

Exercises

- 7.1. Consider the following linear program.

$$\begin{aligned} \text{maximize } & 5x + 3y \\ & 5x - 2y \geq 0 \\ & x + y \leq 7 \\ & x \leq 5 \\ & x \geq 0 \\ & y \geq 0 \end{aligned}$$

Plot the feasible region and identify the optimal solution.

- 7.2. Duckwheat is produced in Kansas and Mexico and consumed in New York and California. Kansas produces 15 shnupells of duckwheat and Mexico 8. Meanwhile, New York consumes 10 shnupells and California 13. The transportation costs per shnupell are \$4 from Mexico to New York, \$1 from Mexico to California, \$2 from Kansas to New York, and \$3 and from Kansas to California.

Write a linear program that decides the amounts of duckwheat (in shnupells and fractions of a shnupell) to be transported from each producer to each consumer, so as to minimize the overall transportation cost.

- 7.3. A cargo plane can carry a maximum weight of 100 tons and a maximum volume of 60 cubic meters. There are three materials to be transported, and the cargo company may choose to carry any amount of each, upto the maximum available limits given below.

- Material 1 has density 2 tons/cubic meter, maximum available amount 40 cubic meters, and revenue \$1,000 per cubic meter.
- Material 2 has density 1 ton/cubic meter, maximum available amount 30 cubic meters, and revenue \$1,200 per cubic meter.
- Material 3 has density 3 tons/cubic meter, maximum available amount 20 cubic meters, and revenue \$12,000 per cubic meter.

Write a linear program that optimizes revenue within the constraints.

- 7.4. Moe is deciding how much Regular Duff beer and how much Duff Strong beer to order each week. Regular Duff costs Moe \$1 per pint and he sells it at \$2 per pint; Duff Strong costs Moe \$1.50 per pint and he sells it at \$3 per pint. However, as part of a complicated marketing scam, the Duff company will only sell a pint of Duff Strong for each two pints or more of Regular Duff that Moe buys. Furthermore, due to past events that are better left untold, Duff will not sell Moe more than 3,000 pints per week. Moe knows that he can sell however much beer he has. Formulate a linear program for deciding how much Regular Duff and how much Duff Strong to buy, so as to maximize Moe's profit. Solve the program geometrically.
- 7.5. The Canine Products company offers two dog foods, Frisky Pup and Husky Hound, that are made from a blend of cereal and meat. A package of Frisky Pup requires 1 pound of cereal and 1.5 pounds of meat, and sells for \$7. A package of Husky Hound uses 2 pounds of cereal and 1 pound of meat, and sells for \$6. Raw cereal costs \$1 per pound and raw meat costs \$2 per pound. It also costs \$1.40 to package the Frisky Pup and \$0.60 to package the Husky Hound. A total of 240,000 pounds of cereal and 180,000 pounds of meat are available each month. The only production bottleneck is that the factory can only package 110,000 bags of Frisky Pup per month. Needless to say, management would like to maximize profit.

- (a) Formulate the problem as a linear program in two variables.
 (b) Graph the feasible region, give the coordinates of every vertex, and circle the vertex maximizing profit. What is the maximum profit possible?
- 7.6. Give an example of a linear program in two variables whose feasible region is infinite, but such that there is an optimum solution of bounded cost.
- 7.7. Find necessary and sufficient conditions on the reals a and b under which the linear program

$$\begin{aligned} \max \quad & x + y \\ \text{s.t.} \quad & ax + by \leq 1 \\ & x, y \geq 0 \end{aligned}$$

