Due: 19th of September 2016 at 11:59pm

COMP 2007 – Assignment 3

All submitted work must be done individually without consulting someone elses solutions in accordance with the Universitys Academic Dishonesty and Plagiarism policies.

IMPORTANT! Questions 1a-g and 2a-c should be submitted via Blackboard as pdf (no handwriting!). The implementation required for Question 2d should be done in Ed, and submitted via Ed.

Questions

1. Consider the knapsack problem as we discussed in class. You are not given the actual input, instead you are given the trace of the execution of the knapsack algorithm. The dynamic programming table M is given below. Recall that $M[k, w]$ is the optimal solution that can be obtained using only the first k items and a maximum allowed total weight of w. **Motivate each answer!**

$k \setminus w$	0	1	2	3	4	5	6	7	8	9	10
1	0	0	0	0	0	18	18	18	18	18	18
2	0	0	9	9	9	18	18	27	27	27	27
3	0	0	9	9	12	18	21	27	27	30	30
4	0	0	9	9	12	18	25	27	34	34	37

   (a) What is the number of items in the input instance? **[1 point]**

   (b) What is the maximum weight limit of the knapsack? **[1 point]**

   (c) What are the weights of all the items? **[10 points]**

   (d) What are the values of all the items? **[10 points]**

   (e) What is the value of the best packing having a total weight of at most 8. **[3 points]**

   (f) Which items are included in an optimal solution for $W = 10$? **[10 points]**

   (g) Assume that you got another knapsack with capacity $W = 12$. Fill in the values in the matrix for $W = 11$ and $W = 12$. **[10 points]**

2. Suppose you want to schedule jobs you perform in order to maximize your profit. In each week i, you have a choice between

   (i) not doing any job,

   (ii) doing an easy job with value $\ell_i \geq 0$, or

   (iii) doing a tough job with value $h_i \geq 0$.

   In order to do a tough job, you must rest the previous week, i.e., if you do a tough job in week i, then you must have done no job in week $i - 1$ (you may assume that you rested in week 0). If you do a lightweight job in week i, you are free to do any kind of job (or no job at all) in week $i - 1$.

   (a) Design and analyze a dynamic programming algorithm to determine the maximum total value you can obtain over the course of n weeks, given the values ℓ_i and h_i for each week. First derive and justify a recurrence relation (be sure to be clear about what exactly you denote by OPT etc.). Then turn it into a bottom-up solution. (A more efficient algorithm gives more points.) **[20 points]**

(b) Argue the correctness of your algorithm. **[5 points]**

(c) Argue the time and space complexity of your algorithm. **[10 points]**

(d) Implement your algorithm in Ed. The input is given as shown in Fig. 1. The first line is n, the number of weeks. The following n lines each consists of a 2-tuple (ℓ_i, h_i) of integers. Your task is to output the maximum profit of the instance (a single integer). **[20 points]**

n	5		
ℓ_1 h_1	3	7	Maximal profit: 20
ℓ_2 h_2	5	8	$[h_1, \ell_2, h_4, \ell_5]$
ℓ_3 h_3	2	5	
ℓ_4 h_4	4	7	
...	1	5	

Figure 1: (left) Generic input. (right) Example input instance and the optimal solution for that instance.