# Problem Set 8 Solutions

This problem set is due **at 11:59pm** on **Friday, April 24, 2015.**

---

**Exercise 8-1.** Read CLRS, Chapter 29.

**Exercise 8-2.** Exercise 29.2-2.

**Exercise 8-3.** Exercise 29.2-4.

**Exercise 8-4.** Read CLRS, Chapter 34.

**Exercise 8-5.** Exercise 34.2-8.

**Exercise 8-6.** Exercise 34.3-5.

---

**Problem 8-1. A Simple Simplex Example** [25 points] Consider a linear program (LP) consisting of two variables $x_1$ and $x_2$ satisfying the following three constraints:

$$x_1 + x_2 \leq 10$$
$$x_2 \geq 4x_1 - 20$$
$$x_1 + 3x_2 \leq 24$$
$$x_1, x_2 \geq 0$$

The goal is to maximize the value of the objective function $p = 4x_1 + x_2$.

**(a)** [5 points] Draw a diagram of the feasible region.

**Solution:**
See Figure 1.

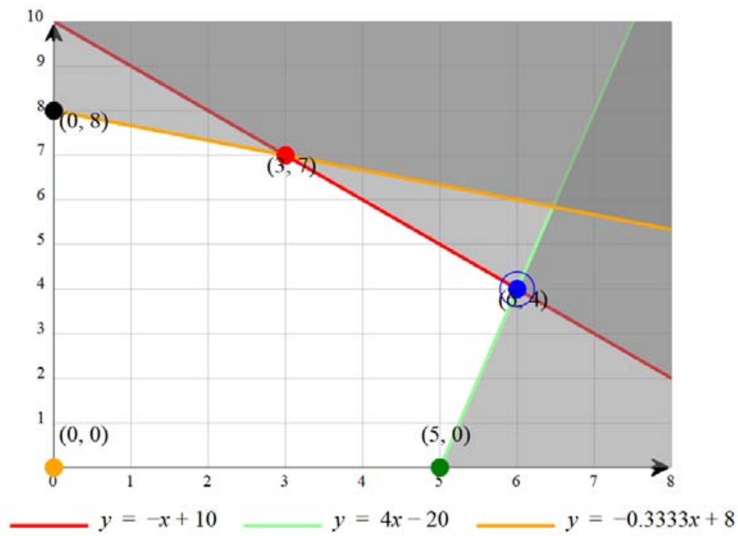


**Figure 1**: The feasible region of the LP problem. Image courtesy of *Finite mathematics & Applied calculus* (http://www.zweigmedia.com).

**(b)** [5 points] Write the given LP in standard form, and transform this standard form representation into slack form.

**Solution:**
Standard form:
Maximize $p = 4x_1 + x_2$, subject to:

$$\begin{aligned} x_1 + x_2 &\leq 10 \\ 4x_1 - x_2 &\leq 20 \\ x_1 + 3x_2 &\leq 24 \\ x_1, x_2 &\geq 0 \end{aligned}$$

Slack form: Introduce three new variables $x_3, x_4, x_5$.
Maximize $p = 4x_1 + x_2$, subject to:

$$\begin{aligned} x_3 &= 10 - x_1 - x_2 \\ x_4 &= 20 - 4x_1 + x_2 \\ x_5 &= 24 - x_1 - 3x_2 \\ x_1, \ldots, x_5 &\geq 0 \end{aligned}$$

**(c)** [10 points] Use Simplex to solve the resulting slack form LP. Identify the pivots you choose and give the resulting modified LPs and the successive feasible solutions. Indicate the successive solutions on your diagram from Part (a).

**Solution:** Start with $x_1 = x_2 = 0$, $x_3 = 10$, $x_4 = 20$, $x_5 = 24$, that is, the solution $(0, 0, 10, 20, 24)$, with objective function value $p = 0$.

