Chapter 10

Introduction to LP-duality

As stated in Chapter , LP-duality theory unifies a large part of the theory of approximation algorithms as we know it today. For a comprehensive introduction to linear programming we recommend [?]. In this chapter, we will review some concepts that we will use critically.

Linear programming is the problem of optimizing (i.e., minimizing or maximizing) a linear function subject to linear inequality constraints; the function being optimized is called the *objective function*. Perhaps the most interesting fact about this problem from our perspective is that it is well-characterized in the sense of Chapter 10. Let us illustrate this through a simple example:

$$
\begin{array}{ll}
\text{minimize} & 7x_1 + x_2 + 5x_3 \\
\text{subject to} & \begin{array}{rcrcrcl} x_1 & - & x_2 & + & 3x_3 & \geq & 10 \\ 5x_1 & + & 2x_2 & - & x_3 & \geq & 6 \end{array} \\
& x_1, x_2, x_3 \geq 0
\end{array}
$$

Notice that in this example all constraints are of the kind "$\geq$" and all variables are constrained to be non-negative. This is the standard form of a minimization linear program; a simple transformation enables one to write any minimization linear program in this manner. The reason for choosing this form will become clear shortly.

Any solution, i.e., a setting for the variables in this linear program, that satisfies all the constraints is said to be a *feasible solution*. Let $\boldsymbol{z}^*$ denote the optimum value of this linear program. Let us consider the question, "Is $\boldsymbol{z}^*$ at most α?" where α is a given rational number. For instance, let us ask whether $\boldsymbol{z}^* \leq 30$. A Yes certificate for this question is simply a feasible solution whose objective function value is at most 30. For example, $\boldsymbol{x} = (2, 1, 3)$ constitutes such a certificate since it satisfies the two constraints of the problem, and the objective function value for this solution is $7 \cdot 2 + 1 + 5 \cdot 3 = 30$. Thus, any Yes certificate to this question provides an upper bound on $\boldsymbol{z}^*$.

How do we provide a No certificate for such a question? In other words, how do we place a good lower bound on $\boldsymbol{z}^*$? In our example, one such bound is given by the first constraint: since the x_i's are restricted to be non-negative, term by term comparison of coefficients shows that $7x_1 + x_2 + 5x_3 \geq x_1 - x_2 + 3x_3$. Since the right hand side of the first constraint is 10, we get that the objective function is at least 10 for any feasible solution. A better lower bound can be obtained by taking the sum of the two constraints: for any feasible solution $\boldsymbol{x}$,

$$7x_1 + x_2 + 5x_3 \geq (x_1 - x_2 + 3x_3) + (5x_1 + 2x_2 - x_3) \geq 16.$$

The idea behind this process of placing a lower bound is that we are finding suitable non-negative multipliers for the constraints so that when we take their sum, the coefficient of each x_i

in the sum is dominated by the coefficient in the objective function. Now, the right hand side of this sum is a lower bound on z^* since any feasible solution has a non-negative setting for each x_i. Notice the importance of ensuring that the multipliers are non-negative: they do not reverse the direction of the constraint inequality.

Clearly, the rest of the game lies in choosing the multipliers in such a way that the right hand side of the sum is as large as possible. Interestingly enough, the problem of finding the best such lower bound can be formulated as a linear program:

$$\begin{array}{ll} \text{maximize} & 10y_1 + 6y_2 \\ \text{subject to} & \begin{array}{rcrcl} y_1 & + & 5y_2 & \leq & 7 \\ -y_1 & + & 2y_2 & \leq & 1 \\ 3y_1 & - & y_2 & \leq & 5 \end{array} \\ & y_1, y_2 \geq 0 \end{array}$$

