Chapter 3

AN ANALYSIS OF SEVERAL HEURISTICS FOR THE TRAVELING SALESMAN PROBLEM*

DANIEL J. ROSENKRANTZ, RICHARD E. STEARNS
AND PHILIP M. LEWIS II
General Electric Research and Development Center, Schenectady, NY 12345, USA

Reprinted from: *SIAM J. Computing*, Vol. 6, No. 3, Sept. 1977, pp. 563–581.

Abstract Several polynomial time algorithms finding "good," but not necessarily optimal, tours for the traveling salesman problem are considered. We measure the closeness of a tour by the ratio of the obtained tour length to the minimal tour length. For the nearest neighbor method, we show the ratio is bounded above by a logarithmic function of the number of nodes. We also provide a logarithmic lower bound on the worst case. A class of approximation methods we call insertion methods are studied, and these are also shown to have a logarithmic upper bound. For two specific insertion methods, which we call nearest insertion and cheapest insertion, the ratio is shown to have a constant upper bound of 2, and examples are provided that come arbitrarily close to this upper bound. It is also shown that for any $n \geq 8$, there are traveling salesman problems with n nodes having tours which cannot be improved by making $n/4$ edge changes, but for which the ratio is $2(1 - 1/n)$.

Keywords: traveling salesman problem, approximation algorithm, k-optimal, minimal spanning tree, triangle inequality

Received by the editors July 19, 1976, and in revised form December 13, 1976.

* An extended abstract of this paper is in the Proceedings of the IEEE Fifteenth Annual Symposium on Switching and Automata Theory, 1974, under the title *Approximate algorithms for the traveling salesperson problem*.

S.S. Ravi, S.K. Shukla (eds.), *Fundamental Problems in Computing*,

1. Introduction

The traveling salesman problem has long been of great interest. The problem has been formulated in several different ways. We use the following formulation:

A traveling salesman graph G is a complete weighted undirected graph specified by a pair (N, d) where N is a set of nodes, d is a *distance function* mapping pairs of nodes (or edges) into real numbers, and d satisfies

(a) $d(i, j) = d(j, i)$ for all i and j in N,

(b) $d(i, j) \geq 0$ for all i and j in N,

(c) $d(i, j) + d(j, k) \geq d(i, k)$ for all i, j, k in N.

Condition (c) is referred to as the *triangle inequality*. The number $d(i, j)$ is called the *length* or *weight* of (i, j).

A *tour* for a traveling salesman graph G is a circuit on the graph containing each node exactly once (i.e. a Hamiltonian circuit). The *length* of a tour is the sum of the lengths of the edges composing the circuit. An *optimal tour* or *solution* for G is a tour of minimal length. The *traveling salesman problem* is to take a traveling salesman graph and find an optimal tour.

The traveling salesman problem is sometimes formulated (Bellmore and Nemhauser [1]) as the problem of finding a minimal length circuit containing each node at least once for an undirected graph in which the distances are not constrained by the triangle inequality. However, a problem stated in this manner can always be reduced (Hardgrave and Nemhauser [6]) to the problem considered here by the technique of changing each $d(i, j)$ to the length of the shortest path between i and j. This conversion can be done in time proportional to the cube of the number of nodes (Floyd [4]). Each tour in the new problem corresponds to a circuit of the same length in the original problem, and the two problems have solutions of the same length. Therefore, our results, which are stated in terms of the new problem, also apply to the original problem.

Another formulation requires that a shortest tour be found for distances not constrained by the triangle inequality. A problem stated this way can always be reduced to the type of problem considered here by adding a suitably large constant k to each distance. The altered problem has the same optimal tour as the original, but the lengths of the optimal tours will differ by the amount $n \cdot k$ where n is the number of nodes. Our results do not apply to this formulation, since our results pertain to the tour lengths.

The best known methods of solving the traveling salesman problem take an amount of time exponential in the number of nodes. Furthermore, the problem is easily seen to be *NP*-hard. Karp [8] shows that determining whether an undirected graph has a Hamiltonian circuit is *NP*-complete. This problem can

be reduced to a traveling salesman problem by forming the complete weighted graph whose edges are of length one if there is a corresponding edge in the original graph, and of length two otherwise. For an n node graph, the minimal tour of the new graph has length n if and only if the original graph has a Hamiltonian circuit.

In view of the computational difficulties in obtaining optimal tours, a number of algorithms have been published which run faster but do not necessarily produce an optimal tour. A number of these approximation algorithms have been experimentally observed to perform well, but there has not been a theoretical characterization of how the obtained tours compare with the optimal.

In this paper, we analyze some of these methods to bound the ratio of the obtained tour length to the optimal tour length. In some cases, these bounds grow as a function of the number of nodes and in other cases a constant bound is found for all traveling salesman problems. In contrast, if the distance function is unconstrained by the triangle inequality then for any constant $k \geq 1$, the problem of finding a tour with a ratio bounded by k is *NP*-complete (Sahni and Gonzalez [16]).

Another approximation method was recently announced and analyzed in Christofides [2]. This method produces a better worst case approximation than the methods analyzed here, but requires more running time.

In the material which follows, we exclude the trivial case where the distance function is identically zero. This assumption together with the triangle inequality implies that every tour has a length greater than zero. We also adopt the convention that OPTIMAL represents the length of the optimal tour. Under the assumption of nontriviality,

$$\text{OPTIMAL} > 0. \tag{1.1}$$

2. Nearest Neighbor Algorithm

The first approximation algorithm we study is the nearest neighbor method (Bellmore and Nemhauser [1]), also called the next best method in Gavett [5]. In this algorithm, a path is constructed as follows:

1. Start with an arbitrary node.

2. Find the node not yet on the path which is closest to the node last added and add to the path the edge connecting these two nodes.

3. When all nodes have been added to the path, add an edge connecting the starting node and the last node added.

We assume that when there are ties in step 2, they can be broken arbitrarily.

We note that the nearest neighbor algorithm can be programmed to operate in time proportional to n^2 where n is the number of nodes. This time is linear in the input length if the input is a list of all distances.

Let NEARNEIBER be the length of the tour obtained by the nearest neighbor algorithm. Let lg denote the logarithm to the base 2, and $\lceil x \rceil$ denote the smallest integer greater than or equal to x.

THEOREM 1. *For a traveling salesman graph with n nodes*

$$\frac{\text{NEARNEIBER}}{\text{OPTIMAL}} \leq \frac{1}{2}\lceil \lg(n) \rceil + \frac{1}{2}.$$

The proof of Theorem 1 is given after the proof of the following lemma.

LEMMA 1. *Suppose that for a n node graph (N, d) there is a mapping assigning each node p a number l_p such that the following two conditions hold*:

(a) $d(p, q) \geq \min(l_p, l_q)$ *for all nodes p and q.*

(b) $l_p \leq \frac{1}{2}$ OPTIMAL *for all nodes p.*

Then $\sum l_p \leq \frac{1}{2}(\lceil \lg(n) \rceil + 1)$ OPTIMAL.