The first nonbasic variable we select for pivoting can be either $x_1$ or $x_2$, since both have positive coefficients in the objective function. To be specific, let's choose $x_2$. As we increase $x_2$, the values of $x_3$ and $x_5$ decrease. The limiting constraint is the one for $x_5$: we can only increase $x_2$ to $8$, because any more would make $x_5$ negative. We exchange $x_2$ with $x_5$: We solve the third constraint for $x_2$, obtaining $x_2 = 8 - \frac{1}{3}x_1 - \frac{1}{3}x_5$. Then we substitute, resulting in the following new LP:

Maximize $p = 8 + \frac{11}{3}x_1 - \frac{1}{3}x_5$, subject to:

$$\begin{aligned} x_3 &= 2 - \frac{2}{3}x_1 + \frac{1}{3}x_5 \\ x_4 &= 28 - \frac{13}{3}x_1 - \frac{1}{3}x_5 \\ x_2 &= 8 - \frac{1}{3}x_1 - \frac{1}{3}x_5 \\ x_1, \ldots, x_5 &\geq 0 \end{aligned}$$

We get a new solution by setting the non-basic variables, $x_1$ and $x_5$ equal to $0$ and calculating the others: $(0, 8, 2, 28, 0)$. The value of the objective function is now $p = 8$.

The only option now for pivoting is $x_1$, because the coefficient of $x_5$ in the objective function is negative. As we increase $x_1$, the values of $x_2$, $x_3$, and $x_4$ all decrease. The limiting constraint is the one for $x_3$; we can only increase $x_1$ to $3$. We exchange $x_1$ with $x_3$, and solve the first constraint for $x_1$, obtaining $x_1 = 3 - \frac{3}{2}x_3 + \frac{1}{2}x_5$. We get the following new LP:

Maximize $p = 19 - \frac{11}{2}x_3 + \frac{3}{2}x_5$, subject to:

$$\begin{aligned} x_1 &= 3 - \frac{3}{2}x_3 + \frac{1}{2}x_5 \\ x_4 &= 15 + \frac{13}{2}x_3 - \frac{5}{2}x_5 \\ x_2 &= 7 + \frac{1}{2}x_3 - \frac{1}{2}x_5 \\ x_1, \ldots, x_5 &\geq 0 \end{aligned}$$

We get a new solution by setting $x_3$ and $x_5$ to $0$, obtaining $(3, 7, 0, 14, 0)$. The value of the objective function is now $p = 19$.

Next, we pivot on $x_5$. Increasing $x_5$ causes $x_4$ and $x_2$ to decrease, with the limiting constraint being the one for $x_4$. We exchange $x_5$ with $x_4$, and obtain $x_5 = 6 + \frac{13}{5}x_3 - \frac{2}{5}x_4$. We obtain the following new LP:

Maximize $p = 28 - \frac{8}{5}x_3 - \frac{3}{5}x_4$,

$$\begin{aligned} x_1 &= 6 - \frac{1}{5}x_3 - \frac{1}{5}x_4 \\ x_5 &= 6 + \frac{13}{5}x_3 - \frac{2}{5}x_4 \\ x_2 &= 4 - \frac{4}{5}x_3 + \frac{1}{5}x_4 \\ x_1, \ldots, x_5 &\geq 0 \end{aligned}$$

We get a new solution by setting $x_3$ and $x_4$ to $0$, obtaining $(6, 4, 0, 0, 6)$, with an objective function value of $28$. At this point no further pivots are possible (their coefficients in the objective function are both negative). Thus, an optimal solution is $x_1 = 6, x_2 = 4$.

In the above process, we started from the basic solution (0,0) meaning $x_1 = 0, x_2 = 0$, gradually improved our estimates through (0,8), (3,7), and finally arrived at the final solution (6,4). In Figure 1, this corresponds to traversing four corners of the white region starting from the origin in the clockwise direction.