Here y_1 and y_2 were chosen to be the non-negative multipliers for the first and the second constraint respectively. Let us call the first linear program the *primal program* and the second the *dual program.* There is a systematic way of obtaining the dual of any linear program; one is a minimization problem and the other is a maximization problem. Further, the dual of the dual is the primal program itself (see [?]). By construction, every feasible solution to the dual program gives a lower bound on the optimum value of the primal. Observe that the reverse also holds: every feasible solution to the primal program gives an upper bound on the optimal value of the dual. Therefore, if we can find feasible solutions for the dual and the primal with matching objective function values, then both solutions must be optimal. In our example, $\boldsymbol{x} = (7/4, 0, 11/4)$ and $\boldsymbol{y} = (2, 1)$ both achieve objective function values of 26, and so both are optimal solutions. The reader may wonder whether our example was ingeniously chosen for this to happen. Surprisingly enough, this is not an exception but the rule! This is the central theorem of linear programming: *the LP-Duality Theorem.*

In order to state this theorem formally, let us consider the following minimization problem, written in standard form, as the primal program; equivalently, we could have started with a maximization problem as the primal program.

$$\begin{array}{lll} \text{minimize} & \displaystyle\sum_{j=1}^{n} c_j x_j & \\ \text{subject to} & \displaystyle\sum_{j=1}^{n} a_{ij} x_j \geq b_i, & i = 1, \ldots, m \\ & x_j \geq 0, & j = 1, \ldots, n \end{array} \tag{10.1}$$

where a_{ij}, b_i, and c_j are given rational numbers.
Then the dual program is:

$$\begin{array}{lll} \text{maximize} & \displaystyle\sum_{i=1}^{m} b_i y_i & \\ \text{subject to} & \displaystyle\sum_{i=1}^{m} a_{ij} y_i \leq c_j, & j = 1, \ldots, n \\ & y_i \geq 0, & i = 1, \ldots, m \end{array} \tag{10.2}$$

Theorem 10.1 (LP-Duality Theorem) *The primal program has finite optimum if and only if its dual has finite optimum. Moreover, if $\boldsymbol{x}^* = (x_1^*, \ldots, x_n^*)$ and $\boldsymbol{y}^* = (y_1^*, \ldots, y_m^*)$ are optimal solutions for the primal and the dual program respectively, then*

$$\sum_{j=1}^{n} c_j x_j^* = \sum_{i=1}^{m} b_i y_i^*.$$

Notice that the LP-Duality Theorem is really a min-max relation, since one program is a minimization problem and the other is a maximization problem. This theorem shows that the linear programming problem is well-characterized: feasible solutions to the primal (dual) provide Yes (No) certificates to the question, "Is the optimum value less than or equal to α?" Thus, as a corollary of this theorem we get that linear programming is in $\mathbf{NP} \cap \text{co-}\mathbf{NP}$.

Going back to our example, by construction, any feasible solution to the dual program gives a lower bound on the optimal value of the primal. So, in fact it also gives a lower bound on the objective function value achieved by any feasible solution to the primal. This is the easy half of the LP-Duality Theorem, sometimes called the *Weak Duality Theorem*. We give a formal proof of this since some steps in the proof will lead to the next important fact. For a proof of the LP-Duality Theorem see [**?**]. The design of several exact algorithms have their basis in the LP-Duality Theorem. In contrast, in approximation algorithms, typically the Weak Duality Theorem suffices.

Theorem 10.2 (Weak Duality Theorem) *If $\boldsymbol{x} = (x_1, \ldots, x_n)$ is a feasible solution for the primal program and $\boldsymbol{y} = (y_1, \ldots, y_m)$ is a feasible solution for the dual, then*

$$\sum_{j=1}^{n} c_j x_j \geq \sum_{i=1}^{m} b_i y_i. \tag{10.3}$$

Proof : Since $\boldsymbol{y}$ is dual feasible and x_j's are non-negative,

$$\sum_{j=1}^{n} c_j x_j \geq \sum_{j=1}^{n} \left(\sum_{i=1}^{m} a_{ij} y_i \right) x_j. \tag{10.4}$$

