Graph Algorithms II (part 2 of CSC 282),

http://www.cs.rochester.edu/~stefanko/Teaching/12CS282

"Homework" problem session is in CSB 601, 5pm-6pm on **Wednesday, Oct. 17**; held by Julian Lunger.

Homework is **due Thursday, Oct 18**.

"Exam" problem session is in CSB 601, 5pm-6pm on **Monday, Oct. 24**.

EXAM #3 will be on **Tuesday, Oct. 25**.

1 Homework - solve and turn in

2.1 (**due Thursday, Oct 18**) Suppose we want to run Dijkstra's algorithm on a graph whose edge weights are in the range $\{0, \ldots, W\}$. Show how Dijkstra's algorithm can be made to run in time $O(W|V| + |E|)$.

2.2 (**due Thursday, Oct 18**) We are given n rectangles of sizes $a_1 \times b_1, \ldots, a_n \times b_n$. We want to build the highest tower out of the rectangles. In a tower, if a rectangle of width w is on top of a rectangle of width w' then we require $w \leq w'$. We are allowed to rotate the rectangles (i. e., an $a \times b$ rectangle can be changed into a $b \times a$ rectangle). Give an $O(n)$ algorithm which finds the height of the highest tower.

(For example if the input is $11 \times 11, 8 \times 2, 1 \times 10$ then the solution is a tower of height $29 = 11 + 8 + 10$.)

2.3 (**due Thursday, Oct 18**) Suppose that in Problem 2.2 we change the requirement $w \leq w'$ to $w < w'$, i. e., a rectangle on top of another rectangle has to be strictly thinner. Note that now it can happen that not all rectangles get used (e. g., if we have two squares of the same size). Give a polynomial-time algorithm for the modified problem. (Hint: try to cast the problem as Max-Weight Matching, see definitions below.)

(For example if the input is $2 \times 11, 2 \times 10, 10 \times 10$ then the solution is a tower of height $22 = 2 + 10 + 10$.)

2 Bonus Homework - solve and turn in

Definitions needed for problems 2.3 - 2.5:

Let $G = (V, E)$ be an undirected graph. A *matching* of G is a set of edges $E' \subseteq E$ such that every vertex of G is in *at most one* edge of E'. A *perfect matching* of G is a set of edges $E' \subseteq E$ such that every vertex of G is in *exactly one* edge of E'.

Let $G = (V, E, w)$ be an undirected graph where w are edge-weights. Let $E' \subseteq E$ be a matching of G. The weight of E' is

$$\sum_{e \in E'} w(e).$$

We will consider the following two problems:

Max-Weight Matching:

INPUT: weighted graph $G = (V, E, w)$

OUTPUT: a matching $E' \subseteq E$ of G with the maximum weight.

MAX-WEIGHT PERFECT MATCHING:

INPUT: weighted graph $G = (V, E, w)$

OUTPUT: a perfect matching $E' \subseteq E$ of G with the maximum weight.

You need the following information for problems 2.3 and 2.6:

FACT: there is a polynomial-time algorithm to find a maximum-weight matching.

2.4 (**due Thursday, Oct 18**) Suppose that you have an algorithm (let's call the algorithm A) which solves the MAX-WEIGHT MATCHING problem on any weighed graph $G = (V, E, w)$ in time $O(m+n)$ (where $n := |V|$, $m := |E|$). Show how you can use algorithm A to solve the MAX-WEIGHT PERFECT MATCHING problem in time $O(m + n)$. (HINT: consider the weighted graph $G' = (V, E, w')$ where $w'(e) = w(e) + C$ where C is some large number.)

2.5 (**due Thursday, Oct 18**) Suppose that you have an algorithm (let's call the algorithm B) which solves the MAX-WEIGHT PERFECT MATCHING problem on any weighed graph $G = (V, E, w)$ in time $O(m + n)$ (where $n := |V|$, $m := |E|$). Show how you can use the algorithm B to solve the MAX-WEIGHT MATCHING problem in time $O(m + n)$.

2.6 (**due Thursday, Oct 18**) We have n jobs and m machines. We are given $n \times m$ table T where T_{ij} is the time to complete job i on machine j. Our task is to schedule the jobs on the machines to minimize the **average completion time** (that is, for each job you take the time when it finished and then average over all jobs). Give a polynomial-time algorithm for the problem. (Hint: try to cast the problem as maximum weight matching.)

(For example if we have 3 jobs and 2 machines and

$$T = \begin{pmatrix} 1 & 3 \\ 3 & 2 \\ 4 & 1 \end{pmatrix},$$

then the following schedule is optimal:

- machine 1: 1;
- machine 2: 3, 2.

The completion times of jobs are: 1, 3, 1, the average completion time is 5/3.)