Branch-and-Bound Algorithms
(Maximization Problem)

Main Ingredients:

1) **Branching scheme (branch-decision tree).**
2) **Upper Bound computation (Problem Relaxations).**
3) **Reduction Procedures.** ***(RSC)***
4) **Dominance Criteria among the nodes of the branch-decision tree.**
5) **Parametric Techniques for the computation of the Upper Bound at each node of the branch-decision tree.**
6) **Lower Bound computation (Heuristic Procedures).**
7) **"Core problem" approach for large-size instances.**

Branch-and-Bound Algorithms (2)

* Given the maximization problem P_0 :

$$(P_0) \quad z(P_0) = \max \{ f(x) : x \in F(P_0)\}$$

* Subdivide P_0 into m subproblems: $P_1, P_2, \ldots, P_m$ $(m > 1)$:

$$z(P_k) = \max \{ f(x) : x \in F(P_k)\} \qquad \text{for } k = 1, 2, \ldots, m$$

(RSC)

(where $F(P_k)$ is the set of the feasible solutions of problem P_k)

so as to have: $F(P_1) \cup F(P_2) \cup \ldots \cup F(P_m) = F(P_0)$

Any feasible solution of problem P_0 must be a feasible solution of at least one of the subproblems $P_1, P_2, \ldots, P_m$ (and viceversa).

Branch-and-Bound Algorithms (3)

* **Generally problem P_0 is "partitioned" into subproblems $P_1, P_2, \ldots, P_m$**
so as to have: $F(P_k) \cap F(P_j) = \emptyset$ **for each pair of subproblems P_k and P_j ($k = 1, 2, \ldots, m$; $j = 1, 2, \ldots, m$; $k \neq j$).**

* $z(P_0) = \max \{z(P_k) : k = 1, 2, \ldots, m\}$

* **If subproblem P_k cannot be directly solved, subdivide it.**

…

Branch-and-Bound Algorithms (4)

* The branching scheme is represented through a "*branch-decision tree*":

a) each "*node*" k of the tree corresponds to a subproblem P_k

b) the "*root node*" (i.e., node 0) corresponds to the original problem P_0

* A node k of the tree (and the corresponding subproblem P_k) can be "*fathomed*" if:

1) P_k is infeasible (i.e., if $F(P_k) = \emptyset$); or

2) $UB(P_k) \leq z^*$ (where: $UB(P_k)$ is an "upper bound" on $z(P_k)$, i.e., $UB(P_k) \geq z(P_k)$, and z^* is the value of the best feasible solution found so far)

Relaxations (Maximization Problem)

* An "*upper bound*" $UB(P_k)$ on $z(P_k)$ can be computed by solving to optimality a "*Relaxed Problem*" R_k :

$$UB(P_k) = z(R_k) = \max \{ g(x) : x \in F(R_k)\}$$

such that:

1) $F(P_k) \subseteq F(R_k)$,
2) $g(x) \geq f(x)$ for $x \in F(P_k)$.

* A "*good*" upper bound $UB(P_k)$ must be:

- as "close" as possible to $z(P_k)$ ($z(R_k)$ small);
- with R_k "easy" to be solved (small computing time to solve R_k to optimality).

Two extreme (useless) cases: a) $UB(P_k) = \infty$; b) $UB(P_k) = z(P_k)$

Relaxations

Given the ILP model:

$$(P) \qquad z(P) = \max \sum_{j=1,n} a_j x_j$$

$$\sum_{j=1,n} b_{ij} x_j \le c_i \qquad (i = 1, \ldots, m)$$

$$\sum_{j=1,n} d_{hj} x_j = e_h \qquad (h = 1, \ldots, k)$$

$$x_j \in \{0, 1\} \qquad (j = 1, \ldots, n)$$

Relaxations: Constraint Elimination

$$(R) \qquad UB = z(R) = \max \sum_{j=1,n} a_j x_j$$

$$\sum_{j=1,n} b_{ij} x_j \le c_i \qquad (i = 1, \ldots, m)$$

$$\xcancel{\sum_{j=1,n} d_{hj} x_j = e_h \qquad (h = 1, \ldots, k)} \qquad (EC)$$

$$x_j \in \{0, 1\} \qquad (j = 1, \ldots, n)$$

* *R*: "well-structured" Relaxed Problem

* If the optimal solution (x) is feasible for problem *P* (i.e., constraints (*EC*) are satisfied), (x) is also optimal for *P*.

