LP-BASED ALGORITHMS

USE OF LP-DUALITY IN THE DESIGN AND ANALYSIS OF APPROXIMATION ALGORITHMS

Linear Programming Key Concepts

Linear programming:

The problem of optimizing (minimizing or maximizing) a linear function (objective function) subject to linear inequality constraints

Example of a linear programming problem (minimization problem):

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

Standard form of a minimization linear program:

- All constraints are of the kind $"\geq"$
- All variables are constrained to be nonnegative

All linear programming problems can be converted to standard form.

Linear Programming Key Concepts

- Any setting for the variables that satisfies all the constraints of the corresponding linear programming problem is said to be a feasible solution.

- A linear programming problem is said to be feasible if the constraint set is not empty. Otherwise is said to be infeasible.

- A feasible linear programming problem is said to be unbounded if the objective function can assume arbitrarily large positive or negative values at feasible solutions; otherwise is said to be bounded.

Possibilities for a linear program: *bounded feasible, unbounded feasible, infeasible*

Certificates for an LP Decision Problem

Let z^* denote the optimum value of the following linear program:

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

"*Is z^* **at most** a ?*"

- a YES certificate for this question : a feasible solution whose objective function value is at most a → this problem is in NP
- Any YES certificate to this question provides an upper bound on z^*
- Can we provide a **NO certificate** for this question so that this problem is in $NP \cap co-NP$? → Yes. The problem is well-characterized

Placing Lower Bounds on the Objective Function Optimal Value

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

- By the first constraint since:

$$7x_1 + x_2 + 5x_3 \geq x_1 - x_2 + 3x_3 \geq 10$$

- A better lower bound can be obtained by taking the sum of the two constraints :

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

- ***The Idea behind the Process of placing lower bounds*** :find suitable nonnegative multipliers for the constraints so that in their sum , the coefficient of each x_i is dominated by the correspondent coefficient in the objective function. The right hand side of the sum is a lower bound on z^*
- *The coefficients are chosen so that the lower bound that is obtained is large as possible.*

LP-Duality

- The problem of finding the coefficients that give the best (highest) lower bound can be formulated as a linear program:

primal program

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

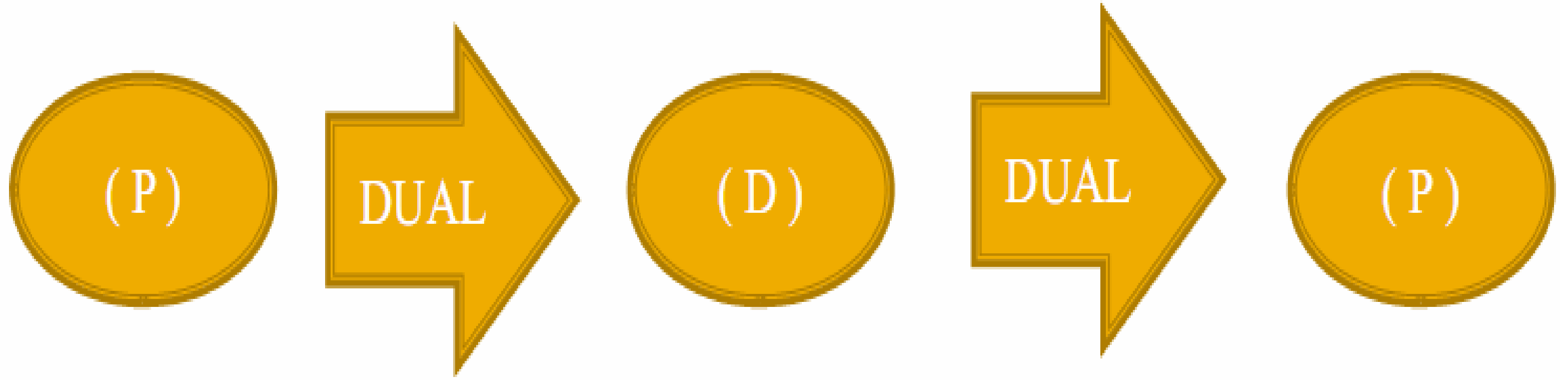
dual program

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

The original problem is called the Primal Problem and the other is called the Dual Problem

