In the name of God

Network Flows

6. Lagrangian Relaxation
6.3 Lagrangian Relaxation and Integer Programming

Fall 2010

Instructor: Dr. Masoud Yaghini

Outline

- Integer Programming
- Branch-and-Bound Technique
- Applications of Lagrangian Relaxation in Integer Programming
- Applications of Lagrangian Relaxation in Uncapacitated Network Design

Integer Programming

Integer Programming

- ***Integer Programming (IP) Problem***
  - A linear programming problem in which the decision variables are required to have integer values
  - The mathematical model for ***integer programming*** is the ***linear programming model*** with the one additional restriction that the variables must have integer values
- ***Integer Linear Programming problem***
  - The more complete name of ***IP problem*** but the adjective linear normally is dropped except when this problem is contrasted with ***integer nonlinear programming problem***

Integer Programming

- ***Mixed Integer Programming (MIP)***
  - If only some of the variables are required to have integer values and the divisibility assumption holds for the rest variables
- ***Pure Integer Programming***
  - If all the variables are required to have integer values
- ***Binary Integer Programming (BIP)***
  - or ***0–1 integer programming problems***
  - IP problems that contain only binary variables

Branch-and-Bound Technique

Branch-and-Bound Technique

- ***Branch-and-bound technique***
  - The basic concept is to ***divide and conquer***.
  - Since the original "large" problem is too difficult to be solved directly, it is divided into smaller and smaller subproblems until these subproblems can be conquered.

Branch-and-Bound Technique

- ***Branch-and-bound technique steps***
  - ***The dividing (branching)***
    - partitioning the entire set of feasible solutions into smaller and smaller subsets.
  - ***The bounding***
    - how good the best solution in the subset can be
  - ***The conquering (fathoming)***
    - discarding the subset if its bound indicates that it cannot possibly contain an optimal solution for the original problem.

Branch-and-Bound Technique

- An example:

$$\text{Maximize} \qquad Z = 9x_1 + 5x_2 + 6x_3 + 4x_4,$$

subject to

$$\begin{array}{lrcrcrcrcr}
(1) & 6x_1 & + & 3x_2 & + & 5x_3 & + & 2x_4 & \le & 10 \\
(2) & & & & & x_3 & + & x_4 & \le & 1 \\
(3) & -x_1 & + & & & x_3 & & & \le & 0 \\
(4) & & & -x_2 & & & + & x_4 & \le & 0
\end{array}$$

and

$$(5) \qquad x_j \text{ is binary}, \qquad \text{for } j = 1, 2, 3, 4.$$

Branch-and-Bound Technique

- **Branching**
  - dividing the whole problem into the two smaller subproblems shown below.
- **Subproblem 1:** Fix $x_1 = 0$

$$\text{Maximize} \qquad Z = 5x_2 + 6x_3 + 4x_4,$$

subject to

$$\begin{array}{lrcl}
(1) & 3x_2 + 5x_3 + 2x_4 & \leq & 10 \\
(2) & x_3 + x_4 & \leq & 1 \\
(3) & x_3 & \leq & 0 \\
(4) & -x_2 + x_4 & \leq & 0 \\
(5) & x_j \text{ is binary,} & & \text{for } j = 2, 3, 4.
\end{array}$$

Branch-and-Bound Technique

- **Subproblem 2** Fix $x_1 = 1$

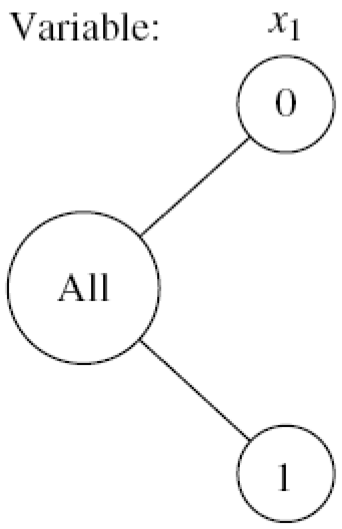
$$\text{Maximize} \quad Z = 9 + 5x_2 + 6x_3 + 4x_4,$$

subject to

$$\begin{array}{lrcl}
(1) & 3x_2 + 5x_3 + 2x_4 & \leq & 4 \\
(2) & x_3 + x_4 & \leq & 1 \\
(3) & x_3 & \leq & 1 \\
(4) & -x_2 + x_4 & \leq & 0 \\
(5) & x_j \text{ is binary,} & & \text{for } j = 2, 3, 4.
\end{array}$$

Branch-and-Bound Technique

- ***Solution tree (or enumeration tree)***
  - The tree, which will continue "growing branches" iteration by iteration
  - The variable used to do this branching at any iteration by assigning values to the variable *is called the* ***branching variable***.



Branch-and-Bound Technique

- ***Selecting branching variables***
  - Sophisticated methods for selecting branching variables are an important part of some branch-and-bound algorithms
  - for simplicity, we select them in their natural order—x_1, x_2, . . . , x_n

Branch-and-Bound Technique

- ***Bounding***
  - For each of these subproblems, we now need to obtain a ***bound*** on how good its best feasible solution can be.
  - The standard way of doing this is to quickly solve a simpler ***relaxation*** of the subproblem.
  - In most cases, a relaxation of a problem is obtained simply by deleting ("relaxing") one set of constraints that had made the problem difficult to solve.
  - For IP problems, the most troublesome constraints are those requiring the respective variables to be integer.
  - Therefore, the most widely used relaxation is the ***LP relaxation*** that deletes this set of constraints.

Branch-and-Bound Technique

- first the whole problem is considered
  - Its LP relaxation is obtained by replacing the

    x_j is binary, for j = (1, 2, 3, 4)
  - by the constraints that

    $x_j \leq 1$ and $x_j \geq 0$ *for j* = (1, 2, 3, 4).
  - Using the ***simplex method*** to quickly solve this LP relaxation yields its optimal solution

$$(x_1, x_2, x_3, x_4) = \left(\frac{5}{6}, 1, 0, 1\right), \qquad \text{with } Z = 16\frac{1}{2}.$$

Branch-and-Bound Technique

- Therefore, $Z \leq 16.5$ for all feasible solutions for the original BIP problem
  - since these solutions are a subset of the feasible solutions for the LP relaxation.
- In fact, this bound of 16.5 can be rounded down to 16, because all coefficients in the objective function are integer, so all integer solutions must have an integer value for Z.
- Bound for whole problem: $Z \leq 16$.

