Jan van Leeuwen, Utrecht
Anneke A. Schoone, Utrecht

UNTANGLING A TRAVELLING SALESMAN TOUR IN THE PLANE

Techn Report RUU-CS-80-11
Dept of Computer Science
Utrecht University
1980

Published in:

J.R. Mühlbacher (ed.),
Proc. 7th Conference on Graphtheoretic
Concepts in Computer Science (WG 81),
Hanser Verlag, München, 1981,
pp 87-98.

Abstract

Known approximation algorithms for the traveling salesman problem in the plane deliver tours which can still contain intersecting edges. We prove that the common procedure to remove all intersections from a tour without increasing its length, is guaranteed to terminate within polynomially many steps.

Zusammenfassung

Näherungsalgorithmen zur Lösing des bekannten Problems des Handlungsreisenden im zweidimensionalen Fall ergeben Touren deren Kanten möglicherweise kreuzen. Wir zeigen dass die Anzahl der Schritte eines bestimmten Algorithmus zur Entfernung kreuzender Kanten ohne Verlängerung der Tour polynomial beschränkt ist.

1. Introduction

Given a set of n points (cities) in the plane, the traveling salesman problem asks for the shortest tour that visits all points exactly once and returns in the point of departure. There are many different versions of the problem. In our case (the "euclidean traveling salesman problem" or ETSP) an edge between two points is the connecting straightline segment in the plane. An edge between two points may "pass through" a third point on the way, but this is not counted as a visit.

The ETSP was proved NP-complete by Papadimitriou [5]. The computational intractability had been observed earlier from experiments, though, and several heuristics have been developed that will normally lead to good tours. Karp (see e.g. [3]) analyzed some methods probabilistically, to assess the expected quality of the tours. Rosenkrantz, Stearns and Lewis [6] observed that there are algorithms that guarantee a tour of length at most α.OPT for some $\alpha \leq 2$, where OPT is the length of the optimal tour, of which the running time nevertheless is bounded by a polynomial. The best polynomial time approximation algorithm presently known for the ETSP is due to Christofides [1], which guarantees a performance ratio $\alpha = 3/2$. The algorithm begins by building a minimum spanning tree T (of length necessarily $<$ OPT) and adding a minimum length matching of nodes of odd degree to it (of length $< 1/2$ OPT). The resulting graph of even degree nodes must admit an Eulerian tour of length $< 3/2$ OPT. Connecting nodes in "first visit" order yields a tour within the same length-bound. A readable survey of known heuristics for solving the traveling salesman problem was given in Christofides [2].

It is easy to see that the tours resulting from some approximation algorithms (viz. Christofides') may well contain crossing edges and thus fail to be optimal in view of the following "folk theorem":

Theorem A. Optimal tours have no intersecting edges.

Hence optimal tours always are simple polygonal paths. Another conclusion is that optimal tours always visit points on the convex hull of the set in the order in which they appear on the convex hull.

An immediate question is whether "approximate" tours can always be improved so as to satisfy the characteristic expressed in theorem A. It is apparently common knowledge how intersections can be eliminated from a given tour, but in the literature no concrete complexity results for this removal process are reported. In this paper we prove that the removal process must end in polynomially many steps.