LP-Duality

- Associated with every linear program there is another program called a "dual" program
- There is a systematic way of obtaining the dual of every linear program
- If the primal program is a minimization program then the dual program is a maximization program
- The dual of the dual is the primal program itself



Obtaining The Dual of a Linear Program

primal program

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

subject to $\sum_{j=1}^{n} a_{ij}\ x_j \geq b_i, \quad i = 1, \ldots, m$

$x_j \geq 0, \qquad j = 1, \ldots, n$

dual program

maximize $\sum_{i=1}^{m} b_i y_i$

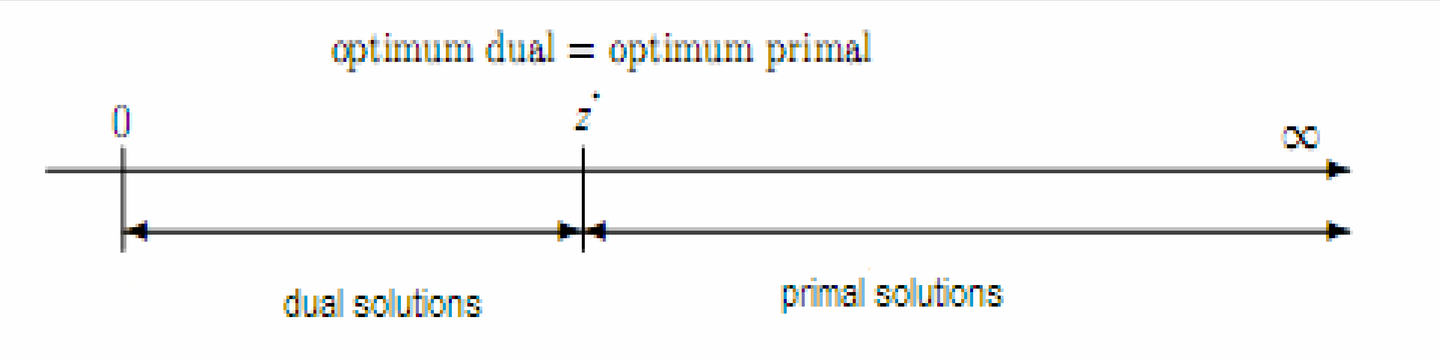
subject to $\sum_{i=1}^{m} a_{ij}\ y_i \leq c_j, \quad j = 1, \ldots, n$

$y_i \geq 0, \qquad i = 1, \ldots, m$

LP Duality Theorem

- By construction, every feasible solution to the dual program gives a lower bound on the optimum value of the primal
- Every feasible solution to the primal program gives an upper bound on the optimal value of the dual

if we can find solutions for the primal and the dual program with **matching objective function values** then both solutions must be optimal



LP Duality Theorem

LP-duality theorem :

The primal program has finite optimum iff its dual has finite optimum. Moreover , if $x^* = (x_1^*, \ldots, x_n^*)$ and $y^* = (y_1^*, \ldots, y_m^*)$ are optimal solutions for the primal and the dual programs respectively, then

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

- The LP-duality theorem is a min-max relation since one program is a minimization problem and the other is a maximization problem.
- A corollary of this theorem is that LP is well-characterized

Weak Duality Theorem

Weak duality theorem:

If $x^* = (x_1^*, \ldots, x_n^*)$ and $y^* = (y_1^*, \ldots, y_m^*)$ are feasible solutions for the primal and the dual program respectively then:

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

Proof:

Since y is dual feasible and x_j are nonnegative : $\sum_{j=1}^{n} c_j x_j \geq \sum_{j=1}^{n} \left(\sum_{i=1}^{m} a_{ij} y_i \right) x_j$

Since x is primal feasible and y_i are nonnegative : $\sum_{i=1}^{m} \left(\sum_{j=1}^{n} a_{ij} x_i \right) y_j \geq \sum_{i=1}^{m} b_i y_i$

The theorem follows by observing that:

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

Complementary Slackness Conditions

Let x and y be primal and dual solutions, respectively. Then x and y are both optimal iff all of the following conditions are satisfied:

- Primal complementary slackness conditions:

