Linear Programming

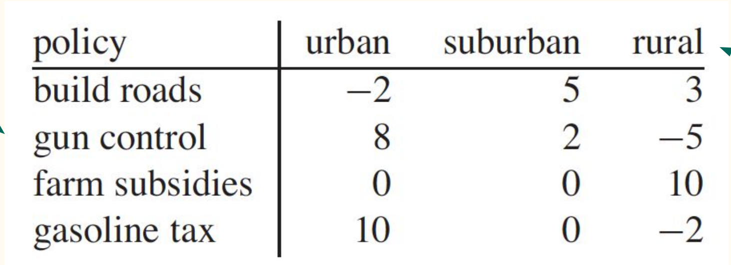
Course: CS 5130 - Advanced Data Structures and Algorithms

Instructor: Dr. Badri Adhikari

A political problem

Suppose that you are a politician trying to win an election. You would like at least half the registered voters in each of the three regions to vote for you.

Your issues



policy	urban	suburban	rural
build roads	−2	5	3
gun control	8	2	−5
farm subsidies	0	0	10
gasoline tax	10	0	−2

A table given to you by your campaign research staff. The number of votes you win/lose (in thousands) by spending $1000 on advertising on each issue.

Your district has three different types of areas. Voters in urban = 100K, suburban = 200K, and rural = 50K.

Goal: Figure out the minimum amount of money that you need to spend in order to win 50K urban votes, 100K suburban votes, and 25K rural votes.

Obviously, same issues are not equally important in all areas!

Trial and error approach:

Build roads = $20K, gun control = $0, farm subsidies = $4K, gasoline tax = $9K (Total advertising = $33K)

Outcome:

Urban = -2 (20) + 8 (0) + 0 (4) + 10 (9) = 50K votes ✓

Suburban = 5 (20) + 2 (0) + 0 (4) + 0 (9) = 100K votes ✓

Rural = 3 (20) + (-5) (0) + 10 (4) + (-2) (9) = 82K votes ✓(?)

Systematic approach:

x_1 = number of thousands of dollars spent on advertisement on building roads

x_2 = number of thousands of dollars spent on advertisement on gun control

x_3 = number of thousands of dollars spent on advertisement on farm subsidies

x_4 = number of thousands of dollars spent on advertisement on gasoline tax

Minimize $x_1 + x_2 + x_3 + x_4$ subject to

$$
\begin{aligned}
-2x_1 + 8x_2 + 0x_3 + 10x_4 &\geq 50 \quad (1)\\
-5x_1 + 2x_2 + 0x_3 + 0x_4 &\geq 50 \quad (2)\\
3x_1 - 5x_2 + 10x_3 - 2x_4 &\geq 50 \quad (3)\\
x_1, x_2, x_3, x_4 &\geq 0 \quad (4)
\end{aligned}
$$

The solution to this linear program yields optimal strategy.

General linear programs

In the general linear-programming problem, we wish to optimize (minimize or maximize) a linear function subject to a set of linear inequalities.

A *linear function f* on real numbers a_1, a_2, etc. can be defined as

$$f(x_1, x_2, \ldots, x_n) = a_1x_1 + a_2x_2 + \cdots + a_nx_n = \sum_{j=1}^{n} a_jx_j$$

Two kinds of *linear constraints*: *linear equalities* and *linear inequalities*.

$$f(x_1, x_2, \ldots, x_n) = b$$

Linear equality

$$f(x_1, x_2, \ldots, x_n) \leq b$$

and

$$f(x_1, x_2, \ldots, x_n) \geq b$$

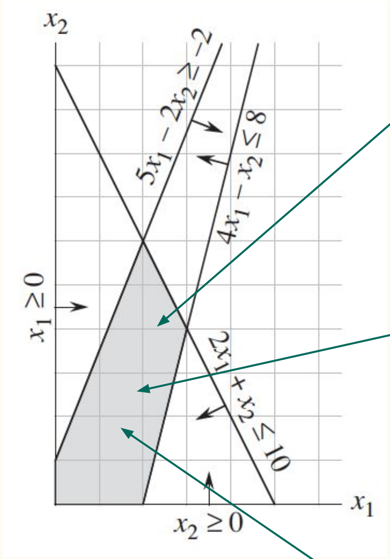
Linear inequalities

The linear program can be a *minimization linear program* or *maximization linear program*.