Proof. We can assume without loss of generality that node set N is $\{i \mid 1 \leq i \leq n\}$ and that $l_i \geq l_j$ whenever $i \leq j$. The key to the proof is the following inequality:

$$\text{OPTIMAL} \geq 2 \sum_{i=k+1}^{\min(2k,n)} l_i \tag{2.1}$$

for all k satisfying $1 \leq k \leq n$.

To prove (2.1), we let H be the complete subgraph defined on the set of nodes

$$\{i \mid 1 \leq i \leq \min(2k, n)\}.$$

We let T be the tour in H which visits the nodes of H in the same order as these nodes are visited in an optimal tour of the original graph. Let LENGTH be the length of T. By the triangle inequality, each edge (b, c) of T must have a length which is less than or equal to the length of the path from b to c used in the optimal tour. Since the edges of T sum to LENGTH and the corresponding paths in the original graph sum to OPTIMAL we conclude that

$$\text{OPTIMAL} \geq \text{LENGTH}. \tag{2.2}$$

By condition (a) of the Lemma, for each (i, j) in T, $d(i, j) \geq \min(l_i, l_j)$. Therefore,

$$\text{LENGTH} \geq \sum_{(i,j)\in T} \min(l_i, l_j) = \sum_{i\in H} \alpha_i l_i \tag{2.3}$$

where α_i is the number of edges (i, j) in T for which $i > j$ (and hence $l_i = \min(l_i, l_j)$).

We want to obtain a lower bound on the right hand side of (2.3). Observe that each α_i is at most 2 (because i is the endpoint of only two edges in tour T) and that the α_i sum to the number of edges in T. Because k is at least half of the number of edges in T, we certainly get a lower bound on the right hand side of (2.3) if we assume that the k largest l_i have $\alpha_1 = 0$ and the remaining $\min(2k, n) - k$ of the l_i have $\alpha_i = 2$. By assumption, the k largest are $\{l_i | 1 \leq i \leq k\}$ so the estimated lower bound is

$$\sum_{i\in H} \alpha_i l_i \geq 2 \sum_{i=k+1}^{\min(2k,n)} l_i \tag{2.4}$$

and (2.2), (2.3), and (2.4) together establish (2.1).

We now sum inequalities (2.1) for all values of k equal to a power of two less than s, namely:

$$\sum_{j=0}^{\lceil \lg(n)\rceil - 1} \text{OPTIMAL} \geq \sum_{j=0}^{\lceil \lg(n)\rceil - 1} 2 \cdot \sum_{i=2^j+1}^{\min(2^{j+1},n)} l_i,$$

which reduces to

$$\lceil \lg(n)\rceil \cdot \text{OPTIMAL} \geq 2 \cdot \sum_{i=2}^{n} l_i. \tag{2.5}$$

Now condition (b) of the lemma implies

$$\text{OPTIMAL} \geq 2 \cdot l_1 \tag{2.6}$$

and (2.5) and (2.6) combine to give the conclusion of the lemma. □

Proof of Theorem 1. For each node p, let l_p be the length of the edge leaving node p and going to the node selected as the nearest neighbor to p. We want to show that the l_p satisfy the conditions of Lemma 1.

If node p was selected by the algorithm before node q, then q was a candidate for the closest unselected node to node p. This means that edge (p, q) is no shorter than the edge selected and hence

$$d(p, q) \geq l_p. \tag{2.7}$$

Conversely, if q was selected before p, then

$$d(p,q) \geq l_q. \tag{2.8}$$

Since one of the nodes was selected before the other, either (2.7) or (2.8) must hold and condition (a) of Lemma 1 must be satisfied.

To prove condition (b) it suffices to prove that for any edge (p,q)

$$d(p,q) \leq \frac{1}{2} \cdot \text{OPTIMAL}. \tag{2.9}$$

The optimal tour can be considered to consist of two disjoint parts, each of which is a path between nodes p and q. From the triangle inequality, the length of any path between p and q cannot be less than $d(p,q)$, establishing (2.9).

Because the l_p are the lengths of the pairs comprising tour T,

$$\sum l_p = \text{NEARNEIBER}. \tag{2.10}$$

The conclusion of Lemma 1 together with (2.10) and (1.1) imply the inequality of Theorem 1. □

THEOREM 2. *For each $m > 3$, there exists a traveling salesman graph with $n = 2^m - 1$ nodes such that*

$$\frac{\text{NEARNEIBER}}{\text{OPTIMAL}} > \frac{1}{3}\lg(n+1) + \frac{4}{9}.$$

Proof. For all $i \geq 1$, we define an incomplete weighted graph F_i with three distinguished nodes. The distinguished nodes are called the *start node*, the *middle node*, and the *right node*. These graphs are defined recursively using Fig. 3.1 where the start node appears to the left, the middle node in the middle, and the right node on the right. Each F_i has a path P_i which goes from the start node to the middle node visiting each node of F_i on the way. The P_i are also defined recursively in Fig. 3.1.

Graph F_1 consists of precisely three nodes with each pair of nodes having an edge of weight 1. Path P_1 consists of two edges, the edge from the start node to the right node and the edge from the right node to the middle node.

To construct graph F_{i+1}, one takes two copies of F_i (which we call the *left copy* and *right copy*) and one additional node (which becomes the middle node of F_{i+1}). This additional node is called D in Fig. 3.1. The additional node D is connected to the right node of the left copy (node C) and the start node of the right copy (node E) by edges of length 1. The additional node D is also connected to the middle node of the right copy (node F) by an edge of length l_i (defined below). Finally, the middle node of the left copy (node B) is

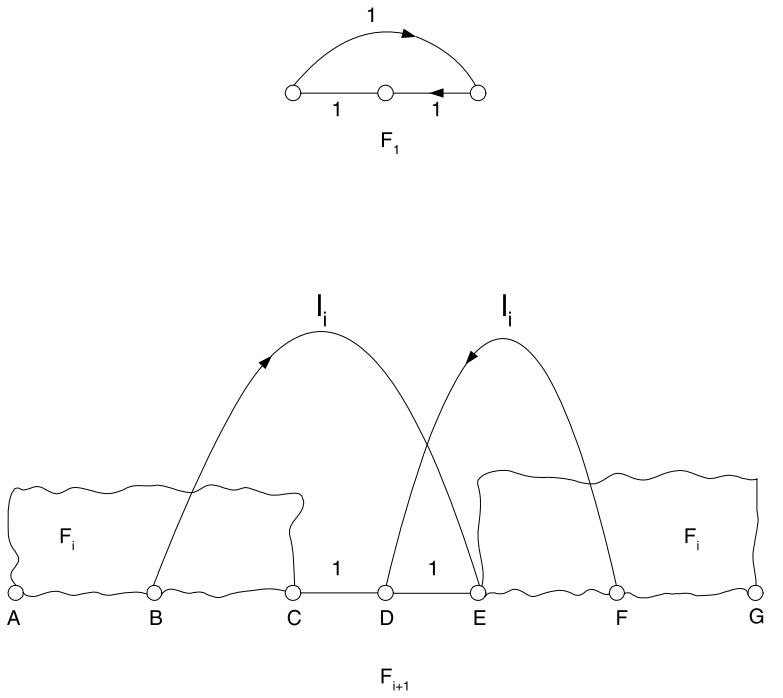

Figure 3.1.

