Algorithms: COMP3121/9101

School of Computer Science and Engineering
University of New South Wales

10. LINEAR PROGRAMMING

Problem:

- You are given a list of food sources $f_1, f_2, \ldots, f_n$;
- for each source f_i you are given:
  - its price per gram p_i;
  - the number of calories c_i per gram, and
  - for each of 13 vitamins $V_1, V_2, \ldots, V_{13}$ you are given the content $v(i, j)$ of milligrams of vitamin V_j in one gram of food source f_i.

- Your task: to find a combination of quantities of food sources such that:
  - the total number of calories in all of the chosen food is equal to a recommended daily value of 2000 calories;
  - the total intake of each vitamin V_j is at least the recommended daily intake of w_j milligrams for all $1 \leq j \leq 13$;
  - the price of all food per day is as low as possible.

Linear Programming problems - Example 1 cont.

- To obtain the corresponding constraints let us assume that we take x_i grams of each food source f_i for $1 \leq i \leq n$. Then:
  - the total number of calories must satisfy

$$\sum_{i=1}^{n} x_i c_i = 2000;$$

  - for each vitamin V_j the total amount in all food must satisfy

$$\sum_{i=1}^{n} x_i v(i,j) \geq w_j \quad (1 \leq j \leq 13);$$

  - an implicit assumption is that all the quantities must be non-negative numbers,

$$x_i \geq 0, \quad 1 \leq i \leq n.$$

- Our goal is to minimise the objective function which is the total cost

$$y = \sum_{i=1}^{n} x_i p_i.$$

- Note that all constraints and the objective function, are **linear**.

Problem:

- Assume now that you are politician and you want to make certain promises to the electorate which will ensure that your party will win in the forthcoming elections.
- You can promise that you will build
  - a certain number of bridges, each 3 billion a piece;
  - a certain number of rural airports, each 2 billion a piece, and
  - a certain number of olympic swimming pools each a billion a piece.
- You were told by your wise advisers that
  - each bridge you promise brings you 5% of city votes, 7% of suburban votes and 9% of rural votes;
  - each rural airport you promise brings you no city votes, 2% of suburban votes and 15% of rural votes;
  - each olympic swimming pool promised brings you 12% of city votes, 3% of suburban votes and no rural votes.
- In order to win, you have to get at least 51% of each of the city, suburban and rural votes.
- You wish to win the election by cleverly making a promise that **appears** that it will blow as small hole in the budget as possible, i.e., that the total cost of your promises is as low as possible.

Linear Programming problems - Example 2

- We can let the number of bridges to be built be x_b, number of airports x_a and the number of swimming pools x_p.
- We now see that the problem amounts to minimising the objective $y = 3x_b + 2x_a + x_p$, while making sure that the following constraints are satisfied:

$$
\begin{aligned}
0.05x_b \qquad\qquad\quad + 0.12x_p &\geq 0.51 \quad \text{(securing majority of city votes)}\\
0.07x_b + 0.02x_a + 0.03x_p &\geq 0.51 \quad \text{(securing majority of suburban votes)}\\
0.09x_b + 0.15x_a \qquad\qquad\quad &\geq 0.51 \quad \text{(securing majority of rural votes)}\\
x_b, x_a, x_p &\geq 0.
\end{aligned}
$$

- However, there is a very significant difference with the first example:
  - you can eat 1.56 grams of chocolate, but
  - you cannot promise to build 1.56 bridges, 2.83 airports and 0.57 swimming pools!

- The second example is an example of an **Integer Linear Programming problem**, which requires all the solutions to be integers.
- Such problems are MUCH harder to solve than the “plain” Linear Programming problems whose solutions can be real numbers.

Linear Programming problems

- In the **standard form** the *objective* to be maximised is given by

$$\sum_{j=1}^{n} c_j x_j$$

- the *constraints* are of the form

$$\sum_{j=1}^{n} a_{ij} x_j \leq b_i, \qquad 1 \leq i \leq m; \tag{1}$$

$$x_j \geq 0, \qquad 1 \leq j \leq n, \tag{2}$$

- Let the boldface $\mathbf{x}$ represent a (column) vector, $\mathbf{x} = \langle x_1 \ldots x_n \rangle^\intercal$.
- To get a more compact representation of linear programs we introduce a partial ordering on vectors $\mathbf{x} \in \mathbf{R}^n$ by $\mathbf{x} \leq \mathbf{y}$ if and only if the corresponding inequalities hold coordinate-wise, i.e., if and only if $x_j \leq y_j$ for all $1 \leq j \leq n$.