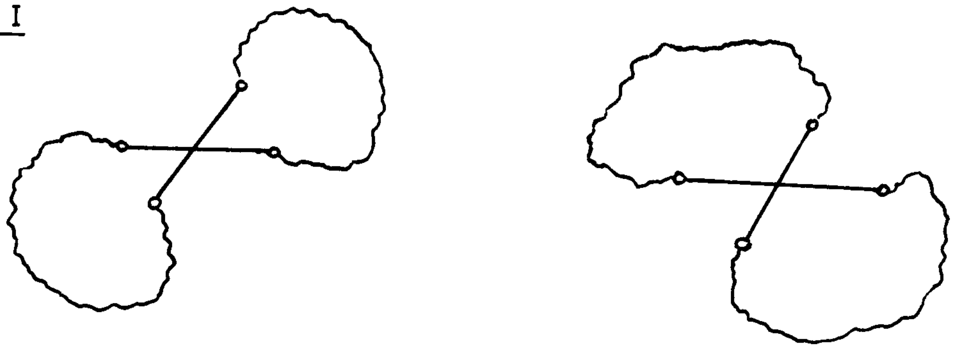
In section 2 we analyse the types of intersections that can occur and the (common) way of removing them. Even though each removal may lead to new intersections, we shall prove in section 3 that any order of eliminating intersections must eventually lead to an intersection-free tour within $O(n^3)$ removals. In section 4 we discuss the result.

2. Crossings, concurrencies and their removal

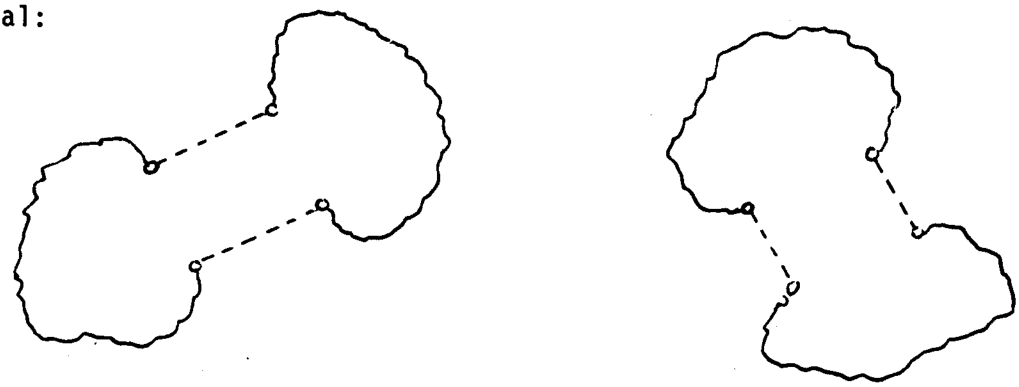
Without loss of generality we assume that the points do not all lie on a single line. In this section we shall take a precise look at the different types of intersections tours can have.

We first look at (pairs of) edges that intersect in only one point.

