IMPERIAL COLLEGE LONDON

DEPARTMENT OF ELECTRICAL AND ELECTRONIC ENGINEERING
EXAMINATIONS 2015

EEE/EIE PART II: MEng, BEng and ACGI

Corrected Copy

ALGORITHMS AND COMPLEXITY

Monday, 15 June 2:00 pm

Time allowed: 1:30 hours

There are TWO questions on this paper.

Answer ALL questions. Question One carries 40% of the marks. Question Two carries 60%.

Any special instructions for invigilators and information for candidates are on page 1.

Examiners responsible First Marker(s) : D.B. Thomas
Second Marker(s) : D.F.M. Goodman

ALGORITHMS AND COMPLEXITY

1. Give a tight bound for each of the following recurrence relations.

   Carefully justify your answers.

   a) $T(n) = 2T(n/3) + 1$,

   [4]

   b) $T(n) = 5T(n/4) + n$,

   [4]

   c) $T(n) = 7T(n/7) + n$,

   [4]

   d) $T(n) = 49T(n/25) + n^{3/2}\log(n)$.

   Hint: This recurrence does not directly fit in the statement of the Master theorem. To derive the asymptotic behaviour of $T(n)$ use the tree decomposition used in the proof of the Master theorem.

   [8]

Master Theorem. Let $T(n)$ be the number of operations performed by an algorithm that takes an input of size n. Assume $T(n)$ satisfies, $T(n) = 0$ for $n = 1$, and for $n \geq 2$

$$T(n) = aT(n/b) + O(n^d),$$

where $a > 0, b > 1$ and $d \geq 0$. Then

$$T(n) = \begin{cases} O(n^d), & \text{if } d > \log_b(a) \\ O(n^d \log(n)), & \text{if } d = \log_b(a) \\ O(n^{\log_b(a)}); & \text{if } d < \log_b(a). \end{cases}$$

2. **Knapsack problem.** We are given n items with values $c_1, c_2, \ldots, c_n$ and weights $w_1, w_2, \ldots, w_n$ respectively. The goal is to pack some of these items in a rucksack in order to maximise the value of the items packed, while satisfying the carrying capacity of the rucksack, given by W. More precisely, we would like to choose a subset $I \subseteq 1..n$ that maximises $\sum_{i \in I} c_i$ given the constraint $\sum_{i \in I} w_i \leq W$.

a) *Greedy approach.* We first propose a greedy approach to solve this problem. More precisely we will consider the items in decreasing order of the ratio of their value to their weight, i.e. c_i/w_i, and add them in this order until we reach the capacity of the rucksack.

i) Derive the complexity of this greedy algorithm. [2]

ii) Does this provide an optimal packing?

Hint: Let $n = 3$, $W = 10$, $w_1 = 6, w_2 = 5, w_3 = 5$ and $c_1 = 7, c_2 = 5, c_3 = 5$.

[3]

iii) Derive a general family of examples for the case $n = 3$ for which the above greedy policy will be suboptimal.

Hint: Consider the case where $w_1 = W$ and $w_2 + w_3 \leq W$ and work out relationships between c_1, c_2, c_3 that lead to a suboptimal packing if we use the above greedy algorithm. [5]

b) We now describe a dynamic programme to solve the knapsack problem optimally. To this end we introduce the following subproblems. Consider the items in some arbitrary order and let $C(v,i)$ be the optimal value one gets from solving the knapsack problem, with the first i items in the chosen order, and where the capacity of the rucksack is given by v. To solve the general problem, we have to find $C(W,n)$.

i) Derive a relationship between $C(v,i)$, $C(v', i-1)$, for some $v' \leq v$. [4]

ii) Propose an algorithm for finding $C(W,n)$ and the corresponding optimal packing. [6]

iii) Derive its complexity in terms of n and W. [4]

iv) Apply the above dynamic programme to the following example: $n = 5$, $W = 11$, $w_1 = 1, w_2 = 2, w_3 = 5, w_4 = 6, w_5 = 7$ and $c_1 = 1, c_2 = 6, c_3 = 18, c_4 = 22, c_5 = 28$, i.e. compute $C(11,5)$ and the items to be packed to achieve optimal packing. [6]