

Problem Set 5—Due Thursday, Feb 19 3:15PM

- (25) **Problem 1.** Suppose you have a 0-1 knapsack problem where the items have the following property: whenever $w_i \leq w_j$ then $v_i \geq v_j$. (thus if we sorted the items by weight they would be in the reverse order of sorting them by value).

Find the set of items of maximum value in this setting in $O(n \log n)$ time. Be sure to justify the correctness of your answer.

- (25) **Problem 2.** Problem 16.1-3 in the text. We want to schedule the n activities using as few lecture halls as possible.

- (10) **Problem 3.** In the STABLE ROOMMATE PROBLEM one is given an even number n and n “preference lists”, L_1, \dots, L_n . Each preference list L_i is a an ordered preference list of the other $n - 1$ people. A *stable matching* of such an instance is a partition of $\{1, \dots, n\}$ into two-element subsets, S , such that there are no dissatisfied pairs: every $\{x, x'\}, \{y, y'\} \in S$ implies:

(x prefers x' to y or y prefers y' to x) AND
(x' prefers x to y or y prefers y' to x') AND
(x prefers x' to y' or y' prefers y to x) AND
(x' prefers x to y' or y' prefers y to x').

(thus we never have one person from each pair wanting to leave to form a new pair). . Prove the following: sometimes there doesn't exist a stable matching by giving a (small) setting where no stable matching exists.

- (20) **Problem 4.** Suppose there are n programs $P_1 P_2 \dots P_n$ running on a computer system. There are also k files $F_1, F_2, \dots F_k$ in this system. Each of the n programs needs exclusive access to two files in order to execute and we denote by a_i and b_i the two files needed by program P_i . A program which has both of the files it needs will run to completion and then free up its two files, but any program which is missing one or both of its files must wait until the missing file(s) become free before it can run.

At any given point in time the state of the system is described by which files have been assigned to each program. A *deadlock* occurs if there is at least one programs which can never finish. For example, if we have two programs both of which need files 1 and 2 and program P_1 has F_1 and program P_2 has file F_2 , then both programs will wait forever for their needed files to become free.

Given the state of the system give an efficient algorithm to determine if a deadlock has occurred. Give the run time of your algorithm and justify both the correctness and time bound.

- (25) **Problem 5.** Suppose we are given an undirected graph $G=(V,E)$. We want to preprocess the graph so we can quickly answer a query of the form: $Path(i,j)$ that returns YES if there is a path from vertex i to vertex j in G , and NO if there is no path.

Your goal is to be able to answer $Path(i,j)$ for any pair i, j in $O(1)$ time.

a) A simple way to solve this problem is to construct an $n \times n$ matrix P such that $P[i, j] = 1$ if there is a path from i to j , otherwise $P[i, j] = 0$. Describe how to construct such a matrix P .(for this part, don't worry about the fastest method).

b) We now want an improved scheme to answer $Path(i,j)$ that only uses $O(n)$ extra space to store the information we use to answer in $O(1)$ time. In addition, we want our preprocessing to be fast.

i) Describe how to do the preprocessing in $O(m + n)$ time (and using $O(n)$ extra space.

ii) Describe how to answer a $Path(i,j)$ query in $O(1)$ time for your method.