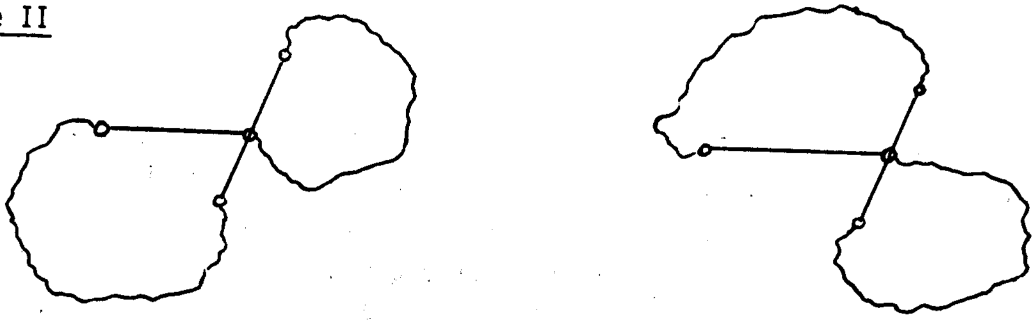
Type I



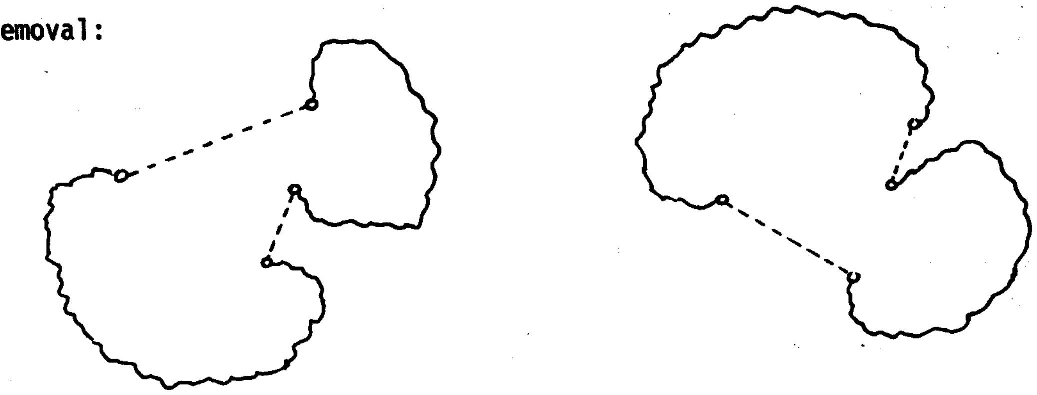
removal:



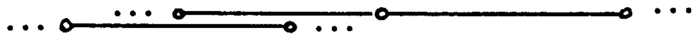
Type II



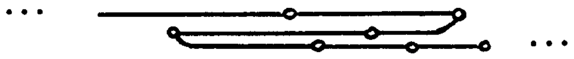
removal:



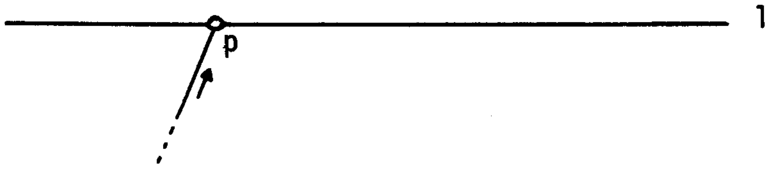
All other intersections are no straight crossings, but sneaky ways in which sections of the tour overlap:
e.g.



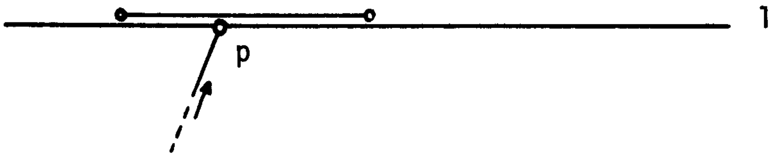
or



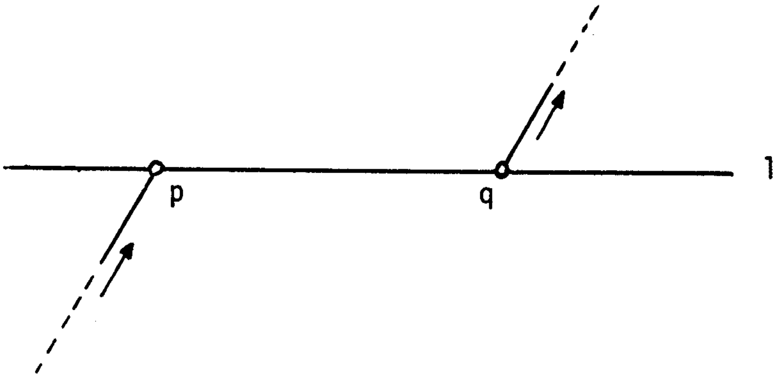
(where the points shown are assumed to lie on the same line). Consider any straight line l on which sections of the tour overlap. Consider any point p where the tour enters l and begins a section along l. (Because not all points lie on one line, such a point must exist.)