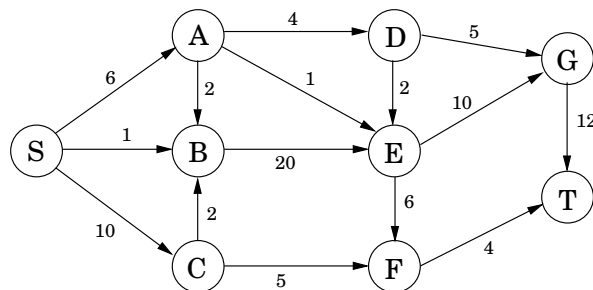
- (a) Is infeasible.
 (b) Is unbounded.
 (c) Has a unique optimal solution.
- 7.8. You are given the following points in the plane:

$$(1, 3), (2, 5), (3, 7), (5, 11), (7, 14), (8, 15), (10, 19).$$

You want to find a line $ax + by = c$ that approximately passes through these points (no line is a perfect fit). Write a linear program (you don't need to solve it) to find the line that minimizes the *maximum absolute error*,

$$\max_{1 \leq i \leq 7} |ax_i + by_i - c|.$$

- 7.9. A *quadratic programming* problem seeks to maximize a quadratic objective function (with terms like $3x_1^2$ or $5x_1x_2$) subject to a set of linear constraints. Give an example of a quadratic program in two variables x_1, x_2 such that the feasible region is nonempty and bounded, and yet none of the vertices of this region optimize the (quadratic) objective.
- 7.10. For the following network, with edge capacities as shown, find the maximum flow from S to T , along with a matching cut.



- 7.11. Write the dual to the following linear program.

$$\begin{aligned} \max \quad & x + y \\ \text{s.t.} \quad & 2x + y \leq 3 \\ & x + 3y \leq 5 \\ & x, y \geq 0 \end{aligned}$$

Find the optimal solutions to both primal and dual LPs.

7.12. For the linear program

$$\begin{aligned} \max \quad & x_1 - 2x_3 \\ \text{s.t.} \quad & x_1 - x_2 \leq 1 \\ & 2x_2 - x_3 \leq 1 \\ & x_1, x_2, x_3 \geq 0 \end{aligned}$$

prove that the solution $(x_1, x_2, x_3) = (3/2, 1/2, 0)$ is optimal.

7.13. *Matching pennies*. In this simple two-player game, the players (call them R and C) each choose an outcome, *heads* or *tails*. If both outcomes are equal, C gives a dollar to R ; if the outcomes are different, R gives a dollar to C .

- (a) Represent the payoffs by a 2×2 matrix.
- (b) What is the value of this game, and what are the optimal strategies for the two players?

7.14. The pizza business in Little Town is split between two rivals, Tony and Joey. They are each investigating strategies to steal business away from the other. Joey is considering either lowering prices or cutting bigger slices. Tony is looking into starting up a line of gourmet pizzas, or offering outdoor seating, or giving free sodas at lunchtime. The effects of these various strategies are summarized in the following payoff matrix (entries are dozens of pizzas, Joey's gain and Tony's loss).

| | | TONY | | |
|------|---------------|---------|---------|-----------|
| | | Gourmet | Seating | Free soda |
| JOEY | Lower price | +2 | 0 | -3 |
| | Bigger slices | -1 | -2 | +1 |

For instance, if Joey reduces prices and Tony goes with the gourmet option, then Tony will lose 2 dozen pizzas worth of business to Joey.

What is the value of this game, and what are the optimal strategies for Tony and Joey?

7.15. Find the value of the game specified by the following payoff matrix.

| | | | |
|----|----|----|----|
| 0 | 0 | -1 | -1 |
| 0 | 1 | -2 | -1 |
| -1 | -1 | 1 | 1 |
| -1 | 0 | 0 | 1 |
| 1 | -2 | 0 | -3 |
| 1 | -1 | -1 | -1 |
| 0 | -3 | 2 | -1 |
| 0 | -2 | 1 | -1 |

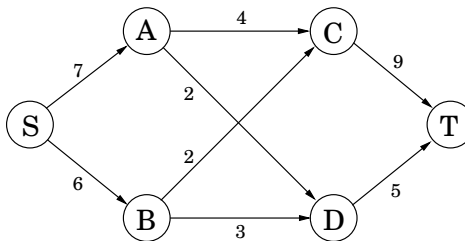
(Hint: Consider the mixed strategies $(1/3, 0, 0, 1/2, 1/6, 0, 0, 0)$ and $(2/3, 0, 0, 1/3)$.)