Similarly, since $\boldsymbol{x}$ is primal feasible and y_i's are non-negative,

$$\sum_{i=1}^{m} \left(\sum_{j=1}^{n} a_{ij} x_j \right) y_i \geq \sum_{i=1}^{m} b_i y_i. \tag{10.5}$$

The theorem follows by observing that

$$\sum_{j=1}^{n} \left(\sum_{i=1}^{m} a_{ij} y_i \right) x_j = \sum_{i=1}^{m} \left(\sum_{j=1}^{n} a_{ij} x_j \right) y_i.$$

□

By the LP-Duality Theorem, $\boldsymbol{x}$ and $\boldsymbol{y}$ are both optimal solutions if and only if (10.3) holds with equality. Clearly, this happens if and only if both (10.4) and (10.5) hold with equality. Hence, we get the following result about the structure of optimal solutions:

Theorem 10.3 (Complementary Slackness Conditions) *Let $\boldsymbol{x}$ and $\boldsymbol{y}$ be primal and dual feasible solutions respectively. Then, $\boldsymbol{x}$ and $\boldsymbol{y}$ are both optimal if and only if all of the following conditions are satisfied:*

Primal complementary slackness conditions
For each $1 \leq j \leq n$: either $x_j = 0$ or $\sum_{i=1}^{m} a_{ij} y_i = c_j$; and

Dual complementary slackness conditions
For each $1 \leq i \leq m$: either $y_i = 0$ or $\sum_{j=1}^{n} a_{ij} x_j = b_i$.

The complementary slackness conditions play a vital role in the design of efficient algorithms, both exact and approximation; see Chapter 13 for details. For a better appreciation of their importance, we recommend that the reader study Edmonds' weighted matching algorithm (see, for example, [**?**]).

Min-max relations and the LP-Duality Theorem

In order to appreciate the role of LP-duality theory in approximation algorithms, it is important to first understand its role in exact algorithms. To do so, we will review some of these ideas in the context of the max-flow min-cut theorem; in particular, we will show how this and other min-max relations follow from the LP-Duality Theorem. Some of the ideas on cuts and flows developed here will also be used in the study of multicommodity flow in Chapters 14 17 and 18.

The problem of computing a maximum flow in a network is: Given a directed[1] graph, $G = (V, E)$ with two distinguished nodes, *source* s and *sink* t, and positive arc capacities, $c : E \rightarrow \mathbf{R}^+$, find the maximum amount of flow that can be sent from s to t subject to

1. *capacity constraint:* for each arc e, the flow sent through e is bounded by its capacity, and

2. *flow conservation:* at each node v, other than s and t, the total flow into v should equal the total flow out of v.

An *s-t cut* is defined by a partition of the nodes into two sets X and $\overline{X}$ so that $s \in X$ and $t \in \overline{X}$, and consists of the set of arcs going from X to $\overline{X}$. The *capacity* of this cut, $c(X, \overline{X})$, is defined to be the sum of capacities of these arcs. Because of the capacity constraints on flow, the capacity of any s-t cut is an upper bound on any feasible flow. So, if the capacity of an s-t cut, say $(X, \overline{X})$ equals the value of a feasible flow, then $(X, \overline{X})$ must be a minimum s-t cut and the flow must be a maximum flow in G. The max-flow min-cut theorem proves that it is always possible to find a flow and an s-t cut so that equality holds.

Let us formulate the maximum flow problem as a linear program. First, introduce a fictitious arc of infinite capacity from t to s, thus converting the flow to a circulation; the objective now is to maximize the flow on this arc, denoted by f_{ts}. The advantage of making this modification is that we can now require flow conservation at s and t as well. If f_{ij} denotes the amount of flow sent through arc $(i, j) \in E$, we can formulate the maximum flow problem as follows:

$$\begin{array}{lll} \text{maximize} & f_{ts} & \\ \text{subject to} & f_{ij} \leq c_{ij}, & (i, j) \in E \end{array}$$

