

# Linear Programming

Advanced Algorithms (CSE 794A)

Instructor: Tamal K. Dey

Scribe: Rohit Dinakar \*

April 30, 2009

## 1 Introduction

Many problems can be formulated as maximizing or minimizing an objective, given limited resources and competing constraints. If we can specify the objective as a linear function of certain variables, and if we can specify the constraints on resources as equalities or inequalities on those variables, then we have a linear programming problem.

## 2 Linear Programs

In the general linear programming problem, we wish to optimize a linear function subject to a set of linear inequalities. Given a set of real numbers  $a_1, a_2, \dots, a_n$  and a set of variables  $x_1, x_2, \dots, x_n$ , a *linear function*  $f$  on those variables is defined by

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

If  $b$  is a real number and  $f$  is a linear function, then the equation

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

is a *linear equality* and the linear inequalities

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

and

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

are *linear inequalities*.

---

\*Department of Computer Science and Engineering, The Ohio State University, Columbus, Ohio, USA.

## 2.1 Example

Let us consider the following linear program with two variables:

Maximize

$$x_1 + x_2 \tag{1}$$

subject to

$$4x_1 - x_2 \leq 8 \tag{2}$$

$$2x_1 + x_2 \leq 10 \tag{3}$$

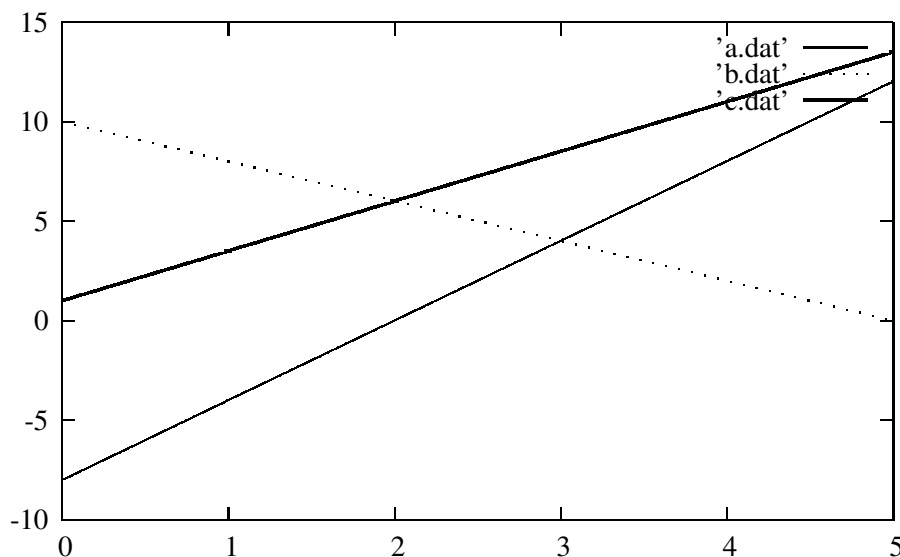
$$5x_1 - 2x_2 \geq -2 \tag{4}$$

$$x_1, x_2 \geq 0 \tag{5}$$

## 2.2 Feasible Solution

We call any setting of the variables  $x_1$  and  $x_2$  that satisfies all the constraints (2)-(5) a *feasible solution* to the linear program.

If we graph the constraints in the  $(x_1, x_2)$ -Cartesian coordinate system, as shown in the figure below, we see that the set of feasible solutions forms a convex region in the two-dimensional space. We call this convex region the *feasible region*. Each constraint is represented by a line. The intersection of the constraints is the feasible region.



## 2.3 Objective Function

The function that we are trying to optimize is called the *objective function*. The value of the objective function at a particular point is called the *objective value*.

## 2.4 Algorithms for linear programming

The **simplex algorithm** takes as input a linear program and returns an optimal solution. It starts at some vertex of the simplex and performs a sequence of iterations. The simplex algorithm terminates when it reaches a local maximum, which is a vertex from which all neighbouring vertices have a smaller objective

value. Because the feasible region is convex and the objective function is linear, this local optimum is actually a global optimum. However, the simplex algorithm can require exponential time in the worst case.