connected to the start node of the right copy (node E) by an edge of length l_i. The start node of F_{i+1} is the start node of the left copy (node A) and the right node is the right node of the right copy (node G). The path P_{i+1} consists of the two copies of path P_i plus the two edges (B, E) and (F, D) of length l_i. The length l_i is given by the formula

$$l_i = \frac{1}{6}(4 \cdot 2^i - (-1)^i + 3). \tag{2.11}$$

Let L_i be the length of path P_i. Length L_i is described by the difference equation

$$L_{i+1} = 2 \cdot L_i + 2 \cdot l_i$$

since P_{i+1} consists of two copies of P_i and two edges of length l_i. Given that $L_1 = 2$, the solution of this difference equation is

$$L_i = \frac{1}{9}(6 \cdot i \cdot 2^i + 8 \cdot 2^i + (-1)^i - 9). \tag{2.12}$$

For each F_i, we define a graph G_i obtained by connecting the start and right nodes of F_i by an edge of length 1, and connecting the middle node to the start node with an edge of length $l_i - 1$. The start node of F_i is then also

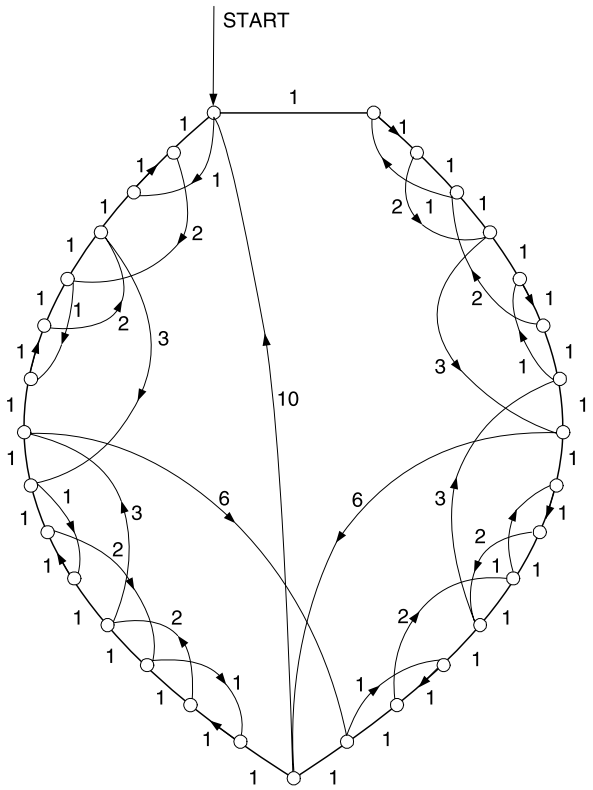

Figure 3.2.

referred to as the start node of G_i. Figure 3.2 is a picture of G_4. We define $\bar{G}_i$ to be the complete graph on the nodes of G_i where $d(a, b)$ is the length of the minimal path from a to b in G_i. Therefore, the distances in $\bar{G}_i$ satisfy the triangle inequality.

Graph $\bar{G}_i$ has two important properties:

(a) the edges of G_i have the same lengths in $\bar{G}_i$ as they have in G_i;

(b) if the nearest neighbor method is started with the start node of G_i, the method can (with suitable resolution of ties) produce path P_i followed by the edge of length $l_i - 1$ returning from the middle node (which is the last node of path P_i) to the start node.

We return to prove properties (a) and (b) after completing the main thread of the proof.

Each $\bar{G}_i$ has an optimal tour whose length is equal to the number of nodes n in $\bar{G}_i$ (namely $2^{i+1} - 1$). This tour is found, starting with the start node, by

visiting the nodes in left to right order and then returning from the right node back to the start node. Each of the edges in this tour has weight one.

The example satisfying the theorem is $\bar{G}_{m-1}$. Its ratio is exactly

$$\frac{\text{NEARNEIBER}}{\text{OPTIMAL}} = (L_i + l_i - 1)/n \quad \text{where } i = \lg(n+1) - 1.$$

This ratio is greater than the ratio indicated in the theorem.

All that remains is to prove properties (a) and (b). Referring back to Fig. 3.1, we first show that for each F_{i+1}

$$\overline{AB} = \overline{BC} = \overline{EF} = \overline{FG} = l_i - 1, \tag{2.13}$$
$$\overline{AC} = \overline{EG} = l_{i+1} - 2, \tag{2.14}$$
$$\overline{BE} = \overline{DF} = l_i, \tag{2.15}$$
$$\overline{AD} = \overline{DG} = l_{i+1} - 1, \tag{2.16}$$
$$\overline{AG} = l_{i+2} - 2. \tag{2.17}$$

The notation $\overline{XY}$ indicates the length of the shortest path between X and Y in F_{i+1}. The equations are routinely verified for $i = 1$. We continue by induction. Assume that (2.12)–(2.16) are true for $i \leq I - 1$ (i.e. for F_I). Figure 3.3 shows the relevant nodes in F_{I+1} before the two copies of F_I are connected. Associated with each pair of nodes from the same copy of F_I, an edge is shown whose weight is the length of the shortest path in F_I connecting these nodes. These shortest path lengths in F_I are specified by the induction hypothesis. For instance, edge (A, B) in Fig. 3.3 connects the start and middle nodes of F_I and from (2.15), the shortest path length in F_I connecting these nodes is $l_I - 1$. Figure 3.4 shows Fig. 3.3 with the addition of the four edges created in the construction of F_{I+1} from the two copies of F_I. Because each of the edge weights in Fig. 3.3 represents a shortest path in F_I, by applying formula (2.11) for l_I to all possible paths in F_{I+1}, we can conclude that each of the edge weights in Fig. 3.4 is the length of the shortest path in F_{I+1} *connecting the end nodes of the edges. This establishes equations* (2.12)–(2.14) *for* F_{I+1}.

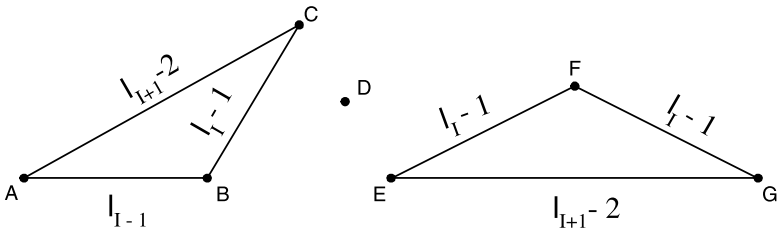

Figure 3.3.

