MTAT.03.286: Advanced Methods in Algorithms **University of Tartu**

Solutions for the final exam

Instructor: Dr. Vitaly Skachek January 13th, 2015

Student name: ______________________________________

Student ID: ____

1. This exam contains 10 pages. Check that no pages are missing.
2. It is possible to collect up to 110 points. Try to collect as many points as possible.
3. Justify and prove all your answers (where applicable).
4. All facts and results that were proved or stated in the class can be used in your solution without a proof. Such results need to be rigorously formulated.
5. Any printed and written material is allowed in the class. No electronic devices are allowed.
6. Exam duration is 3 hours.
7. Good luck!

Question 1	
Question 2	
Question 3	
Question 4	
Total	

Question 1 (20 points).

By using Fast Fourier Transform (FFT) algorithm, evaluate the polynomial $P(x) = x^3 - 2x^2 + 2x - 1$ at the complex 4th roots of unity. Show at least one level of recursion.

Answer:

The 4th roots of unity are 1, i, -1 and $-i$, where $i = \sqrt{-1}$.

$$\begin{pmatrix} P(1) \\ P(i) \\ P(-1) \\ P(-i) \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & i & -1 & -i \\ 1 & -1 & 1 & -1 \\ 1 & -i & -1 & i \end{pmatrix} \begin{pmatrix} -1 \\ 2 \\ -2 \\ 1 \end{pmatrix}$$
$$= \left(\frac{\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} -1 \\ -2 \end{pmatrix} + \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 2 \\ 1 \end{pmatrix}}{\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} -1 \\ -2 \end{pmatrix} + \begin{pmatrix} -1 & 0 \\ 0 & -i \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 2 \\ 1 \end{pmatrix}} \right).$$

We apply the algorithm recursively to compute

$$\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} -1 \\ -2 \end{pmatrix} = \begin{pmatrix} -3 \\ 1 \end{pmatrix}$$

and

$$\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 2 \\ 1 \end{pmatrix} = \begin{pmatrix} 3 \\ 1 \end{pmatrix}.$$

By substituting the result back, we obtain

$$\begin{pmatrix} P(1) \\ P(i) \\ P(-1) \\ P(-i) \end{pmatrix} = \left(\frac{\begin{pmatrix} -3 \\ 1 \end{pmatrix} + \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix} \begin{pmatrix} 3 \\ 1 \end{pmatrix}}{\begin{pmatrix} -3 \\ 1 \end{pmatrix} + \begin{pmatrix} -1 & 0 \\ 0 & -i \end{pmatrix} \begin{pmatrix} 3 \\ 1 \end{pmatrix}} \right) = \begin{pmatrix} 0 \\ 1+i \\ -6 \\ 1-i \end{pmatrix}.$$

We obtain that $P(1) = 0$, $P(i) = 1 + i$, $P(-1) = -6$ and $P(-i) = 1 - i$. $\square$

Question 2 (30 points). You are given the following linear-programming (LP) problem:

$$\begin{array}{lrcrcrcl} \textbf{max} & x_1 & - & 2x_2 & + & x_3 & & \\ \textbf{s.t.} & x_1 & + & x_2 & & & \leq & 3 \\ & x_1 & + & & & 2x_3 & \leq & 5 \\ & & & x_2 & - & x_3 & \leq & 2 \\ & \multicolumn{7}{c}{x_1 \geq 0, \;\; x_2 \geq 0, \;\; x_3 \geq 0} \end{array}$$

(a) Solve this LP problem by using the simplex method.

(b) Formulate the dual LP problem.

(c) Verify that your solution of (a) is optimal by the substitution of the point $(\frac{1}{2}, \frac{1}{2}, 0)$ into the dual problem. Explain your answer.

Answer:

(a) We have:

$$\left(\begin{array}{ccc|ccc|c} \textcircled{1} & 1 & 0 & 1 & 0 & 0 & 3 \\ 1 & 0 & 2 & 0 & 1 & 0 & 5 \\ 0 & 1 & -1 & 0 & 0 & 1 & 2 \\ \hline \textcircled{1} & -2 & 1 & 0 & 0 & 0 & 0 \end{array}\right) \begin{array}{l} \\ (2) \leftarrow (2) - (1) \\ \\ (\star) \leftarrow (\star) - (1) \end{array}$$