For each $1 \le j \le n$ either $x_j = 0$ or $\sum_{i=1}^{m} a_{ij} y_i = c_j$;

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

- Dual complementary slackness conditions:

For each $1 \le j \le m$ either $y_i = 0$ or $\sum_{j=1}^{n} a_{ij} x_j = b_i$;

(So that $\sum_{i=1}^{m} \left(\sum_{j=1}^{n} a_{ij} x_i \right) y_j = \sum_{i=1}^{m} b_i y_i$)

MAXFLOW Problem

The MAXFLOW problem:

Given a directed graph $G=(V,E)$ with two distinguished nodes source s and sink t and positive capacities $c : E \to \mathbb{R}^+$, find the maximum amount of flow that can be sent from s to t subject to:

- Capacity constraint: for each arc e, the flow sent through e is bounded by its capacity
- Flow conservation: for each node u other that s and t the total flow into u should equal the total flow out of u

The MAXFLOW Problem as a Linear Program

- Introduction of a fictitious arc of infinite capacity from t to s ⇒ conversion of the flow to a circulation ⇒ we can require flow conservation at s and t as well
- The objective is to maximize the flow on arc from t to s.
- We formulate the maximum flow problem as follows:

$$\begin{aligned} \text{maximize} \quad & f_{ts} & & \\ \text{subject to} \quad & f_{ij} \le c_{ij}, & (i,j) \in E \quad & \text{capacity constraints} \\ & \sum_{j:(j,i)\in E} f_{ji} - \sum_{j:(i,j)\in E} f_{ij} \le 0, & i \in V \quad & \text{flow conservation constraints} \\ & f_{ij} \ge 0, & (i,j) \in E \quad & \end{aligned}$$

f_{ij} denotes the amount of flow sent through arc $(i, j) \in E$

- <u>The trick to get the MAXFLOW problem formulation as a linear program in standard form</u>:

  If the second inequality holds at each node then in fact it must be satisfied with equality at each node

LP-Duality Theory and max-flow min-cut Theorem

- ***S-t cut***: defined by a partition of nodes into two sets X and $\overline{X}$ so that $s \in X$ and $t \in \overline{X}$. It consists of the set of arcs going from X to $\overline{X}$
- The ***capacity of a s-t cut***, denoted $c(X, \overline{X})$ is defined to be the sum of capacities of the arcs in the cut
- *The capacity of any s-t cut is an upper bound on any feasible flow*
- If the capacity of an s-t cut, say $(X, \overline{X})$ equals the value of a feasible flow, then (X, X) must be a minimum s-t cut and the flow must be a maximum flow in the graph.

The max-flow min-cut theorem proves that it is always possible to find flow and an s-t cut so that equality holds.

The Dual of MAXFLOW Problem

- Introduction of the variables

d_{ij} :distance labels on arcs

p_i :potentials on nodes

The dual program:

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

The dual program as integer: program

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

The Dual of MAXFLOW Problem

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

$\Rightarrow$ $p_s^* - p_t^* \geq 1 \Rightarrow p_s^* = 1, p_t^* = 0$

- This solution defines a cut $(X, \overline{X})$ where X is the set of nodes with potential 1 and $\overline{X}$ is the set of nodes with potential 0.
- Consider an arc (i, j) with $i \in X$ and $j \in \overline{X}$ $\Rightarrow$ $p_i^* = 1, p_j^* = 0 \Rightarrow d_{ij}^* \geq 1 \Rightarrow d_{ij}^* = 1$
- The distance label for each of the remaining arcs can be either 0 or 1 without violating the first constraint $\Rightarrow$ set to 0 in order to minimize the objective function
- the objective function value must be equal to the capacity of the cut $(X, \overline{X})$ and $(X, \overline{X})$ must be a minimum cut $\Rightarrow$ the integer program is a formulation of the min-cut problem!

Relaxation of a Linear Program

- What about the following dual program?

