DS 574 Algorithmic Mechanism Design **Lecture #8**
Fall 2023 Prof. Kira Goldner

Introduction to Linear Programming and Duality

Why Linear Programming rocks:

- Incredibly general: Every problem we've seen so far can be formulated as a linear program.
- Computational tractable
  - In theory: Can be solved in polynomial time
  - In practice: Fast with input sizes up into the millions!
- Contains many properties that can be turned into useful algorithmic paradigms and analysis:
  - Duality:
    - Solve an easier equivalent problem.
    - How do we know when we're done?
  - Complementary Slackness and Strong Duality: something is optimal!

How to Think About Linear Programming

Comparison to Systems of Linear Equations

Think back to linear systems of equations. Such a system consists of m linear equations in real-valued variables $x_1, \ldots, x_n$:

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \cdots + a_{12}x_n &= b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{22}x_n &= b_2 \\ &\vdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{m2}x_n &= b_m. \end{aligned}$$

The a_{ij}'s and the b_i's are given; the goal is to check whether or not there are values for the x_j's such that all m constraints are satisfied. We used Gaussian elimination; "solved" meant that the algorithm returns a feasible solution, or correctly reports that no feasible solution exists.

What about inequalities? The point of linear programming is to solve systems of linear equations *and inequalities*. Moreover, when there are multiple feasible solutions, we would like to compute the "best" one.

Ingredients of a Linear Program

Using the language of linear programming, we can express many of the computational problems that we know.

Ingredients of a Linear Program

a. *Decision variables* $x_1, \ldots, x_n \in \mathbb{R}$.

b. *Linear constraints*, each of the form

$$\sum_{j=1}^{n} a_j x_j \quad (*) \quad b_i,$$

where $(*)$ could be $\leq, \geq$, or $=$.

c. A *linear objective function* of the form

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

or

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

- The a_{ij}'s, b_i's, and c_j's are *constants*, part of the input.
- The x_j's are *variables*, what the algorithm is trying to set.
- When specifying constraints, there is no need to make use of both "$\leq$" and "$\geq$"inequalities—one can be transformed into the other just by multiplying all the coefficients by -1 (the a_{ij}'s and b_i's are allowed to be positive or negative).
- Equality constraints can be turned into two inequalities.
- min and max can easily be converted from one to another by multiplying by -1.

What's not allowed in a linear program? Non-linear variables—terms like $x_j^2, x_j x_k, \log(1 + x_j)$, etc. So whenever a decision variable appears in an expression, it is alone, possibly multiplied by a constant (and then summed with other such terms).

A Simple Example

To make linear programs more concrete and develop your geometric intuition about them, let's look at a toy example. (Many "real" examples of linear programs are coming shortly.) Suppose there are two decision variables x_1 and x_2—so we can visualize solutions as points (x_1, x_2) in the

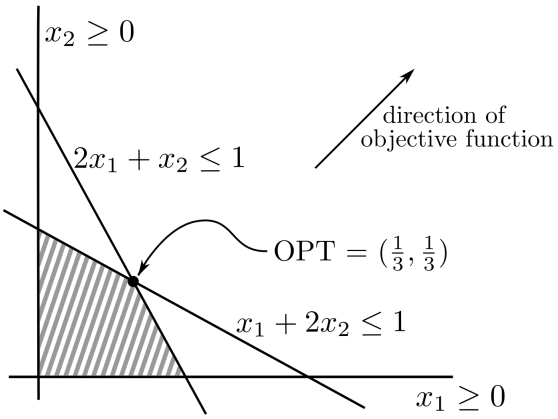

Figure 1: A toy example of a linear program.

plane. See Figure 1. Let's consider the (linear) objective function of maximizing the sum of the decision variables:

$$\max x_1 + x_2.$$

We'll look at four (linear) constraints:

$$\begin{aligned} x_1 &\geq 0 \\ x_2 &\geq 0 \\ 2x_1 + x_2 &\leq 1 \\ x_1 + 2x_2 &\leq 1. \end{aligned}$$

The feasible region is shaded in Figure . Geometrically, the objective function asks for the feasible point furthest "northeast" in the direction of the coefficient vector $(1, 1)$. Eyeballing, this point is $(\frac{1}{3}, \frac{1}{3})$, for an optimal objective function value of $\frac{2}{3}$.