7.16. A salad is any combination of the following ingredients: (1) tomato, (2) lettuce, (3) spinach, (4) carrot, and (5) oil. Each salad must contain: (A) at least 15 grams of protein, (B) at least 2 and at most 6 grams of fat, (C) at least 4 grams of carbohydrates, (D) at most 100 milligrams of sodium. Furthermore, (E) you do not want your salad to be more than 50% greens by mass. The nutritional contents of these ingredients (per 100 grams) are

| ingredient | energy (kcal) | protein (grams) | fat (grams) | carbohydrate (grams) | sodium (milligrams) |
|------------|------------------|--------------------|----------------|-------------------------|------------------------|
| tomato | 21 | 0.85 | 0.33 | 4.64 | 9.00 |
| lettuce | 16 | 1.62 | 0.20 | 2.37 | 8.00 |
| spinach | 371 | 12.78 | 1.58 | 74.69 | 7.00 |
| carrot | 346 | 8.39 | 1.39 | 80.70 | 508.20 |
| oil | 884 | 0.00 | 100.00 | 0.00 | 0.00 |

Find a linear programming applet on the Web and use it to make the salad with the fewest calories under the nutritional constraints. Describe your linear programming formulation and the optimal solution (the quantity of each ingredient and the value). Cite the Web resources that you used.

7.17. Consider the following network (the numbers are edge capacities).



- Find the maximum flow f and a minimum cut.
- Draw the residual graph G_f (along with its edge capacities). In this residual network, mark the vertices reachable from S and the vertices from which T is reachable.
- An edge of a network is called a *bottleneck edge* if increasing its capacity results in an increase in the maximum flow. List all bottleneck edges in the above network.
- Give a very simple example (containing at most four nodes) of a network which has no bottleneck edges.
- Give an efficient algorithm to identify all bottleneck edges in a network. (*Hint*: Start by running the usual network flow algorithm, and then examine the residual graph.)

7.18. There are many common variations of the maximum flow problem. Here are four of them.

- There are many sources and many sinks, and we wish to maximize the total flow from all sources to all sinks.
- Each *vertex* also has a capacity on the maximum flow that can enter it.
- Each edge has not only a capacity, but also a *lower bound* on the flow it must carry.
- The outgoing flow from each node u is not the same as the incoming flow, but is smaller by a factor of $(1 - \epsilon_u)$, where ϵ_u is a loss coefficient associated with node u .

Each of these can be solved efficiently. Show this by reducing (a) and (b) to the original max-flow problem, and reducing (c) and (d) to linear programming.

7.19. Suppose someone presents you with a solution to a max-flow problem on some network. Give a *linear* time algorithm to determine whether the solution does indeed give a maximum flow.

7.20. Consider the following generalization of the maximum flow problem.

You are given a directed network $G = (V, E)$ with edge capacities $\{c_e\}$. Instead of a single (s, t) pair, you are given multiple pairs $(s_1, t_1), (s_2, t_2), \dots, (s_k, t_k)$, where the s_i are sources of G and the t_i are sinks of G . You are also given k demands d_1, \dots, d_k . The goal is to find k flows $f^{(1)}, \dots, f^{(k)}$ with the following properties:

- $f^{(i)}$ is a valid flow from s_i to t_i .
- For each edge e , the total flow $f_e^{(1)} + f_e^{(2)} + \dots + f_e^{(k)}$ does not exceed the capacity c_e .
- The size of each flow $f^{(i)}$ is at least the demand d_i .
- The size of the *total* flow (the sum of the flows) is as large as possible.

How would you solve this problem?

7.21. An edge of a flow network is called *critical* if decreasing the capacity of this edge results in a decrease in the maximum flow. Give an efficient algorithm that finds a critical edge in a network.

7.22. In a particular network $G = (V, E)$ whose edges have integer capacities c_e , we have already found the maximum flow f from node s to node t . However, we now find out that one of the capacity values we used was wrong: for edge (u, v) we used c_{uv} whereas it should have been $c_{uv} - 1$. This is unfortunate because the flow f uses that particular edge at full capacity: $f_{uv} = c_{uv}$.

