

Approximation Algorithms for Routing Problems

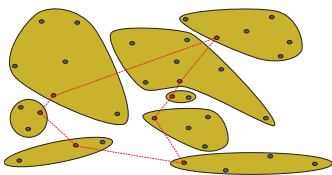
René Sitters
sitters@mpi-inf.mpg.de

Euclidean Group TSP

Instance: A set of disjoint objects and a set of points in the Euclidean plane.

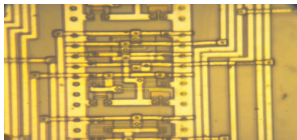
Solution: A tour containing at least one point in each object.

Objective: Minimize the length of the tour.



An instance and the GREEDY solution.

The Group TSP is a generalization of the classical Traveling Salesman Problem. We don't have to connect all points but just one in each group (object). Problems like this appear in VLSI design.



VLSI design.

Algorithm GREEDY:

Step 1. Order the objects by their diameter.

Step 2. Pick any point in the smallest object.

Step 3. For every next object: pick the point that is nearest to the already chosen points.

Step 4. Connect the points by a tour.

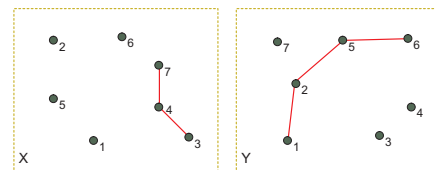
We prove that algorithm GREEDY gives a $9/\alpha$ -approximation, where α is the fatness of the objects. In VLSI design the objects are usual thin, that means, α is almost zero. Finding good approximation algorithm for non-fat objects remains an open problem.

On-line 2-server problem

Instance: Points x_1, x_2, \dots, x_n in metric space X and points y_1, y_2, \dots, y_n in metric space Y .

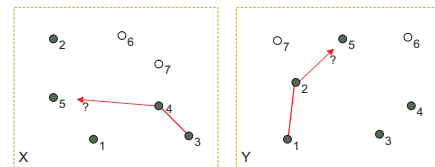
Solution: Paths $u_1, \dots, u_n \in X$ and $v_1, \dots, v_n \in Y$ such that for all i , $u_i = x_i$ or $v_i = y_i$.

Objective: Minimize the sum of path-lengths.



An instance and a feasible solution.

An optimal solution can be found efficiently by dynamic programming. However, we consider the on-line setting in which we have to choose between x_i and y_i without any knowledge of the future pairs $(x_{i+1}, y_{i+1}), (x_{i+2}, y_{i+2}), \dots$.



On-line setting; Future requests are unknown.

Any deterministic on-line algorithm is worse than 10-competitive, that means, it will produce for some instance a solution that is longer than 10 times the optimal solution. We prove that the so called generalized work function algorithm is 70-competitive.



Illustrating the problem; The harddisks contain the same information. With every request we have to choose the disk to read from. Future requests are unknown. The objective is to minimize the total movement.