If the tour "passes through" p for a second time while running on l (perhaps

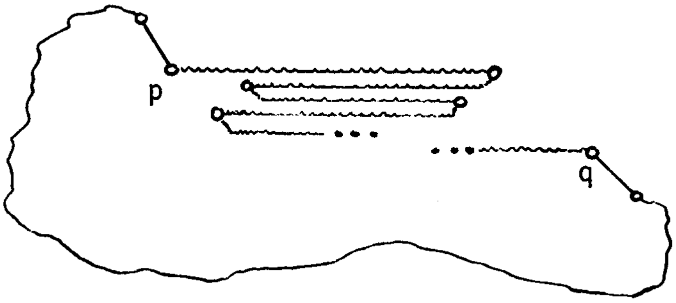


after having left l in the meantime), a type II intersection is created, which can consequently be removed. Now assume p is not passed a second time on l. Follow the tour starting at p as it runs along l and let q be the (first) point where this section departs from l. (Again, such a q must exist.)



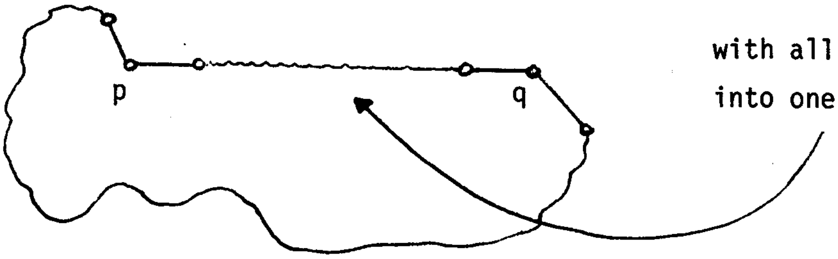
We only need to consider the case that q is not "passed through" in another section on l (or else we had a type II intersection again). Thus the section from p to q either has no internal overlaps and is intersection-free or has an intersection of type III which we show how to remove below (while shortening the tour). We use ... ○∿∿∿∿∿○ ... to denote a connecting chain of 1 or more edges.

Type III



(with all points shown between p and q on the same straight line l)

removal:



with all points merged into one chain

This concludes the case analysis. The types cover all possible ways in which a tour can intersect (cross or stick with) itself. It gives us a concrete proof of the following, more precise version of theorem A:

Theorem A. Assuming points do not all lie on a same straight line, an optimal traveling salesman tour connecting the points must be a simple (closed) polygonal path.

Proof

Suppose there was an optimal tour T for which it wasn't true. The preceding analysis shows that any tour T that is not intersection-free, must contain an intersection of one of the 3 types distinguished which can, consequently, be removed. Since all of the removal procedures shorten the tour by a non-zero amount, it would mean that T wasn't optimal. Contradiction.

□

3. A bound on the number of removals needed.

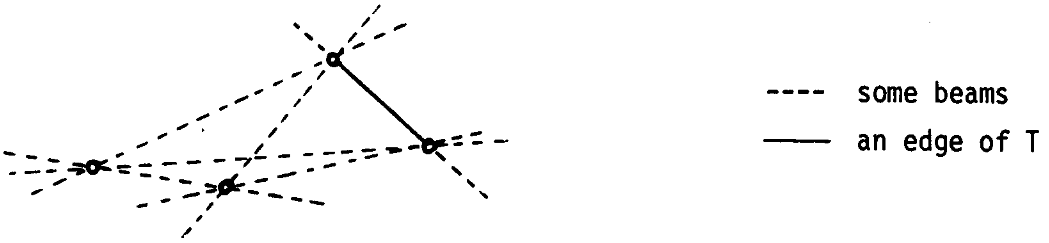
Consider an arbitrary (i.e., not necessarily optimal) tour T which visits all points exactly once and returns in the point of departure. If T is not a simple polygonal path, then it can be shortened as indicated in Section 2. Any intersection of type I, II or III will be called a tangle. In this section we consider a simple algorithm to remove all tangles:

```
            algorithm J:
            while the tour isn't simple polygonal yet do
            begin
                      locate a tangle;
   "phase"            determine its type;
                      remove it
            end.
```