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

- The former program can be viewed as a relaxation of the integer program where:

$$d_{ij} \in \{0,1\} \rightarrow 1 \geq d_{ij} \geq 0, \quad (i,j) \in E$$

$$p_i \in \{0,1\} \rightarrow 1 \geq p_i \geq 0, \quad i \in V$$

- The constraints $1 \geq p_i$ and $1 \geq d_{ij}$ are redundant

Fractional s-t cuts

- Consider an s-t cut C in G.
- Any path from s to t in G contains at least one edge of C.
- Thus any feasible solution to the dual problem can be interpreted 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.
- Consider an s-t path: $(s = u_0, u_1, \ldots, u_k = t)$
- Sum the potential differences on the endpoints of arcs on this path:

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

- The sum of the distance labels on the arcs must add up to $p_s - p_t \geq 1$

We define the capacity of this fractional s-t cut to be the dual objective function value achieved by it

Max-flow min-cut Theorem

- the best fractional s-t cut could have lower capacity than the best integral cut? NO
- Consider the polyhedron defining the set of feasible solutions to the dual program
- A feasible solution is an ***extreme point solution*** if it cannot be expressed as a convex combination of two feasible solutions
- For any objective function ***there is an extreme point solution that is optimal***
- It can be proven that ***each extreme point solution of the polyhedron for the integer dual program is integral***
- 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.
- A minimum fractional s-t cut equals the capacity of an s-t cut

Max-flow min cut theorem

Most min-max relations arise from LP-relaxations that always have integral optimal solutions

Two Fundamental Algorithm Design Techniques

- **Why linear programming is so useful in approximation algorithms?**

Many combinatorial problems can be stated as integer programs. Once this is done, the linear relaxation of this program provides a natural way of lower bounding the cost of the optimal solution.

A feasible solution to the relaxed problem can be thought as a fractional solution to the original problem.

In the case of an NP-hard problem we cannot expect the polyhedron defining the set of feasible solutions to have integer vertices ➡ we look for a near optimal integral solution

There are two basic techniques for obtaining approximation algorithms using linear programming:

- **LP-rounding**: Solve the linear program ➡ convert the fractional solution obtained into an integral solution. The approximation guarantee is established by *comparing the cost of the integral and fractional solutions.*
- **Primal-dual schema**: an integral solution to the primal program and a feasible solution to the dual are constructed iteratively. The approximation guarantee is established by *comparing the cost of the two solutions.*

- ***Is the primal-dual schema inferior to LP-rounding?***

Integrality Gap of an LP-relaxation

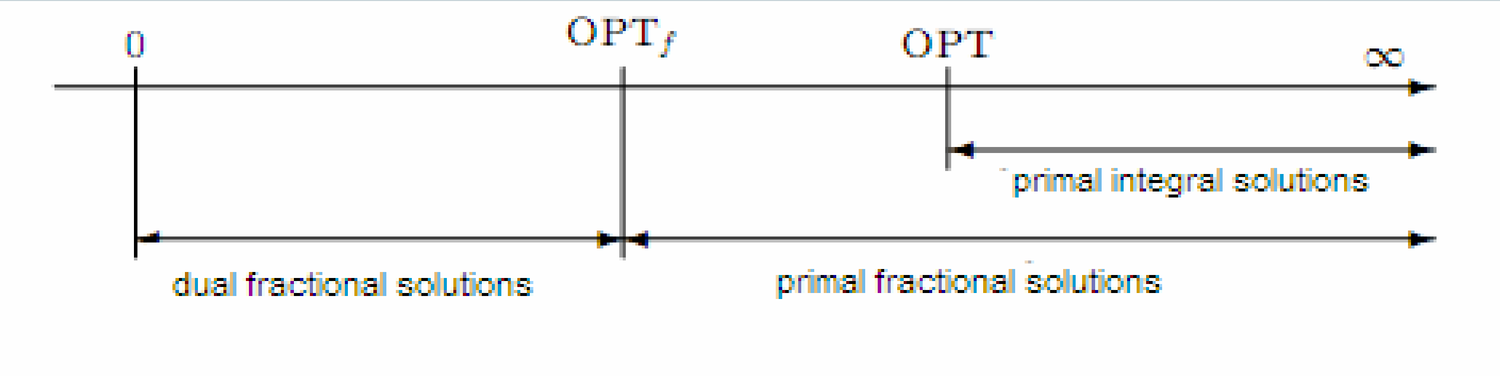
Integrality gap of an LP- relaxation:

Given an LP-relaxation for a minimization problem Π, let OPT_f denote the cost of an optimal fractional solution to instance I .