**(d)** [5 points] Give the dual LP of your standard-form LP from Part (b) and give its optimal value. (*Hint: Use your solution to Part (c).*)

**Solution:**

The standard-form LP from Part (b) is: Maximize $p = 4x_1 + x_2$, subject to:

$$\begin{aligned} x_1 + x_2 &\leq 10 \\ 4x_1 - x_2 &\leq 20 \\ x_1 + 3x_2 &\leq 24 \\ x_1, x_2 &\geq 0 \end{aligned}$$

As in CLRS p. 880, the dual LP uses new variables $y_1$, $y_2$, and $y_3$. The LP is: Minimize $10y_1 + 20y_2 + 24y_3$, subject to:

$$\begin{aligned} y_1 + 4y_2 + y_3 &\geq 4 \\ y_1 - y_2 + 3y_3 &\geq 1 \\ y_1, y_2, y_3 &\geq 0 \end{aligned}$$

By LP duality (Theorem 29.10), the minimum value for this LP is $28$. This value is attained when $y_1 = \frac{8}{5}$, $y_2 = \frac{3}{5}$, and $y_3 = 0$. You can find this solution manually, or by considering the final slack form LP in your solution in Part (c) and using formula (29.91) on p. 882.

**Problem 8-2. NP-Completeness** [25 points]

In this problem, you will prove NP-completeness of a few decision problems. To prove NP-hardness, you may reduce from any problem that has been shown, in class or in CLRS, to be NP-complete.

**(a)** [5 points]

Let TRIPLE-SAT denote the following decision problem: given a Boolean formula $\phi$, decide whether $\phi$ has at least three distinct satisfying assignments. Prove that TRIPLE-SAT is NP-complete.

**Solution:** To show that TRIPLE-SAT is in NP, for any input formula $\phi$, we need only guess three distinct assignments and verify that they satisfy $\phi$.

To show that TRIPLE-SAT is NP-hard, we reduce SAT to it. Let $\phi$ denote the input Boolean formula to a SAT problem and suppose that the set of variables in $\phi$ are $X = \{x_1, \ldots, x_n\}$. We construct a TRIPLE-SAT problem with a Boolean formula $\phi'$ over a new variable set $X'$ as follows:

- $X' = \{x_1, \ldots, x_n, y, z\}$.
- $\phi' = \phi$.

Now we claim $\phi$ is satisfiable iff $\phi'$ has at least 3 satisfying assignments. If $\phi$ is satisfiable, then we can augment any particular assignment by adding any of the $4$ possible pairs of values for $\{y, z\}$ to give at least four satisfying assignments overall. On the other hand, if $\phi$ is not satisfiable, then neither is $\phi'$.

**(b)** [10 points] In Problem Set 1, we considered how one might locate donut shops at some of the vertices of a street network, modeled as an arbitrary undirected graph $G = (V, E)$. Each vertex $u$ has a nonnegative integer value $p(u)$, which describes the potential profit obtainable from a shop located at $u$. Two shops cannot be located at adjacent vertices. The problem was to design an algorithm that outputs a subset $U \subseteq V$ that maximizes the total profit $\sum_{u \in U} p(u)$. No doubt, you found an algorithm with time complexity that was exponential in the graph parameters. Now we will see why.

Define DONUT to be the following decision problem: given an undirected graph $G = (V, E)$, given a mapping $p$ from vertices $u \in V$ to nonnegative integer profits $p(u)$, and given a nonnegative integer $k$, decide whether there is a subset $U \subseteq V$ such that no two vertices in $U$ are neighbors in $G$, and such that $\sum_{u \in U} p(u) \geq k$. Prove that DONUT is NP-hard. (*Hint:* Try a reduction from 3SAT.)

Also, explain why this implies that, if there is a polynomial-time algorithm to solve the original problem, i.e., to output a subset $U$ that maximizes the total profit, then P = NP.

**Solution:**

Let $\phi = C_1 \wedge C_2 \wedge \ldots C_m$ be the input formula to a 3SAT problem, where each clause $C_c$ has three literals chosen from $\{x_i, \bar{x}_i \| 1 \leq i \leq n\}$. We construct a DONUT problem $\langle G, p, k \rangle$ as follows.