Geometric Intuition

While it's always dangerous to extrapolate from two or three dimensions to an arbitrary number, the geometric intuition above remains valid for general linear programs, with an arbitrary number of dimensions (i.e., decision variables) and constraints. Even though we can't draw pictures when there are many dimensions, the relevant algebra carries over without any difficulties. Specifically:

a. A linear constraint in n dimensions corresponds to a halfspace in $\mathbb{R}^n$. Thus a feasible region is an intersection of halfspaces, the higher-dimensional analog of a polygon.[1]

b. When there is a unique optimal solution, it is a vertex (i.e., "corner") of the feasible region.

A few edge cases:

[1] A finite intersection of halfspaces is also called a "polyhedron;" in the common special case where the feasible region is bounded, it is called a "polytope."

a. There might be no feasible solutions at all. For example, if we add the constraint $x_1 + x_2 \geq 1$ to our toy example, then there are no longer any feasible solutions. Linear programming algorithms correctly detect when this case occurs.

b. The optimal objective function value is unbounded ($+\infty$ for a maximization problem, $-\infty$ for a minimization problem). Note a necessary but not sufficient condition for this case is that the feasible region is unbounded. For example, if we dropped the constraints $2x_1 + x_2 \leq 1$ and $x_1 + 2x_2 \leq 1$ from our toy example, then it would have unbounded objective function value. Again, linear programming algorithms correctly detect when this case occurs.

c. The optimal solution need not be unique, as a "side" of the feasible region might be parallel to the levels sets of the objective function. Whenever the feasible region is bounded, however, there always exists an optimal solution that is a vertex of the feasible region.[2]

A Case Study: Maximum-Weight Bipartite Matching

The Maximum-Weight Matching Problem

Given a graph $G = (V, E)$ choose a maximum weight matching—a set of edges S with maximum weight such that no vertex is covered by more than one edge.

a. *Decision variables:* What are we try to solve for? A set of edges S that is our matching. So our variables are x_e for each edge e, where we want $x_e = 1$ if e is in our matching S and 0 otherwise.

b. *Constraints:* We cannot put more than 1 edge that is incident to a vertex into our matching, so

$$\sum_{e:v \in e} x_e \leq 1 \quad \forall v$$

and similarly, we can never take a negative quantity of an edge, so

$$x_e \geq 0 \quad \forall e \in E.$$

c. *Objective function:* We want to maximize the weight of our matching:

$$\max \sum_{e \in E} x_e w_e$$

Note that this is again a linear function.

[2]There are some annoying edge cases for unbounded feasible regions, for example the linear program $\max(x_1 + x_2)$ subject to $x_1 + x_2 = 1$.

Maximum-Weight Matching as an Integer Program

$$\begin{aligned} \max \quad & \sum_{e \in E} x_e w_e & & \\ \text{subject to} \quad & \sum_{e : v \in e} x_e \leq 1 & \forall v \quad & \text{(vertex matched at most once)} \\ & x_e \in \{0, 1\} & \forall e \quad & \text{(integral)} \end{aligned}$$

Maximum-Weight Matching as a Linear Program

$$\begin{aligned} \max \quad & \sum_{e \in E} x_e w_e & & \\ \text{subject to} \quad & \sum_{e : v \in e} x_e \leq 1 & \forall v \quad & \text{(vertex matched at most once)} \\ & x_e \geq 0 & \forall e \quad & \text{(non-negativity)} \end{aligned}$$

Writing Problems as Linear Programs

Example 1: Grain Nutrients

Suppose BU has hired you to optimize nutrition for campus dining. There are two possible grains they can offer, grain 1 and grain 2, and each contains the macronutrients found in the table below, plus cost per kg for each of the grains.

Macros	Starch	Proteins	Vitamins	Cost ($/kg)
Grain 1	5	4	2	0.6
Grain 2	7	2	1	0.35

The nutrition requirement per day of starch, proteins, and vitamins is 8, 15, and 3 respectively. Determine how much of each grain to buy such that BU spends as little but meets its nutrition requirements.

Decision variables: amount of grain 1 (y_1) and grain 2 (y_2).

Objective: Minimize cost.

$$\min 0.6y_1 + 0.35y_2$$

Constraints:

$$\begin{aligned} 5y_1 + 7y_2 &\geq 8 && \text{(starch)} \\ 4y_1 + 2y_2 &\geq 15 && \text{(proteins)} \\ 2y_1 + 1y_2 &\geq 3 && \text{(vitamins)} \\ y_1, y_2 &\geq 0 && \text{(non-negativity)} \end{aligned}$$

Example 2: Transportation

You're working for a company that's producing widgets among two different factories and selling them from three different centers. Each month, widgets need to be transported from the factories to the centers. Below are the transportation costs from each factory to each center, along with the monthly supply and demand for each factory and center respectively. Determine how to route the widgets in a way that minimizes transportation costs.