Define the integrality gap to be:

$$\sup_{I} \frac{OPT(I)}{OPT_f(i)}$$

- ***The integrality gap of the min-max relations*** which arise from LP-relaxations that ***always have integral solutions is 1***. We call such an LP –relaxation an ***exact*** LP-relaxation.



A Comparison of the two Techniques

- If the cost of the solution found by the algorithm is compared directly to the cost of an optimal fractional solution (or a feasible dual solution), ***the best approximation factor we can hope to prove is the integrality gap of the relaxation.***
- Both techniques have been successful in yielding algorithms having guarantees ***essentially equal to the integrality gap of the relaxation***.
- Main difference between the techniques ➡ running time

LP-Rounding:

- needs to find an optimal solution to the linear programming relaxation
- polynomial time if the relaxation has polynomially many constraints.
- The running time is high

Primal-dual schema:

- better running times
- it provides only an abroad outline of the algorithm
- it leaves enough space to exploit the special combinatorial structure of individual problems
- a combinatorial algorithm is more malleable than an algorithm that requires an LP-solver

Dual fitting-based analysis

- the method of dual fitting helps analyze combinatorial algorithms using LP-duality theory
- We will present an analysis of the natural greedy algorithm for the set cover problem
- The power of this approach will become apparent when we show the ease with which it extends to solving several generalizations of the set cover problem
- Description of the dual-fitting method:

One shows that the primal integral solution found by the algorithm is fully paid by the dual computed.

By *fully paid for* we mean that the objective function value of the primal solution found is at most the objective function value of the dual computed, however the dual is infeasible

Main step in the analysis: divide the dual by a suitable factor and show that the shrunk dual is feasible, i.e. it fits into the given instance

The shrunk dual is then a lower bound on OPT, and the factor is the approximation guarantee of the algorithm

SET COVER formulation as an integer program

- Assign a variable x_s for each set $s \in S$ which is allowed 0/1 values.
- The constraint is that for each element $e \in U$ we want that at least one of the sets containing it to be picked.

$$
\begin{aligned}
\text{minimize} \quad & \sum_{s \in S} c(s) x_s \\
\text{subject to} \quad & \sum_{s: e \in s} x_s \geq 1, \quad e \in U \\
& x_s \in \{0,1\} \quad s \in S
\end{aligned}
\qquad \xrightarrow[(1 \geq x_s \geq 0)]{\text{LP-relaxation}} \qquad
\begin{aligned}
\text{minimize} \quad & \sum_{s \in S} c(s) x_s \\
\text{subject to} \quad & \sum_{s: e \in s} x_s \geq 1, \quad e \in U \\
& x_s \geq 0 \quad s \in S
\end{aligned}
$$

The upper bound on x_s is redundant because the algorithm does not select more than once the same set. Thus, by omitting $1 \geq x_s$ we don't lose any better solution and we get the program in a standard form

Obtaining the dual of the SET COVER LP-relaxed program

$$\text{minimize} \quad \sum_{s \in S} c(s) x_s$$

$$\text{subject to} \quad \sum_{s: e \in s} x_s \geq 1, \quad e \in U$$

$$x_s \geq 0 \quad s \in S$$

Introduce variables y_e corresponding to each element $e \in U$

$$\text{maximize} \quad \sum_{e \in U} y_e$$

$$\text{subject to} \quad \sum_{e: e \in s} y_e \leq c(s), \quad s \in S$$

$$y_e \geq 0, \qquad e \in U$$

- An intuitive way of thinking about the dual of SET COVER is that it is *packing stuff into elements* trying to maximize the total amount packed
- The constraint is that no set is overpacked
- *A set is said to be overpacked if the total amount packed into its elements exceeds the cost of the set.*
- Whenever the coefficients in the constraint matrix, objective function, and right-hand side are all nonnegative , the minimization LP is called a *covering LP* and the maximization LP is called a *packing LP*.