- Letting $\mathbf{c} = \langle c_1 \ldots c_n \rangle^{\intercal} \in \mathbb{R}^n$ and $\mathbf{b} = \langle b_1 \ldots b_m \rangle^{\intercal} \in \mathbb{R}^m$, and letting A be the matrix $A = (a_{ij})$ of size $m \times n$, we get that the above problem can be formulated simply as:
  - maximize $\mathbf{c}^{\intercal}\mathbf{x}$
  - subject to the following two (matrix-vector) constraints:

$$A\mathbf{x} \leq \mathbf{b}$$

and

$$\mathbf{x} \geq \mathbf{0}.$$

- Thus, to specify a Linear Programming optimisation problem we just have to provide a triplet $(A, \mathbf{b}, \mathbf{c})$;
- This is the usual form which is accepted by most standard LP solvers.

- The value of the objective for any value of the variables which makes the constraints satisfied is called a *feasible solution* of the LP problem.
- Equality constraints of the form $\sum_{i=1}^{n} a_{ij}x_i = b_j$ can be replaced by two inequalities: $\sum_{i=1}^{n} a_{ij}x_i \geq b_j$ and $\sum_{i=1}^{n} a_{ij}x_i \leq b_j$; thus, we can assume that all constraints are inequalities.
- In general, a “natural formulation” of a problem as a Linear Program does not necessarily produce the non-negativity constrains for all of the variables.
- However, in the standard form such constraints are required for all of the variables.
- This poses no problem, because each occurrence of an unconstrained variable x_j can be replaced by the expression $x_j' - x_j^*$ where x_j', x_j^* are new variables satisfying the constraints $x_j' \geq 0$, $x_j^* \geq 0$.
- If $\mathbf{x} = (x_1, \ldots, x_n)$ is a vector, we let $|\mathbf{x}| = (|x_1|, \ldots, |x_n|)$. Some problems are naturally translated into constraints of the form $|A\mathbf{x}| \leq \mathbf{b}$. This also poses no problem because we can replace such constraints with two linear constraints: $A\mathbf{x} \leq \mathbf{b}$ and $-A\mathbf{x} \leq \mathbf{b}$ because $|x| \leq y$ if and only if $x \leq y$ and $-x \leq y$.

- Standard Form: maximize $\mathbf{c}^\intercal \mathbf{x}$ subject to $A\mathbf{x} \le \mathbf{b}$ and $\mathbf{x} \ge \mathbf{0}$.
- Any vector $\mathbf{x}$ which satisfies the two constraints is called a *feasible solution*, regardless of what the corresponding objective value $\mathbf{c}^\intercal \mathbf{x}$ might be.

- As an example, let us consider the following optimisation problem.

maximize $$z(x_1, x_2, x_3) = 3x_1 + x_2 + 2x_3 \tag{3}$$

subject to the constraints

$$x_1 + x_2 + 3x_3 \le 30 \tag{4}$$
$$2x_1 + 2x_2 + 5x_3 \le 24 \tag{5}$$
$$4x_1 + x_2 + 2x_3 \le 36 \tag{6}$$
$$x_1, x_2, x_3 \ge 0 \tag{7}$$

- How large can the value of the objective $z(x_1, x_2, x_3) = 3x_1 + x_2 + 2x_3$ be, without violating the constraints?
- If we add inequalities (4) and (5), we get

$$3x_1 + 3x_2 + 8x_3 \le 54 \tag{8}$$

- Since all variables are constrained to be non-negative, we are assured that

$$3x_1 + x_2 + 2x_3 \le 3x_1 + 3x_2 + 8x_3 \le 54$$

$$\begin{aligned}
\text{maximize:} \quad & z(x_1, x_2, x_3) = 3x_1 + x_2 + 2x_3 && (3)\\
\text{with constraints:} \quad & x_1 + x_2 + 3x_3 \leq 30 && (4)\\
& 2x_1 + 2x_2 + 5x_3 \leq 24 && (5)\\
& 4x_1 + x_2 + 2x_3 \leq 36 && (6)\\
& x_1, x_2, x_3 \geq 0 && (7)
\end{aligned}$$

- Thus the objective $z(x_1, x_2, x_3)$ is bounded above by 54, i.e., $z(x_1, x_2, x_3) \leq 54$.

- Can we obtain a tighter bound? We could try to look for coefficients $y_1, y_2, y_3 \geq 0$ to be used to for a linear combination of the constraints:

$$\begin{aligned}
y_1(x_1 + x_2 + 3x_3) &\leq 30y_1\\
y_2(2x_1 + 2x_2 + 5x_3) &\leq 24y_2\\
y_3(4x_1 + x_2 + 2x_3) &\leq 36y_3
\end{aligned}$$