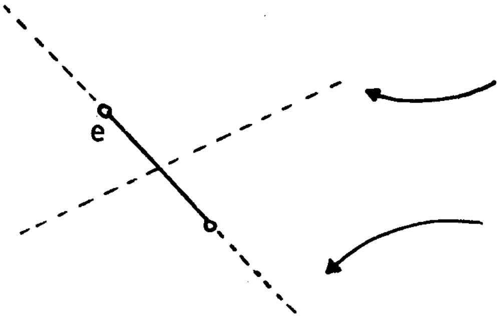
Note that tangles are not removed in any particular order, although an implementation could always take (say) the leftmost tangle. Any execution of the loop-body is called a phase. The complexity of algorithm J depends on the number of phases required to remove all tangles. The correctness of algorithm J follows from the observations in Section 2 that show that any tour that is free of tangles (i.e., of intersections of types I, II or III) really has no intersections whatsoever.

There is an obvious complication, though. Whenever one tangle is removed, the replacing edges may lead to new tangles. It is possible even that after some phases a tangle occurs again that was removed earlier. Fortunately a configuration cannot repeat as a whole (because of the decreasing length of the tour) and algorithm J must eventually terminate, regardless of the order in which tangles are removed. We shall argue that the number of phases is, in fact, polynomially bounded. To this end we shall introduce a function c(T) to measure changes from one configuration to a next.

Consider the set of n points. With any two points of the set we associate a beam, which is the infinite straight line determined by these two points. Even when beams "coincide", we treat them distinctly (i.e., beams will have a multiplicity). The beam associated with an edge is the beam determined by the two end-points of the edge. We assume that edges are segments, i.e., the end-points are considered part of an edge.



Definition. The index c(e) of an edge e of T is equal to the number of beams that hit it, counting every beam that hits e in its own direction twice!



this beam is counted once in c(e)

any such beam is counted twice in c(e)

Definition. The index c(T) of a tour T is defined as $c(T) = \sum_{\substack{\text{edges } e\\ \text{of } T}} c(e)$.

Proposition 3.1. For any tour $c(T) < n^3$.

Proof

There are $\frac{n(n-1)}{2}$ beams. As some beams have to be counted twice in c(e), it follows that for each edge $c(e) \leq 2.\frac{n(n-1)}{2} < n^2$. Hence $c(T) < n.n^2 = n^3$.

□

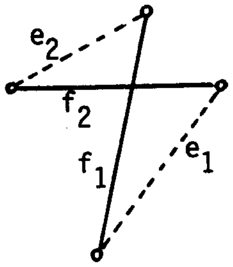
There are tours T for which $c(T) = \Omega(n^3)$. The following observation is crucial.

Lemma 3.2. Let T have at least one tangle. Let T' be a tour obtained after removing one tangle from T. Then we have $c(T') < c(T)$.

Proof

By considering each of the 3 possible types of tangles and their removal procedure separately.

Case 1 (Type I)



—— edges removed

---- edges added

Observe that $c(T) - c(T') = \{c(f_1) + c(f_2)\} - \{c(e_1) + c(e_2)\}$. We shall first argue that any beam b not in the direction of any of the 4 edges shown which contributes to $c(e_1) + c(e_2)$, contributes to $c(f_1) + c(f_2)$ with at least the same count. After all, any such b which hits e_1 or e_2 must hit at least one of f_1 and f_2 as well. If b hits both e_1 and e_2, then it must hit both f_1 and f_2. It follows that

$\{c(f_1) + c(f_2)\} - \{c(e_1) + c(e_2)\} \geq$ difference in count over beams that hit either f_1, f_2, e_1 or e_2 in their full direction.

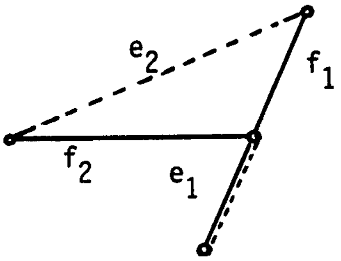
Notation. Let $\lambda(a)$ be the number of beams that hit an edge a in its direction. (Note: $\lambda(a) \geq 1$.)