A linear program with two variables

$$
\begin{array}{lrcrcr}
\text{maximize} & x_1 & + & x_2 & & \\
\text{subject to} & & & & & \\
& 4x_1 & - & x_2 & \leq & 8 \\
& 2x_1 & + & x_2 & \leq & 10 \\
& 5x_1 & - & 2x_2 & \geq & -2 \\
& & x_1, x_2 & & \geq & 0
\end{array}
$$

The function we wish to maximize is the *objective function*.

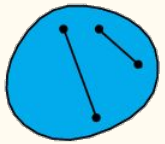


Value of the objective function at a particular point in the feasible region is *objective value*.

Feasible solutions in 2D form a *convex region*!

We call any setting of the variables x_1 and x_2 that satisfies all the constraints a *feasible solution* to the linear program.

Convex region: The segment between any two distinct points of the region is completely included in the shape



Algorithms for linear programming

1. **Simplex algorithm** - requires exponential time
2. Ellipsoid algorithm - polynomial time* algorithm but runs slow
3. Interior-point methods

Professional tools (available in the market) do not use simplex method (most of them).

When the requirement is that all variables take on integer values, the problem is *integer linear program*. It is NP-hard problem.

* its running time on inputs of size n is at most Cn^k.

Formulating 'shortest-paths' problem as a linear program

Single-pair shortest-path problem (it can be extended to the more general single-source shortest-paths problem).

Given: a weighted, directed graph $G = (V, E)$; source s and destination t.

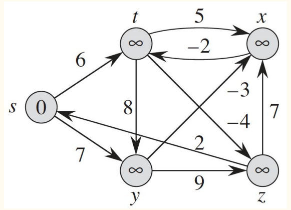
Goal: compute d_t, the weight of a shortest path from s to t.

To express this problem as a linear program, we need to determine a <u>set of variables</u> and <u>constraints</u>. $|V|$ variables - d_v, one for each vertex. $|E| + 1$ constraints - for each edge and $d_s = 0$.

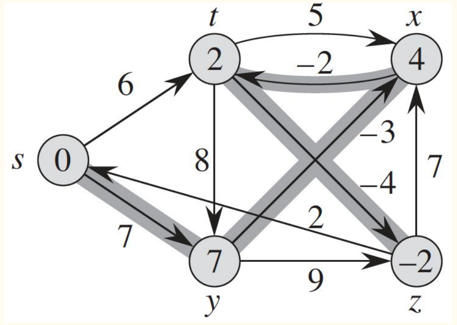
When the Bellman-Ford algorithm terminates, it computes, for each vertex v, a value d_v such that $d_v \le d_u + w(u, v)$. Initially $d_s = 0$, and it never changes.

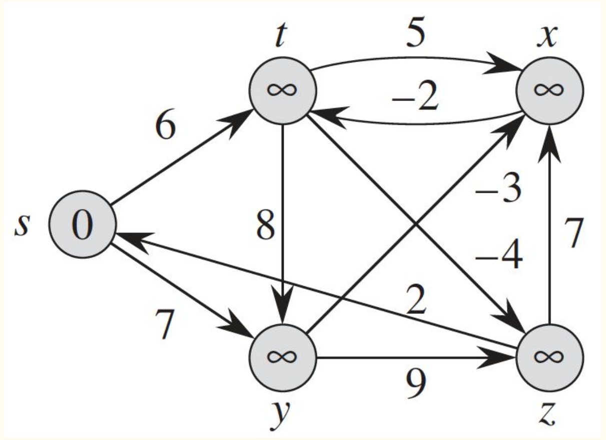
Following will be the linear program to compute the shortest-path weight from s to t:

```
Minimize?       d_t
subject to      d_v ≤ d_u + w(u, v)
                d_s = 0
```