Transit Cost	Center 1	Center 2	Center 3
Factory 1	5	5	3
Factory 2	6	4	1

- The supply per factory is 6 and 9 respectively.
- The demand per center is 8, 5, and 2 respectively.

Decision variables: x_{ij} is the number of widgets transported from factory i to center j.

Objective: Minimize cost.

$$\min \quad 5x_{11} + 5x_{12} + 3x_{13} + 6x_{21} + 4x_{22} + 1x_{23}$$

Constraints:

$$\begin{aligned} x_{11} + x_{12} + x_{13} &= 6 && \text{(Factor 1 supply)} \\ x_{21} + x_{22} + x_{23} &= 9 && \text{(Factor 2 supply)} \\ x_{11} + x_{21} &= 8 && \text{(Center 1 demand)} \\ x_{12} + x_{22} &= 5 && \text{(Center 2 demand)} \\ x_{13} + x_{23} &= 2 && \text{(Center 3 demand)} \\ x_{ij} &\geq 0 && \text{(non-negativity)} \end{aligned}$$

Converting to Normal Form

The "Normal Form" of a Linear Program looks like:

$$\begin{aligned} \max \quad & \mathbf{c}^T\mathbf{x} \\ \text{subject to} \quad & \mathbf{A}\mathbf{x} \leq \mathbf{b} \end{aligned}$$

Our Transportation problem had the LP:

$$\begin{aligned} \min \quad & 5x_{11} + 5x_{12} + 3x_{13} + 6x_{21} + 4x_{22} + 1x_{23} \\ \text{subject to} \quad & x_{11} + x_{12} + x_{13} = 6 && \text{(Factor 1 supply)} \\ & x_{21} + x_{22} + x_{23} = 9 && \text{(Factor 2 supply)} \\ & x_{11} + x_{21} = 8 && \text{(Center 1 demand)} \\ & x_{12} + x_{22} = 5 && \text{(Center 2 demand)} \\ & x_{13} + x_{23} = 2 && \text{(Center 3 demand)} \\ & x_{ij} \geq 0 && \text{(non-negativity)} \end{aligned}$$

How can we convert it to normal form—a maximization problem with all less-than-or-equal-to constraints?

First observe that $x_{11} + x_{12} + x_{13} = 6$ is equivalent to having both inequalities

$$x_{11} + x_{12} + x_{13} \leq 6 \qquad \text{and} \qquad x_{11} + x_{12} + x_{13} \geq 6.$$

But, we need both to be $\leq$ inequalities! We transform them to

$$x_{11} + x_{12} + x_{13} \leq 6 \qquad \text{and} \qquad -x_{11} - x_{12} - x_{13} \leq -6.$$

The resulting LP in normal form is:

$$\begin{aligned}
\max \quad & -5x_{11} - 5x_{12} - 3x_{13} - 6x_{21} - 4x_{22} - 1x_{23} & \\
\text{subject to} \quad & x_{11} + x_{12} + x_{13} \leq 6 & \text{(Factor 1 supply)} \\
& -x_{11} - x_{12} - x_{13} \leq -6 & \text{(Factor 1 supply)} \\
& x_{21} + x_{22} + x_{23} \leq 9 & \text{(Factor 2 supply)} \\
& -x_{21} - x_{22} - x_{23} \leq -9 & \text{(Factor 2 supply)} \\
& x_{11} + x_{21} \leq 8 & \text{(Center 1 demand)} \\
& -x_{11} - x_{21} \leq -8 & \text{(Center 1 demand)} \\
& x_{12} + x_{22} \leq 5 & \text{(Center 2 demand)} \\
& -x_{12} - x_{22} \leq -5 & \text{(Center 2 demand)} \\
& x_{13} + x_{23} \leq 2 & \text{(Center 3 demand)} \\
& -x_{13} - x_{23} \leq -2 & \text{(Center 3 demand)} \\
& x_{ij} \geq 0 & \text{(non-negativity)}
\end{aligned}$$

The Dual of a Linear Program

Every linear program has a *dual* linear program. We call the original linear program the *primal*. There are a bunch of amazing properties that come from LP duality.

