CS60035/CS60086 Selected Topics in Algorithms, Autumn 2016–2017

Class Test

01–September–2016 | CSE-120, 6:00–7:00pm | Maximum marks: 20

Roll no: ________________ **Name:** ______________________________

[*Write your answers in the question paper itself. Be brief and precise. Answer all questions.*]

1. Let $a_1, a_2, \ldots, a_n$ be positive integers with $\sum_{i=1}^{n} a_i = 2S$. The *partition problem* deals with deciding whether there exists a subset $\{i_1, i_2, \ldots, i_k\}$ of indices such that $a_{i_1} + a_{i_2} + \cdots + a_{i_k} = S$. The partition problem is known to be NP-Complete. We develop a dynamic-programming algorithm to solve the partition problem. Assume that each $a_i \leqslant S$, otherwise the output is trivially *false*. We prepare a two-dimensional table $T = (T_{i,j})_{\substack{1 \leqslant i \leqslant n \\ 0 \leqslant j \leqslant S}}$ of Boolean values (*true* and *false*) such that $T_{i,j} = \textit{true}$ if and only if an exact sum of j can be realized by a sub-collection of the first i items $a_1, a_2, \ldots, a_i$. We fill T in a row-major order.

(a) Specify how to initialize the first row, that is, the entries $T_{1,j}$ for $0 \leqslant j \leqslant S$. **(2)**

Solution $T_{1,j} = \textit{true}$ if $j = 0$ or if $j = a_1$, $T_{1,j} = \textit{false}$ otherwise.

(b) Explain how to populate the i-th row for $i \geqslant 2$. **(3)**

Solution $T_{i,j} = \textit{true}$ if $T_{i-1,j} = \textit{true}$ or if $j \geqslant a_i$ and $T_{i-1,j-a_i} = \textit{true}$, $T_{i,j} = \textit{false}$ otherwise.

(c) How do you solve the partition problem after the table T is fully populated? **(2)**

Solution Return $T_{n,S}$.

(d) Prove that the running time of this algorithm is pseudo-polynomial. **(3)**

Solution Since each cell of T can be computed in $O(1)$ time, the running time of this algorithm is $O(nS)$. Let $M = \max_{i=1}^{n} a_i$. Then $S = \frac{1}{2}\sum_{i=1}^{n} a_i \leqslant \frac{1}{2}\sum_{i=1}^{n} M = \frac{1}{2}nM$, so the running time is $O(n^2 M)$ which is polynomial in n and the maximum argument M.

2. Let U be an optimization problem, and $\mathscr{M}$ the set of all feasible solutions for an input I. This exercise focuses on solving U using a local-search algorithm.

(a) What is meant by a neighborhood in the space $\mathscr{M}$? **(1)**

Solution A function $N : \mathscr{M} \to 2^{\mathscr{M}}$ such that for all x, we have $x \in N(x)$, and for all $x, y \in \mathscr{M}$, we have $y \in N(x)$ if and only if $x \in N(y)$.

(b) What are local and global optima for the instance I of U? **(2)**

Solution An $x \in \mathscr{M}$ is called a local optimum if the cost of x is the best in $N(x)$. An $x \in \mathscr{M}$ is called a global optimum if the cost of y is the best in the entire space $\mathscr{M}$.

(c) When is a neighborhood in $\mathscr{M}$ called exact? **(2)**

Solution A neighborhood is called exact if every local optimum is also a global optimum.

Let us now specialize to the problem of sorting n integers $a_1, a_2, \ldots, a_n$, that is, to finding a permutation $\pi = (\pi_1, \pi_2, \ldots, \pi_n)$ of $(1, 2, \ldots, n)$ such that the sequence $a_{\pi_1}, a_{\pi_2}, \ldots, a_{\pi_n}$ is sorted (in non-decreasing order). This is equivalent to maximizing $\sum_{i=1}^{n} i a_{\pi_i}$ over all permutations π of $(1, 2, \ldots, n)$. The feasible-solution space $\mathscr{M}$ is thus the set of all permutations of $(1, 2, \ldots, n)$.

(d) Define a polynomial-sized (in n) neighborhood in this solution space. **(2)**

Solution For any permutation π, the neighborhood $N(\pi)$ consists of all permutations obtained by swapping π_i and π_j for any two $i, j \in \{1, 2, \ldots, n\}$. We allow $i = j$, that is, π is in the neighborhood of itself.

(e) Prove/Disprove: The neighborhood you proposed in Part (d) is exact. **(3)**

Solution Yes, the neighborhood proposed in Part (d) is exact. Suppose that $\pi = (\pi_1, \pi_2, \ldots, \pi_n)$ is not a global optimum. Then, the sequence $a_{\pi_1}, a_{\pi_2}, \ldots, a_{\pi_n}$ is not sorted, that is, there exist indices i, j with $i < j$ such that $a_{\pi_i} > a_{\pi_j}$. This in turn implies that $(j - i)(a_{\pi_i} - a_{\pi_j}) > 0$, that is, $j a_{\pi_i} + i a_{\pi_j} > i a_{\pi_i} + j a_{\pi_j}$. The permutation obtained from π by swapping π_i and π_j is in the neighborhood of π, that is, π is not a local optimum too.