Branch-and-Bound Technique

- Let us obtain the bounds for the two subproblems
- Replacing the constraints that x_j is binary for j = (2, 3, 4) by the constraints $0 \leq x_j \leq 1$ *for* j = (2, 3, 4), plus the fixed value of x_1
- Applying the simplex method then yields their optimal solutions:
  - LP relaxation of subproblem 1:

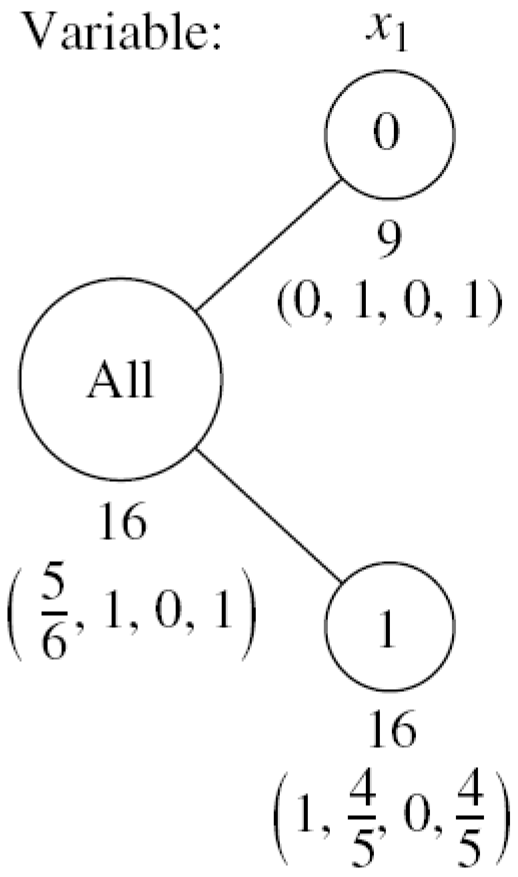
$$(x_1, x_2, x_3, x_4) = (0, 1, 0, 1) \qquad \text{with } Z = 9.$$

  - LP relaxation of subproblem 2:

$$(x_1, x_2, x_3, x_4) = \left(1, \frac{4}{5}, 0, \frac{4}{5}\right) \qquad \text{with } Z = 16\frac{1}{5}.$$

Branch-and-Bound Technique

- The resulting bounds for the subproblems then are:
  - Bound for subproblem 1: $Z \leq 9$
  - Bound for subproblem 2: $Z \leq 16$


Variable:
x_1
0
9
(0, 1, 0, 1)
All
16
$\left(\frac{5}{6}, 1, 0, 1\right)$
1
16
$\left(1, \frac{4}{5}, 0, \frac{4}{5}\right)$

Branch-and-Bound Technique

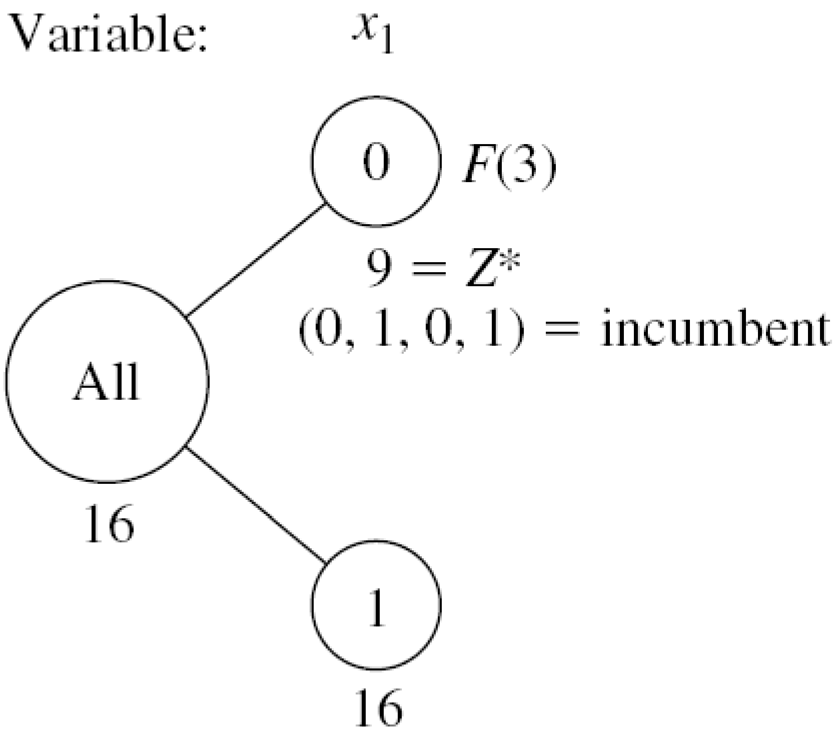
- ***Three ways of fathoming a subproblem***
  - (1) The optimal solution for its LP relaxation is an integer solution.
    - If necessary, the best-so-far integer solution or ***incumbent*** (Z*) must be updated.
  - (2) Its bound is less than or equal ***best-so-far integer solution***
  - (3) A subproblem's LP relaxation has no ***feasible solutions***

Branch-and-Bound Technique

- ***Fathoming Tests***. A subproblem is *fathomed* if
  - **Test 1**: Its bound $\leq Z^*$,
    or
  - **Test 2**: Its LP relaxation has no feasible solutions,
    or
  - **Test 3**: The optimal solution for its LP relaxation is integer.
    - (Z^* should be updated if necessary)

Branch-and-Bound Technique

- The results of applying fathoming tests



Branch-and-Bound Technique

- ***Optimality test:***
  - Stop when there are no remaining subproblems;
  - the current incumbent is optimal.

Integer Programming

- **The BIP Branch-and-Bound Algorithm**
- *Initialization:*
  - Set $Z^* = -\infty$.
  - Apply the bounding step, fathoming step, and optimality test to the whole problem.
  - If not fathomed, classify this problem as the one remaining "subproblem" for performing the first full iteration below.

Branch-and-Bound Technique

- ***Steps for each iteration:***
  - ***Branching***:
    - Select the one remaining (unfathomed) subproblems
    - Branch from the node for this subproblem to create two new subproblems by fixing the next variable (the branching variable) at either 0 or 1.
  - ***Bounding:***
    - For each new subproblem, obtain its bound.
  - ***Fathoming:***
    - For each new subproblem, apply the three fathoming tests, and discard those subproblems that are fathomed by any of the tests.