Example





Conversion into a linear program $\longrightarrow$

```
maximize dt subject to
ds = 0,
dt - ds <= 6, dt - dx <= -2,
dy - ds <= 7, dy - dt <= 8,
dx - dt <= 5, dx - dy <= -3, dx - dz <=
7,
dz - dt <= -4, dz - dy <= 9,
ds - dz <= 2
```

https://en.wikipedia.org/wiki/CPLEX

https://www.zweigmedia.com/simplex/simplex.php?lang=en

IBM CPLEX

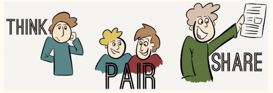
Formulating ‘maximum-flow’ as a linear program

Each edge has a nonnegative capacity $c(u,v) \geq 0$. Flow is a nonnegative function that satisfies (a) capacity constraint, and (b) flow conservation. Maximum flow = flow out of the source - flow into the source. Also, $c(u,v) = 0$ if $(u,v) \notin E$.

We can express the maximum-flow problem as a linear program:

$$
\begin{array}{llcll}
\text{maximize} & \displaystyle\sum_{v\in V} f_{sv} & - & \displaystyle\sum_{v\in V} f_{vs} & \\
\text{subject to} & & & & \\
 & f_{uv} & \leq & c(u,v) & \text{for each } u,v \in V\ , \\
 & \displaystyle\sum_{v\in V} f_{vu} & = & \displaystyle\sum_{v\in V} f_{uv} & \text{for each } u \in V - \{s,t\}\ , \\
 & f_{uv} & \geq & 0 & \text{for each } u,v \in V\ .
\end{array}
$$

$|V|^2$ variables corresponding to the flow between each pair of vertices, and $2\,|V|^2 + |V| - 2$ constraints.



Why formulate problems as linear programs?

Efficient algorithms that are designed specifically like

- Dijkstra's algorithm for the single-source shortest paths problem, or
- Push-relabel method (better than Ford-Fulkerson method) for maximum flow,

will be more efficient (in practice and in theory).

Is it useful at all to use LP for these problems? [Think-pair-share]

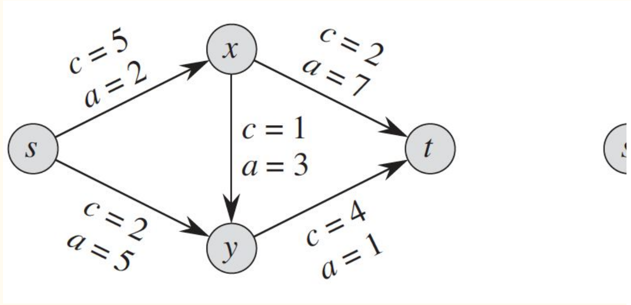
The real power of linear programming comes from the ability to solve new problems. For example, the problem faced by a politician (earlier slides).

No algorithms exist to solve such problems (unless someone designs one specifically for the problem).

LP is also particularly useful for solving variants of problems (like maximum flow).

Minimum-cost flow problem

A generalization of maximum-flow problem: G = (V, E). (u,v) ∉ E if c(u,v) = 0. Nonnegative capacities. No antiparallel edges. In addition to capacity c(u, v) for each edge (u,v) we are given a real-valued cost a(u,v). If we send f_{uv} units of flow over edge (u,v), we incur a cost of a(u,v) f_{uv}. We are also given a flow demand d. We wish to send d units of flow from s to t while minimizing the total cost $\sum_{(u,v)\in E} a(u,v) f_{uv}$ incurred by the flow.



