

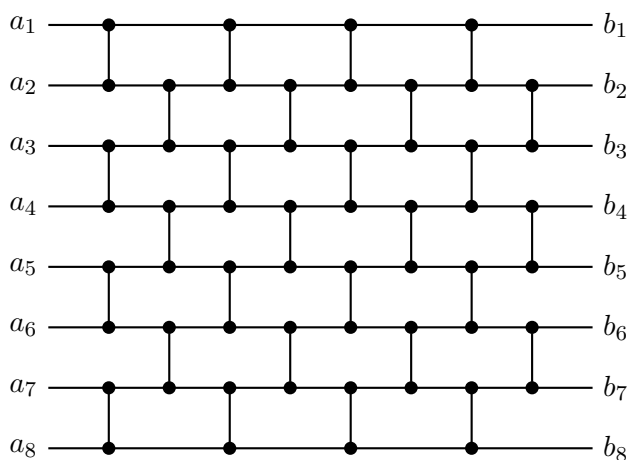
# 1 Original Questions

**Question 1** (CLRS, Question 27.2-2). Prove that a comparison network with  $n$  inputs correctly sorts the input sequence  $\langle n, n-1, \dots, 1 \rangle$  if and only if it correctly sorts the  $n-1$  zero-one sequences  $\langle 1, 0, 0, \dots, 0, 0 \rangle, \langle 1, 1, 0, \dots, 0, 0 \rangle, \dots, \langle 1, 1, 1, \dots, 1, 0 \rangle$ .

**Question 2.** [CLRS, Question 27.2-5] Prove that an  $n$ -input sorting network must contain at least one comparator between the  $i$ -th and  $(i+1)$ -st lines for all  $i = 1, 2, \dots, n-1$ .

**Question 3.** [CLRS, Question 27.4-3] Show that any network that can merge 1 item with  $n-1$  sorted items to produce a sorted sequence of length  $n$  must have depth at least  $\log n$ .

**Question 4.** [CLRS, Problem 27-1] An **odd-even-sorting network** on  $n$  inputs  $\langle a_1, a_2, \dots, a_n \rangle$  is a transposition sorting network with  $n$  levels of comparators connected in the “brick-like” pattern illustrated below.



As can be seen in the figure, for  $i = 1, 2, \dots, n$  and  $d = 1, 2, \dots, n$ , line  $i$  is connected by a depth- $d$  comparator to line  $j = i + (-1)^{i+d}$  if  $1 \leq j \leq n$ .

Prove that odd-even sorting networks actually sort.

**Question 5.** [CLRS, Question 4.2-7] Show how to multiply the complex numbers  $a + bi$  and  $c + di$  using only three multiplications of real numbers. The algorithm should take  $a, b, c$ , and  $d$  as input and produce the real component  $ac - bd$  and the imaginary component  $ad + bc$  separately.

**Question 6.** What can be said about the relation between the time complexity for multiplying two arbitrary square matrices  $A$  and  $B$  and the time complexity for multiplying a matrix  $C$  with itself?

**Question 7.** [CLRS, Question 4.2-2] Write pseudocode for Strassen’s algorithm.

**Question 8.** [CLRS: 29.1-5] Convert the following linear program into slack form:

$$\begin{array}{llllll}
\text{maximize} & 2x_1 & & & - & 6x_3 \\
\text{subject to} & & & & & \\
& x_1 & + & x_2 & - & x_3 & \leq & 7 \\
& 3x_1 & - & x_2 & & & \geq & 8 \\
& -x_1 & + & 2x_2 & + & 2x_3 & \geq & 0 \\
& & & x_1, x_2, x_3 & & & \geq & 0
\end{array}$$

What are the basic and non-basic variables?

**Question 9.** [CLRS: 29.1-6] Show that the following linear program is infeasible:

$$\begin{array}{llllll}
\text{maximize} & 3x_1 & - & 2x_2 \\
\text{subject to} & & & \\
& x_1 & + & x_2 & \leq & 2 \\
& -2x_1 & - & 2x_2 & \leq & -10 \\
& & & x_1, x_2 & \geq & 0
\end{array}$$

**Question 10.** [CLRS: 29.1-7] Show that the following linear program is unbounded:

$$\begin{array}{llllll}
\text{maximize} & x_1 & - & x_2 \\
\text{subject to} & & & \\
& -2x_1 & + & x_2 & \leq & -1 \\
& -x_1 & - & 2x_2 & \leq & -2 \\
& & & x_1, x_2 & \geq & 0
\end{array}$$

**Question 11.** [CLRS: 29.1-8] Suppose that we have a general linear program with  $n$  variables and  $m$  constraints, and suppose we convert it into standard form. Give an upper bound on the number of variables and constraints in the resulting linear program.