Continuous (LP) Relaxation (2)

$$(R) \qquad UB = z(R) = \max \sum_{j=1,n} a_j\, x_j$$

$$\sum_{j=1,n} b_{ij}\, x_j \le c_i \qquad (i = 1, \ldots, m)$$

$$\sum_{j=1,n} d_{hj}\, x_j = e_h \qquad (h = 1, \ldots, k) \qquad (EC)$$

$$\cancel{x_j \in \{0, 1\}} \quad 0 \le x_j \le 1 \;\; (j = 1, \ldots, n)$$

* ***R*: *Linear Programming* (*LP*) Problem**

- **If the optimal solution (x) is feasible for problem P (i.e., (x) is integer), (x) is also optimal for P.**

- **If the coefficients (a_j) are integer: UB $= \lfloor z(R) \rfloor$**

Surrogate Relaxation (1)

Consider an array (s_i) of m non-negative elements (surrogate multipliers) associated with the "inequality" constraints:

$$(R(s)) \qquad UB(s) = \max \sum_{j=1,n} a_j x_j$$

$$s_i \sum_{j=1,n} b_{ij} x_j \le s_i c_i \qquad (i = 1, \ldots, m) \qquad (SC)$$

$$\sum_{j=1,n} d_{hj} x_j = e_h \qquad (h = 1, \ldots, k)$$

$$x_j \in \{0, 1\} \qquad (j = 1, \ldots, n)$$

*** If "equality" constraints are considered, the corresponding surrogate multipliers can take any value.**

Surrogate Relaxation (2)

Consider an array (s_i) of m non-negative elements:

$$(R(s)) \qquad UB(s) = \max \sum_{j=1,n} a_j x_j$$

$$s_i \sum_{j=1,n} b_{ij} x_j \le s_i c_i \qquad (i = 1, \ldots, m) \qquad (SC)$$

$$\sum_{j=1,n} d_{hj} x_j = e_h \qquad (h = 1, \ldots, k)$$

$$x_j \in \{0, 1\} \qquad (j = 1, \ldots, n)$$

$$\sum_{i=1,m} s_i \sum_{j=1,n} b_{ij} x_j \le \sum_{i=1,m} s_i c_i \qquad (RSC)$$

Surrogate Relaxation (3)

Consider an array (s_i) of m non-negative elements:

$$(R(s)) \qquad UB(s) = \max \sum_{j=1,n} a_j x_j$$

$$\sum_{i=1,m} s_i \sum_{j=1,n} b_{ij} x_j \le \sum_{i=1,m} s_i c_i \qquad (RSC)$$

$$\sum_{j=1,n} d_{hj} x_j = e_h \qquad (h = 1, \ldots, k)$$

$$x_j \in \{0, 1\} \qquad (j = 1, \ldots, n)$$

$$\sum_{j=1,n} B_j x_j \le C \qquad (RSC)$$

with $C = \sum_{i=1,m} s_i c_i$, $B_j = \sum_{i=1,m} s_i b_{ij}$

Surrogate Relaxation (4)

$(R(s))$ $\quad UB(s) \;=\; \max \;\; \sum_{j=1,n} a_j\, x_j$

$$\sum_{j=1,n} B_j\, x_j \;\le\; C \qquad\qquad (RSC)$$

$$\sum_{j=1,n} d_{hj}\, x_j \;=\; e_h \qquad\qquad (h = 1, \ldots, k)$$

$$x_j \in \{0, 1\} \qquad\qquad (j = 1, \ldots, n)$$

with $C = \sum_{i=1,m} s_i\, c_i$, $\qquad B_j = \sum_{i=1,m} s_i\, b_{ij}$

* Well-structured Problem

$UB(s)$ is a valid Upper Bound for any non-negative array (s_i).

