Algorithms and Data Structures 2017/18

Coursework 2

This coursework is due by **4:00pm, on Fri, 16 March 2018** at the ITO. (This is a firm deadline. Please hand in your solution, written on paper, by that time to the ITO.)

This coursework 2 is **summative**. It counts for 25% of the overall course grade.


**Conduct policy**: [http://web.inf.ed.ac.uk/infweb/admin/policies/academic-misconduct](http://web.inf.ed.ac.uk/infweb/admin/policies/academic-misconduct)

1. (a) State the CountingSort algorithm in pseudocode and explain informally how it works. What is the running time of CountingSort? [10 marks]

(b) Explain how CountingSort sorts the following list of keyed objects. The first column contains the key, and the second column contains the object data (in this case just a letter).

<table>
<thead>
<tr>
<th>key</th>
<th>data</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>A</td>
</tr>
<tr>
<td>3</td>
<td>B</td>
</tr>
<tr>
<td>4</td>
<td>C</td>
</tr>
<tr>
<td>5</td>
<td>D</td>
</tr>
<tr>
<td>1</td>
<td>E</td>
</tr>
<tr>
<td>3</td>
<td>F</td>
</tr>
</tbody>
</table>

(c) Explain what a comparison based sorting algorithm is and name two such algorithms. Moreover, state the lower bound on the running time of comparison based deterministic sorting algorithms. Explain in detail why this lower bound does not apply to CountingSort. [5 marks]

(d) Let $n$ denote the number of objects and let $m$ denote the number of keys. For what ratios $m/n$ is CountingSort faster in the worst case than Quicksort if Quicksort chooses the last element of the input list as the pivot? [10 marks]

(e) Let $x_1, \ldots, x_n$ be (not necessarily distinct) integers in the range \{1, \ldots, n\}. Let $(a_1, b_1), \ldots, (a_k, b_k)$ be pairs of integers. Furthermore, for any $1 \leq i \leq k$ let $Z_i$ be the number of indices $j$ such that $a_i \leq x_j \leq b_i$. Devise an algorithm that, given $x_1, \ldots, x_n$ and $(a_1, b_1), \ldots, (a_k, b_k)$, computes the numbers $Z_1, \ldots, Z_k$ in time $O(n + k)$. [15 marks]
2. In this question we use $N$ to denote a network $N = (G = (V, E), c, s, t)$, where $c : V \times V \rightarrow \mathbb{R}^\geq 0$ is the capacity function, $s$ is the source vertex and $t$ is the sink vertex. Let $n = |V|$ and $m = |E|$.

A network flow $f : V \times V \rightarrow \mathbb{R}$ is a function satisfying the following conditions:

(i) Capacity constraints: $f(u, v) \leq c(u, v)$ for all $u, v \in V$.

(ii) Skew-symmetry: $f(u, v) = -f(v, u)$ for all $u, v \in V$.

(iii) Flow conservation: For all $u \in V \setminus \{s, t\}$,

$$\sum_{v \in V} f(u, v) = 0.$$ 

The value of a network flow $f$ is defined as $|f| = \sum_{v \in V} f(s, v)$.

(a) Let $f$ be a flow in $N$. Define the residual network (with respect to $f$) $N_f$. [6 marks]

(b) Let $f$ be a flow in $N$, and $g$ be a flow in the residual network $N_f$. Prove that $f + g : V \times V \rightarrow \mathbb{R}$ is a flow in $N$, by individually demonstrating that the three conditions ((i), (ii), (iii)) above hold for $f + g$. [12 marks]

(c) Describe the Ford-Fulkerson algorithm for computing the Maximum Flow of a given network. [8 marks]

(d) Describe the Edmonds-Karp heuristic for Network Flow. Briefly explain why the running time of the Ford-Fulkerson algorithm is $O(nm^2)$ when the Edmonds-Karp heuristic is used. [8 marks]

(e) A safe-path graph $H = (\mathcal{V}, \mathcal{E})$ is a directed graph with a source vertex $s$ and a sink vertex $t$. The graph is $k$-safe if there are $k$ vertex-disjoint paths from $s$ to $t$ - that is, $k$ paths from $s$ to $t$ such that no two paths share any vertex except $s, t$. The $k$-safe problem has applications to network routing.

Describe an algorithm that, given a directed graph $H = (\mathcal{V}, \mathcal{E})$ as input, runs in $O(|\mathcal{V}| \cdot |\mathcal{E}|)$ time and determines the maximum value $k$ for which $H$ is $k$-safe. Justify the correctness and the running time of your algorithm.

**Hint:** The naive application of network flow to the original graph $H$ will not work, as it does not produce vertex-disjoint paths.