We could redo the flow computation from scratch, but there's a faster way. Show how a new optimal flow can be computed in $O(|V| + |E|)$ time.

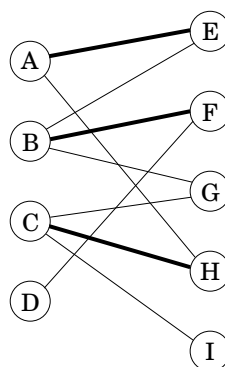
7.23. A *vertex cover* of an undirected graph $G = (V, E)$ is a subset of the vertices which touches every edge—that is, a subset $S \subset V$ such that for each edge $\{u, v\} \in E$, one or both of u, v are in S .

Show that the problem of finding the minimum vertex cover in a *bipartite* graph reduces to maximum flow. (*Hint*: Can you relate this problem to the minimum cut in an appropriate network?)

7.24. *Direct bipartite matching*. We've seen how to find a maximum matching in a bipartite graph via reduction to the maximum flow problem. We now develop a direct algorithm.

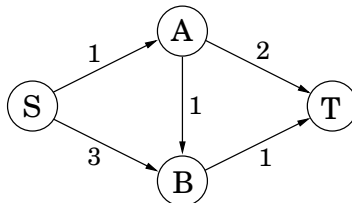
Let $G = (V_1 \cup V_2, E)$ be a bipartite graph (so each edge has one endpoint in V_1 and one endpoint in V_2), and let $M \in E$ be a matching in the graph (that is, a set of edges that don't touch). A vertex is said to be *covered* by M if it is the endpoint of one of the edges in M . An *alternating path* is a path of odd length that starts and ends with a non-covered vertex, and whose edges alternate between M and $E - M$.

(a) In the bipartite graph below, a matching M is shown in bold. Find an alternating path.



- (b) Prove that a matching M is maximal if and only if there does not exist an alternating path with respect to it.
- (c) Design an algorithm that finds an alternating path in $O(|V| + |E|)$ time using a variant of breadth-first search.
- (d) Give a direct $O(|V| \cdot |E|)$ algorithm for finding a maximal matching in a bipartite graph.

7.25. *The dual of maximum flow.* Consider the following network with edge capacities.



- (a) Write the problem of finding the maximum flow from S to T as a linear program.
- (b) Write down the dual of this linear program. There should be a dual variable for each edge of the network and for each vertex other than S, T .

Now we'll solve the same problem in full generality. Recall the linear program for a general maximum flow problem (Section 7.2).

- (c) Write down the dual of this general flow LP, using a variable y_e for each edge and x_u for each vertex $u \neq s, t$.
- (d) Show that any solution to the general dual LP must satisfy the following property: for any directed path from s to t in the network, the sum of the y_e values along the path must be at least 1.
- (e) What are the intuitive meanings of the dual variables? Show that any $s - t$ cut in the network can be translated into a dual feasible solution whose cost is exactly the capacity of that cut.

7.26. In a satisfiable system of linear inequalities

$$\begin{array}{rcl}
 a_{11}x_1 + \cdots + a_{1n}x_n & \leq & b_1 \\
 & \vdots & \\
 a_{m1}x_1 + \cdots + a_{mn}x_n & \leq & b_m
 \end{array}$$

we describe the j th inequality as *forced-equal* if it is satisfied with equality by *every* solution $\mathbf{x} = (x_1, \dots, x_n)$ of the system. Equivalently, $\sum_i a_{ji}x_i \leq b_j$ is *not* forced-equal if there exists an \mathbf{x} that satisfies the whole system and such that $\sum_i a_{ji}x_i < b_j$.

For example, in

$$\begin{array}{rcl}
 x_1 + x_2 & \leq & 2 \\
 -x_1 - x_2 & \leq & -2 \\
 x_1 & \leq & 1 \\
 -x_2 & \leq & 0
 \end{array}$$

the first two inequalities are forced-equal, while the third and fourth are not. A solution \mathbf{x} to the system is called *characteristic* if, for every inequality I that is not forced-equal, \mathbf{x} satisfies I without equality. In the instance above, such a solution is $(x_1, x_2) = (-1, 3)$, for which $x_1 < 1$ and $-x_2 < 0$ while $x_1 + x_2 = 2$ and $-x_1 - x_2 = -2$.