[1] The maximum flow problem in undirected graphs reduces to that in directed graphs: replace each edge (u, v) by two directed edges, $(u \rightarrow v)$ and $(v \rightarrow u)$, each of the same capacity as (u, v).

$$\begin{aligned} \sum_{j:\ (j,i)\in E} f_{ji} - \sum_{j:\ (i,j)\in E} f_{ij} \le 0, &\qquad i \in V \\ f_{ij} \ge 0, &\qquad (i,j) \in E \end{aligned}$$

The second set of inequalities say that for each node i, the total flow into i is at most the total flow out of i. Notice that if this inequality holds at each node, then in fact it must be satisfied with equality at each node, thereby implying flow conservation at each node (this is so because a deficit in flow balance at one node implies a surplus at some other node). With this trick, we get a linear program in standard form.

To obtain the dual program we introduce variables d_{ij} and p_i corresponding to the two types of inequalities in the primal. We will view these variables as distance labels on arcs and potentials on nodes respectively. The dual program is:

$$\begin{array}{lll} \text{minimize} & \sum_{(i,j)\in E} c_{ij} d_{ij} & \\ \text{subject to} & d_{ij} - p_i + p_j \ge 0, & (i,j) \in E \\ & p_s - p_t \ge 1 & \\ & d_{ij} \ge 0, & (i,j) \in E \\ & p_i \ge 0, & i \in V \end{array}$$

For developing an intuitive understanding of the dual program, it will be best to first transform it into an integer program that seeks 0/1 solutions to the the variables:

$$\begin{array}{lll} \text{minimize} & \sum_{(i,j)\in E} c_{ij} d_{ij} & \\ \text{subject to} & d_{ij} - p_i + p_j \ge 0, & (i,j) \in E \\ & p_s - p_t \ge 1 & \\ & d_{ij} \in \{0,1\}, & (i,j) \in E \\ & p_i \in \{0,1\}, & i \in V \end{array}$$

Let $(\boldsymbol{d}^*, \boldsymbol{p}^*)$ be an optimal solution to this integer program. The only way to satisfy the inequality $p_s^* - p_t^* \ge 1$ with a 0/1 substitution is to set $p_s^* = 1$ and $p_t^* = 0$. So, this solution naturally defines an s-t cut $(X, \overline{X})$, where X is the set of potential 1 nodes, and $\overline{X}$ the set of potential 0 nodes. Consider an arc (i,j) with $i \in X$ and $j \in \overline{X}$. Since $p_i^* = 1$ and $p_j^* = 0$, by the first constraint, $d_{ij}^* \ge 1$. But since we have a 0/1 solution, $d_{ij}^* = 1$. The distance label for each of the remaining arcs can be set to either 0 or 1 without violating the first constraint; however, in order to minimize the objective function value, it must be set to 0. So, the objective function value is precisely the capacity of the cut $(X, \overline{X})$, and hence $(X, \overline{X})$ must be a minimum s-t cut.

Thus, this integer program is a formulation of the minimum s-t cut problem! What about the dual program? The dual program can be viewed as a relaxation of the integer program where the integrality constraint on the variables is dropped. This leads to the constraints $1 \ge d_{ij} \ge 0$ for $(i,j) \in E$ and $1 \ge p_i \ge 0$ for $i \in V$. Next, we notice that the upper bound constraints on the variables are redundant; their omission cannot give a better solution. Dropping these constraints gives the dual program in the form given above. We will say that this program is the *LP relaxation* of the integer program.

Consider an s-t cut C. Set C has the property that any path from s to t in G contains at least one edge of C. Using this observation, we can interpret any feasible solution to the dual program as a *fractional s-t cut*: the distance labels it assigns to arcs satisfy the property that on any path from s to t the distance labels add up to at least 1. To see this, consider an s-t path $(s = v_0, v_1, \ldots, v_k = t)$. Now, the sum of the potential differences on the end points of arcs on this path,

$$\sum_{i=0}^{k-1} (p_i - p_{i+1}) = p_s - p_t.$$

So, by the first constraint, the sum of the distance labels on the arcs must add up to at least $p_s - p_t$, which is ≥ 1. Let us define the *capacity* of this fractional s-t cut to be the dual objective function value achieved by it.