$$\Longrightarrow \left(\begin{array}{ccc|ccc|c} 1 & 1 & 0 & 1 & 0 & 0 & 3 \\ 0 & -1 & \textcircled{2} & -1 & 1 & 0 & 2 \\ 0 & 1 & -1 & 0 & 0 & 1 & 2 \\ \hline 0 & -3 & \textcircled{1} & -1 & 0 & 0 & -3 \end{array}\right) \begin{array}{l} \\ (2) \leftarrow \frac{1}{2} \cdot (2) \\ (3) \leftarrow (3) + \frac{1}{2} \cdot (2) \\ (\star) \leftarrow (\star) - \frac{1}{2} \cdot (2) \end{array}$$

$$\Longrightarrow \left(\begin{array}{ccc|ccc|c} 1 & 1 & 0 & 1 & 0 & 0 & 3 \\ 0 & -\frac{1}{2} & 1 & -\frac{1}{2} & \frac{1}{2} & 0 & 1 \\ 0 & \frac{1}{2} & 0 & -\frac{1}{2} & \frac{1}{2} & 1 & 3 \\ \hline 0 & -2\frac{1}{2} & 0 & -\frac{1}{2} & -\frac{1}{2} & 0 & -4 \end{array}\right)$$

All coefficients in the last row of the matrix are non-positive. We stop here.

We obtain that $x_1 = 3$, $x_2 = 0$, $x_3 = 1$. The maximum value of the objective function is 4.

(b) The dual problem is:

$$\begin{array}{lrcrcrcl} \textbf{min} & 3y_1 & + & 5y_2 & + & 2y_3 & & \\ \textbf{s.t.} & y_1 & + & y_2 & & & \geq & 1 \\ & y_1 & + & & & y_3 & \geq & -2 \\ & & & 2y_2 & - & y_3 & \geq & 1 \\ & \multicolumn{7}{c}{y_1 \geq 0, \;\; y_2 \geq 0, \;\; y_3 \geq 0} \end{array}$$

(c) We see that the point $(\frac{1}{2}, \frac{1}{2}, 0)$ satisfies all dual constraints and therefore it is a feasible solution. The value of the dual objective function at this point is

$$3 \cdot \frac{1}{2} + 5 \cdot \frac{1}{2} + 2 \cdot 0 = 4 \, ,$$

and it is equal to the maximum of the primal problem found in part (a).

We know that the value of the objective of the primal problem is always less or equal to the value of the objective function of the dual problem, and they are equal only at the point, which is the maximum of the primal and minimum of the dual. Therefore, indeed, 4 is the maximum of the primal problem. □

Question 3 (30 points). Let $\boldsymbol{A}$ be an $n \times n$ matrix, whose entries are '0' and '1'. A *generalized diagonal* in $\boldsymbol{A}$ is a set of exactly n entries in $\boldsymbol{A}$, such that:

- Exactly one entry is selected from each row and from each column;
- Every selected entry is '1'.

Propose an algorithm that finds a generalized diagonal in a given matrix $\boldsymbol{A}$ as above. If no generalized diagonal exists, the algorithm prints a corresponding message. The required time complexity is $O(n^3)$.

Proof correctness of your algorithm and analyze its time complexity.

Answer:

Algorithm

1. Enumerate the rows of $\boldsymbol{A}$ as $1, 2, \cdots, n$, and also the columns of $\boldsymbol{A}$ as $1, 2, \cdots, n$.

2. Construct a flow network $\mathcal{N}(\mathcal{G}(\mathcal{V}, \mathcal{E}), s, t, c)$ as follows:

$$\begin{aligned}
\mathcal{V} &= \{u_i \ : \ i = 1, 2, \cdots, n\} \cup \{v_j \ : \ j = 1, 2, \cdots, n\} \cup \{s, t\} \ ; \\
\mathcal{E} &= \{(u_i, v_j) \ : \ \text{the entry in row } i \text{ and column } j \text{ in } \boldsymbol{A} \text{ is '1'}\} \cup \\
&\qquad \{(s, u_i) \ : \ i = 1, 2, \cdots, n\} \cup \{(v_j, t) \ : \ j = 1, 2, \cdots, n\} \ .
\end{aligned}$$

   Capacity function is $c(e) = 1$ for all edges $e \in \mathcal{E}$.

3. Find a maximum flow F in $\mathcal{N}$ by using Ford-Fulkerson algorithm.

4. Output:
   - If $F < n$ then "there is no generalized diagonal".
   - Otherwise, if the flow $f((u_i, v_j)) = 1$, then the entry in row i and column j of $\boldsymbol{A}$ belongs to the generalized diagonal.

Proof of correctness

Statement 1. If there is a flow of size n, then the entries in rows i and columns j of $\boldsymbol{A}$, such that $f((u_i, v_j)) = 1$, form the generalized diagonal.