Simple example

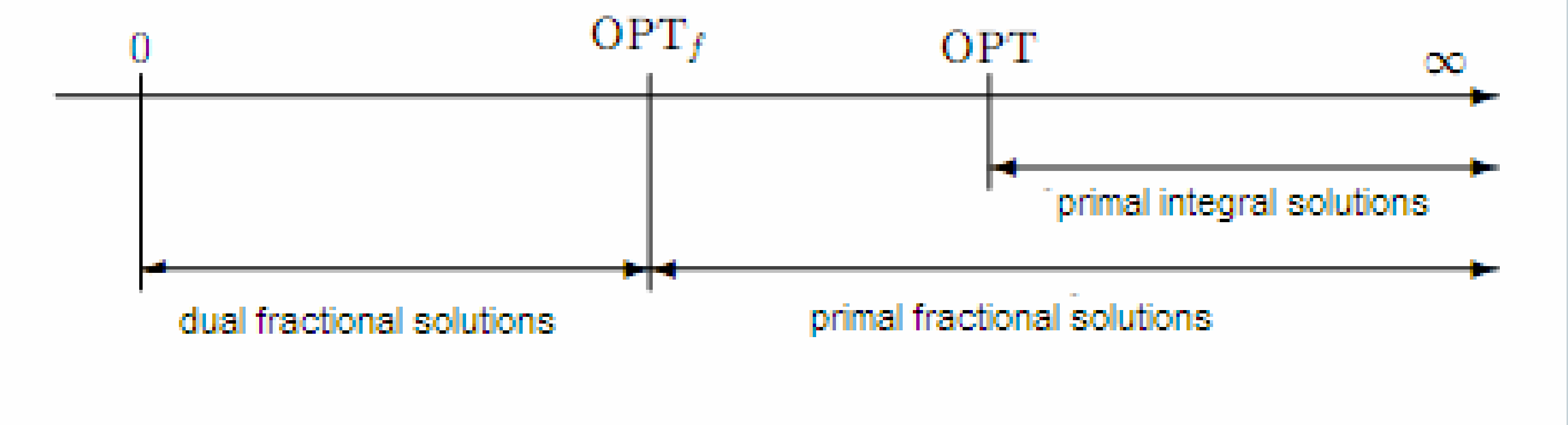
- *A fractional set cover may be cheaper than the optimal integral set cover*

Example:

Let $U = \{e, f, g\}$ and the specific sets be $S_1 = \{e, f\}$, $S_2 = \{f, g\}$ and $S_3 = \{g, e\}$ each of unit cost.

Integral cover: picking two of the sets for a cost of 2

Fractional cover: picking each set to the extend of ½ gives a cost of 3/2.



OPT_f :the cost of an optimal fractional set

OPT :the cost of an optimal integral set cover

$\Rightarrow$ $OPT_f \leq OPT$

The cost of any feasible solution to the dual program is a lower bound on OPT_f*, and hence also on* OPT .

Greedy SET COVER algorithm

Algorithm 2.2 (Greedy set cover algorithm)

1. $C \leftarrow \emptyset$
2. while $C \neq U$ do

   Find the most cost-effective set in the current iteration, say S.

   Let $\alpha = \frac{\text{cost}(S)}{|S \cap C|}$, i.e., the cost-effectiveness of S.

   Pick S, and for each $e \in S - C$, $\text{price}(e) \leftarrow \alpha$.
3. Output the picked sets.

- The algorithm 2.2 defines dual variables price(e) for each element e
- The cover picked by the algorithm is fully paid for by its dual solution
- In general this dual solution is not feasible
- If this dual is shrunk by a factor H_n no set is overpacked
- For each element e define : $y_e = \frac{\text{price}(e)}{H_n}$
- Algorithm 2.2 uses the dual feasible solution y as the lower bound on OPT

y is a feasible solution for the dual program

Lemma: "the vector **y** is a feasible solution for the dual program"

Proof: (*need to show that no set is overpacked by the solution y*)