In principle, the best fractional s-t cut could have lower capacity than the best integral cut. Surprisingly enough, this does not happen. Consider the polyhedron defining the set of feasible solutions to the dual program. Let us call a feasible solution a *vertex solution* if it is a vertex of this polyhedron. From linear programming theory we know that for any objective function, i.e., assignment of capacities to the arcs of G, there is a vertex solution that is optimal (for this discussion let us assume that for the given objective function, an optimal solution exists). Now, it can be proven that each vertex solution is integral, with each coordinate being 0 or 1. (This follows from the fact that the constraint matrix of this program is totally unimodular[2]; see [**?**] for a proof.) Thus, the dual program always has an integral optimal solution.

By the LP-Duality Theorem maximum flow in G must equal capacity of a minimum fractional s-t cut. But since the latter equals the capacity of a minimum s-t cut, we get the max-flow min-cut theorem.

So, the max-flow min-cut theorem is a special case of the LP-duality theorem; it holds because the dual polyhedron has integral vertices. In fact, most min-max relations in combinatorial optimization hold for analogous reasons.

Finally, let us illustrate the usefulness of complementary slackness conditions by utilizing them to derive additional properties of optimal solutions to the flow and cut programs. Let $\boldsymbol{f}^*$ be an optimum solution to the primal LP (i.e., a maximum s-t flow). Also, let $(\boldsymbol{d}^*, \boldsymbol{p}^*)$ be an integral optimum solution to the dual LP, and let $(X, \overline{X})$ be the cut defined by $(\boldsymbol{d}^*, \boldsymbol{p}^*)$. Consider an arc (i, j) such that $i \in X$ and $j \in \overline{X}$. We have proven above that $d^*_{ij} = 1$. Since $d^*_{ij} \neq 0$, by the dual complementary slackness condition, $f^*_{ij} = c_{ij}$. Next, consider an arc (k, l) such that $k \in \overline{X}$ and $l \in X$. Since $p^*_k - p^*_l = -1$, and $d^*_{kl} \in \{0, 1\}$, the constraint $d^*_{kl} - p^*_k + p^*_l \geq 0$ must be satisfied as a strict inequality. So, by the primal complementary slackness condition, $f^*_{kl} = 0$. Thus we have proven that arcs going from X to $\overline{X}$ are saturated by $\boldsymbol{f}^*$, and the reverse arcs carry no flow. (Observe that it was not essential to invoke complementary slackness conditions to prove these facts; they also follow from the fact that flow across cut $(X, \overline{X})$ equals its capacity.)

Exercise 10.4 Show that the dual of the dual of a linear program is the program itself.

Exercise 10.5 Show that any minimization program can be transformed into an equivalent program in standard form, i.e., the form of LP (10.1).

Exercise 10.6 Change some of the constraints of the primal program (10.1) into equalities, i.e., so they are of the form

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

[2] A matrix A is said to be *totally unimodular* if the determinant of every square submatrix of A is $1, -1$ or 0.

Show that the dual of this program involves modifying program (10.2) so that the corresponding dual variables y_i are *unconstrained*, i.e., they are not constrained to be non-negative. Additionally, if some of the variables x_j in program (10.1) are unconstrained, then the corresponding constraints in the dual become equalities.

Exercise 10.7 In this exercise, you will derive von Neumann's Minimax Theorem in game theory from the LP-Duality Theorem. A finite two-person zero-sum game is specified by an $m \times n$ matrix $\boldsymbol{A}$ with real entries. In each round, the row player, R, selects a row, say i; simultaneously, the column player, C, selects a column, say j. The *payoff* to R at the end of this round is a_{ij}. Thus, $|a_{ij}|$ is the amount that C pays R (R pays C) if a_{ij} is positive (a_{ij} is negative); no money is exchanged if a_{ij} is zero. *Zero sum game* refers to the fact that the total amount of money possessed by R and C together is conserved.