Branch-and-Bound Technique

- **Iteration 2.**
  - The only remaining subproblem corresponds to the $x_1 = 1$ node, so we shall branch from this node to create the two new subproblems
- ***Subproblem 3***: Fix $x_1 = 1$, $x_2 = 0$

$$\text{Maximize} \qquad Z = 9 + 6x_3 + 4x_4,$$

subject to

$$\begin{array}{ll}
(1) & 5x_3 + 2x_4 \leq 4 \\
(2) & x_3 + x_4 \leq 1 \\
(3) & x_3 \leq 1 \\
(4) & x_4 \leq 0 \\
(5) & x_j \text{ is binary,} \qquad \text{for } j = 3, 4.
\end{array}$$

Branch-and-Bound Technique

- ***Subproblem 4:*** Fix $x_1 = 1$, $x_2 = 1$

$$\text{Maximize} \quad Z = 14 + 6x_3 + 4x_4,$$

subject to

$$\begin{aligned}
&(1) \quad 5x_3 + 2x_4 \le 1 \\
&(2) \quad x_3 + x_4 \le 1 \\
&(3) \quad x_3 \le 1 \\
&(4) \quad x_4 \le 1 \\
&(5) \quad x_j \text{ is binary}, \quad \text{for } j = 3, 4.
\end{aligned}$$

Branch-and-Bound Technique

- LP relaxation of subproblem 3:

$$(x_1, x_2, x_3, x_4) = \left(1, 0, \frac{4}{5}, 0\right) \qquad \text{with } Z = 13\frac{4}{5},$$

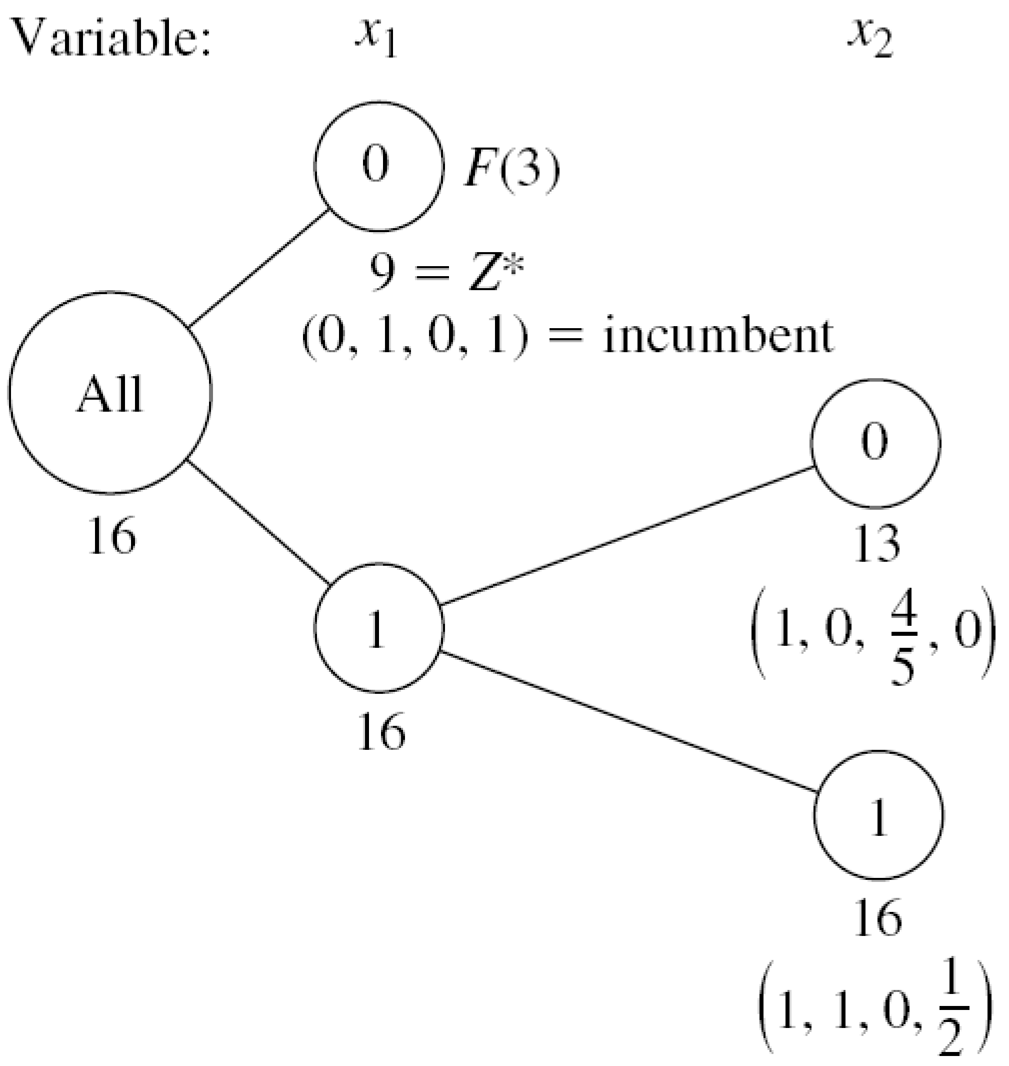
- LP relaxation of subproblem 3:

$$(x_1, x_2, x_3, x_4) = \left(1, 1, 0, \frac{1}{2}\right) \qquad \text{with } Z = 16.$$

- The resulting bounds for the subproblems are
  - Bound for subproblem 3: $Z \leq 13$,
  - Bound for subproblem 4: $Z \leq 16$.