The first polynomial-time algorithm for linear programming was the **ellipsoid algorithm**, which runs slowly in practice. A second class of polynomial-time algorithms are known as **interior-point methods**. For large inputs, the performance of interior-point algorithms can be competitive with, and sometimes faster than, the simplex algorithm. If we add to a linear program the additional requirement that all variables take on integer values, we have an **integer linear program**. Finding a feasible solution to this problem is NP-hard.

### 3 Standard Form

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

$$\sum_{j=1}^n c_j x_j \tag{6}$$

subject to

$$\sum_{j=1}^n a_{i,j} x_j \leq b_i \text{ for } i = 1, 2, \dots, m \tag{7}$$

$$x_j \geq 0 \text{ for } j = 1, 2, \dots, n. \tag{8}$$

Generalizing the terminology we introduced for the two-variable linear program, we call expression (6) the *objective function* and the  $n + m$  inequalities in lines (7) and (8) the *constraints*. The  $n$  constraints in line (8) are called the *nonnegativity constraints*.

A linear program may not be in standard form for one of four possible reasons:

- (i) The objective function may be a minimization rather than a maximization.
- (ii) There may be variables without nonnegativity constraints.
- (iii) There may be equality constraints, which have an equal sign rather than a less-than-or-equal-to sign.
- (iv) There may be inequality constraints, but instead of having a less-than-or-equal-to sign, they have a greater-than-or-equal-to sign.

#### 3.1 Example

Minimize

$$-2x_1 + 3x_2 \tag{9}$$

subject to

$$x_1 + x_2 = 7 \tag{10}$$

$$x_1 - 2x_2 \leq 4 \tag{11}$$

$$x_1 \geq 0 \tag{12}$$

and we negate the coefficients of the objective function, we obtain

Maximize

$$2x_1 - 3x_2 \tag{13}$$

subject to

$$x_1 + x_2 = 7 \tag{14}$$

$$x_1 - 2x_2 \leq 4 \tag{15}$$

$$x_1 \geq 0 \tag{16}$$

Continuing the example, we want to ensure that each variable has a corresponding nonnegativity constraint. Variable  $x_1$  has such a constraint, but variable  $x_2$  does not. Therefore, we replace  $x_2$  by two variables  $x'_2$  and  $x''_2$ , and we modify the linear program to obtain

Maximize

$$2x_1 - 3x'_2 + 3x''_2 \tag{17}$$

subject to

$$x_1 + x'_2 - x''_2 = 7 \tag{18}$$

$$x_1 - 2x'_2 + 2x''_2 \leq 4 \tag{19}$$

$$x_1, x'_2, x''_2 \geq 0 \tag{20}$$

Finishing our example, we replace the equality in constraint (18) by two inequalities, obtaining

Maximize

$$2x_1 - 3x'_2 + 3x''_2 \tag{21}$$

subject to

$$x_1 + x'_2 - x''_2 \leq 7 \tag{22}$$

$$x_1 + x'_2 - x''_2 \geq 7 \tag{23}$$

$$x_1 - 2x'_2 + 2x''_2 \leq 4 \tag{24}$$

$$x_1, x'_2, x''_2 \geq 0 \tag{25}$$

Finally, we negate constraint (23), obtaining the standard form

Maximize

$$2x_1 - 3x'_2 + 3x''_2 \tag{26}$$

subject to

$$-x_1 - x'_2 + x''_2 \leq -7 \tag{27}$$

$$x_1 + x'_2 - x''_2 \leq 7 \tag{28}$$

$$x_1 - 2x'_2 + 2x''_2 \leq 4 \tag{29}$$

$$x_1, x'_2, x''_2 \geq 0 \tag{30}$$

Sometimes we find it convenient to express a linear program in a more compact form. If we create an  $m * n$  matrix  $A = (a_{ij})$ , an  $m$ -dimensional vector  $b = (b_i)$ , an  $n$ -dimensional vector  $x = (x_j)$ , then we can rewrite a linear program as

Maximize

$$c^T x \tag{31}$$

subject to

$$Ax \leq b \tag{32}$$

$$x \geq 0. \tag{33}$$

Here  $c^T x$  is the inner product of two vectors,  $Ax$  is a matrix-vector product, and  $x \geq 0$  means that each entry of the vector  $x$  must be nonnegative.