The *strategy* of each player is specified by vector whose entries are non-negative and add up to one, giving the probabilities with which the player picks each row or column. Let R's strategy be given by m-dimensional vector $\boldsymbol{x}$, and C's strategy be given by n-dimensional vector $\boldsymbol{y}$. Then, the excected pay off to R in a round is $\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{y}$. The job of each player is to pick a strategy that *guarantees* maximum possible expected winnings (equivalently, minimum possible expected losses), regardless of the strategy chosen by the other player. If R chooses strategy $\boldsymbol{x}$, he can be sure of winning only $\min_{\boldsymbol{y}} \boldsymbol{x}^T\boldsymbol{A}\boldsymbol{y}$, where the minimum is taken over all possible strategies of C. So, the optimal choice for R is given by $\max_{\boldsymbol{x}} \min_{\boldsymbol{y}} \boldsymbol{x}^T\boldsymbol{A}\boldsymbol{y}$. Similarly, C will minimize her losses by choosing the strategy given by $\min_{\boldsymbol{y}} \max_{\boldsymbol{x}} \boldsymbol{x}^T\boldsymbol{A}\boldsymbol{y}$. The Minimax Theorem states that for every matrix $\boldsymbol{A}$, $\max_{\boldsymbol{x}} \min_{\boldsymbol{y}} \boldsymbol{x}^T\boldsymbol{A}\boldsymbol{y} = \min_{\boldsymbol{y}} \max_{\boldsymbol{x}} \boldsymbol{x}^T\boldsymbol{A}\boldsymbol{y}$.

Let us say that a strategy is *pure* if it picks a single row or column, i.e., the vector corresponding to it consists of one 1 and the rest 0's. A key observation is that for any strategy $\boldsymbol{x}$ of R, $\min_{\boldsymbol{y}} \boldsymbol{x}^T\boldsymbol{A}\boldsymbol{y}$ is attained for a pure strategy of C: Suppose the minimum is attained for strategy $\boldsymbol{y}$. Consider the pure strategy corresponding to any non-zero component of $\boldsymbol{y}$. The fact that the components of $\boldsymbol{y}$ are non-negative and add up to one leads to an easy proof that this pure strategy attains the same minimum. So, R's optimum strategy is given by $\max_{\boldsymbol{x}} \min_j \sum_{i=1}^m a_{ij}x_i$. The second critical observation is that the problem of computing R's optimal strategy can be expressed as a linear program:

$$
\begin{array}{lll}
\text{maximize} & z & \\
\text{subject to} & z - \displaystyle\sum_{i=1}^{m} a_{ij}x_i \leq 0, & j = 1, \ldots, n \\
& \displaystyle\sum_{i=1}^{m} x_i = 1 & \\
& x_i \geq 0, & i = 1, \ldots, m
\end{array}
$$

Find the dual of this LP, and show that it computes the optimal strategy for C. (Use the fact that for any strategy $\boldsymbol{y}$ of C, $\max_{\boldsymbol{x}} \boldsymbol{x}^T\boldsymbol{A}\boldsymbol{y}$ is attained for a pure strategy of R.) Hence, prove the Minimax Theorem using the LP-Duality Theorem.

Chapter 13

The primal-dual schema

LP-duality theory not only provides a method of lower bounding the cost of the optimal solution for several **NP**-hard problems, but also provides a general schema for obtaining the approximation algorithm itself: the primal-dual schema. In this chapter, we will first present the central ideas behind this schema, and then illustrate them in a simple setting by again obtaining an f factor algorithm for the minimum set cover problem, where f is the frequency of the most frequent element. An algorithm achieving this factor, using rounding, was presented in Chapter 11.