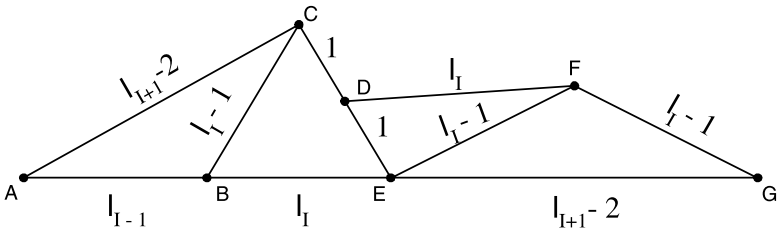

Figure 3.4.

Equations (2.15) and (2.16) are established by a similar consideration of all paths in Fig. 3.4. The path of length $l_{I+2} - 2$ from A to G is *ABEG*.

We note also that (2.12)–(2.15) also hold when F_{I+1} is converted to G_{I+1}. This is proven by connecting A and G in Fig. 3.4 by an edge of weight one and A and D by an edge of length $l_{I+1} - 1$ and again checking the paths. Note also that the shortest path from A to D is the edge (A, D).

Now we return to property (a). Equation (2.14) shows that, as each F_{i+1} is constructed, the newly added edges constitute the shortest paths between their endpoints. All distances among points in an F_i are maintained when that F_i is embedded in F_{i+1}, because the distances among the three exit points at F_i are maintained. (Compare (2.12) with (2.15) and (2.13) with (2.16).)

We have already noted that the final edge added in constructing a G_i is also a shortest path and the fact that the length one edges are also shortest paths requires no argument. Thus property (a) is established.

Property (b) is established by observing that the middle node of an F_i is reached only after all of the nodes of F_i have been visited, and the node at the end of the edge of length l_i is at least as close as any node reached by a path through the start or right nodes. These nodes are already distance l_{i-1} away from the middle node and are at least distance 1 from a node not yet selected. □

One way to improve a nearest neighbor result is to repeat the method for each possible starting node and then take the minimum solution among these. This idea is described in Gavett [5]. However, for the examples used to prove Theorem 2, the result of this method (with suitable resolution of ties) is also proportional to $\lg(n)$.

3. Insertion Methods

We now consider a class of methods we call insertion methods. The basic idea of these methods is to construct the approximation tour by a sequence of steps in which tours are constructed for progressively larger subsets of the nodes.

DEFINITION. Given a traveling salesman graph (N, d), a tour T on a subset S of N will be called a *subtour* of (N, d). We write $a \in T$ to mean $a \in S$. We treat a one node subset as a tour without edges.

DEFINITION. Given a traveling salesman graph (N, d), a subtour T, and a node k in N which is not in T, we define TOUR(T, k) to be a subtour obtained as follows:

if T passes through more than one point, then

(a) find an edge (x, y) in T which minimizes

$$d(x, k) + d(k, y) - d(x, y), \tag{3.1}$$

(b) delete edge (x, y) and add edges (x, k) and (k, y) to obtain $\mathrm{TOUR}(T, k)$;

if T passes through a single node i, then make TOUR(T, k) the two node tour consisting of edges (i, k) and (k, i).

In either case, we say that $\mathrm{TOUR}(T, k)$ is obtained by *inserting* k into T.

Formula (3.1) represents the difference in length between tour T and the tour obtained by replacing (x, y) by (x, k) and (k, y). Thus, when T has two or more nodes, $\mathrm{TOUR}(T, k)$ is the shortest tour that can be obtained from T and k by the alteration described in step (b). When T has only one node, $\mathrm{TOUR}(T, k)$ is the only tour that can be made from k and the point in T.

DEFINITION. An approximation method is called an *insertion method* if it takes a traveling salesman graph (N, d) with n nodes and constructs a sequence of subtours $T_1, \ldots, T_n$ so that

1. T1 consists of a single node a_0,
2. for each $i < n$, there is a node a_i not in T_i such that

$$T_{i+1} = \mathrm{TOUR}(T_i, a_i), \tag{3.2}$$

3. T_n is the approximation.

In later sections, we consider specific selection criteria for choosing the nodes a_i. Here we are concerned with results which hold regardless of the selection method.

DEFINITION. Given a subtour T and a node k not in T, we define $\mathrm{COST}(T, k)$ to be the length of $\mathrm{TOUR}(T, k)$ minus the length of T.

An important consequence of the triangle inequality is the following:

LEMMA 2. *If (N, d) is a traveling salesman graph, T is a subtour, k a node not in T, and j a node in T, then*

$$\text{COST}(T, k) \leq 2 \cdot d(k, j). \tag{3.3}$$

Proof. In the case that T has only one node, the result is obvious. When T consists of more than one node, j is an endpoint of some edge (i, j) in T. Because k is inserted to minimize (3.1),

$$\text{COST}(T, k) \leq d(i, k) + d(k, j) - d(i, j) \tag{3.4}$$

where the right-hand side is (3.1) with (i, j) substituted for (x, y). The triangle inequality says

$$d(i, k) - d(i, j) \leq d(j, k). \tag{3.5}$$

Inequalities (3.4) and (3.5) together with $d(j, k) = d(k, j)$ give (3.3). □

We let INSERT represent the length of a path constructed by an insertion algorithm.

THEOREM 3. *For a traveling salesman graph with n nodes,*

$$\frac{\text{INSERT}}{\text{OPTIMAL}} \leq \lceil \lg(n) \rceil + 1. \tag{3.6}$$

Proof. Let (N, d) be the graph and let T_i for $1 \leq i \leq n$ and a_i for $0 \leq i < n$ be the subtours and nodes referred to in the definition of an insertion method. An obvious consequence of the definition of cost is

$$\text{INSERT} = \sum_{i=1}^{n-1} \text{COST}(T_i, a_i) \tag{3.7}$$

For each node a_i in $N - \{a_0\}$, define

$$l_{a_i} = \frac{1}{2} \cdot \text{COST}(T_i, a_i) \tag{3.8}$$

and define

$$l_{a_0} = 0. \tag{3.9}$$

We want to show that the l_p for p in N satisfy the hypothesis of Lemma 1. To verify condition (a), consider two nodes a_i and a_j with $i > j$. By our naming conventions, $i > j$ means that a_j belongs to T_i and a_i was inserted in T_i. By Lemma 2,

$$\text{COST}(T_i, a_i) \leq 2 \cdot d(a_i, a_j). \tag{3.10}$$

With (3.8) this implies

$$l_{a_i} \leq d(a_i, a_j), \tag{3.11}$$

which implies condition (a).