Only counting beams in the direction of the 4 edges shown, we have

$$\begin{array}{lcl}
c(f_1) & = & \lambda(e_1) + 2\lambda(f_1) + \lambda(f_2) + \lambda(e_2) \\
c(f_2) & = & \lambda(e_1) + \lambda(f_1) + 2\lambda(f_2) + \lambda(e_2) \\
c(e_1) & = & 2\lambda(e_1) + \lambda(f_1) + \lambda(f_2) \\
c(e_2) & = & \lambda(f_1) + \lambda(f_2) + 2\lambda(e_2)
\end{array}$$

It follows that $\{c(f_1) + c(f_2)\} - \{c(e_1) + c(e_2)\} \geq \lambda(f_1) + \lambda(f_2) \geq 2$.

Case 2 (Type II)



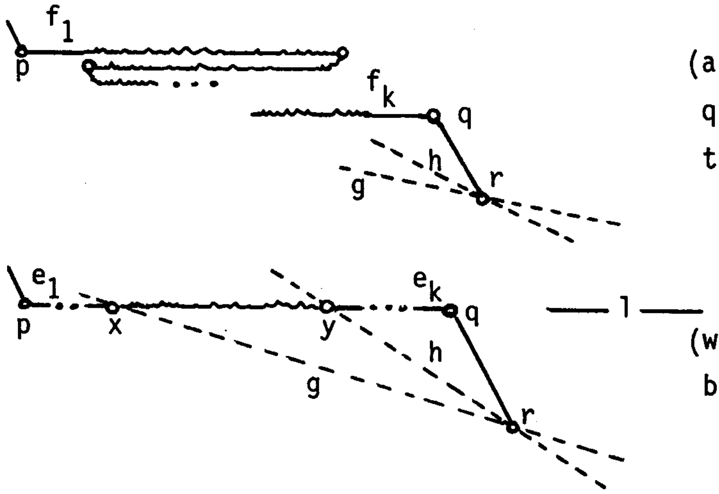
—— edges removed
---- edges added

Again $c(T) - c(T') = \{c(f_1) + c(f_2)\} - \{c(e_1) + c(e_2)\}$. And as in case 1 one can argue that one only has to take beams b into consideration that hit any of the 4 edges shown in full direction. Now note that $\lambda(e_1) = \lambda(f_1)$. Only counting such beams, we have

$$\begin{aligned}
c(f_1) &\equiv 2\lambda(f_1) + \lambda(f_2) + \lambda(e_2)\\
c(f_2) &\equiv \lambda(f_1) + 2\lambda(f_2) + \lambda(e_2)\\
c(e_1) &\equiv 2\lambda(f_1) + \lambda(f_2)\\
c(e_2) &\equiv \lambda(f_1) + \lambda(f_2) + 2\lambda(e_2)
\end{aligned}$$

It follows that $\{c(f_1) + c(f_2)\} - \{c(e_1) + c(e_2)\} \geq \lambda(f_2) \geq 1$.

Case 3 (Type III)



(all points between p and q are assumed to lie on the same straight line l)

(with x and y as explained below)