We wish to send 4 units of flow from s to t. What is the best flow (min cost)? What is the total cost?

minimize $\sum_{(u,v)\in E} a(u,v) f_{uv}$

subject to

$$
\begin{array}{rcll}
f_{uv} & \le & c(u,v) & \text{for each } u, v \in V \, , \\
\sum_{v\in V} f_{vu} - \sum_{v\in V} f_{uv} & = & 0 & \text{for each } u \in V - \{s,t\} \, , \\
\sum_{v\in V} f_{sv} - \sum_{v\in V} f_{vs} & = & d \, , & \\
f_{uv} & \ge & 0 & \text{for each } u, v \in V \, .
\end{array}
$$

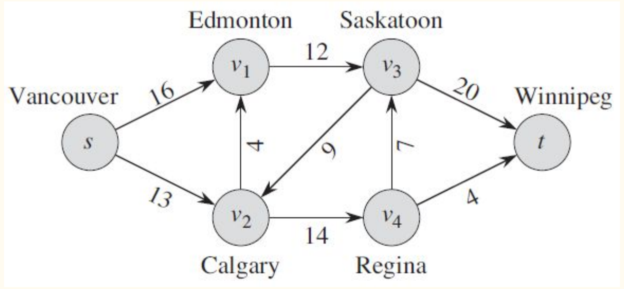
Q. What will be the linear program for the maximum flow problem aside?

Multicommodity flow

Lucky Puck company decides to diversify its product line - ship pucks, sticks and hockey helmets as well.

Each piece of equipment is manufactured in its own factory, has its own warehouse, and must be shipped, each day, from factory to warehouse.

Example: Sticks are manufactured in Vancouver and must be shipped to Saskatoon, Helmets are manufactured in Edmonton and must be shipped to Regina.



The capacity of the shipping network does not change. Different items (commodities) share the same network.

Multicommodity flow

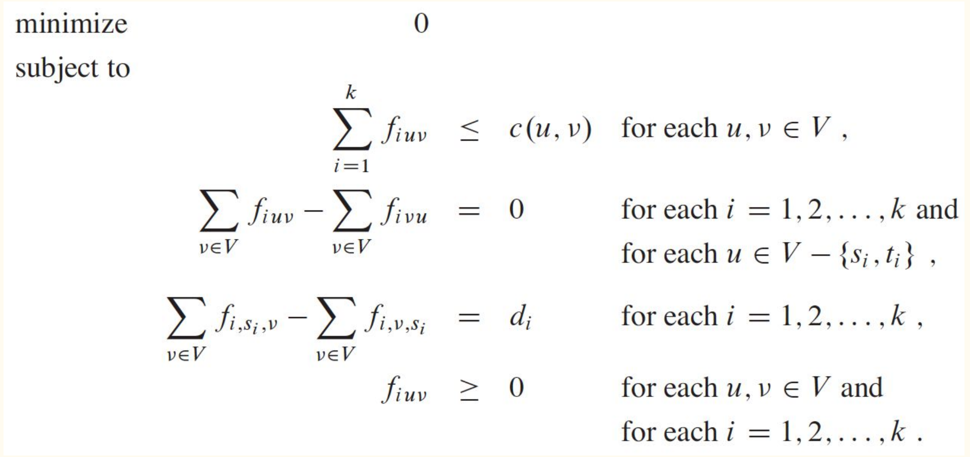
The problem: G = (V, E). (u,v) ∉ E if c(u,v) = 0. Nonnegative capacities. No antiparallel edges. We have k different commodities, K_1, K_2, ..., K_k.

Each commodity can be defined by a triplet $K_i = (s_i, t_i, d_i)$. s_i is the source. t_i is the sink. d_i is the demand.

Flow for a commodity i is f_i, so that f_{iuv} is a flow of commodity i from u to v.

Each f_{iuv} satisfy flow conservation. f_{uv} is an *aggregate flow*, $f_{uv} = \Sigma^k_{i=1} f_{iuv}$.