Proof. First, since all edges have integral capacities, the total flow is integral. Since every vertex u_i has only one incoming edge, then the outgoing flow from u_i is at most 1, and therefore only one entry from row i will be taken into the generalized diagonal. Similarly, since every vertex v_j has only one outgoing edge, then the incoming flow into each v_j is at most 1, and therefore only one entry from column j will be taken into the generalized diagonal.

Since the total flow is n, all rows and columns of $\boldsymbol{A}$ will be used. Therefore the selected entries form a generalized diagonal. □

Statement 2. If the maximum flow is of size less than n, then there is no generalized diagonal in $\boldsymbol{A}$.

Proof. By contrary, assume that there is a generalized diagonal in $\boldsymbol{A}$. For each entry in row i and and column j define the following flow in $\mathcal{N}$:

$$f((s, u_i) = f((u_i, v_j)) = f((v_j, t)) = 1 \ .$$

For all other edges e define flow $f(e) = 0$. It is a legal flow: vertex and edge rules are preserved. The total flow is equal to the number of elements in the generalized diagonal, which is n. Contradiction. Therefore, there is no generalized diagonal in $\boldsymbol{A}$. □

Time complexity

- The network $\mathcal{N}$ contains $O(n)$ vertices and $O(n^2)$ edges. Construction of $\mathcal{N}$ requires $O(n^2)$ operations.
- Since there are $O(n)$ vertices in the network, the length of any augmenting path is bounded from above by $O(n)$. Therefore, finding one augmenting path by using Ford-Fulkerson algorithm takes $O(n)$ time.
- Since each augmenting path improves the total flow by one, and the total flow is bounded from above by $O(n^2)$, the algorithm finds at most $O(n^2)$ augmenting paths.
- We find that the total complexity is $O(n^3)$. □

Question 4 (30 points).

A tourist wants to place n objects $a_1, a_2, \cdots, a_n$ into k suitcases. For each $i = 1, 2, \cdots, n$, the weight of a_i is w_i kilograms. The capacity of each suitcase is not limited. The goal is to minimize the weight of the most heavy suitcase.

The following greedy algorithm is used (it is the same algorithm as in Questions 3 and 4 of Homework 5). The objects are ordered in an arbitrary order. The tourist always places the object under consideration into the least heavy suitcase.

(a) Show that, for $k = 3$, at the end of the algorithm run, the weight of the most heavy suitcase is at most $\frac{5}{3}$ times the optimum.

(b) Generalize the result from part (a) and conclude that for general k, the weight of the most heavy suitcase is at most $2 - \frac{1}{k}$ times the optimum. (Note that it is a better approximation factor than in Question 3 of Homework 5).

Answer:

(a) Let OPT be the weight of the most heavy suitcase in the optimal solution. Let x the weight of the *last* object to be placed in the *most heavy* suitcase. Denote by $S = \sum_{i=1}^{n} w_i$ the total weight of all the objects.

- We have $x \leq \text{OPT}$.
- We also have that $S/3 \leq \text{OPT}$, i.e. the optimum cannot be smaller than the average weight of the suitcases.
- Before the tourist places the last object in the most heavy suitcase, the total weight of the objects that are already placed in the suitcases is at most $S - x$. Therefore, the weight of the suitcase, where that object is placed, before that object is placed, is at most
$$\frac{S-x}{3} \, .$$
When the last object is placed, the weight of that suitcase becomes at most
$$\frac{S-x}{3} + x = \frac{S+2x}{3} \leq \frac{3 \cdot \text{OPT} + 2 \cdot \text{OPT}}{3} = \frac{5}{3} \cdot \text{OPT} \, .$$

□

(b) Similarly to part (a), let OPT be the weight of the most heavy suitcase in the optimal solution. Let x the weight of the *last* object to be placed in the *most heavy* suitcase. Denote $S = \sum_{i=1}^{n} w_i$.

- We have $x \leq \text{OPT}$ and $S/k \leq \text{OPT}$.
- Before the tourist places the last object in the most heavy suitcase, the total weight of the objects that are already placed in the suitcases is at most $S - x$. Therefore, the weight of the suitcase, where that object is placed, before that object is placed, is at most
$$\frac{S-x}{k} \, .$$

When the last object is placed, the weight of that suitcase becomes at most

$$\frac{S-x}{k} + x = \frac{S+(k-1)x}{k} \leq \frac{k \cdot \mathrm{OPT} + (k-1) \cdot \mathrm{OPT}}{k} = \left(2 - \frac{1}{k}\right) \cdot \mathrm{OPT}\ .$$

□