Let r be the point visited immediately after q. It is implicit in a type III tangle that r does not lie on l. In fact, neither does the point visited just before p, but we will ignore it (the point could be identical to r!). Clearly $c(T) - c(T') = \sum_1^k \lambda(f_i) - \sum_1^k \lambda(e_i)$. Observe that necessarily $k \geq 3$ in this type of tangle. It is straightforward to see that any beam b contributing to $\Sigma\lambda(e_i)$ contributes with at least the same count to $\Sigma\lambda(f_i)$, regardless its direction. For two beams we can make a more precise statement than that. Let x be the leftmost point (on l) visited after the first turn left during the interval from p to q and let y be the rightmost such point visited before the last turn right. Then $x \neq y$ and T must "pass" both x and y at least three times. (Note that the "change of direction" that must occur in both x and y accounts for two and the first pass "over" x and the last pass "over" y, respectively, yield the third.) Let g be the beam determined by r and x and let h be the beam determined by r and y. It follows that beam g is counted at least three times in $\Sigma\lambda(f_i)$, as is beam h. On the other hand, in the new tour beam g only hits the two e-edges incident to x and thus contributes a total count of 2 to $\sum_1^k\lambda(e_i)$. For a similar reason, beam h contributes a total of 2 to the new count. Hence $c(T) - c(T') \geq$ $\geq (3 - 2) + (3 - 2) = 2$.

This completes the analysis and shows that in all cases of possible tangles, the tour T' resulting after a removal must satisfy $c(T) - c(T') \geq 1$, i.e., $c(T) > c(T')$.

□

It is to be expected that after one removal $c(T) - c(T')$ will jump by a larger amount than just 1.

The important conclusion from 3.2. is a bound on the number of phases algorithm J ever needs to perform.

Theorem 3.3. Given an arbitrary traveling salesman tour T, algorithm J needs to go through no more than c(T) phases to remove any tangles T may have, i.e., to transform T into an intersection-free tour of the same length or less.

Corollary 3.4. Algorithm J never needs to perform more than $O(n^3)$ phases, regardless of the order in which tangles are detected and removed.

Proof

From 3.3. and 3.1.

□

We conclude with our main result.

Theorem B. Given an arbitrary traveling salesman tour T, one can remove any intersections it may have and transform it into an intersection-free T' of length less than or equal to T's in polynomial time.

Proof

Algorithm J does the job. It never needs more than $O(n^3)$ phases and it is easy to see that each single phase can be performed in, say, $O(n^2)$ steps by looking at each single edge and testing whether it "participates" in a tangle.

□

4. Discussion

The traveling salesman problem comes in many different guises and has many relevant "companions". (See, e.g., Lenstra and Rinnooy Kan [4] for a survey of complexity results for the problem and similar routing problems.) We have shown that any euclidean traveling salesman tour that intersects itself can be transformed into a shorter one without intersections in only polynomially many steps.

Theorem B enables us to remove all intersections from any approximate solution (tour) to the traveling salesman problem. In particular we can strengthen Christofides' result [1] to

Theorem 4.1. There is a polynomial time algorithm to obtain an intersection-free tour (i.e., a simple closed polygonal path) that is guaranteed to be within 3/2 of the length of the optimum solution of the traveling salesman problem in the plane.

The result obviously holds for any approximation method and preserves whatever performance factor it had.

5. References

[1] Christofides, N., Worst-case analysis of a new heuristic for the traveling salesman problem, Techn. Rep., Grad. School of Industr. Adm., Carnegie-Mellon Univ., Pittsburgh, 1976.

[2] Christofides, N., The traveling salesman problem, in: N. Christofides et.al. (editors), Combinatorial Optimization, Wiley & Sons, Chichester, 1979, Ch 6, pp. 131-149.

[3] Karp, R.M., Probabilistic analysis of partitioning algorithms for the traveling salesman problem in the plane, Math. Oper. Res. 2 (1977) 209-224.

[4] Lenstra, J.K. and A.H.G. Rinnooy Kan, Complexity of vehicle routing and scheduling problems, Report BW 111/79, Dept. of Operations Research, Mathem. Centre, Amsterdam, 1979.

[5] Papadimitriou, C., The euclidean traveling salesman problem is NP-complete, Theor. Comp. Sci. 4 (1977) 237-244.

[6] Rosenkrantz, D., R.E. Stearns and P.M. Lewis, An analysis of several heuristics for the traveling salesman problem, SIAM J. Comput. 6 (1977) 563-581.