Goal: Not trying to minimize any objective function. Need to determine whether such a flow exists. Our objective function will be "null."



$$\begin{array}{llll}
\text{minimize} & 0 & & \\
\text{subject to} & & & \\
& \displaystyle\sum_{i=1}^{k} f_{iuv} & \leq \; c(u,v) & \text{for each } u,v \in V \,, \\
& \displaystyle\sum_{v \in V} f_{iuv} - \sum_{v \in V} f_{ivu} & = \; 0 & \text{for each } i = 1,2,\ldots,k \text{ and} \\
& & & \text{for each } u \in V - \{s_i, t_i\} \,, \\
& \displaystyle\sum_{v \in V} f_{i,s_i,v} - \sum_{v \in V} f_{i,v,s_i} & = \; d_i & \text{for each } i = 1,2,\ldots,k \,, \\
& f_{iuv} & \geq \; 0 & \text{for each } u,v \in V \text{ and} \\
& & & \text{for each } i = 1,2,\ldots,k \,.
\end{array}$$

SPECIAL BONUS

Bonus points (2.5 points) Deadline Nov 27th

In a **minimum-cost multicommodity flow** problem, we are given directed graph $G = (V, E)$ in which each edge $(u, v) \in E$ has a nonnegative capacity $c(u, v) \geq= 0$ and a cost $a(u, v)$. As in the multicommodity-flow problem, we are given k different commodities, $K_1, K_2, ..., K_k$, where commodity i is specified by the triple $K_i = (s_i, t_i, d_i)$. We define the flow f_i for commodity i and the aggregate flow $f(u, v)$ on edge (u, v) as in the multicommodity-flow problem. A feasible flow is one in which the aggregate flow on each edge (u, v) is no more than the capacity of edge (u, v). The cost of a flow is $\Sigma_{u, v \in V}\, a(u, v)\, f(u, v)$, and the goal is to find the feasible flow of minimum cost.

Your Task: Create an specific example of your own. Express it as a linear program. Solve it using CPLEX or some online LP solver. Verify your answer (on your own).

Motivation (Example): The *minimum-cost multicommodity flow problem* simultaneously ships multiple commodities through a single network so the total flow obeys the arc capacity constraints and has minimum cost. For example, in telephone networks, calls between different locations must be routed through the same telephone cables which have finite capacities and differing costs.

Standard form of linear programs

In a standard form, we are given n real numbers $c_1, c_2, \ldots, c_n$; m real numbers $b_1, b_2, \ldots, b_m$; and $m\ n$ real numbers a_{ij} for $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, n$. We wish to find n real numbers $x_1, x_2, \ldots, x_n$ that

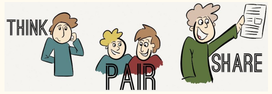
$$
\begin{array}{lrcll}
\text{maximize} & \displaystyle\sum_{j=1}^{n} c_j x_j & & & \\
\text{subject to} & & & & \\
 & \displaystyle\sum_{j=1}^{n} a_{ij} x_j & \leq & b_i & \text{for } i = 1, 2, \ldots, m \\
 & x_j & \geq & 0 & \text{for } j = 1, 2, \ldots, n .
\end{array}
$$

Nonnegativity constraints

$$\xrightarrow{\text{Compact form}}$$

$$
\begin{array}{lrcl}
\text{maximize} & c^{\mathrm{T}} x & & \\
\text{subject to} & & & \\
 & Ax & \leq & b \\
 & x & \geq & 0
\end{array}
$$

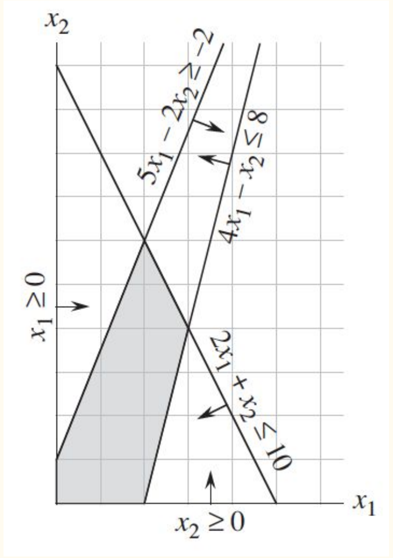
Which of these variables are matrix and which are scalars?



Feasible/infeasible solutions

We call a setting of the variables ‘x’ that satisfies all the constraints a *feasible solution*.