The vertices $V$ of $G$ are $\{v_{c,j} \| 1 \leq c \leq m, 1 \leq j \leq 3\}$, where $v_{c,j}$ corresponds to literal $j$ in clause $C_c$. We label each vertex $v_{c,j}$ with $x_i$ or $\bar{x}_i$, whichever appears in position $j$ of clause $C_c$. The edges $E$ of $G$ are of two types:

- For each clause $C_c$, an edge between each pair of vertices corresponding to literals in clause $C_c$, that is, between $v_{c,j_1}$ and $v_{c,j_2}$ for $j_1 \neq j_2$.
- For each $i$, an edge between each pair of vertices for which one is labeled by $x_i$ and the other by $\bar{x}_i$.

The function $p$ maps all vertices to 1. The threshold $k$ is equal to $m$. We claim that $\phi$ is satisfiable iff the total profit in the DONUT problem $\langle G, p, k \rangle$ can be at least $k$.

First, suppose that $\phi$ is satisfiable. Then there is some truth assignment $A$ mapping the variables to $\{\mathit{true}, \mathit{false}\}$. $A$ must make at least one literal per clause true; for each clause, select the vertex corresponding to one such literal to be in the set $U$. Since there are $m$ clauses, this yields exactly $m = k$ vertices, so the total profit is $k$. Moreover, we claim that $U$ cannot contain two neighboring vertices in $G$. Suppose for contradiction that $u, v \in U$ and $(u, v) \in E$. Then the edge $(u, v)$ must be of one of the two types above. But $u$ and $v$ cannot correspond to literals in the same clause because we selected only one vertex for each clause. And $u$ and $v$ cannot be labeled by $x_i$ and $\bar{x}_i$ for the same $i$, because $A$ cannot make both a variable and its negation true. Since neither possibility can hold, $U$ cannot contain two neighboring vertices. $U$ achieves a total profit of $k$ for the DONUT problem $\langle G, p, k \rangle$.

Conversely, suppose that there exists $U \subseteq V$, $|U| \geq k = m$ containing no two neighbors in $G$. Since $U$ does not contain neighbors, it cannot contain two vertices from the same clause. Therefore, we must have $|U| = m$, with exactly one vertex from each clause. Now define a truth assignment $A$ for the variables: $A(x_i) = \mathit{true}$ if some vertex with label $x_i$ is in $U$, and $A(x_i) = \mathit{false}$ if some vertex with label $\bar{x}_i$ is in $U$. For other variables the truth value can be arbitrary. Also since $U$ does not contain neighbors, $U$ cannot contain two vertices with contradictory labels, so assignment $A$ is well-defined. $A$ satisfies all clauses by making one literal corresponding to a vertex in $U$ true in each clause. Therefore, $A$ satisfies $\phi$.

For the last question, suppose that there is a polynomial-time algorithm to solve the original problem, i.e., to output a subset $U$ that maximizes the total profit. Then this algorithm can be easily adapted to a polynomial-time algorithm for DONUT: for any $\langle G, p, k \rangle$, simply run the assumed algorithm and obtain an optimal subset $U$. Then output $\mathit{true}$ if $k \leq |U|$, and $\mathit{false}$ otherwise. Since we have already shown that DONUT is NP-hard, this implies that P= NP.

**(c)** [10 points] Suppose we have one machine and a set of $n$ tasks $a_1, a_2, \ldots, a_n$. Each task $a_j$ requires $t_j$ units of time on the machine, yields a profit of $p_j$, and has a deadline $d_j$. Here, the $t_j$, $p_j$, and $d_j$ values are nonnegative integers. The machine can process only one task at a time. Not all tasks have to be run, but if a task starts running, it must run without interruption and must complete by its deadline.

A *schedule* for a subset of the tasks describes when each of the tasks in the subset starts running. A schedule must observe the constraints given above. The *profit* for the schedule is the sum of all the $p_j$ values for the tasks $a_j$ in the schedule.