The primal-dual schema has its origins in the design of exact algorithms. In that setting, this schema yielded the most efficient known algorithms to some of the cornerstone problems in **P**, including matching, network flow and shortest paths. These problems have the property that their LP-relaxations have integral optimal solutions. By Theorem 10.3 we know that optimal solutions to linear programs are characterized by fact that they satisfy all the complementary slackness conditions. In fact, the primal-dual schema is driven by these conditions: starting with initial feasible solutions to the primal and dual programs, it iteratively satisfies more and more complementary slackness conditions; when they are all satisfied, both solutions must be optimal. During the iterations, the primal is always modified integrally, so that eventually we get an integral optimal solution.

In the case of **NP**-hard problems, we cannot be looking for an optimal solution to the LP-relaxation since, in general, none of these solutions may be integral. Does this rule out a complementary slackness condition driven approach? Interestingly enough, the answer is "No". It turns out that the algorithm can be driven by a suitable relaxation of these conditions!

Overview of the schema

Let us consider the following primal program, written in standard form:

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

where a_{ij}, b_i, and c_j are specified in the input. The dual program is:

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

All known approximation algorithms using the primal-dual schema run by ensuring the primal complementary slackness conditions and relaxing the dual conditions. They find primal and dual feasible solutions satisfying:

Primal complementary slackness conditions
For each $1 \leq j \leq n$: either $x_j = 0$ or $\sum_{i=1}^{m} a_{ij} y_i = c_j$; and

Relaxed dual complementary slackness conditions
For each $1 \leq i \leq m$: either $y_i = 0$ or $\sum_{j=1}^{n} a_{ij} x_j \leq \alpha \cdot b_i$,

where $\alpha > 1$ is a constant; if α were set to 1, we would get the usual condition.

Proposition 13.1 *If $\boldsymbol{x}$ and $\boldsymbol{y}$ are primal and dual feasible solutions satisfying the conditions stated above then*

$$\sum_{j=1}^{n} c_j x_j \leq \alpha \cdot \sum_{i=1}^{m} b_i y_i.$$

Proof :

$$\sum_{j=1}^{n} c_j x_j = \sum_{j=1}^{n} \left(\sum_{i=1}^{m} a_{ij} y_i \right) x_j = \sum_{i=1}^{m} \left(\sum_{j=1}^{n} a_{ij} x_j \right) y_i \leq \alpha \sum_{i=1}^{m} b_i y_i. \tag{13.1}$$

The first equality is obtained by applying the primal conditions, and the inequality follows by applying the relaxed dual conditions. The second equality follows by simply changing the order of summation. □

The algorithm starts with a primal infeasible solution and a dual feasible solution; these are usually the trivial solutions $\boldsymbol{x} = \mathbf{0}$ and $\boldsymbol{y} = \mathbf{0}$. It iteratively improves the feasibility of the primal solution, and the optimality of the dual solution, ensuring that in the end a primal feasible solution is obtained and all conditions stated above are satisfied. The primal solution is always extended integrally, thus ensuring that the final solution is integral. The improvements to the primal and the dual go hand-in-hand: the current primal solution is used to determine the improvement to the dual, and vice versa. Finally, the cost of the dual solution is used as a lower bound on OPT, and by Proposition 13.1, the approximation guarantee of the algorithm is α.

Despite being so general, this schema leaves sufficient scope for exploiting the special combinatorial structure of the specific problem: in designing the solution improving algorithms and in comparing the final solutions, thereby yielding very good approximation guarantees. Since numerous **NP**-hard problems can be expressed as integer programming problems, the primal-dual schema should find even greater applicability in the setting of approximation algorithms than exact algorithms.

Primal-dual schema applied to set cover

Let us obtain a factor f algorithm for the set cover problem using the primal-dual schema. A statement of this problem appears in Chapter 11, its LP-relaxation is given in (11.2) and dual program is given in (12.1). As shown in Chapter 11, the weighted vertex cover problem can be stated as a set cover problem with $f = 2$.

Let us start by stating the primal complementary slackness conditions and relaxing the dual conditions appropriately.