## 4 Converting linear programs into slack form

To efficiently solve a linear program with the simplex algorithm, we prefer to express it in a form in which some of the constraints are equality constraints. More precisely, we shall convert it into a form in which the nonnegativity constraints are the only inequality constraints, and the remaining constraints are equalities. Let

$$\sum_{j=1}^n a_{ij} x_j \leq b_i$$

be an inequality constraint. We introduce a new variable  $s$  and rewrite the above inequality as the two constraints

$$s = b_i - \sum_{j=1}^n a_{ij} x_j,$$

$$s \geq 0$$

We call  $s$  a *slack variable* because it measures the slack, or difference, between the left-hand and right-hand sides of the equation. When converting from standard to slack form, we shall use  $x_{n+i}$  (instead of  $s$ ) to denote the slack variable associated with the  $i$ th inequality. The  $i$ th constraint is therefore

$$x_{n+i} = b_i - \sum_{j=1}^n a_{ij} x_j,$$

along with the nonnegativity constraint  $x_{n+i} \geq 0$ .

### 4.1 Example

Applying this constraint to each constraint of a linear program in standard form, we obtain a linear program in a different form. For example, for the linear program described in the standard form above, we introduce slack variables  $x_4$ ,  $x_5$ , and  $x_6$ , obtaining

Maximize

$$2x_1 - 3x_2 + 3x_3 \tag{34}$$

subject to

$$x_4 = 7 - x_1 - x_2 + x_3 \tag{35}$$

$$x_5 = -7 + x_1 + x_2 - x_3 \tag{36}$$

$$x_6 = 4 - x_1 + 2x_2 - 2x_3 \tag{37}$$

$$x_1, x_2, x_3, x_4, x_5, x_6 \geq 0 \tag{38}$$

In this linear program, all the constraints except for the nonnegativity constraints are equalities, and each variable is subject to a nonnegativity constraint.

## 5 Formulating problems as linear programs

### 5.1 Max flow

A flow is a real-valued function  $f : V * V \rightarrow \mathbf{R}$  that satisfies three properties: capacity constraints, skew symmetry, and flow conservation. A maximum flow is a flow that satisfies these constraints and maximizes the flow value, which is the total flow coming out of the source. A flow, therefore, satisfies linear constraints, and the value of a flow is a linear function. We can express the maximum-flow problem as a linear program:

Maximize

$$\sum_{v \in V} f(s, v) \quad (39)$$

subject to

$$f(u, v) \leq c(u, v) \text{ for each } u, v \in V, \quad (40)$$

$$f(u, v) = -f(v, u) \text{ for each } u, v \in V, \quad (41)$$

$$\sum_{v \in V} f(u, v) = 0 \text{ for each } u \in V - \{s, t\}. \quad (42)$$

### 5.2 Min-cost flow

Suppose that each edge  $(u, v)$  has, in addition to a capacity  $c(u, v)$ , a real-valued cost  $a(u, v)$ . If we send  $f(u, v)$  units of flow over edge  $(u, v)$ , we incur a cost of  $a(u, v)f(u, v)$ . We are also given a flow target  $d$ . We wish to send  $d$  units of flow from  $s$  to  $t$  in such a way that the total cost incurred by the flow is minimized. This problem is known as the **minimum-cost-flow problem**. The linear program looks similar to the one for the maximum-flow problem with the additional constraint that the value of the flow be exactly  $d$  units, and with the new objective function of minimizing the cost:

Minimize

$$\sum_{(u,v) \in E} a(u, v)f(u, v) \quad (43)$$

subject to

$$f(u, v) \leq c(u, v) \text{ for each } u, v \in V, \quad (44)$$

$$f(u, v) = -f(v, u) \text{ for each } u, v \in V, \quad (45)$$

$$\sum_{v \in V} f(u, v) = 0 \text{ for each } u \in V - \{s, t\}. \quad (46)$$

$$\sum_{v \in V} f(s, v) = d. \quad (47)$$

## 6 Conclusion

Thus we have introduced the basic concepts in linear programming, and how linear programs can be converted into standard and slack forms. Finally, we also saw how common real time problems can be formulated in terms of linear programs.