Condition (b) is trivial for l_{a_0}. For other l_{a_i}, (3.8) requires us to prove

$$\text{COST}(T_i, a_i) \leq \text{OPTIMAL}. \tag{3.12}$$

In the case of a_1, this cost is just $2d(a_0, a_1)$ and by the triangle inequality, OPTIMAL is at least as large as the distance between two points and back. For $i > 1$, a_i is inserted between two distinct points x and y with cost

$$d(x, a_i) + d(a_i, y) - d(x, y), \tag{3.13}$$

which is the length of the added edges minus the length of the deleted edge. There is a subpath of the optimal tour between x and a_i which does not contain y and a disjoint subpath between a_i and y not containing x. By the triangle inequality, these subpaths are no shorter than $d(x, a_i)$ and $d(a_i, y)$ respectively and hence (3.13) must be no greater than OPTIMAL and condition (b) is established.

Lemma 1 together with (3.8), (3.9), (3.7), and (1.1) imply the theorem. □

We do not know if the logarithmic growth permitted by Theorem 3 can actually be achieved. In fact, we know of no examples such that INSERT/OPTIMAL > 4 so there could even be a constant upper bound. In the next section we present some insertion methods for which we can establish a constant upper bound.

4. Nearest Insertion and Cheapest Insertion

We now consider two insertion methods which produce a tour no longer than twice the optimal regardless of the number of nodes in the problem. We call these two methods the nearest insertion method and the cheapest insertion method.

Given a subtour T and a node p, we define the distance $d(T, p)$ between T and p as

$$\min\{d(x, p) \text{ for } x \text{ in } T\}. \tag{4.1}$$

We say that a tour is constructed by *nearest insertion* if each a_i,$1 \leq i < n$, in the definition of an insertion method satisfies

$$d(T_i, a_i) = \min\{d(T_i, x) \text{ for } x \text{ in } N - T_i\}. \tag{4.2}$$

We say a tour is constructed by *cheapest insertion* if the a_i satisfy

$$\text{COST}(T_i, a_i) = \min\{\text{COST}(T_i, x) \text{ for } x \text{ in } N - T_i\}. \tag{4.3}$$

The nearest insertion method is easily programmed to run in a time proportional to n^2. The only programming trick is to compute the value of $d(T_{i+1}, x)$ as the minimum of the two numbers $d(T_i, x)$ and $d(a_i, x)$. Thus the nearest insertion method runs in time proportional to the nearest neighbor method.

The cheapest insertion method is described in Nicholson [12]. The fastest algorithm we have devised for the cheapest insertion method runs proportional to $n^2 \cdot \log(n)$. Each time a node a_i is inserted in T_i, the new subtour T_{i+1} contains two new edges not in T_i. For each new edge (x, a_i) in T_{i+1}, the algorithm involves performing a sort of the $n - (i + 1)$ values of

$$d(x, k) + d(k, a_i) - d(x, a_i)$$

obtained for all k in $N - T_{i+1}$.

THEOREM 4. *If a tour of length* INSERT *is obtained by nearest insertion or cheapest insertion, then*

$$\frac{\text{INSERT}}{\text{OPTIMAL}} < 2. \tag{4.4}$$

We prove this theorem after proving the following lemma:

LEMMA 3. *Suppose that, for a traveling salesman graph (N, d) with n nodes, a tour of length* INSERT *is constructed by the insertion method of Sect. 3. Suppose further that for i satisfying $1 \leq i < n$, the tour T_i and node a_i selected by the insertion method satisfy*

$$\text{COST}(T_i, a_i) \leq 2 \cdot d(p, q) \tag{4.5}$$

for all nodes p and q such that p is in T_i and q is not in T_i. Then

$$\text{INSERT} \leq 2 \cdot \text{TREE}, \tag{4.6}$$

where TREE *is the length of a minimal spanning tree for (N, d).*

Proof. Let M be a minimal spanning tree. The idea of the proof is to establish a correspondence between steps in the insertion procedure and edges of M. For the step of inserting node a_i into T_i, the corresponding edge of M will have one endpoint in T_i and the other endpoint in $N - T_i$. Thus (4.5) can be used to show that the cost of each step is no more than twice the corresponding edge.

First, since M is a tree, there is a unique path in M connecting each pair of nodes. For each node a_i with $i > 0$, we say that node a_j is *compatible* with node a_i if $j < i$ and all the intermediate nodes in the unique path in M connecting a_i and a_j have indices greater than i. Thus a_j compatible with a_i

implies that a_j is the first node in T_i encountered in the path from a_i to a_j. For each a_i with $i > 0$, the *critical node* is the node with the largest index that is compatible with a_i. The *critical path* for a_i is the unique path in M between a_i and its critical node. The *critical edge* for a_i is the edge in the critical path, one of whose endpoints is the critical node. Observe that the critical edge for a_i has one endpoint (the critical node) in T_i, and the other endpoint in $N - T_i$.

We now show that no two nodes can have the same critical edge. Assume to the contrary that a_i and a_j (with $j > i$) have the same critical edge. Let the endpoints of this critical edge be a_k and a_l with $l > k$. For any critical edge, the node with the lower index is the critical node and the node with the higher index is on the critical path, so node a_k is the critical node for both a_i and a_j. Thus, the critical paths for a_i and a_j both pass through a_l before reaching a_k. Therefore, there is a path P in M connecting a_j and a_i, such that every edge in P belongs to either the critical path for a_j or the critical path for a_i (or both). Therefore every intermediate node on P has an index greater than i. Since the path P from a_j reaches a node of lower index (a_i), some node a_m along path P is compatible with a_j. Now $m \geq i$ because a_m is on path P and $i > k$ because a_k is a compatible node for a_i. This implies $m > k$ and so a_m is a compatible node for a_j with a higher index than a_k. This contradicts the assumption that a_k is critical for a_j. Therefore no two nodes can have the same critical edge. Thus given a minimal spanning tree we can associate a unique edge in that tree with each node inserted by the insertion method.

Let e_i be the critical edge for node a_i. Since one endpoint of e_i is in T_i and the other endpoint is not, by (4.5).

$$\text{COST}(T_i, a_i) \leq 2 \cdot d(e_i). \tag{4.7}$$

Summing (4.7) gives

$$\sum_{i=1}^{n-1} \text{COST}(T_i, a_i) \leq 2 \cdot \sum_{i=1}^{n-1} d(e_i). \tag{4.8}$$

The left-hand side of (4.8) is INSERT by (3.8). Since M consists of $n - 1$ edges, and each e_i is distinct, the right-hand side of (4.8) is $2 \cdot$ TREE. Thus (4.8) implies (4.6). □

Proof of Theorem 4. We first show that, for both insertion methods, (4.5) holds. For the nearest insertion, there is for each i by (4.2) a node y_i in T_i such that

$$d(y_i, a_i) \leq d(p, q) \tag{4.9}$$

for all p in T_i and q in $N - T_i$. Lemma 2 says that

$$\text{COST}(T_i, a_i) \leq 2 \cdot d(y_i, a_i) \tag{4.10}$$

and (4.9) and (4.10) imply (4.5). For the cheapest insertion, the cost of inserting a_i is by (4.3) even less than the cost of inserting an a_i chosen to satisfy (4.2). Therefore (4.5) must also hold in this case and Lemma 3 applies to both cases.