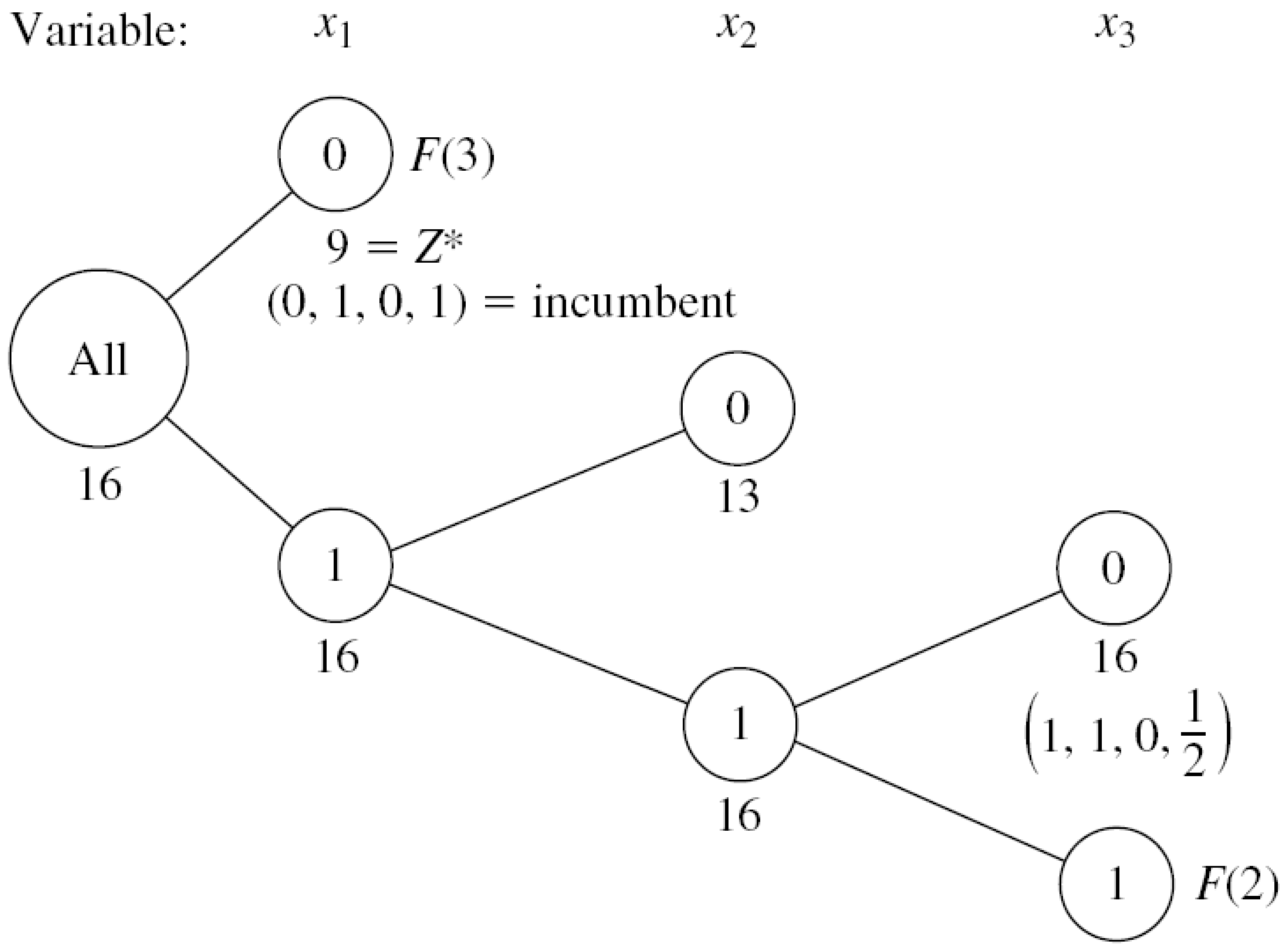
Branch-and-Bound Technique

- The solution tree after iteration 2


Variable:
x_1
x_2
All
16
0
$F(3)$
$9 = Z^*$
$(0, 1, 0, 1)$ = incumbent
1
16
0
13
$\left(1, 0, \frac{4}{5}, 0\right)$
1
16
$\left(1, 1, 0, \frac{1}{2}\right)$

