average of 49 seconds on a CDC 7600. The maximum time observed was only 82 seconds.

Since exact approaches to the TSP are, in general, computationally burdensome for large TSP's, a variety of heuristic approaches have found wide use. The next section reviews heuristic techniques for the TSP.

2.3. TSP—HEURISTIC APPROACHES

Heuristics examined. The heuristics we examine fall into three broad classes—tour construction procedures, tour improvement procedures, and composite procedures. *Tour construction procedures* generate an approximately optimal tour from the distance matrix. *Tour improvement procedures* attempt to find a better tour given an initial tour. *Composite procedures* construct a starting tour from one of the tour construction procedures and then attempt to find a better tour using one or more of the tour improvement procedures. Most of these procedures are described in the literature and, hence, will be sketched only briefly; but newer procedures will be studied in more detail. We assume for the sake of simplicity that the costs are symmetric, (i.e. $c_{ij} = c_{ji}$) satisfy the triangle inequality, and are defined for each (i, j) pair, unless otherwise specified.

2.3.1 *Tour construction procedures*

(*a*) *Nearest neighbor procedure* (Rosenkrantz, Stearns, and Lewis[568]).

Step 1. Start with any node as the beginning of a path.

Step 2. Find the node closest to the last node added to the path. Add this node to the path.

Step 3. Repeat step 2 until all nodes are contained in the path. Then, join the first and last nodes.

Worst case behavior:

$$\frac{\text{length of nearest neighbor tour}}{\text{length of optimal tour}} \leq \frac{1}{2}\left\lceil \lg(n) \right\rceil + \frac{1}{2}$$

where lg denotes the logarithm to the base 2, $\lceil X \rceil$ is the smallest integer $\geq X$, and n is the number of nodes in the network.

Number of computations. The nearest neighbor algorithm requires on the order of n^2 computations.

Comments. In a computational setting, the procedure outlined above may be repeated n times, each time with a new node selected as the starting node. The best solution obtained would then be listed as the answer. Notice that this strategy runs in an amount of time proportional to n^3.

(b) *Clark and Wright Savings* (Clark and Wright[145], Golden [291]).

Procedure

Step 1. Select any node as the central depot which we denote as node 1.

Step 2. Compute savings $s_{ij} = c_{1i} + c_{1j} - c_{ij}$ for $i, j = 2,3, \ldots, n$.

Step 3. Order the savings from largest to smallest.

Step 4. Starting at the top of the savings list and moving downwards, form larger subtours by linking appropriate nodes i and j. Repeat until a tour is formed.

Worst case behavior. The worst case behavior for this approach is known for both a sequential and concurrent version. Golden[289] demonstrates that for a sequential version of this algorithm where at each step we select the best savings from the last node added to the subtour, the worst case ratio is bounded by a linear function in $\lg(n)$. Ong[517] has derived a similar result for the concurrent version.

Number of computations. The calculation of the matrix $S = [s_{ij}]$ in step 2 requires about cn^2 operations for some constant c. Next, in step 3, savings can be sorted into nonincreasing order via the "Heapsort" method of Williams[674] and Floyd[226] in a maximum of $cn^2 \lg(n)$ comparisons and displacements. Step 4 involves at most n^2 operations since there are that many savings to consider. Thus, the Clark and Wright savings procedure requires on the order of $n^2\lg(n)$ computations.