- (a) Show that any satisfiable system has a characteristic solution.
- (b) Given a satisfiable system of linear inequalities, show how to use linear programming to determine which inequalities are forced-equal, and to find a characteristic solution.

7.27. Show that the *change-making problem* (Exercise 6.17) can be formulated as an integer linear program. Can we solve this program as an LP, in the certainty that the solution will turn out to be integral (as in the case of bipartite matching)? Either prove it or give a counterexample.

7.28. A *linear program for shortest path*. Suppose we want to compute the shortest path from node s to node t in a directed graph with edge lengths $l_e > 0$.

- (a) Show that this is equivalent to finding an $s - t$ flow f that minimizes $\sum_e l_e f_e$ subject to $\text{size}(f) = 1$. There are no capacity constraints.
- (b) Write the shortest path problem as a linear program.
- (c) Show that the dual LP can be written as

$$\begin{aligned} \max \quad & x_s - x_t \\ & x_u - x_v \leq l_{uv} \text{ for all } (u, v) \in E \end{aligned}$$

- (d) An interpretation for the dual is given in the box on page 223. Why isn't our dual LP identical to the one on that page?

7.29. *Hollywood*. A film producer is seeking actors and investors for his new movie. There are n available actors; actor i charges s_i dollars. For funding, there are m available investors. Investor j will provide p_j dollars, but only on the condition that certain actors $L_j \subseteq \{1, 2, \dots, n\}$ are included in the cast (*all* of these actors L_j must be chosen in order to receive funding from investor j).

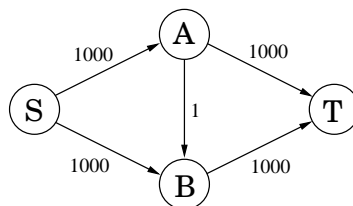
The producer's profit is the sum of the payments from investors minus the payments to actors. The goal is to maximize this profit.

- (a) Express this problem as an integer linear program in which the variables take on values $\{0, 1\}$.
- (b) Now relax this to a linear program, and show that there must in fact be an *integral* optimal solution (as is the case, for example, with maximum flow and bipartite matching).

7.30. *Hall's theorem*. Returning to the matchmaking scenario of Section 7.3, suppose we have a bipartite graph with boys on the left and an equal number of girls on the right. Hall's theorem says that there is a perfect matching if and only if the following condition holds: any subset S of boys is connected to at least $|S|$ girls.

Prove this theorem. (*Hint*: The max-flow min-cut theorem should be helpful.)

7.31. Consider the following simple network with edge capacities as shown.



- (a) Show that, if the Ford-Fulkerson algorithm is run on this graph, a careless choice of updates might cause it to take 1000 iterations. Imagine if the capacities were a million instead of 1000!

We will now find a strategy for choosing paths under which the algorithm is guaranteed to terminate in a reasonable number of iterations.

Consider an arbitrary directed network $(G = (V, E), s, t, \{c_e\})$ in which we want to find the maximum flow. Assume for simplicity that all edge capacities are at least 1, and define the capacity of an $s - t$ path to be the smallest capacity of its constituent edges. The *fattest path* from s to t is the path with the most capacity.

- (b) Show that the fattest $s - t$ path in a graph can be computed by a variant of Dijkstra's algorithm.
- (c) Show that the maximum flow in G is the sum of individual flows along at most $|E|$ paths from s to t .
- (d) Now show that if we always increase flow along the fattest path in the residual graph, then the Ford-Fulkerson algorithm will terminate in at most $O(|E| \log F)$ iterations, where F is the size of the maximum flow. (*Hint:* It might help to recall the proof for the greedy set cover algorithm in Section 5.4.)

In fact, an even simpler rule—finding a path in the residual graph using breadth-first search—guarantees that at most $O(|V| \cdot |E|)$ iterations will be needed.