Going back to our nutrition example, we want to find the dual linear program. A maximization problem's dual is a minimization problem. Here, we have a minimization problem, so the dual will be a maximization problem.

To take the dual: Label each primal constraint with a new dual variable. In our new linear program, each dual constraint will correspond to a primal variable. For the left-hand side, count up the appearances of this constraint's primal variable (e.g., x_1) in each of the primal constraints and multiply them by the dual variable for those constraints. That is, if x_1 appears 5 times ($5x_1$) in constraint for y_1, then add $5y_1$ to x_1's constraint. Don't forget to include its appearance in the primal's objective function, but this will be the right-hand side of the constraint. Finally, the dual objective function is given by the right-hand side coefficients and their correspondence to the dual variables via the constraints in the primal. (See below).

Primal:

$$\begin{aligned} \min \quad & 0.6y_1 + 0.35y_2 & & \\ \text{subject to} \quad & 5y_1 + 7y_2 \geq 8 & \text{(starch)} \quad & (x_1) \\ & 4y_1 + 2y_2 \geq 15 & \text{(proteins)} \quad & (x_2) \\ & 2y_1 + 1y_2 \geq 3 & \text{(vitamins)} \quad & (x_3) \\ & y_1, y_2 \geq 0 & \text{(non-negativity)} & \end{aligned}$$

Dual:

$$\begin{aligned} \max \quad & 8x_1 + 15x_2 + 3x_3 & & \\ \text{subject to} \quad & 5x_1 + 4x_2 + 2x_3 \leq 0.6 & \text{(grain 1)} \quad & (y_1) \\ & 7x_1 + 2x_2 + 1x_3 \leq 0.35 & \text{(grain 2)} \quad & (y_2) \\ & x_1, x_2, x_3 \geq 0 & \text{(non-negativity)} & \end{aligned}$$

Sometimes, the dual can even be interpreted as a related problem, as we will see.

The following is the normal form for a maximization problem primal and its dual:

$$\begin{aligned} \max \quad & \mathbf{c}^T\mathbf{x} \\ \text{subject to} \quad & \mathbf{A}\mathbf{x} \leq \mathbf{b} \\ & \mathbf{x} \geq 0 \end{aligned} \qquad \begin{aligned} \min \quad & \mathbf{y}^T\mathbf{b} \\ \text{subject to} \quad & \mathbf{A}^T\mathbf{y} \geq \mathbf{c} \\ & \mathbf{y} \geq 0 \end{aligned}$$

For the above example:

$$\mathbf{A} = \begin{bmatrix} 5 & 4 & 2 \\ 7 & 2 & 1 \end{bmatrix} \quad \mathbf{b} = \begin{bmatrix} 0.6 \\ 0.35 \end{bmatrix} \quad \mathbf{c} = \begin{bmatrix} 8 \\ 15 \\ 3 \end{bmatrix}$$

Example 3: Unweighted Maximum Matching

Given a graph $G = (V, E)$ choose a maximum size matching—a set of edges S such that no vertex is covered by more than one edge.

Decision variables: x_e indicating whether edge e is in the matching.

Primal Linear Program:

$$\begin{aligned} \max \quad & \sum_{e \in E} x_e & & & \\ \text{subject to} \quad & \sum_{e : v \in e} x_e \leq 1 & \forall v \quad & \text{(vertex matched at most once)} \quad & (y_v) \\ & x_e \geq 0 & \forall e \quad & \text{(non-negativity)} & \end{aligned}$$

Taking the dual of the above primal, we get the following linear program:

$$\begin{aligned} \min \quad & \sum_{v \in V} y_v & & \\ \text{subject to} \quad & \sum_{v \in e} y_v \geq 1 & \forall e \quad & \text{(edge covered)} \quad (x_e) \\ & y_v \geq 0 & \forall v \quad & \text{(non-negativity)} \end{aligned}$$

What problem is this? (Fractional) Vertex Cover!

Conditions for Optimality

Weak Duality

Theorem 1 (Weak Duality)**.** *If* $\mathbf{x}$ *is feasible in (P) and* $\mathbf{y}$ *is feasible in (D) then* $\mathbf{c}^T\mathbf{x} \leq \mathbf{b}^T\mathbf{y}$.