The optimal tour can be made into a tree by deleting its longest edge and this longest edge has a length at least OPTIMAL$/n$ where n is the number of nodes in the problem. Since the minimal spanning tree is no longer than this tree,

$$\text{TREE} \leq \left(1 - \frac{1}{n}\right) \cdot \text{OPTIMAL}. \tag{4.11}$$

Equations (4.11), (4.6), and (1.1) imply (4.4). □

COROLLARY. *For a traveling salesman graph on n nodes,* (4.4) *in Theorem* 4 *may be replaced by*

$$\frac{\text{INSERT}}{\text{OPTIMAL}} \leq 2 \cdot \left(1 - \frac{1}{n}\right). \tag{4.12}$$

For the nearest insertion method, a simpler correspondence than that in the proof of Lemma 3 can be established between the cost of the insertion steps and the edges of a minimal spanning tree. Since each a_i is selected in accordance with (4.2), there is an edge (x, a_i) such that x is in T_i and

$$d(x, a_i) = \min\{d(p, q) \text{ for } p \text{ in } T_i \text{ and } q \text{ in } N - T_i\}. \tag{4.13}$$

Let e_i be this edge (x, a_i) and observe from Lemma 2 that

$$\text{COST}(T_i, a_i) \leq 2 \cdot e_i.$$

Moreover, the set of edges $\{e_i \mid 1 \leq i < n\}$ constitute a minimal spanning tree since the method of selecting edges satisfying (4.13) is a method of constructing a minimal spanning tree (Kruskal [9], Prim [13]).

We now show that there exist traveling salesman graphs for which the bound (4.12) is actually achieved. The examples can be interpreted as cities placed uniformly on a circular road. The case for 8 nodes is shown in Fig. 3.5. The optimal path is simply to go around the circle. The insertion methods may construct a path such as that in the figure, a path which goes almost all the way around and then doubles back on itself. Thus, each edge of the circle (except one) is traveled twice instead of the one time actually required, and the ratio of the path obtained to OPTIMAL is roughly two.

THEOREM 5. *For $n \geq 6$, there exists a traveling salesman graph on n nodes such that*

$$\frac{\text{INSERT}}{\text{OPTIMAL}} = 2 \cdot \left(1 - \frac{1}{n}\right) \tag{4.14}$$

for either the nearest insertion or cheapest insertion methods.

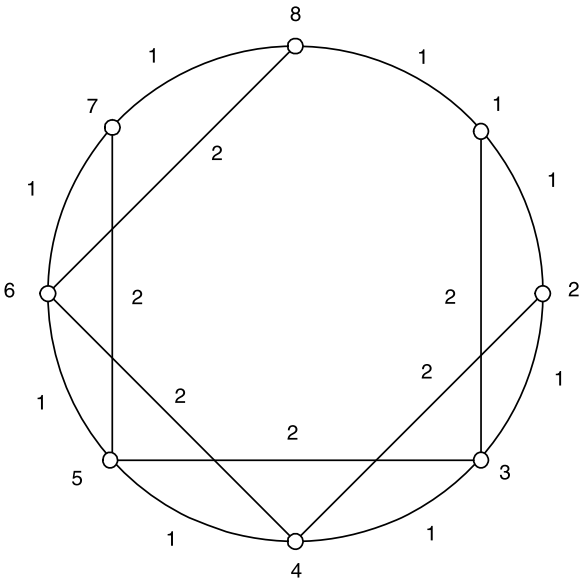

Figure 3.5.

Proof. We define graph (N_n, d_n) as follows:

$$N_n = \{i \mid 1 \leq i \leq n\},$$
$$d_n(i,j) = \text{smallest nonnegative integer } m \text{ such that } i - j \equiv m \pmod{n} \text{ or } j - i \equiv m \pmod{n}.$$

We define T_1 to be the tour on set $\{1\}$, we define

$$T_2 = \{(1,2),(2,1)\}$$

and for $3 \leq i \leq n$ we define

$$T_i = \{(1,2),(i-1,i)\} \cup \{(j,j+2) | 1 \leq j \leq i-2\}.$$

Figure 3.5 shows T_8 for the case $n = 8$.

We define

$$a_i = i + 1 \quad \text{for } 0 \leq i < n.$$

Obviously the T_i defined above are tours. We will show that the T_i together with the a_i satisfy (3.2), (4.2), and (4.3).

T_{i+1} is obtained from T_i by deleting edge $(i-1,i)$ and adding edges $(i-1, i+1)$ and $(i, i+1)$.

We compute that

$$\text{COST}(T_i, a_i) = 2$$

and that $(i-1, i)$ is the edge in T_i which minimizes (3.1); this proves (3.2). We also compute that

$$d(T_i, a_i) = 1$$

because $d(a_{i-1}, a_i) = (i+1) - i = 1$ and (4.2) is satisfied because 1 is the shortest distance between distinct nodes. It can also be calculated that no insertion can cost less than 2 and so (4.2) holds. We omit these calculations but note that they require the assumption $n \geq 6$.

We note finally that the approximation T_n has two edges of length one and $n-2$ edges of length two for a total length of $2 \cdot (n-1)$. The optimal tour is obviously the tour of length n that starts with node 1 and visits the nodes in numerical order. Equation (4.14) is obtained by dividing these two lengths. □

For $i > 3$ in the above proof, the insertion of a_i into T_i to obtain T_{i+1} involves a tie between edges $(i-1, i)$ and $(i-2, i)$, both of which minimize (3.1). An example with no ties in (3.1) is obtained from the example by decreasing the length of all edges greater than 1 by a small number ε. The choice $(i-1, i)$ of the proof becomes the unique choice and the construction proceeds as in the proof. The resulting ratio is very close (depending on ε) to (4.14).

Theorem 5 shows that, in the worst case, nearest insertion can create paths which double back on themselves and are roughly twice as long as necessary. We examined a number of problems with nodes placed randomly on a plane, and observed that the nearest insertion method often produced paths in which portions doubled back on themselves.

5. Farthest Insert

There is another insertion method which has some intuitive and empirical appeal, a method we call farthest insertion.

We say that a tour is constructed by *farthest insertion* if each a_i, $1 \leq i < n$, in the definition of an insertion method satisfies

$$d(T_i, a_i) = \max\{d(T_i, x) \text{ for } x \text{ in } N - T_i\}. \tag{5.1}$$

Contrasting (5.1) with (4.2), we observe that farthest insertion has a max where nearest insertion has a min. The intuitive appeal is that the method establishes the general outline of the approximate tour at the outset and then fills in the details. The early establishment of a general outline is appealing because we expect better performance when the number of nodes is small. Inserting nearby points late in the approximation is appealing because the short edges used late in the procedure are less likely to be accidentally deleted by some still later insertion.

The empirical appeal is that, in a series of experiments, we found that farthest insertion usually produced a better tour than nearest insertion, cheapest

insertion, and the nearest neighbor. For example, when tried on problems obtained by placing 50 nodes randomly on a unit square, nearest insertion was from 7 to 22% worse than farthest insertion, nearest neighbor was from 0 to 38% worse, and cheapest insertion ranged from 7% better to 12% worse. The usual ranking was thus farthest insertion first, cheapest insertion second, nearest insertion third, and nearest neighbor last.