A setting of variables ‘x’ that fails to satisfy at least one constraint is an *infeasible solution*. (LP has no feasible solutions)

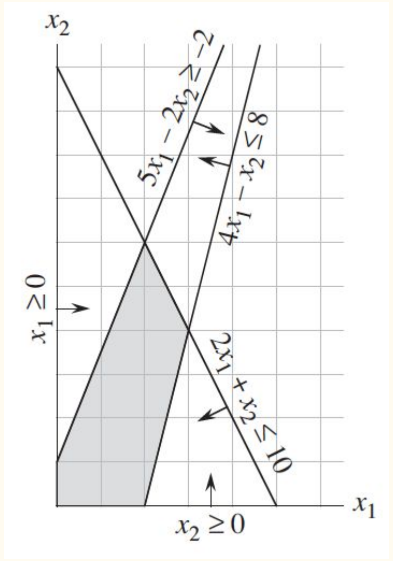


$$
\begin{array}{lrcrcr}
\text{maximize} & 3x_1 & - & 2x_2 & & \\
\text{subject to} & & & & & \\
& x_1 & + & x_2 & \leq & 2 \\
& -2x_1 & - & 2x_2 & \leq & -10 \\
& & x_1, x_2 & & \geq & 0
\end{array}
$$

Is this linear program feasible or not? Show it (provide example).

Bounded/Unbounded solutions

If a linear program has some feasible solutions but does not have a finite optimal objective value, it is *unbounded*.



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

Is this linear program bounded or not? Show it (provide example).

Classwork

Consider the following linear program:

```
Maximize −x
with respect to
x + 2y ≥ 3, x − 2y ≥ −1 , x , y ≥ 0.
```

Does it have a bounded or unbounded solution?

Does it have a finite optimal objective value?

Converting linear programs into standard form

$$\begin{array}{lrcll} \text{maximize} & \sum_{j=1}^{n} c_j x_j & & & \\ \text{subject to} & & & & \\ & \sum_{j=1}^{n} a_{ij} x_j & \leq & b_i & \text{for } i = 1, 2, \ldots, m \\ & x_j & \geq & 0 & \text{for } j = 1, 2, \ldots, n \end{array}$$

Standard form

It is always possible to convert a linear program, given as a minimizing or maximizing of a linear function subject to linear constraints, into standard form.

A linear program might not be in standard form for any of four possible reasons:

1. The objective function might be a minimization rather than a maximization.
2. There might be variables without nonnegativity constraints.
3. There might be *equality constraints*, which have an equal sign rather than a less-than-or-equal-to sign.
4. There might be *inequality constraints*, but instead of having a less-than-or-equal-to sign, they have a greater-than-or-equal-to sign.

Converting linear programs into standard form

When converting one linear program L into another linear program L′, we would like the property that <u>an optimal solution to L′ yields an optimal solution to L</u>.

We say that two maximization linear programs L and L′ are *equivalent* if

> (a) for each feasible solution $\bar{x}$ to L with objective value z, there is a corresponding feasible solution $\bar{x}'$ to L′ with objective value z, and
>
> (b) for each feasible solution $\bar{x}'$ to L′ with objective value z, there is a corresponding feasible solution $\bar{x}$ to L with objective value z. (Not necessarily one-to-one).

A minimization linear program L and a maximization linear program L′ are *equivalent* if for each feasible solution $\bar{x}$ to L with objective value z, there is a corresponding feasible solution $\bar{x}'$ in L′ with objective value $-z$, and vice versa.

$$\begin{array}{lrcll} \text{maximize} & \sum_{j=1}^{n} c_j x_j & & & \\ \text{subject to} & & & & \\ & \sum_{j=1}^{n} a_{ij} x_j & \leq & b_i & \text{for } i = 1, 2, \ldots, m \\ & x_j & \geq & 0 & \text{for } j = 1, 2, \ldots, n \end{array}$$

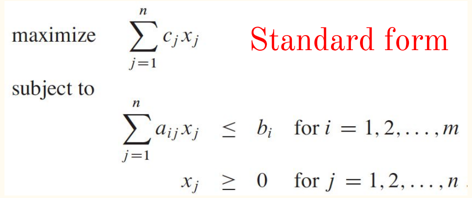
Standard form

Converting linear programs into standard form

(1) To convert minimization linear program L into an equivalent maximization linear program L′, we simply negate the coefficients in the objective function.