Proof.

$$\mathbf{c}^T\mathbf{x} \overset{1}{\leq} (\mathbf{A}^T\mathbf{y})\mathbf{x} = \mathbf{y}^T\mathbf{A}\mathbf{x} \overset{2}{\leq} \mathbf{y}^T\mathbf{b} = \mathbf{b}^T\mathbf{y}.$$

Where (1) follows by the dual constraints $\mathbf{A}^T\mathbf{y} \geq \mathbf{c}$ and (2) follows by the primal constraints $\mathbf{A}\mathbf{x} \leq \mathbf{b}$. □

This theorem says that *any* feasible solution to the primal is a *lower bound* to *any* feasible solution to the dual, and likewise, any feasible solution to the dual is an *upper bound* to the primal.

That is, fractional vertex cover gives an upper bound on how large the (fractional) maximum matching can be, and likewise, fractional maximum matching gives a lower bound on how small the minimum (fractional) vertex cover can be.

What is not trivial (or by definition) is *strong duality*, and in fact, it is so involved that we will not even prove the hard direction: that an optimal solution always exists.

Conditions for Optimality

Strong Duality

Strong duality states that everything in fact needs to hold with equality to be optimal.

Theorem 2 (Strong Duality)**.** *A pair of solutions* $(\mathbf{x}^*, \mathbf{y}^*)$ *are optimal for the primal and dual respectively if and only if* $\mathbf{c}^T\mathbf{x}^* = \mathbf{b}^T\mathbf{y}^*$.

Proof. ($\Leftarrow$) The *if* direction is easy to see: we know that the dual gives an upper bound on the primal, so if these objectives are equal, then the primal objective that we are trying to maximize could not possible get any larger, as it's always *at most* the dual's objective. This is *as tight as possible*.

($\Rightarrow$) The *only if* direction is harder to prove, and we'll skip it for now. □

Complementary Slackness

We rewrite the primal and dual with each constraint separated, and then formalize another condition for optimality called *complementary slackness*, which states that for each corresponding constraint and variable, at most one can be slack in an optimal solution.

Primal (P):

$$\begin{aligned} \max \quad & \mathbf{c}^T\mathbf{x} \\ \text{subject to} \quad & \sum_i a_{ji}x_i \le b_j \quad \forall j \quad (y_j) \\ & x_i \ge 0 \quad \forall i \end{aligned}$$

Dual (D):

$$\begin{aligned} \min \quad & \mathbf{y}^T\mathbf{b} \\ \text{subject to} \quad & \sum_i a_{ij}y_i \ge c_i \quad \forall i \quad (x_i) \\ & y_j \ge 0 \quad \forall j \end{aligned}$$

Theorem 3 (Complementary Slackness)**.** *A pair of solutions* $(\mathbf{x}^*, \mathbf{y}^*)$ *are optimal for the primal and dual respectively if and only if the following complementary slackness conditions (1) and (2) hold:*

$$\sum_i a_{ji}x_i = b_j \quad \textit{or} \quad y_j = 0 \qquad (1) \qquad\qquad \sum_i a_{ij}y_i = c_i \quad \textit{or} \quad x_i = 0. \qquad (2)$$

Proof. ($\Rightarrow$) According to complementary slackness, by rearranging our constraint, either $\sum_i a_{ji}x_i - b_j = 0$ or $y_j = 0$. This ensures that the multiplied quantity $\left(\sum_i a_{ji}x_i - b_j\right) y_j = 0$, as *one* of the two terms on the left-hand side must be 0. Then multiplying out and rearranging gives that $y_j \sum_i a_{ji}x_i = y_j b_j$. This process with all rows gives the equality from complementary slackness that $\mathbf{y}^T\mathbf{A}\mathbf{x} = \mathbf{y}^T\mathbf{b}$.

Similarly, using the condition that $\sum_i a_{ij}y_i = c_i$ or $x_i = 0$ gives that $\mathbf{c}^T\mathbf{x} = (\mathbf{A}^T\mathbf{y})\mathbf{x}$.