Primal conditions: These can be written as:

$$\forall S \in \mathcal{S} : x_S \neq 0 \Rightarrow \sum_{e:\ e \in S} y_e = c(S).$$

Set S will said to be *tight* if $\sum_{e:\ e\in S} y_e = c(S)$. Since we will increment the primal variables integrally, we can state the conditions as: *Pick only tight sets in the cover.*
Clearly, in order to maintain dual feasiblity, we are not allowed to overpack any set.

Dual conditions: The dual conditions will be relaxed with $\alpha = f$.

$$\forall e : y_e \neq 0 \Rightarrow \sum_{S:\ e \in S} x_S \leq f$$

Since we will find a 0/1 solution for $\boldsymbol{x}$, these conditions are equivalently to:
Cover each element with non-zero dual at most f times.
Since each element is in at most f sets, this condition is trivially satisfied for all elements.

The two sets of conditions naturally suggest the following algorithm:

Algorithm 13.2 (Set cover – factor f)

1. **Initialization:** $\boldsymbol{x} \leftarrow \mathbf{0}$; $\boldsymbol{y} \leftarrow \mathbf{0}$
2. Until all elements are covered, do:

   Pick an uncovered element, say e, and raise y_e until some set goes tight.

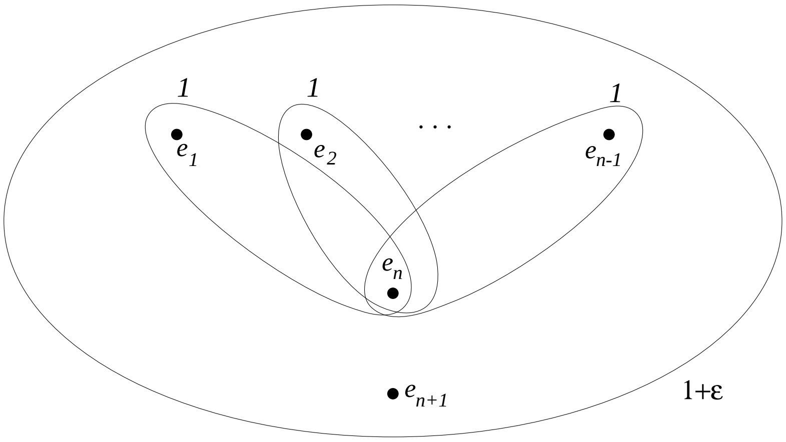
   Pick all tight sets in the cover and update $\boldsymbol{x}$.

   Declare all the elements occurring in these sets as "covered".
3. Output the set cover $\boldsymbol{x}$.

Theorem 13.3 *Algorithm 13.2 achieves an approximation factor of f.*

Proof : Clearly there will be no uncovered elements and no overpacked sets at the end of the algorithm. So, the primal and dual solutions will both be feasible. Since they satisfy the relaxed complementary slackness conditions with $\alpha = f$, by Proposition refprimal-dual.prop-relax the approximation factor is f. □

Example 13.4 A tight example for this algorithm is provided by the following set system:



Here, $\mathcal{S}$ consists of $n-1$ sets of cost 1, $\{e_1, e_n\}, \ldots, \{e_{n-1}, e_n\}$, and one set of cost $1+\epsilon$, $\{e_1, \ldots, e_{n+1}\}$, for a small $\epsilon > 0$. Since e_n appears in all n sets, this set system has $f = n$.

Suppose the algorithm raises y_{e_n} in the first iteration. When y_{e_n} is raised to 1, all sets $\{e_i, e_n\}$, $i = 1, \ldots, n-1$ go tight. They are all picked in the cover, thus covering the elements $e_1, \ldots, e_n$. In the second iteration, $y_{e_{n+1}}$ is raised to ϵ and the set $\{e_1, \ldots, e_{n+1}\}$ goes tight. The resulting set cover has a cost of $n + \epsilon$, whereas the optimum cover has cost $1 + \epsilon$. $\square$