**Question 12.** [CLRS: 29.1-9] Give an example of a linear program for which the feasible region is not bounded, but the optimal objective value is finite.

**Question 13.** [CLRS: 29.2-5] Rewrite the linear program for maximum flow so that it uses only  $O(V + E)$  constraints.

**Question 14.** [CLRS: 29.3-6] Solve the following linear program using SIMPLEX:

$$\begin{array}{llllll}
\text{maximize} & 5x_1 & - & 3x_2 \\
\text{subject to} & & & \\
& x_1 & - & x_2 & \leq & 1 \\
& 2x_1 & + & x_2 & \leq & 2 \\
& & & x_1, x_2 & \geq & 0
\end{array}$$

**Question 15.** [CLRS: 29.5-5] Solve the following linear program using SIMPLEX:

$$\begin{array}{llllll}
\text{maximize} & x_1 & + & 3x_2 & & \\
\text{subject to} & & & & & \\
& x_1 & - & x_2 & \leq & 8 \\
& -x_1 & - & x_2 & \leq & -3 \\
& -x_1 & + & 4x_2 & \leq & 2 \\
& x_1, x_2 & & & \geq & 0
\end{array}$$

**Question 16.** Let  $G = (V, E)$  be an undirected graph with maximum degree  $\Delta$ . A dominating set is a subset of vertices  $S \subseteq V$  so that for every vertex  $u \in V$  there exists a vertex  $v \in S$  with  $\{u, v\} \in E(G)$ . The goal is to find a dominating set as small as possible. Design an approximation algorithm based on greedy for the problem and analyse the quality of its solution.

**Question 17.** Given an undirected graph  $G = (V, E)$ , a *vertex cover* of  $G$  is a set of vertices  $C \subseteq V$  so that each edge in  $G$  is incident to at least one vertex in  $C$ . A minimum vertex cover is a vertex cover with smallest possible size  $|C|$ . Consider a greedy approach which iteratively adds the vertex with the highest degree to  $C$  and then removes all covered edges from  $E$ . Find an example that shows that this greedy algorithm does not always find the optimum solution.

**Question 18.** [CLRS: 35.1-3, this one improves on the previous question and is marked with a “★” in CLRS] Professor Bündchen proposes the following heuristic to solve the vertex-cover problem. Repeatedly select a vertex of highest degree, and remove all of its incident edges. Give an example to show that the professor’s heuristic does not have an approximation ratio of 2. (*Hint:* Try a bipartite graph with vertices of uniform degree on the left and vertices of varying degree on the right.)

**Question 19.** [CLRS: 35.3-3] Show how to implement GREEDY-SET-COVER in such a way that it runs in time  $O(\sum_{S \in \mathcal{F}} |S|)$ .

**Question 20.** [CLRS: 35.2-1] Suppose that a complete undirected graph  $G = (V, E)$  with at least 3 vertices has a cost function that satisfies the triangle inequality. Prove that  $c(u, v) \geq 0$  for all  $u, v \in V$ .

**Question 21.** [CLRS: 35.2-5] Suppose that the vertices for an instance of the travelling-salesman problem are points in the plane and that the cost  $c(u, v)$  is the euclidean distance between points  $u$  and  $v$ . Show that an optimal tour never crosses itself.

**Question 22.** [CLRS: 35.2-3] Show how in polynomial time we can transform one instance of the travelling-salesman problem into another instance whose cost function satisfies the triangle inequality. The two instances must have the same set of optimal tours. Explain why such a polynomial-time transformation does not contradict the inapproximability result (Theorem 35.3), assuming that  $P \neq NP$ .

**Question 23.** [CLRS: 35.4-2] The **MAX-CNF satisfiability problem** is like the MAX-3-CNF satisfiability problem, except that it does not restrict each clause to have exactly 3 literals. Give a randomized 2-approximation algorithm for the MAX-CNF satisfiability problem.