The largest example we tried was 2000 points placed uniformly at random in the unit square. The score was farthest insertion 36.8, nearest insertion 41.4, and nearest neighbor 39.9. A path of length 37.2 was obtained by randomly selecting the order in which points were chosen for insertion. The farthest insertion path was no more than 1.25 times the optimal since the minimal spanning tree had length 29.5.

The advantage of picking random or arbitrary points for insertion is that virtually no computation time is needed to select an arbitrary point. On the 2000 city problem, the nearest neighbor tour was constructed in 751 seconds, the arbitrary insertion in 820 seconds, and the nearest and farthest insertions in 1628 seconds each.

Theorems 2 and 4 tell us that, in the worst case, the nearest neighbor paths become progressively worse than the nearest insertion paths as the number of nodes increase. We found no evidence of such a trend in our experiments. For example, in the 2000 node example described above, nearest neighbor actually did better than nearest insertion.

Altogether, our experiments suggest that the performance of the methods is not strongly tied to their worst case behavior.

6. Some Other Approximation Methods

There are a variety of other approximation methods for which the cost of each step in the construction of the tour corresponds to a unique edge in a minimal spanning tree and for which the reasoning of Lemma 3 and Theorem 4 can be used to demonstrate a worst case ratio bound of 2. In this section, we discuss two such methods.

The first method, which we call *nearest addition*, is similar to nearest insertion. The nearest addition method takes a traveling salesman graph (N, d) with n nodes and constructs a sequence of subtours $T_1, T_2, \ldots, T_n$ so that

1. T_1 consists of a single node a_0;
2. for each $i < n$ there are nodes a_i in $N - T_i$ and b_i in T_i such that
$$d(b_i, a_i) = \min\{d(y, x) \text{ for } y \text{ in } T_i \text{ and } x \text{ in } N - T_i\}. \tag{6.1}$$
and T_{i+1} is constructed from T_i by deleting some edge (c, b_i) from T_i and adding the two edges (c, a_i) and (b_i, a_i);
3. T_n is the approximation.

At each step of the procedure, the closest node is selected and added to the subtour next to the node to which it is the closest.

The increase in length between T_i and T_{i+1} is

$$d(c, a_i) + d(b_i, a_i) - d(c, b_i). \tag{6.2}$$

From the triangle inequality

$$d(c, a_i) \leq d(c, b_i) - d(b_i, a_i) \tag{6.3}$$

so that (6.2) is bounded by $2 \cdot d(b_i, a_i)$. The set of edges (b_i, a_i) selected in accordance with (6.1) is identical to the set of edges that would be selected for the nearest insertion method in accordance with (4.2), and constitutes a minimal spanning tree. Therefore results similar to Lemma 3 and Theorem 4 apply, and the ratio of the obtained tour length to the optimal tour length is bounded by 2.

Another approximation method is one we call *nearest merger*. First, given two disjoint subtours (i.e., subtours having no nodes in common) T_1 and T_2, their merger MERGE(T_1, T_2) is defined as follows:

(a) If T_1 consists of a single node k, then

$$\text{MERGE}(T_1, T_2) = \text{TOUR}(T_2, k)$$

else if T_2 consists of a single node k_1, then

$$\text{MERGE}(T_1, T_2) = \text{TOUR}(T_1, k).$$

(b) If T_1 and T_2 each contain at least two nodes, let a, b, c, d be nodes such that (a, b) is an edge in T_1, (c, d) is an edge in T_2 and

$$d(a, c) + d(b, d) - d(a, b) - d(c, d) \tag{6.4}$$

is minimized. Then MERGE(T_1, T_2) is the tour obtained from T_1 and T_2 by deleting (a, b) and (c, d) and adding (a, c) and (b, d).

The *nearest merger* method takes a problem (N, d) with n nodes and constructs a sequence $S_1, \ldots, S_n$ such that each S_i is a set of $n - i + 1$ disjoint subtours covering all the nodes in N. The sequence is constructed as follows:

1. S_1 consists of n subtours, each containing a single node.
2. For each $i < n$, find an edge (a_i, c_i) such that

$$d(a_i, c_i) = \min\{d(x, y) \text{ for } x \text{ and } y \text{ in different subtours in } S_i\}. \tag{6.5}$$

Then S_{i+1} is obtained from S_i by merging the subtours containing a_i and c_i.

At each step in the procedure, the two closest subtours are merged.

Observe that in a merger corresponding to (6.4), from the triangle inequality

$$d(b,d) \leq d(b,a) + d(a,c) + d(c,d)$$

so that (6.4) is bounded by $2 \cdot d(a,c)$. Also observe that the set of edges (a_i, c_i) chosen in accordance with (6.5) form a minimal spanning tree (Kruskal [9]). From these facts, results similar to Lemma 3 and Theorem 4 can be proved for nearest merger, and so the ratio of the obtained tour length to the optimal tour length is bounded by 2.

We also observe that Theorem 5 is also true for both nearest addition and nearest merger. For the examples in the proof of Theorem 5 both of these methods produce the same approximate tour as nearest insertion and cheapest insertion.

One possible way to improve nearest insertion, cheapest insertion, and nearest addition is to repeat each of these methods for each possible starting node and then take the minimum solution among these. However, for the examples in the proof of Theorem 5, these methods produce tours of the same length for all starting nodes. Therefore the approach of trying all starting nodes does not improve the worst case ratio.

The methods of this section and Sect. 4 are all proven to have constant bounds because of comparisons with the minimal spanning tree. There are also known bounded methods which actually construct a tour by first constructing the minimal spanning tree. One widely known but unpublished method is to construct the minimal spanning tree, double its edges to obtain an Eulerian circuit containing each point at least once, and then make the circuit into a tour by removing extra occurrences of each node. This method also has an upper bound of 2.

The method of Christofides [2] also starts with the minimal spanning tree, but this is converted into an Eulerian circuit by solving the matching problem among the nodes of odd order. This method has an upper bound of $\frac{3}{2}$, an improvement on the bounds for the methods studied here. However, the running time of this method is n^3, which is slower than the n^2 methods studied here.

7. k-Optimality

One approach to obtaining approximate solutions is to first find some tour and then perturb it somewhat to see if a better tour results. If a better tour does result, the original tour is discarded and perturbations on the new tour are tried. Methods of this kind are described in Croes [3], Lin [10], Reiter and Sherman [14], Roberts and Flores [15] and Nicholson [12]. The local optimum obtained by these perturbation methods can be further adjusted to obtain a global optimum (Croes [3]). Lin and Kernighan [11] generalize these techniques in a powerful way.

Define a *k-change* of a tour as the deletion of k edges and their replacement by k other edges so that another tour is obtained.

Define a tour as *k-optimal* (Lin [10]) if no k-change produces a better tour.

