the type of the dissection algorithm.

Acknowledgements—This paper is the result of a collaboration with the group of Professor J. Nievergelt at the ETH–Zurich. I acknowledge all his group for the support I received; in particular I am grateful to A. Brüngger who made available his Branch and Bound code to calculate optimal TSP tours. I would also like to thank B. Bäumle from the MUSIC group and Dr M. Mächler of the ETH Seminar für Statistik group for his useful comments about statistical models. Last but not least, I would like to warmly acknowledge Professor W. Ukovich from the University of Trieste for having read and carefully corrected this paper.

REFERENCES

1. E. L. Lawler, J. K. Lenstra, A. H. G. Rinnoy Kan and D. B. Shmoys, *The Traveling Salesman Problem.* John Wiley, New York (1985).
2. D. S. Johnson, Local optimization and the traveling salesman problem. In *Proc. 11th Colloquim Automata, Languages and Programming*, Springer, Berlin, pp. 446–461 (1990).
3. G. Laporte, The traveling salesman problem: an overview of exact and approximate algorithms. *Eur. J. Op. Res.* **59**, 231–247 (1992).
4. R. M. Karp, Probabilistic analysis of partitioning algorithms for the traveling salesman problem in the plane. *Math. Ops Res.* **2**, 209–224 (1977).
5. M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness.* Freeman (1979).
6. B. Korte, Applications of combinatorial optimization. In *13th Int. Mathematical Programming Symp.* (1988).
7. R. G. Bland and D. F. Shallcross, Large traveling salesman problems arising from experiments in X-ray crystallography: a preliminary report on computation. *Ops Res. Lett.* **8**, 125–128 (1989).
8. J. D. Litke, An improved solution to the traveling salesman problem with thousands of nodes. *ACM Commun.* **27**, 1227–1236 (1984).
9. D. S. Johnson and C. H. Papadimitriou, Performance guarantee for heuristics. In *The Traveling Salesman Problem* (Edited by E. L. Lawler, J. K. Lenstra, A. H. G. Rinnoy Kan and D. B. Shmoys), John Wiley, New York (1985).
10. R. M. Karp and J. M. Steele, Probabilistic analysis of heuristics. In *The Traveling Salesman Problem* (Edited by E. L. Lawler, J. K. Lenstra, A. H. G. Rinnoy Kan and D. B. Shmoys), John Wiley, New York (1985).
11. B. L. Golden and Jr Steward, Empirical analysis of heuristics. In *The Traveling Salesman Problem* (Edited by E. L. Lawler, J. K. Lenstra, A. H. G. Rinnoy Kan and D. B. Shmoys), John Wiley, New York (1985).
12. F. Glover and A. P. Punnen, The traveling salesman problem: new solvable cases and linkages with the development of approximation algorithms. Technical report, Graduate School of Business, University of Colorado, Boulder (1994).
13. A. P. Punnen and F. Glover, TSP revisited: new heuristics with combinatorial leverage of exponential power. In *ORSA/TIMS–Detroit* (1994).
14. S. Lin and B. W. Kernighan, An effective heuristic algorithm for the traveling salesman problem. *Ops Res.* **21**, 498–516 (1973).
15. V. Cerny, A thermodynamical approach to the traveling salesman problem: an efficient simulation. *J. Optimization Theory Applications* **45**, 41–51 (1985).
16. S. Kirkpatrick, C. D. Gelatt and M. P. Vecchi, Optimization by simulated annealing. *Science* **220**, 671–680 (1983).
17. J. Grefenstette, R. Gopal, B. Rosmaita and D. Van Gucht, Genetic algorithms for the traveling salesman problem. In *Proc. Int. Conf. Genetic Algorithms Applications* (1985).
18. C-N. Fiechter, A parallel tabu search algorithm for large traveling salesman problems. Technical Report ORWP 90/1, DMA–Ecole polytechnique Fédérale de Lausanne, Lausanne, Switzerland (1990).
19. F. Glover. Tabu search—part 1. *J. Comput.* **1**, 190–200 (1989).
20. M. Gendreau, A. Hertz and G. Laporte, New insertion and post-optimization procedures for the traveling salesman problem. *Ops Res.* **40**, 1086–1094 (1992).
21. J. L. Bentley, K–d trees for semidynamic point sets. In *6th Annual A.C.M. Symp. Computational Geometry* (1990).
22. J. L. Bentley, Fast algorithms for geometric traveling salesman problems. *ORSA J. Computing* **4**, 387–411 (1992).
23. M. L. Fredman, D. S. Johnson, L. A. McGeoch and G. Ostheimer, Data structures for traveling salesman (1993).
24. G. Reinelt, Fast heuristics for large geometric traveling salesman problems. *ORSA J. Computing* **4**, 206–217 (1992).
25. G. A. Kindervater, J. K. Lenstra and D. B. Shmoys, The parallel complexity of TSP heuristics. *J. Algorithms* **10**, 249–270 (1988).
26. C. P. Ravikumar, Parallel techniques for solving large scale travelling salesperson problems. *Microprocessors Microsystems* **16**, 149–157 (1992).
27. N. Christofides, Worst case analysis of a new heuristic for the traveling salesman problem. Technical Report 388, Graduate School of Industrial Administartion Carnegie–Mellon University, Pittsburgh (1976).
28. J. Beardwood, J. H. Halton and J. M. Hammersley, The shortest path through many points. In *Proc. Cambridge Philosophical Soc.* **55**, 299–327 (1959).
29. B. Bäumle and the MUSIC Group, *The MUSIC Tutorial.* Swiss Federal Institute of Technology ETH, Zurich, Switzerland (1992).
30. A. Gunzinger, U. Müller, W. Scott, B. Bäumle, P. Kohler and W. Guggenbühl, Architecture and realization of a multi signal processor system. In *Proc. Int. Conf. Application Specific Array Processors*, pp. 327–340 (1992).
31. G. Cesari, A parallel implementation of dissection algorithms for the traveling salesman problem. In *Proc. AICA94 Conf.* (1994).
32. A. Brüngger, private communication. ETH Zurich.
33. D. S. Johnson, private communication.
34. H. L. Hong and H. C. Huang, Asymptotical expected performance of some TSP heuristic: an empirical evaluation. *Eur. J. Op. Res.* **43**, 231–238 (1989).
35. G. S. Fishman and L. R. Moore, A statistical evaluation of multiplicative congruential random number generators with modulus $2^{31}-1$. *J. Am. Statist. Assoc.* **77**, 129–136 (1982).