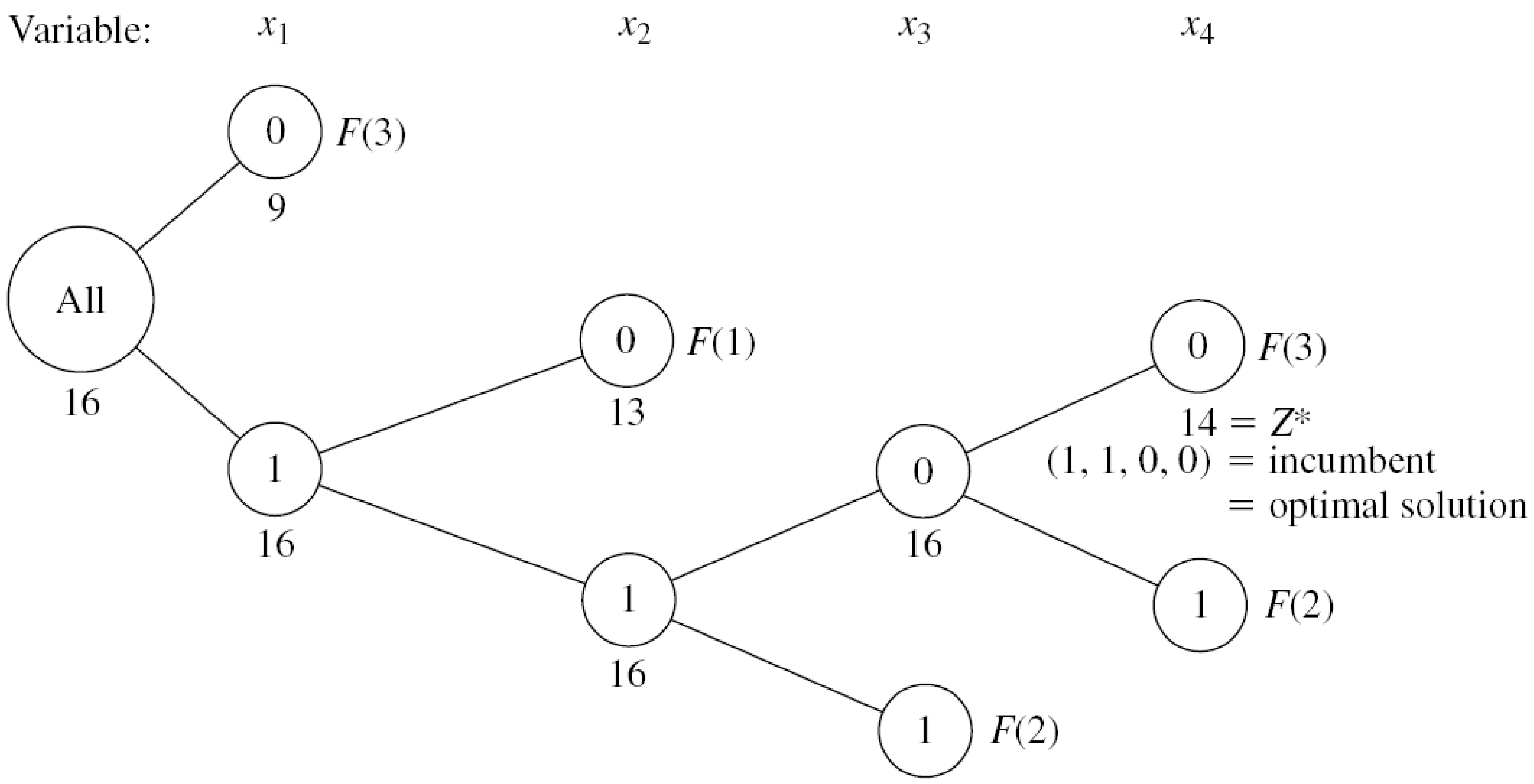
Branch-and-Bound Technique

- The solution tree after iteration 3



Integer Programming

- The solution tree after iteration 4



Applications of Lagrangian Relaxation in Integer Programming

Lagrangian Relaxation and Integer Programming

- The primary use of the ***Lagrangian relaxation*** technique is to obtain lower bounds on the objective function values of ***discrete optimization problems***.
- By relaxing the ***integrality constraints*** in the integer programming formulation of a discrete optimization problem, thereby creating a linear programming relaxation.
- The lower bound obtained by the Lagrangian relaxation technique is at least as sharp as that obtained by using a linear programming relaxation.

Lagrangian Relaxation and Integer Programming

- **Theorem:**
  - Suppose that we apply the Lagrangian relaxation technique to a linear programming problem (P') defined as
    $$min\{cx : \mathcal{A}x = b, \mathcal{D}x \leq q, x \geq 0\}$$
  - by relaxing the constraints $\mathcal{A}x = b$.
  - Then the optimal value L^* of the Lagrangian multiplier problem equals the optimal objective function value of (P').

Lagrangian Relaxation and Integer Programming

- ***Applying Lagrangian relaxation to a discrete optimization problem (P)***
  - Consider a discrete optimization problem (P):

    $$\min\{cx : \mathcal{A}x = b, x \in X\}$$

  - We assume that the discrete set X is specified as

    $$X = \{x : \mathcal{D}x \le q, x \ge 0 \text{ and integer}\}$$

    - for an integer matrix $\mathcal{D}$ and an integer vector q.
  - Consequently, the problem (P) becomes

    $$z^* = \min\{cx : \mathcal{A}x = b, \mathcal{D}x \le q, x \ge 0 \text{ and integer}\}$$

Lagrangian Relaxation and Integer Programming

- ***Linear Programming Relaxation***
  - Let (LP) denote the ***linear programming relaxation*** of the problem (P) and let z^o denote its optimal objective function value:

$$z^o = \min\{cx : \mathcal{A}x = b, \mathcal{D}x \le q, x \ge 0\} \qquad \text{(LP)}$$

  - Clearly, $z^o \le z^*$ because the set of feasible solutions of (P) lies within the set of feasible solutions of (LP).
  - Therefore, the linear programming relaxation provides a valid lower bound on the optimal objective function value of (P).

Lagrangian Relaxation and Integer Programming

- The Lagrangian multiplier problem also gives a lower bound ***L****** on the optimal objective function value of (P).
- Lagrangian relaxation yields a lower bound that is at least as good as that obtained from the linear programming relaxation, i.e, $z^o \leq L^*$
- We establish this result by showing that the ***Lagrangian multiplier problem*** also solves a linear programming problem but that the solution space for this problem is contained within the solution space of the problem (LP).

Lagrangian Relaxation and Integer Programming

- The Lagrangian multiplier problem solves uses ***convexification*** of the solution space:

$$X = \{x : \mathcal{D}x \leq q, x \geq 0 \text{ and integer}\}$$

- We assume that $X = \{x^1, x^2, \ldots, x^K\}$ is a finite set.
- A solution x is a ***convex combination*** of the solutions $x^1, x^2, \ldots, x^K$ if

$$x = \sum_{k=1}^{K} \lambda_k x^k$$

  - for some nonnegative weights $\lambda_1, \lambda_2, \ldots, \lambda_K$ satisfying the condition:

$$\sum_{k=1}^{K} \lambda_k = 1$$

Lagrangian Relaxation and Integer Programming

- Let $\mathcal{H}(X)$ denote the ***convex hull*** of X
  - i.e., the set of all convex combinations of X.
- **Property:**
  - (a) The set $\mathcal{H}(X)$ is a ***polyhedron***, that is, it can be expressed as a solution space defined **by a finite number of linear inequalities**.
  - (b) Each extreme point solution of the polyhedron $\mathcal{H}(X)$ lies in X, and if we optimize a linear objective function over $\mathcal{H}(X)$, some solution in X will be an optimal solution.
  - (c) The set $\mathcal{H}(X)$ is contained in the set of solutions

$$\{\boldsymbol{x} : \mathcal{D}\boldsymbol{x} \leq \boldsymbol{q}, \boldsymbol{x} \geq \mathbf{0}\}$$