- Then, summing up all these inequalities and factoring, we get

$$x_1(y_1 + 2y_2 + 4y_3) + x_2(y_1 + 2y_2 + y_3) + x_3(3y_1 + 5y_2 + 2y_3) \leq 30y_1 + 24y_2 + 36y_3$$

$$\text{maximize:} \quad z(x_1, x_2, x_3) = 3x_1 + x_2 + 2x_3 \tag{3}$$
$$\text{with constraints:} \quad x_1 + x_2 + 3x_3 \le 30 \tag{4}$$
$$2x_1 + 2x_2 + 5x_3 \le 24 \tag{5}$$
$$4x_1 + x_2 + 2x_3 \le 36 \tag{6}$$
$$x_1, x_2, x_3 \ge 0 \tag{7}$$

- So we got

$$x_1(y_1 + 2y_2 + 4y_3) + x_2(y_1 + 2y_2 + y_3) + x_3(3y_1 + 5y_2 + 2y_3) \le 30y_1 + 24y_2 + 36y_3 \tag{9}$$

- If we compare this with our objective (3) we see that if we choose y_1, y_2 and y_3 so that:

$$y_1 + 2y_2 + 4y_3 \ge 3$$
$$y_1 + 2y_2 + y_3 \ge 1$$
$$3y_1 + 5y_2 + 2y_3 \ge 2$$

then

$$3x_3 + x_2 + 2x_3 \le x_1(y_1 + 2y_2 + 4y_3) + x_2(y_1 + 2y_2 + y_3) + x_3(3y_1 + 5y_2 + 2y_3)$$

Combining this with (9) we get:

$$30y_1 + 24y_2 + 36y_3 \ge 3x_1 + x_2 + 2x_3 = z(x_1, x_2, x_3)$$

- Consequently, in order to find as tight upper bound for our objective $z(x_1, x_2, x_3)$ of the problem P:

$$
\begin{aligned}
\text{maximize:} \quad & z(x_1, x_2, x_3) = 3x_1 + x_2 + 2x_3 && (3)\\
\text{with constraints:} \quad & x_1 + x_2 + 3x_3 \leq 30 && (4)\\
& 2x_1 + 2x_2 + 5x_3 \leq 24 && (5)\\
& 4x_1 + x_2 + 2x_3 \leq 36 && (6)\\
& x_1, x_2, x_3 \geq 0 && (7)
\end{aligned}
$$

we have to look for y_1, y_2, y_3 which solve problem P^*:

$$
\begin{aligned}
\text{minimise:} \quad & z^*(y_1, y_2, y_3) = 30y_1 + 24y_2 + 36y_3 && (10)\\
\text{with constraints:} \quad & y_1 + 2y_2 + 4y_3 \geq 3 && (11)\\
& y_1 + 2y_2 + y_3 \geq 1 && (12)\\
& 3y_1 + 5y_2 + 2y_3 \geq 2 && (13)\\
& y_1, y_2, y_3 \geq 0 && (14)
\end{aligned}
$$

then $z^*(y_1, y_2, y_3) = 30y_1 + 24y_2 + 36y_3 \geq 3x_1 + x_2 + 2x_3 = z(x_1, x_2, x_3)$ will be a tight upper bound for $z(x_1, x_2, x_3)$

- The new problem P^* is called the *dual problem* for the problem P.

- Let us now repeat the whole procedure with P^* in place of P, i.e., let us find the dual program $(P^*)^*$ of P^*.
- We are now looking for $z_1, z_2, z_3 \geq 0$ to multiply inequalities (11)-(13) and obtain

$$z_1(y_1 + 2y_2 + 4y_3) \geq 3z_1$$
$$z_2(y_1 + 2y_2 + y_3) \geq z_2$$
$$z_3(3y_1 + 5y_2 + 2y_3) \geq 2z_3$$

- Summing these up and factoring produces

$$y_1(z_1 + z_2 + 3z_3) + y_2(2z_1 + 2z_2 + 5z_3) + y_3(4z_1 + z_2 + 2z_3) \geq 3z_1 + z_2 + 2z_3 \tag{15}$$

- If we choose multipliers z_1, z_2, z_3 so that

$$z_1 + z_2 + 3z_3 \leq 30 \tag{16}$$
$$2z_1 + 2z_2 + 5z_3 \leq 24 \tag{17}$$
$$4z_1 + z_2 + 2z_3 \leq 36 \tag{18}$$

we will have:

$$y_1(z_1 + z_2 + 3z_3) + y_2(2z_1 + 2z_2 + 5z_3) + y_3(4z_1 + z_1 + 2z_3) \leq 30y_1 + 24y_2 + 36y_3$$

- Combining this with (15) we get

$$3z_1 + z_2 + 2z_3 \leq 30y_1 + 24y_2 + 36y_3$$

- Consequently, finding the dual program $(P^*)^*$ of P^* amounts to maximising the objective $3z_1 + z_2 + 2z_3$ subject to the constraints

$$\begin{aligned} z_1 + z_2 + 3z_3 &\leq 30 \\ 2z_1 + 2z_2 + 5z_3 &\leq 24 \\ 4z_1 + z_2 + 2z_3 &\leq 36 \end{aligned}$$

- But note that, except for having different variables, $(P^*)^*$ is exactly our starting program P. Thus, the dual program $(P^*)^*$ for program P^* is just P itself, i.e., $(P^*)^* = P$.