**Question 24.** [CLRS: Problem 35-1] Suppose that we are given a set of  $n$  objects, where the size  $s_i$  of the  $i$ th object satisfies  $0 < s_i < 1$ . We wish to pack all the objects into the minimum number of unit-size bins. Each bin can hold any subset of the objects whose total size does not exceed 1.

The **first-fit** heuristic takes each object in turn and places it into the first bin that can accommodate it. Let  $S := \sum_{i=1}^n s_i$ .

1. Argue that the optimal number of bins required is at least  $\lceil S \rceil$ .
2. Argue that the first-fit heuristic leaves at most one bin less than half full.
3. Prove that the number of bins used by the first-fit heuristic is never more than  $\lceil 2S \rceil$ .
4. Prove an approximation ratio of 2 for the first-fit heuristic.
5. Give an efficient implementation of the first-fit heuristic, and analyse its running time.

**Question 25.** Consider the following algorithm for MAX-CUT on an unweighted, undirected graph  $G = (V, E)$ , which can be regarded as an iterative colouring procedure with three colors possible, grey (=unassigned), red (assigned to  $S$ ) and blue (assigned to  $V \setminus S$ ). Initially, all vertices are grey. Then the algorithm does the following in each step: If there is a grey vertex  $u$  which has more blue than red neighbours color it blue, if there is a grey vertex  $u$  which has more red than blue neighbours color it red. Otherwise, take a grey vertex and color it arbitrarily. Prove that this algorithm returns a 2-approximation.

## 2 Additional Questions

**Question 26.** Prove that any sorting network must have depth  $\Omega(\log n)$ .

**Question 27.** Prove that the set of feasible solutions of a linear program in standard form forms a convex set.

**Question 28.** [Thanks to the student for mentioning this question (and answer).] Find a linear program which has at least one optimal solution that is not a vertex.

**Question 29.** [Thanks to the student for mentioning this question.] Consider the linear program for the minimum-weight shortest-path from  $s$  to  $t$  from the lecture notes (Slide 23 from III Linear Programming).

1. What happens if there exists a negative-weight cycle?
2. Prove that, if there are no negative-weight cycles, the optimal solution  $\bar{d}_t$  of the linear program equals the correct distance  $d_t$ .
3. Find a counter-example in which the linear program does not compute all values  $d_v$  correctly. How would you formulate the single-source-shortest path problem as a linear program?

**Question 30.** How can you implement APPROX-VERTEX-COVER in time  $O(V + E)$ ?

**Question 31.** [CLRS: Problem 35.3-3] Consider the analysis of GREEDY-SET-COVER (Theorem 35.4). Show that the following weaker form of Theorem 35.4 is trivially true:

$$|\mathcal{C}| \leq |\mathcal{C}^*| \cdot \max\{|S| : S \in \mathcal{F}\}$$

**Question 32.** Consider the problem SUBSET-SUM. Design a simple Greedy algorithm which runs in polynomial-time and achieves an approximation ratio of 2.

**Question 33.** Consider the algorithm APPROX-SUBSET-SUM from the lecture. Prove formally that for every element  $y$ , at most  $t$ , which can be written as a sum of a subset of  $\{x_1, x_2, \dots, x_n\}$ , there exists an element  $z \in L_n$  (the list in iteration  $n$  *after* the trimming operation), such that

$$\frac{y}{(1 + \delta)^n} \leq z,$$

where  $0 < \delta < 1$  is the trimming parameter.

**Question 34.** Recall the subtour elimination procedure from Lecture 10: In order to eliminate a subtour going through cities in  $S$  only, we add the following constraint:

$$\sum_{i \in S, j \notin S} x(\max(i, j), \min(i, j)) \geq 2.$$

Prove that adding this constraint to the linear program is equivalent to adding the constraint

$$\sum_{i \in S, j \in S, i < j} x(i, j) \leq |S| - 1.$$

**Question 35.** Consider the following problem. Given an undirected, connected graph  $G = (V, E)$  with non-negative, integral edge capacities  $c(u, v)$  for each edge  $(u, v) \in E(G)$  and  $|E| \geq |V| = n$ , the goal is to find a subset  $E' \subseteq E$  with  $|E'| = n$  so that (i)  $E'$  connects all vertices and (ii)  $\sum_{e \in E'} c(e)$  is minimized. Either prove that this problem is NP-hard or design a polynomial-time algorithm.

**Question 36.** Find an example of a graph in the Euclidean space, with as few vertices as possible, so that the optimal TSP tour does not include a minimum spanning tree.