The problem is to produce a schedule for a subset of the tasks that returns the greatest possible amount of profit. State this problem as a decision problem and show that it is NP-complete. In showing this, you may reduce from any problem that has been shown, in class or in CLRS, to be NP-complete.

**Solution:**

Define SCHED to be the following decision problem: given $\langle T, P, D, k\rangle$ where $T$, $P$, and $D$ are sequences $\{t_j\}$, $\{p_j\}$ and $\{d_j\}$, each of length $n$, decide whether there exists a subset $I \subseteq \{1, \ldots, n\}$ and an ordering $i_1, i_2, \ldots, i_m$ of $I$, such that:

1. For every $j$, $\sum_{l=1}^{j} T(i_l) \leq D(i_j)$. That is, the sum of the running times of the first $j$ tasks being run is no greater than the deadline for the $j^{th}$ task. That means that, when run in the given order, all tasks meet their deadlines.
2. $\sum_{l=1}^{m} P(i_l) \geq k$. That is, the total profit is at least $k$.

To see that SCHED is in NP, given an instance $\langle T, P, D, k\rangle$, we can guess a subset of the tasks and a sequence of start times, and verify that it meets all the constraints for a schedule and that it finishes within the given time $k$.

To show that SCHED is NP-hard, we can reduce from SUBSET-SUM, defined on p. 1097 of CLRS. Given an instance $\langle S, t\rangle$ of SUBSET-SUM, where $|S| = n$, order the elements of $S$ arbitrarily, as $s_1, \ldots, s_n$. We construct an instance $\langle T, P, D, k\rangle$ of SCHED as follows: Let $T = P = \{s_1, \ldots, s_n\}$ in that order, let $D$ be a length-$n$ sequence consisting of $t$ in every position, and let $k = t$. We claim that the answer to the SUBSET-SUM problem $\langle S, t\rangle$ is "yes", iff the answer to the SCHED problem $\langle T, P, D, k\rangle$ is "yes".

First, the answer to the SUBSET-SUM problem $\langle S, t\rangle$ is "yes". Then there is a subset $S'$ of the elements in $S$ whose sum is exactly $t$. Let $I$ be the set of indices of the $S'$ elements within the sequence $s_1, \ldots, s_n$, and let $i_1, i_2, \ldots, i_m$ order $I$ in increasing order. Then the sum of the running times of all tasks is exactly $t$, that is, $\sum_{l=1}^{m} T(i_l) = t$. This implies that all tasks meet their deadlines. Moreover, the total profit is exactly $\sum_{l=1}^{m} P(i_l) = t = k$. Therefore, the answer to the SCHED problem $\langle T, P, D, k\rangle$ is "yes".

Conversely, suppose the answer to the SCHED problem $\langle T, P, D, k\rangle$ is "yes". Then there is a subset $I$ of $1, \ldots, n$ and an ordering $i_1, i_2, \ldots, i_m$ of $I$, such that:

1. For every $j$, $\sum_{l=1}^{j} T(i_l) \leq D(i_j)$.

2. $\sum_{l=1}^{m} P(i_l) \geq k$.

Since all of the deadlines are equal to $k = t$, this is the same as saying:

1. $\sum_{l=1}^{m} T(i_l) \leq t$.
2. $\sum_{l=1}^{m} P(i_l) \geq t$.

Since each $T(i_j) = P(i_j)$, this says that $\sum_{l=1}^{m} T(i_l) = \sum_{l=1}^{m} P(i_l) = t$. Now let $S'$ be the subset of $S$ corresponding to the indices in $I$, that is, $S' = \{s_i \| i \in I\}$. Then the sum of the elements of $S'$ is exactly $t$. Therefore, the answer to the SUBSET-SUM problem $\langle S, t \rangle$ is "yes".

MIT OpenCourseWare
http://ocw.mit.edu

6.046J / 18.410J Design and Analysis of Algorithms
Spring 2015

For information about citing these materials or our Terms of Use, visit: http://ocw.mit.edu/terms.