- So, at the first sight, looking for the multipliers y_1, y_2, y_3 did not help much, because it only reduced a maximisation problem to an equally hard minimisation problem.
- It is now useful to remember how we proved that the Ford - Fulkerson Max Flow algorithm in fact produces a **maximal flow**, by showing that it terminates only when we reach the capacity of a **minimal cut**.

- The original, *primal* Linear Program P and its *dual* Linear Program can be easily described in the most general case:

$$P: \quad \text{maximize} \quad z(\mathbf{x}) = \sum_{j=1}^{n} c_j x_j,$$

$$\text{subject to the constraints} \quad \sum_{j=1}^{n} a_{ij} x_j \leq b_i; \quad 1 \leq i \leq m$$

$$x_1, \ldots, x_n \geq 0;$$

$$P^*: \quad \text{minimize} \quad z^*(\mathbf{y}) = \sum_{i=1}^{m} b_i y_i,$$

$$\text{subject to the constraints} \quad \sum_{i=1}^{m} a_{ij} y_i \geq c_j; \quad 1 \leq j \leq n$$

$$y_1, \ldots, y_m \geq 0,$$

or, in matrix form,

$$P: \quad \text{maximize} \quad z(\mathbf{x}) = \mathbf{c}^{\intercal}\mathbf{x}, \quad \text{subject to the constraints} \quad A\mathbf{x} \leq \mathbf{b} \quad \text{and} \quad \mathbf{x} \geq 0;$$

$$P^*: \quad \text{minimize} \quad z^*(\mathbf{y}) = \mathbf{b}^{\intercal}\mathbf{y}, \quad \text{subject to the constraints} \quad A^{\intercal}\mathbf{y} \geq \mathbf{c} \quad \text{and} \quad \mathbf{y} \geq 0.$$

Weak Duality Theorem

- Recall that any vector $\mathbf{x}$ which satisfies the two constraints, $A\mathbf{x} \leq \mathbf{b}$ and $\mathbf{x} \geq 0$ is called a *feasible solution*, regardless of what the corresponding objective value $\mathbf{c}^\intercal \mathbf{x}$ might be.
- **Theorem** If $x = \langle x_1 \dots x_n \rangle$ is any basic feasible solution for P and $y = \langle y_1 \dots y_m \rangle$ is any basic feasible solution for P^*, then:

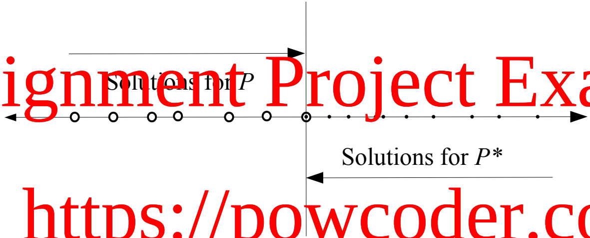
$$z(x) = \sum_{j=1}^{n} c_j x_j \leq \sum_{i=1}^{n} b_i y_i = z^*(y)$$

Proof: Since x and y are basic feasible solutions for P and P^* respectively, we can use the constraint inequalities, first from P^* and then from P to obtain

$$z(x) = \sum_{j=1}^{n} c_j x_j \leq \sum_{j=1}^{n} \left(\sum_{i=1}^{m} a_{ij} y_i \right) x_j = \sum_{i=1}^{m} \left(\sum_{j=1}^{n} a_{ij} x_j \right) y_i \leq \sum_{i=1}^{n} b_i y_i = z^*(y)$$

- Thus, the value of (the objective of P^* for) any feasible solution of P^* is an upper bound for the set of all values of (the objective of P for) all feasible solutions of P, and
- every feasible solution of P is a lower bound for the set of feasible solutions for P^*.

Weak Duality Theorem



- Thus, if we find a feasible solution for P which is equal to a feasible solution to P^*, such solution must be the maximal feasible value of the objective of P and the minimal feasible value of the objective of P^*.
- If we use a search procedure to find an optimal solution for P we know when to stop: when such a value is also a feasible solution for P^*.
- This is why the most commonly used LP solving method, the SIMPLEX method, produces optimal solution for P, because it stops at a value of the primal objective which is also a value of the dual objective.
- See the Lecture Notes for the details and an example of how the SIMPLEX algorithm runs.

Five sisters are alone in their house. Sharon is reading a book, Jennifer is playing chess, Cathrine is cooking and Ana is doing laundry. What is Helen, the fifth sister, doing?