Lagrangian Relaxation and Integer Programming

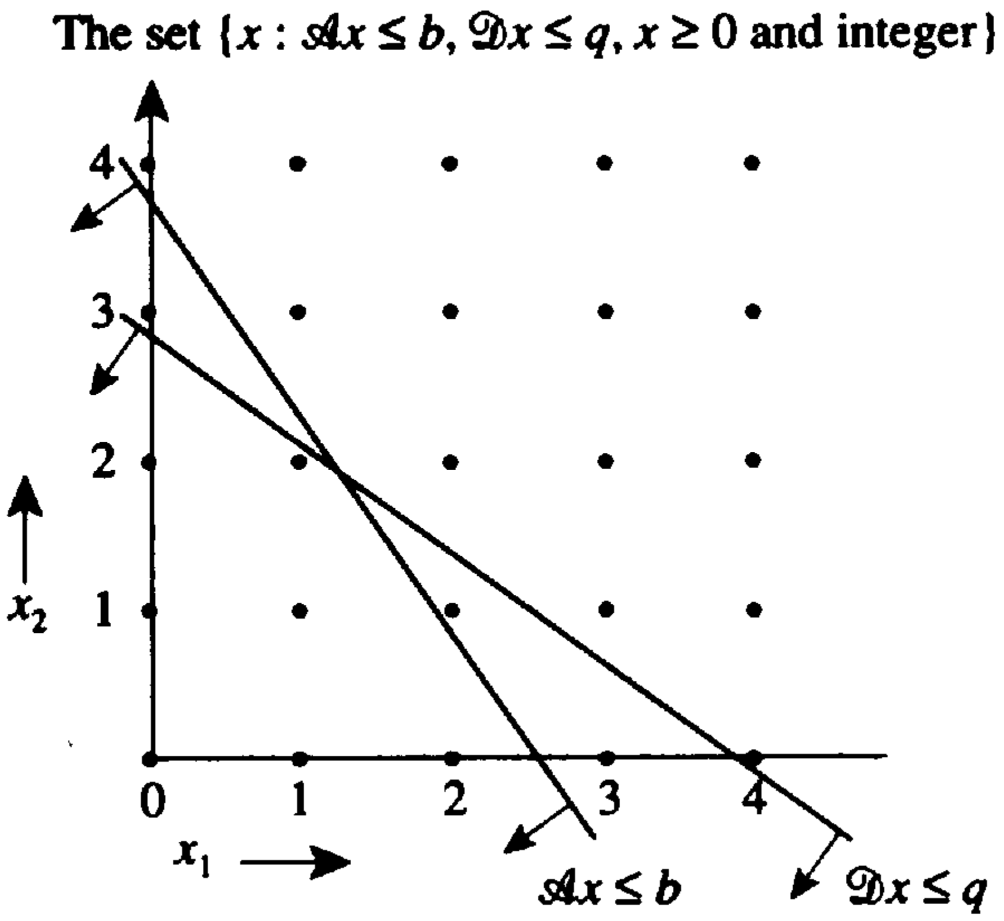
- ***Convexified Problem***
  - We refer to the below problem as the ***convexified problem*** (CP) of problem (P)

$$min\{cx : \mathcal{A}x = b, x \in \mathcal{H}(X)\}.$$

- **Theorem.**
  - The optimal objective function value L^* of the Lagrangian multiplier problem equals the optimal objective function value of the convexified program

Lagrangian Relaxation and Integer Programming

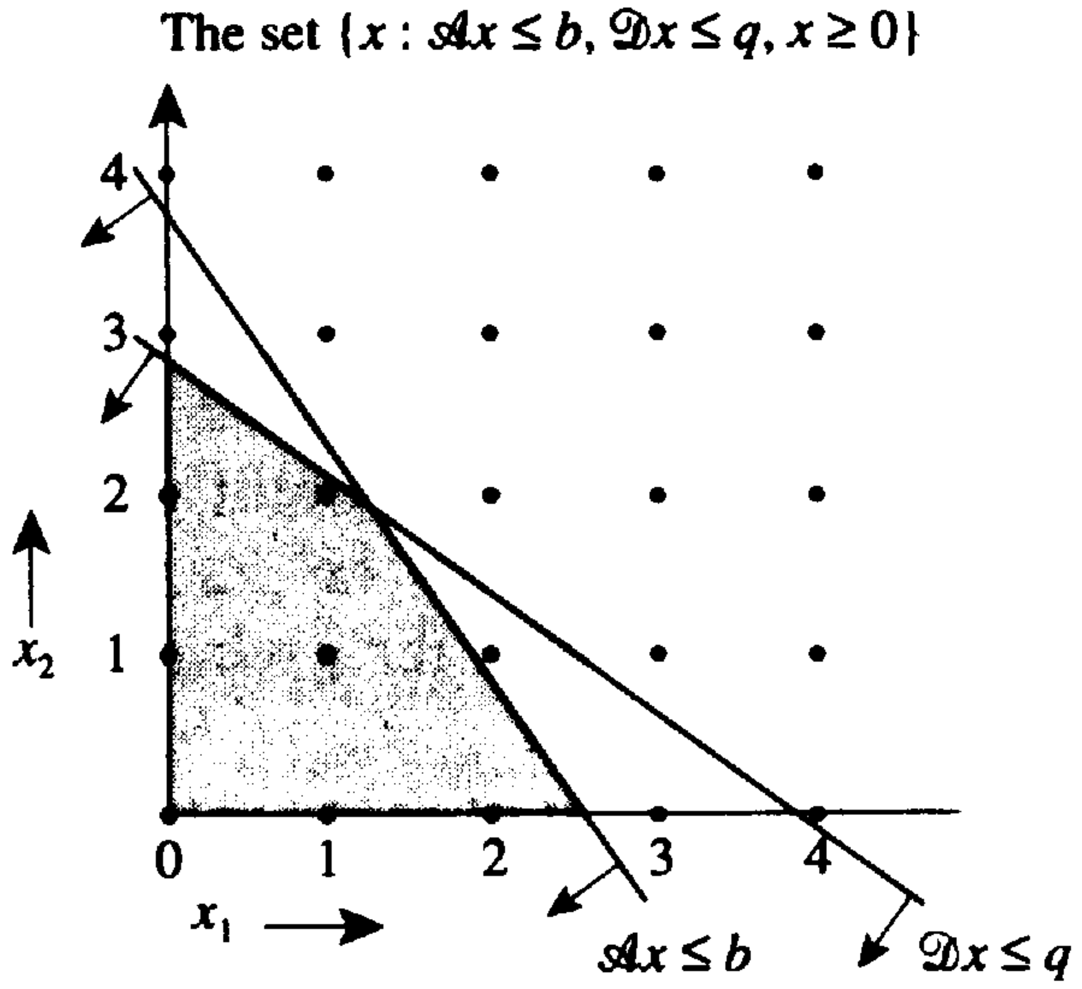
- **An Example:** We consider a two-variable problem with the constraints $\mathcal{A}x \leq b$ and $\mathcal{A}x \leq q$:



The set $\{x : \mathcal{A}x \leq b, \mathcal{D}x \leq q, x \geq 0$ and integer$\}$
4
3
2
1
x_2
0
1
2
3
4
x_1
$\mathcal{A}x \leq b$
$\mathcal{D}x \leq q$

