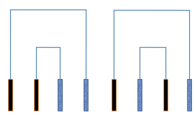
Purdue University — Hambrusch
CS 381 — Fall 2008

Assignment 5

Due: Friday, October 24, 2008 (before class)

1) (12 pts.) You are given the positions of $2n$ pins, equally spaced on a horizontal line. Half of the pins are of type A and the other half are of type B. The goal is to create n pairs, each connecting a type A pin with a type B pin. Every pin is used exactly once. If one pin is at position i and the second is at position j, connecting is done by using a wire of horizontal length $|i - j|$. The objective is to minimize the total horizontal wire length used. The example below shows a feasible solution for 8 pins which does, however, not minimize the total wire length.



Describe an efficient greedy algorithm generating a pairing of the A-B pins achieving minimum total wire length. Assume the A (resp B) pins are given by a list L_A (resp. L_B) giving their positions in increasing order. Analyze the time bound of your algorithm and argue its correctness.

2) (10 pts.) Let $G = (V, E)$ be an n-vertex, m-edge, undirected, weighted graph. Assume all edge weights are distinct. Let T be minimum-cost spanning tree of G.

(i) Assume the weight of every edge in G is increased by a quantity δ (i.e., $w_{(u,v)}$ changes to $w_{(u,v)} + \delta$. Is T still a minimum cost spanning tree? Explain your answer.

(ii) Assume the weight of every edge in G is squared; i.e., it changes from $w_{(u,v)}$ to $w^2_{(u,v)}$. Is T still a minimum cost spanning tree? Explain your answer.

3) (14 pts.) An undirected n-vertex graph $G = (V, E)$ is called a *c+tree* if G is connected and $|E| = n + c$. Describe an $O(n)$ time algorithm for finding a minimum-cost spanning tree in a weighted 8+tree. You can assume that all edge weights are unique.

4) (14 pts.) You just started your internship at a web traffic distribution optimization company. Your supervisor is delighted when she finds out that the algorithms class was your favorite class. She immediately hands you a problem the last intern was unable to solve: "In an undirected graph, the degree of a vertex is the number of adjacent vertices. Given n integers, does there exist a graph whose vertex degrees are identical to this sequence. Design an efficient algorithm that is given a sequence $D = d_1, d_2, ..., d_n$, determines whether there exists an n-vertex graph whose degrees are identical to the entries in D. The graph has no self-loops or multiple edges between the same pair of vertices."

Your supervisor also hands you a theorem (independently proven by S.L. Hakimi and V. Havel in the late 1950's) which she believes to be helpful. It states: *There exists a simple graph with degrees* $d_1 \geq d_2 \geq .... \geq d_n$ *if and only if there exists a simple graph with degrees* $d_2 - 1, d_3 - 1, ..., d_{d_1} - 1, d_{d_1+1} - 1, d_{d_1+2}, ..., d_n$.

Describe an efficient algorithm deciding whether the given degrees can be realized and analyze the time bound. If the degrees can be realized, generate the graph as well.