$$\begin{array}{lrcrcl} \text{minimize} & -2x_1 & + & 3x_2 & & \\ \text{subject to} & & & & & \\ & x_1 & + & x_2 & = & 7 \\ & x_1 & - & 2x_2 & \leq & 4 \\ & x_1 & & & \geq & 0 \end{array} \longrightarrow \begin{array}{lrcrcl} \text{maximize} & 2x_1 & - & 3x_2 & & \\ \text{subject to} & & & & & \\ & x_1 & + & x_2 & = & 7 \\ & x_1 & - & 2x_2 & \leq & 4 \\ & x_1 & & & \geq & 0 \end{array}$$

Converting linear programs into standard form



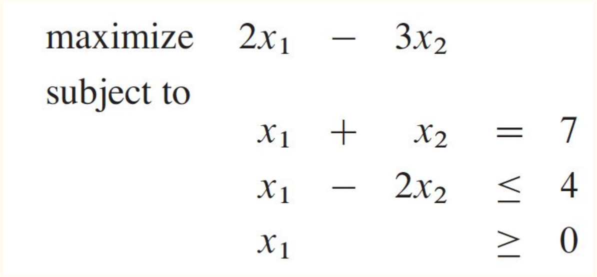
$$\begin{array}{lrcll} \text{maximize} & \sum_{j=1}^{n} c_j x_j & & & \\ \text{subject to} & & & & \\ & \sum_{j=1}^{n} a_{ij} x_j & \leq & b_i & \text{for } i = 1, 2, \ldots, m \\ & x_j & \geq & 0 & \text{for } j = 1, 2, \ldots, n \end{array}$$

Standard form

(2) How to convert a linear program in which some of the variables do not have nonnegativity constraints into one in which each variable has a non-negativity constraint?

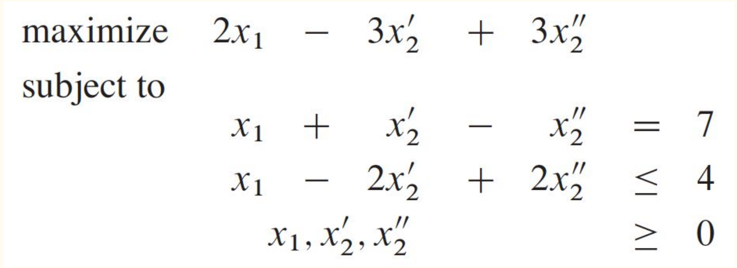
We replace the occurrence of x_j by $x'_j - x''_j$, and add two nonnegativity constraints $x'_j \geq 0$ and $x''_j \geq 0$. Consequently, we replace $c_j x_j$ with $c_j x'_j - c_j x''_j$.

Apply this conversion scheme to each variable that does not have nonnegativity constraints.



$$\begin{array}{lrcrcl} \text{maximize} & 2x_1 & - & 3x_2 & & \\ \text{subject to} & & & & & \\ & x_1 & + & x_2 & = & 7 \\ & x_1 & - & 2x_2 & \leq & 4 \\ & x_1 & & & \geq & 0 \end{array}$$

$\longrightarrow$



$$\begin{array}{lrcrcrcl} \text{maximize} & 2x_1 & - & 3x'_2 & + & 3x''_2 & & \\ \text{subject to} & & & & & & & \\ & x_1 & + & x'_2 & - & x''_2 & = & 7 \\ & x_1 & - & 2x'_2 & + & 2x''_2 & \leq & 4 \\ & & & x_1, x'_2, x''_2 & & & \geq & 0 \end{array}$$

$$\begin{array}{lrcll} \text{maximize} & \sum_{j=1}^{n} c_j x_j & & & \\ \text{subject to} & & & & \\ & \sum_{j=1}^{n} a_{ij} x_j & \le & b_i & \text{for } i = 1, 2, \ldots, m \\ & x_j & \ge & 0 & \text{for } j = 1, 2, \ldots, n \end{array}$$

Standard form

Converting linear programs into standard form

(3) How to convert equality constraints into inequality constraints?