- Consider a set $s \in S$ consisting of k elements
- Number the elements in the order in which they are covered by the algorithm, say $e_1, \ldots, e_k$
- Consider the iteration at which the algorithm covers the element e_i :

  s contains at least (k-i+1) uncovered elements
- In this iteration s can cover e_i at an average cost of at most $c(s)/(k-i+1)$
- The algorithm chooses the most effective set in this iteration

$$price(e_i) \leq c(s)/(k-i+1) \quad \Longrightarrow \quad y_{ei} \leq \frac{1}{H_n} \cdot \frac{c(s)}{k-i+1}$$

- Summing over all elements in S:

$$\sum_{i=1}^{k} y_{ei} \leq \frac{c(s)}{H_n} \cdot \left(\frac{1}{k} + \frac{1}{k-1} + \ldots + \frac{1}{1}\right) = \frac{H_k}{H_n} \cdot c(s) \leq c(s)$$

<u>Theorem</u>: "the approximation guarantee of the greedy set cover algorithm is H_n "

Proof:

The cost of the set cover picked is

$$\sum_{e \in U} \mathrm{price}(e) = H_n \left(\sum_{e \in U} y_e \right) \leq H_n \cdot \mathrm{OPT}_f \leq H_n \cdot \mathrm{OPT}$$

Generalizations of SET COVER

- **Set multicover**: each element e needs to be covered a specific integer number of times.

  Objective: cover all the elements up to their coverage requirements at minimum cost. The cost of picking a set S, k times is k*c(S).

- **Multiset cover**: a collection of multisets of U is given, which contain a specific number of copies of each element. Let $M(S,e)$ denote the multiplicity of element e in set S. the instance satisfies the condition that the multiplicity of an element in a set is at most its coverage requirement

- **Covering integer programs**: programs of the form

$$\begin{aligned} &\text{minimize} \quad c \cdot x \\ &\text{subject to} \quad Ax \geq b \end{aligned}$$

  where all entries in A, b ,c are nonnegative and x is required to be nonnegative and integral.

Constrained SET MULTICOVER: dual fitting analysis

- Constrained SET MULTICOVER: SET MULTICOVER with the additional constraint that each set can be picked at most once
- Let $r_e \in \mathbb{Z}^+$ be the coverage requirement for each element $e \in U$

$$\begin{aligned} \text{minimize} \quad & \sum_{s \in S} c(s) x_s \\ \text{subject to} \quad & \sum_{s: e \in s} x_s \geq r_e, \quad e \in U \\ & x_s \in \{0,1\} \quad s \in S \end{aligned}$$

$$\begin{aligned} \text{minimize} \quad & \sum_{s \in S} c(s) x_s \\ \text{subject to} \quad & \sum_{s: e \in s} x_s \geq r_e, \quad e \in U \\ & -x_s \geq -1, \quad s \in S \\ & x_s \geq 0, \quad s \in S \end{aligned}$$

In the LP-relaxation problem the constraints $1 \geq x_s$ *are no longer redundant so there are negative numbers in the constraint matrix and the problem is not a linear covering problem*

Constrained SET MULTICOVER: dual fitting analysis

- the additional constraints in the primal relaxed program lead to new variables for the dual z_s
- The dual has also negative numbers in the constraint matrix and is not therefore a packing problem
- A set S can be overpacked with the y_e 's.

$$
\begin{aligned}
\text{maximize} \quad & \sum_{e \in U} r_e y_e - \sum_{s \in S} z_s \\
\text{subject to} \quad & \left(\sum_{e : e \in s} y_e \right) - z_s \le c(s), && s \in S \\
& y_e \ge 0, && e \in U \\
& z_s \ge 0, && s \in S
\end{aligned}
$$

Constrained SET MULTICOVER: dual fitting analysis

Description of the greedy algorithm for SET MULTICOVER

- An element e is alive if it occurs in fewer than r_e of the picked sets.
- The cost-effectiveness of a set is defined to be the average cost at which it covers alive elements
- The algorithm is greedy and at each iteration it picks from amongst the currently unpicked sets the most cost-effective set.
- The algorithm halts when there are no more alive elements
- When a set is picked, its cost is distributed equally among the alive elements it covers as follows: If s covers element e for the $j th$ time we set $\text{price}(e, j)$ to the current cost-effectiveness of s.
- For each element : $\text{price}(e,1) \le \text{price}(e,2) \le \ldots \le \text{price}(e,r_e)$