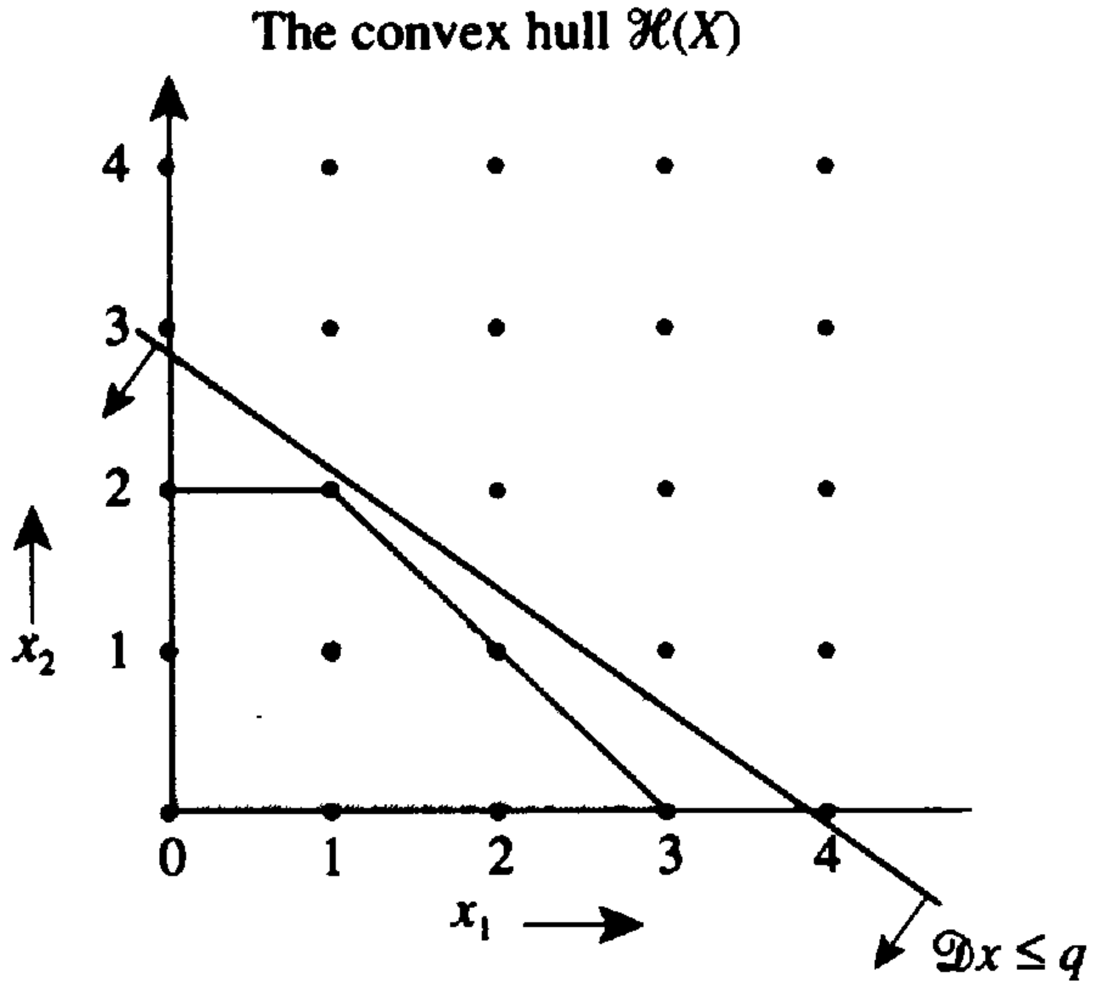
Lagrangian Relaxation and Integer Programming

- The solution space of the linear programming relaxation (LP) of the problem.


The set $\{x : \mathcal{A}x \le b, \mathcal{D}x \le q, x \ge 0\}$
4
3
2
1
0
1
2
3
4
x_2
x_1
$\mathcal{A}x \le b$
$\mathcal{D}x \le q$

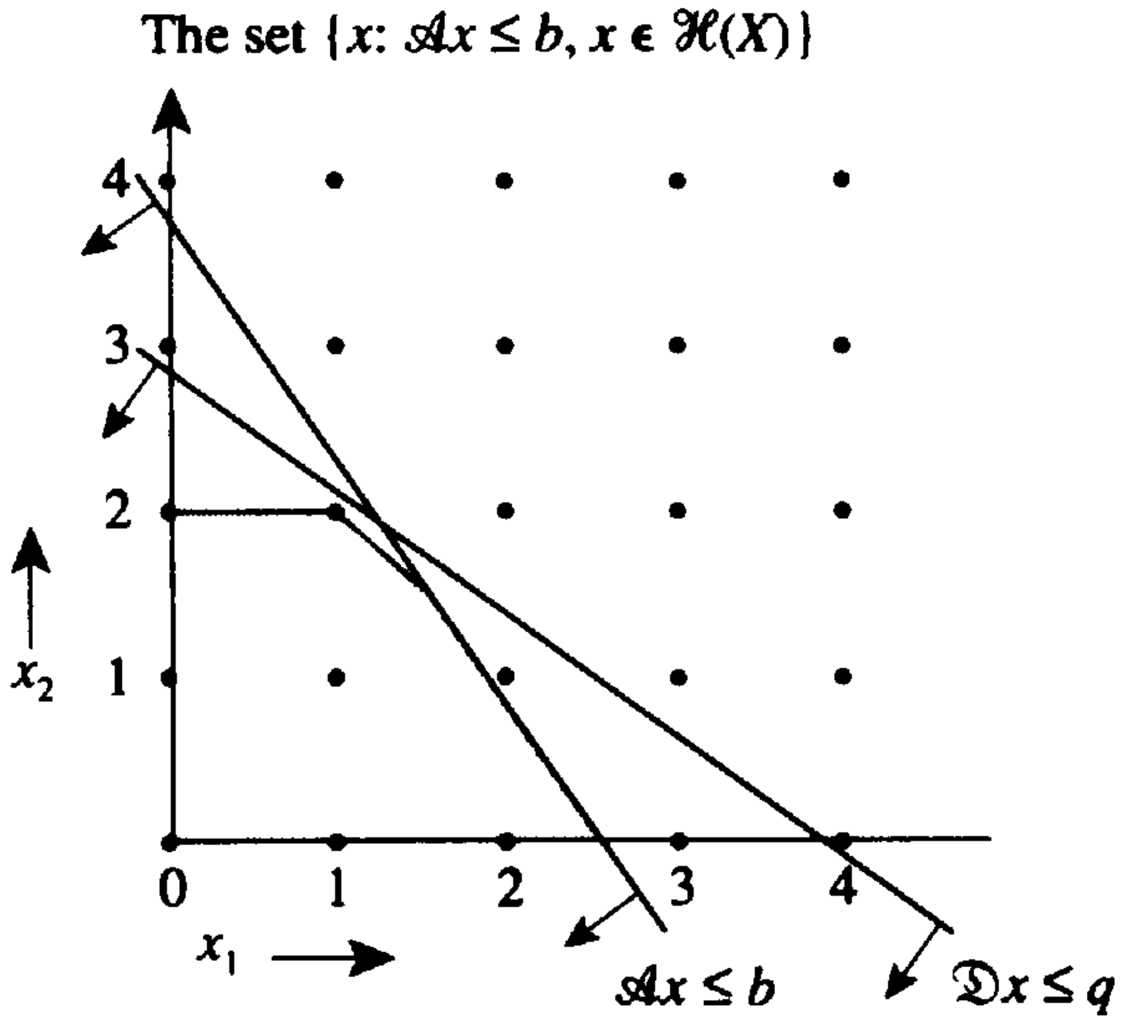
Lagrangian Relaxation and Integer Programming