Replace the equality constraint with a pair of inequality constraints.

$$\begin{array}{lrcrcrcl} \text{maximize} & 2x_1 & - & 3x_2' & + & 3x_2'' & & \\ \text{subject to} & & & & & & & \\ & x_1 & + & x_2' & - & x_2'' & = & 7 \\ & x_1 & - & 2x_2' & + & 2x_2'' & \le & 4 \\ & & & x_1, x_2', x_2'' & & & \ge & 0 \end{array} \longrightarrow \begin{array}{lrcrcrcl} \text{maximize} & 2x_1 & - & 3x_2' & + & 3x_2'' & & \\ \text{subject to} & & & & & & & \\ & x_1 & + & x_2' & - & x_2'' & \le & 7 \\ & x_1 & + & x_2' & - & x_2'' & \ge & 7 \\ & x_1 & - & 2x_2' & + & 2x_2'' & \le & 4 \\ & & & x_1, x_2', x_2'' & & & \ge & 0 \end{array}$$

Converting linear programs into standard form

$$\begin{array}{lrcll} \text{maximize} & \displaystyle\sum_{j=1}^{n} c_j x_j & & & \\ \text{subject to} & & & & \\ & \displaystyle\sum_{j=1}^{n} a_{ij} x_j & \leq & b_i & \text{for } i = 1, 2, \ldots, m \\ & x_j & \geq & 0 & \text{for } j = 1, 2, \ldots, n \end{array}$$

Standard form

(4) How to convert greater-than-or-equal-to sign into a less-than-or-equal-to sign in a constraint?

Multiply the constraint through by -1.

$$\begin{array}{lrcrcrcr} \text{maximize} & 2x_1 & - & 3x'_2 & + & 3x''_2 & & \\ \text{subject to} & & & & & & & \\ & x_1 & + & x'_2 & - & x''_2 & \leq & 7 \\ & x_1 & + & x'_2 & - & x''_2 & \geq & 7 \\ & x_1 & - & 2x'_2 & + & 2x''_2 & \leq & 4 \\ & & & x_1, x'_2, x''_2 & & & \geq & 0 \end{array} \longrightarrow \begin{array}{lrcrcrcr} \text{maximize} & 2x_1 & - & 3x_2 & + & 3x_3 & & \\ \text{subject to} & & & & & & & \\ & x_1 & + & x_2 & - & x_3 & \leq & 7 \\ & -x_1 & - & x_2 & + & x_3 & \leq & -7 \\ & x_1 & - & 2x_2 & + & 2x_3 & \leq & 4 \\ & & & x_1, x_2, x_3 & & & \geq & 0 \end{array}$$

Also rename x'_2 to x_2 and x''_2 to x_3

$$\begin{array}{lrcll} \text{maximize} & \displaystyle\sum_{j=1}^{n} c_j x_j & & & \\ \text{subject to} & & & & \\ & \displaystyle\sum_{j=1}^{n} a_{ij} x_j & \leq & b_i & \text{for } i = 1, 2, \ldots, m \\ & x_j & \geq & 0 & \text{for } j = 1, 2, \ldots, n \end{array}$$

Standard form

Classwork

Convert the following linear program into standard form:

$$\begin{array}{lrcrcrcr} \text{minimize} & 2x_1 & + & 7x_2 & + & x_3 & & \\ \text{subject to} & & & & & & & \\ & x_1 & & & - & x_3 & = & 7 \\ & 3x_1 & + & x_2 & & & \geq & 24 \\ & & & x_2 & & & \geq & 0 \\ & & & & & x_3 & \leq & 0 . \end{array}$$

Note: In exam, the standard form won't be provided.

Summary

Linear programming is powerful. It can solve problems like shortest-paths, maximum-flow, multi-commodity flow, and many others.

Industries widely use LP tools like CPLEX in their operations research.

Once you obtain a linear program for your problem, you can easily convert it into the standard form, in order to feed your linear program to tools like CPLEX.