Then following our inequalities in the proof of weak duality, they now all hold with equality, so by Strong Duality, $(\mathbf{x}, \mathbf{y})$ are optimal solutions to the primal and dual.

$$\mathbf{c}^T\mathbf{x} \quad = \quad (\mathbf{A}^T\mathbf{y})\mathbf{x} \quad = \quad \mathbf{y}^T\mathbf{A}\mathbf{x} \quad = \quad \mathbf{y}^T\mathbf{b} \quad = \quad \mathbf{b}^T\mathbf{y}.$$

($\Leftarrow$) Similarly, if Strong Duality holds, the above inequalities hold with equality, in which case it must be that $y_j \sum_i a_{ji}x_i = y_j b_j$ for all j and $\sum_i a_{ij}y_i x_i = c_i x_i$ for all i, and hence that either $\sum_i a_{ji}x_i - b_j = 0$ or $y_j = 0$ for all j and that either $\sum_i a_{ij}y_i = c_i$ or $x_i = 0$ for all i. □

Maximizing Welfare in the Unit Demand Setting

Given n unit-demand bidders and m items, determine the allocation rule that maximizes welfare. We do this by formulating a linear program, determining our objective, decision variables, and constraints:

$$
\begin{aligned}
\max \quad & \sum_{i=1}^{n}\sum_{j=1}^{m} v_{ij}x_{ij} & & \\
\text{subject to} \quad & \sum_{i=1}^{n} x_{ij} \leq 1 & \forall j \quad & \text{(items allocated at most once)} \\
& \sum_{j=1}^{m} x_{ij} \leq 1 & \forall i \quad & \text{(bidders unit demand)} \\
& x_{ij} \geq 0 & \forall i,j \quad & \text{(non-negativity)}
\end{aligned}
$$

We then formulate the dual:

$$
\begin{aligned}
\min \quad & \sum_{i=1}^{n} u_i + \sum_{j=1}^{m} p_j & & \\
\text{subject to} \quad & u_i + p_j \geq v_{ij} & \forall (i,j) \quad & \text{(IC)} \\
& u_i, p_j \geq 0 & \forall i,j \quad & \text{(non-negativity)}
\end{aligned}
$$

By rewriting our first constraint as

$$u_i \geq v_{ij} - p_j,$$

we reinterpret it as an incentive compatibility constraint. Instead of determining an allocation, we determine buyer utilities and item prices, the sum of which we minimize.

On our homework, we will use this to prove that for gross substitutes valuations, the optimal primal allocation $\mathbf{x}$ and dual solution $\mathbf{p}$ form a Walrasian equilibria, and this is precisely the valuation class for which welfare can be maximized in polynomial time.

We will need to use the following theory of when there exist polynomial-time algorithms for linear programs.

Separation Oracles

Fact 1 (Ellipsoid Algorithm)**.** Every linear program that admits a polynomial-time separation oracle can be solved in polynomial time.

Consider a linear program such that:

- **a.** There are n decision variables.
- **b.** There are any number of constraints, for example, exponential in n. These constraints are not provided explicitly as input.
- **c.** There is a polynomial-time *separation oracle* for the set of constraints. By "polynomial-time," we mean running time polynomial in n and the maximum number of bits of precision required.

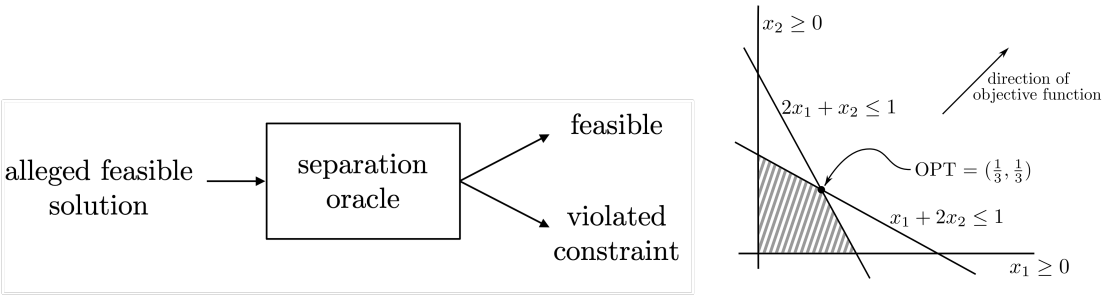

Figure 2: Left: A sketch of a separation oracle. For example, in the toy example on the right, on the alleged feasible solution $(\frac{1}{3}, \frac{1}{2})$, the separation oracle may return the violated constraint $x_1 + 2x_2 \leq 1$.

> A separation oracle (Figure 1) is a subroutine that takes as input an alleged feasible solution to the LP, and either (i) correctly declares the solution to be feasible, or (ii) correctly declares the solution to be infeasible, and more strongly provides a proof of infeasibility in the form of a constraint that the proposed solution violates.

(The ellipsoid algorithm is not actually practical, but there are other algorithms that *are* often practically useful that rely on a separation oracle, such as cutting plane methods.)