Lin [10] describes a method whereby several random initial tours are obtained, each is improved until a 3-optimal tour is obtained and the best of these 3-optimal tours is used.

In this section, we investigate how far a k-optimal tour can be from the optimal tour.

THEOREM 6. *For each $n \geq 8$ there exists a traveling salesman graph having a tour which is k-optimal for all $k \leq n/4$, and for which the length of that tour,* LOCALOPT*, satisfies*

$$\frac{\text{LOCALOPT}}{\text{OPTIMAL}} = 2 \cdot \left(1 - \frac{1}{n}\right). \tag{7.1}$$

Proof. The example is the graph (N_n, d_n) and tour T_n constructed in the proof of Theorem 5. In particular, the tour shown in Fig. 3.5 will be shown to be 2-optimal.

For each n, define the set of edges

$$E_n = \{(1, n)\} \cup \{(i, i+1) \text{ for } 1 \leq i < n\}$$

E_n is the set of edges which have length one. Because of the way function d_n is defined, each pair of points (a, b) from N_n is connected by some path in E_n of length equal to $d_n(a, b)$. For each tour T, there is a circuit $\alpha(T)$ obtained by replacing each edge of T by a path of equal length from E_n. Circuit $\alpha(T)$ has the same length as T and visits each node at least once. Circuit $\alpha(T_n)$ visits node 1 and n once and every other node twice.

For each edge e in E_n and each tour T, we let COUNT(e, T) be the number of times edge e occurs in circuit $\alpha(T)$. For tour T_n we have

$$\text{COUNT}((i, i+1), T_n) = 2 \quad \text{for } 1 \leq i < n, \tag{7.2}$$

$$\text{COUNT}((1, n), T_n) = 0. \tag{7.3}$$

Because the edges of E_n are of unit length, the length $L(T)$ of tour T is given by

$$L(T) = \sum_{e \text{ in } E_n} \text{COUNT}(e, T). \tag{7.4}$$

We say that a tour T is *even* if COUNT(e, T) is even for all e in E_n. We say that a tour T is *odd* if COUNT(e, T) is odd for all e in E_n. We next show that any tour must be either odd or even.

By construction, each node a is the endpoint of exactly two edges of E_n, namely $(a, a+1)$ and $(a, a-1)$ (mod n). Since each occurrence of a in $\alpha(T)$ is associated with two edges of $\alpha(T)$

$$\text{COUNT}((a, a+1), T) + \text{COUNT}((a, a-1), T) = 2 \cdot j_a \qquad (7.5)$$

where j_a is the number of times node a occurs in circuit $\alpha(T)$. Therefore, COUNT$((a, a+1), T)$ and COUNT$((a, a-1), T)$ sum to an even number and are either both even or both odd. Since the edges in E_n form a connected graph, if T were neither odd nor even, some node would have one incident edge with an odd count and its other incident edge with an even count. This contradicts (7.5), so T is either odd or even.

For any tour T, there can be only one edge e in E_n such that COUNT(e, T) = 0 since otherwise the tour could not be connected. Therefore, T_n with its one edge of count 0 and other edges of count 2 (see (7.1) and (7.2)) is the shortest even tour. Consequently, any tour improving on T_n must be odd.

Now suppose that tour T_n is changed by a k-change to an odd tour. Since the largest edges of T_n, are of length 2, the decrease resulting from deleting k edges is at most $2k$. Since at most $2k$ of the counts in $\alpha(T)$ are reduced, and since E_n has n edges, $n - 2k$ edges of E_n do not get their counts decreased. When edges are added to complete the k-change, the counts for the edges not decreased must in fact be increased in order to change from an even number to an odd number. Therefore, the increases are at least $n - 2k$. If the k-change is to improve the tour length, the decreases must be greater than the increases or

$$2k > n - 2k.$$

This inequality is only true when $k > n/4$ so T is indeed k-optimal for $k \leq n/4$.

We already know from the proof of Theorem 5 that T_n and the optimal tour have ratio $2 \cdot (1 - 1/n)$ so the theorem is proved. □

COROLLARY. *For any k and n such that $4 \cdot k \leq n$, the nearest insertion and cheapest insertion methods can result in a k-optimal tour such that*

$$\frac{\text{INSERT}}{\text{OPTIMAL}} = 2 \cdot \left(1 - \frac{1}{n}\right).$$

Proof. We have just shown that the example used to establish Theorem 5 is also k-optimal if $4 \cdot k \leq n$. □

References

[1] M. Bellmore and G. L. Nemhauser. The traveling salesman problem: A survey. *Operations Res.*, 16:538–558, 1968.

[2] N. Christofides. Worst-case analysis of a new heuristic for the traveling salesman problem. In *Symp. on New Directions and Recent Results in Algorithms and Complexity*. Carnegie-Mellon Univ., Pittsburgh, 1976.

[3] G. A. Croes. A method for solving traveling salesman problems. *Operations Res.*, 6:791–812, 1958.

[4] R. Floyd. Algorithm 97, Shortest path. *Comm. ACM*, 5:345, 1962.

[5] J. Gavett. Three heuristic rules for sequencing jobs to a single production facility. *Management Sci.*, 11:166–176, 1965.

[6] W. W. Hardgrave and G. L. Nemhauser. On the relation between the traveling salesman and the longest path problem. *Operations Res.*, 10:647–657, 1962.

[7] L. L. Karg and G. L. Thompson. A heuristic approach to solving traveling salesman problems. *Management Sci.*, 10:225–248, 1964.

[8] R. M. Karp. Reducibility among combinatorial problems. In R. E. Miller and J. W. Thatcher, editors, *Complexity of Computer Computations*, pages 85–103. Plenum, New York, 1972.

[9] J. B. Kruskal. On the shortest spanning subtree of a graph and the traveling salesman problem. *Proc. Amer. Math. Soc.*, 2:48–50, 1956.

[10] S. Lin. Computer solution of the traveling salesman problem. *Bell System Tech. J.*, 44:2245–2269, 1965.

[11] S. Lin and B. W. Kernighan. An effective heuristic algorithm for the traveling salesman problem. *Operations Res.*, 21:498–516, 1973.

[12] T. A. J. Nicholson. A sequential method for discrete optimization problems and its application to the assignment, traveling salesman, and three machine scheduling problems. *J. Inst. Math. Appl.*, 3:362–375, 1967.

[13] R. C. Prim. Shortest connection networks and some generalizations. *Bell System Tech. J.*, 36:1389–1401, 1957.

[14] S. Reiter and G. Sherman. Discrete optimizing. *J. Soc. Indust. Appl. Math.*, 13:864–889, 1965.

[15] S. M. Roberts and B. Flores. An engineering approach to the traveling salesman problem. *Management Sci.*, 13:269–288, 1966.

[16] S. Sahni and T. Gonzales. P-complete problems and approximate solutions. In *IEEE Fifteenth Ann. Symp. on Switching and Automata Theory*, pages 28–32, 1974.