Heuristics

We have seen a couple of ways of solving hard integer programs. Depending on the application, these algorithms may be fast to find an optimal solution or, on the contrary, prohibitively time consuming for a computer. For example we know that, in some cases, branch and bound can take an enormous amount of time. When it is impractical to compute an optimal solution, one has to settle for a good (but not necessarily optimal) solution. Such solution procedures are called *heuristic methods* or *heuristics*. Heuristics often have an intuitive justification but they are not guaranteed to produce an optimal solution nor even a good solution, in many cases.

In this chapter, we illustrate the heuristic approach on the traveling salesperson problem. Remember that the problem is the following. A traveling salesperson must visit once each of **n** cities before returning home. He knows the distance between each of the cities and wishes to minimize the total distance traveled. In what order should he visit the cities? We have seen in earlier chapters that, although this problem can in principle be solved by branch and bound combined with cutting planes or by dynamic programming, the time required to find an optimal solution may explode exponentially as the number of cities **n** becomes large. This is an ideal situation for trying heuristics.

There are two main types of heuristics for the traveling salesperson problem: those that construct a tour from scratch and those that improve on a previously obtained solution.

Tour Construction Heuristics

1. **Nearest neighbor:** A starting city is chosen at random and then a tour is constructed by going from the current city to the nearest city which has not yet been visited. When the last city is reached, the path returns to the starting city, thereby completing the tour.

2. **Nearest insertion:** This heuristic starts with a tour that visits two cities very close together. Then the remaining cities are inserted one at a time by removing one edge of the existing tour and linking in the new city between the two ends. The next city to be introduced at each stage is chosen to be one which increases the length of the tour by the smallest possible amount.

3. **Furthest insertion:** This heuristic starts with a tour that visits two cities which are furthest apart. The next city to be inserted is the one that increases the length of the current tour the most when this city is inserted in the best position on the current tour. The rationale behind the furthest insertion heuristic is that some cities will be expensive to insert and we might as well bite the bullet early since all cities have to be visited eventually.

4. **Sweep:** This heuristic locates the center of the map and then rotates a half line around this center. The cities are visited in the order in which they are met by the line as it sweeps around the center

point. This is the only tour construction heuristic which will guarantee a noncrossing tour for points in the plane.

Tour Improvement Heuristics

1. **Two-opt:** This heuristic systematically looks at each pair of nonadjacent edges of the current tour and determines whether the tour length would be decreased by removing these two edges and adding the other possible pair of edges which would result in a tour. If so, the exchange is performed and the search for an improving pair of edges continues. This will result in a noncrossing tour for points in the plane. Moreover, a noncrossing tour will often have its length reduced by this heuristic.

2. **Three-opt:** This is similar to the two-opt heuristic, except that now three edges are removed and the resulting three tour segments are recombined to form a tour.

3. **Lin-Kernighan:** This is a powerful interchange heuristic which proceeds as follows. An edge is removed from the current tour, thereby producing a path. Then one end of this path is joined to some internal node and an edge is removed, thereby yielding a new path. This add/remove operation is then repeated. Let the "gain sum" be the sum of the lengths of the removed edges (except the last) minus the sum of the lengths of the added edges. The add/remove operation continues as long as the gain sum is positive and there still remain unadded/unremoved edges. For each path constructed, the cost of the tour obtained by joining its endpoints is also computed and if it is less than that of the original tour, we keep track of this improved solution. Then when a sequence of possible interchanges has been exhausted, if an improvement was found at any point, the best tour found replaces the original tour and the procedure is repeated.

Exercise 9 *Apply the various tour construction heuristics to the following traveling salesperson problem. Then apply the 2-opt heuristic in each case. The intercity distances are given in the next table. Figure 6.1 provides the locations of the cities on a map.*

	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	6	16	5	11	10	14	8	10	12	15	18	12	15	23
2		10	6	6	6	9	9	8	11	12	14	13	14	18
3			14	7	10	7	16	13	16	12	11	20	17	14
4				8	6	11	4	6	7	11	14	8	11	19
5					3	4	9	6	9	7	9	13	11	13
6						5	6	3	6	6	9	10	8	13
7							11	7	10	5	5	14	10	9
8								4	4	9	13	5	7	17
9									4	6	10	7	6	13
10										7	11	4	4	14
11											4	11	6	8
12												15	10	4
13													6	18
14														12

Michael A. Trick
Thu Oct 26 12:05:11 EDT 1995