- The convex hull $\mathcal{H}(X)$, $X = \{x : \mathcal{D}x \leq q, x \geq 0 \text{ and integer}\}$



Lagrangian Relaxation and Integer Programming

- The solution space of the ***convexified problem*** (CP). The solution space of (CP) is a subset of the solution space of (LP).



Lagrangian Relaxation and Integer Programming

- Since $\mathcal{H}(X)$ is contained in the set

$$\{x : \mathcal{D}x \leq q, x \geq 0\}$$

- the set of solutions of problem (CP) given by

$$\{x : \mathcal{A}x = b, x \in \mathcal{H}(X)\}$$

- is contained in the set of solutions of (LP) given by

$$\{x : \mathcal{A}x = b, \mathcal{D}x \leq q, x \geq 0\}$$

- Since optimizing the same objective function over a smaller solution space cannot improve the objective function value, hence, $\mathbf{z^o} \leq L^*$.

Lagrangian Relaxation and Integer Programming

- **Theorem:**
  - When applied to an integer program stated in minimization form, the lower bound obtained by the Lagrangian relaxation technique is always as large (or, sharp) as the bound obtained by the linear programming relaxation of the problem; that is, $\mathbf{z}^{\mathbf{o}} \leq \boldsymbol{L^*}$.

Lagrangian Relaxation and Integer Programming

- The situations will the Lagrangian bound equal the linear programming bound ($z^o = L^*$)
  - If the problems

$$\min\{cx + \mu(\mathcal{A}x - b) : \mathcal{D}x \le q,\ x \ge 0 \text{ and integer}\}$$

  - and

$$\min\{cx + \mu(\mathcal{A}x - b) : \mathcal{D}x \le q,\ x \ge 0\}$$

  - have the same optimal objective function values for every choice of the Lagrange multiplier μ.

Lagrangian Relaxation and Integer Programming

- For example, if the constraints $\mathcal{D}x \leq q$ are the mass balance constraints of a minimum cost flow problem (or any of its special cases, such as the maximum flow, shortest path, and assignment problems), the problem

$$\min\{cx + \mu(\mathcal{A}x - b) : \mathcal{D}x \leq q, x \geq 0\}$$

- will always have an integer optimal solution and imposing integrality constraints on the variables will not increase the optimal objective function value.

Applications of Lagrangian Relaxation in Uncapacitated Network Design

Uncapacitated Network Design

- Notation:
  - $G = (N, A)$: a directed network and can introduce an arc (i, j) or not into the design of the network
  - f_{ij} : we incur a design (construction) cost
  - k : a commodity that has a single source node s^k and a single destination node d^{k} .
  - x^k : the vector of flows of commodity k on the network.
  - $x^k{}_{ij}$: the fraction of the flow of commodity k on arc (i, j)
  - c^k : the cost vector for commodity k
  - y_{ij} : be a zero-one vector indicating whether or not we select arc (i, j) as part of the network design.

Uncapacitated Network Design

- Using this notation, we can formulate the network design problem as follows:

$$\text{Minimize} \quad \sum_{1 \le k \le K} c^k x^k + fy$$

subject to

$$\sum_{\{j:(i,j)\in A\}} x_{ij}^k - \sum_{\{j:(j,i)\in A\}} x_{ji}^k = \begin{cases} 1 & \text{if } i = s^k \\ -1 & \text{if } i = d^k \\ 0 & \text{otherwise} \end{cases} \quad \text{for all } i \in N,\ k = 1, 2, \ldots, K,$$

$$x_{ij}^k \le y_{ij} \qquad \text{for all } (i, j) \in A,\ k = 1, 2, \ldots, K,$$

$$x_{ij}^k \ge 0 \qquad \text{for all } (i, j) \in A \text{ and all } k = 1, 2, \ldots, K,$$

$$y_{ij} = 0 \text{ or } 1 \qquad \text{for all } (i, j) \in A.$$

Uncapacitated Network Design

- **Aim**: to find the design that minimizes the total systems cost-that is, the sum of the design cost and the routing cost.

- In this formulation, the ***forcing constraints***

$$x_{ij}^k \leq y_{ij} \qquad \text{for all } (i, j) \in A,\ k = 1, 2, \ldots, K,$$

- state that
  - if we do not select arc (i, j) as part of the design, we cannot flow any fraction of commodity k's demand on this arc, and
  - if we do select arc (i, j) as part of the design, we can flow as much of the demand of commodity k as we like on this arc.

Uncapacitated Network Design

- Note that if we remove the forcing constraints from this model, the resulting model in the flow variables x^k decomposes into a set of independent shortest path problems, one for each commodity k.
- Consequently, the model is another attractive candidate for the application of Lagrangian relaxation.

Uncapacitated Network Design

- To see why this type of solution approach is attractive:
  - consider a typically sized problem with **50 nodes** and **500 candidate arcs**
  - Suppose that we have a separate commodity for each pair of nodes. Then we have 50(49) = **2450 commodities**.
  - Since each commodity can flow on each arc, the model has 2450(500) = **1,225,000 flow variables**
  - Since (1) each flow variable defines a forcing constraint, and (2) each commodity has a flow balance constraint at each node, the model has 1,225,000 + 2450(50) = **1,347,500 constraints**.
  - In addition, it has **500 zero-one variables**.

Uncapacitated Network Design

- So even as a linear program, this model is very big.
- By decomposing the problem, however, for each choice of the vector of Lagrange multipliers, we will solve **2450 small shortest path problems**.

The End