At the end of the algorithm the dual variables are set as follows:

For each $e \in U$: $a_e = \text{price}(e, r_e)$

and for each $s \in S$ that is picked by the algorithm :

$$\beta_s = \sum_{e \text{ covered by s}} (\text{price}(e, r_e) - \text{price}(e, j_e))$$

Constrained SET MULTICOVER: dual fitting analysis

Lemma:

"The multicover picked by the algorithm is fully paid by the dual solution (α,β)"

- The cost of the sets picked by the algorithm is distributed among the covered elements
- The total cost of the multicover produced by the algorithm: $\sum_{e \in U} \sum_{j=1}^{r_e} \mathrm{price}(e, j)$
- The objective function value of the dual solution (α, β) :

$$\sum_{e \in U} r_e \alpha_e - \sum_{s \in S} \beta_s = \sum_{e \in U} \sum_{j=1}^{r_e} \mathrm{price}(e, j)$$

The lemma follows.

The dual solution (α,β) is in general infeasible but when scaled by a factor H_n a feasible solution occurs :

For each $e \in U$: $y_e = \dfrac{\alpha_e}{H_n}$ and each $s \in S$: $z_s = \dfrac{\beta_s}{H_n}$

Constrained SET MULTICOVER: dual fitting analysis

- **Lemma**:

"The pair (y, z) is a feasible solution for the dual program"

- Consider a set $s \in S$ consisting of k elements
- Number the elements in the order in which their requirements are fulfilled: $e_1, \ldots, e_k$
- **Assume s is not picked by the algorithm**
- When the algorithm is about to cover the last copy of e_i , s contains at least k-i+1 alive elements, so: $\text{price}(e_i, r_{e_i}) \le c(s)/(k-i+1)$

- Since $z_s = 0$: $$(\sum_{i=1}^{k} y_{e_i}) - z_s = \frac{1}{H_n} \sum_{i=1}^{k} \text{price}(e_i, r_{e_i})$$

$$\le \frac{c(s)}{H_n} \cdot (\frac{1}{k} + \frac{1}{k-1} + \ldots + \frac{1}{1}) \le c(s)$$

Constrained SET MULTICOVER: dual fitting analysis

- **Assume that s is picked by the algorithm.** Before this happens $k' \geq 0$ elements of S are *completely* covered. Then :

$$(\sum_{i=1}^{k} y_{e_i}) - z_s =$$

$$= \frac{1}{H_n} \cdot \left[\sum_{i=1}^{k} \text{price}(e_i, r_{e_i}) - \sum_{i=k'+1}^{k} (\text{price}(e_i, r_{e_i}) - \text{price}(e_i, j_i)) \right]$$

$$= \frac{1}{H_n} \cdot \left[\sum_{i=1}^{k'} \text{price}(e_i, r_{e_i}) + \sum_{i=k'+1}^{k} \text{price}(e_i, j_i) \right]$$

Where s covers the jth copy of e_i, for each $i \in \{k'+1, \ldots, k\}$

But $\sum_{i=k'+1}^{k} \text{price}(e_i, j_i) = c(s)$

Finally consider elements e_i, $i \in \{1, \ldots, k'\}$

- When the last copy of e_i is being covered , s is not yet picked and covers at least k-i+1 alive elements. Thus $\text{price}(e_i, r_{e_i}) \leq \frac{c(s)}{k - i + 1}$
- Therefore:

$$(\sum_{i=1}^{k} y_{e_i}) - z_s \leq \frac{c(s)}{H_n} \cdot (\frac{1}{k} + \ldots + \frac{1}{k - k' + 1} + 1) \leq c(s)$$

Constrained SET MULTICOVER: dual fitting analysis

Theorem

"The greedy algorithm achieves an approximation guarantee of H_n for the constrained multicover problem"

- By the two former lemmas the total cost of the multicover produced by the algorithm is :

$$\sum_{e \in U} r_e \alpha_e - \sum_{s \in S} \beta_s = H_n \cdot \left[\sum_{e \in U} r_e y_e - \sum_{s \in S} z_s \right] \leq H_n \cdot OPT$$

$\Downarrow$

Thus, the integrality gap of LP is bounded by H_n