Surrogate Relaxation (5)

$(R(s))\qquad UB(s) = \max \sum_{j=1,n} a_j\, x_j$

$$\sum_{j=1,n} B_j\, x_j \le C \qquad (RSC)$$

$$\sum_{j=1,n} d_{hj}\, x_j = e_h \qquad (h = 1, \dots, k)$$

$$x_j \in \{0, 1\} \qquad (j = 1, \dots, n)$$

with $C = \sum_{i=1,m} s_i\, c_i$, $\quad B_j = \sum_{i=1,m} s_i\, b_{ij}$

$UB(s)$ is a valid Upper Bound for any non-negative array (s_i).

*** If the optimal solution (x) is feasible for problem P (i.e., the m constraints (SC) are satisfied), (x) is also optimal for P.**

Surrogate Relaxation (6)

$$(R(s)) \qquad UB(s) = \max \sum_{j=1,n} a_j x_j$$

$$\sum_{j=1,n} B_j x_j \le C \qquad (RSC)$$

$$\sum_{j=1,n} d_{hj} x_j = e_h \qquad (h = 1, \ldots, k)$$

$$x_j \in \{0, 1\} \qquad (j = 1, \ldots, n)$$

* $UB(s)$ is a valid Upper Bound for any non-negative array (s_i).

* Find (s^*_i) $(i = 1, 2, \ldots, m)$ so as to have:

$$UB(s^*) = \text{Min}\,\{UB(s) : s_i \ge 0 \text{ for } i = 1, 2, \ldots, m\}$$

* Surrogate Dual Problem (exact or heuristic procedures)

Lagrangian Relaxation of Equality Constraints (1)

Consider an array (u_h) of k elements (Lagrangian multipliers) associated with the "equality" constraints, and modify the objective function as follows:

$$(S(u)) \quad z(u) = \max \sum_{j=1,n} a_j x_j + \sum_{h=1,k} u_h \left(\sum_{j=1,n} d_{hj} x_j - e_h \right)$$

$$\sum_{j=1,n} b_{ij} x_j \leq c_i \qquad (i = 1, \ldots, m)$$

$$\sum_{j=1,n} d_{hj} x_j = e_h \qquad (h = 1, \ldots, k) \qquad (LC)$$

$$x_j \in \{0, 1\} \qquad (j = 1, \ldots, n)$$

*** Note that: $z(u) = z(P)$ for any array (u_h).**

Lagrangian Relaxation of Equality Constraints (2)

* Eliminate the equality constraints (*LC*):

$$(R(u))\quad UB(u) = \max \sum_{j=1,n} a_j x_j + \sum_{h=1,k} u_h \left(\sum_{j=1,n} d_{hj} x_j - e_h\right)$$

$$\sum_{j=1,n} b_{ij} x_j \le c_i \qquad (i = 1, \ldots, m)$$

$$x_j \in \{0, 1\} \qquad (j = 1, \ldots, n)$$

•$UB(u) = \max \left(\sum_{j=1,n} a_j(u)\, x_j\right) - \sum_{h=1,k} u_h e_h$

with $a_j(u) = a_j + \sum_{h=1,k} u_h d_{hj} \quad (j = 1, \ldots, n)$

* Well-structured Relaxed Problem

Lagrangian Relaxation of Equality Constraints (3)

*** Eliminate the equality constraints (*LC*):**

$$(R(u)) \quad UB(u) = \max \sum_{j=1,n} a_j x_j + \sum_{h=1,k} u_h \left(\sum_{j=1,n} d_{hj} x_j - e_h \right)$$

$$\sum_{j=1,n} b_{ij} x_j \leq c_i \qquad (i = 1, \ldots, m)$$

$$x_j \in \{0, 1\} \qquad (j = 1, \ldots, n)$$

•$UB(u) = \max \left(\sum_{j=1,n} a_j(u) x_j \right) - \sum_{h=1,k} u_h e_h$

with $a_j(u) = a_j + \sum_{h=1,k} u_h d_{hj} \quad (j = 1, \ldots, n)$

*** If the optimal solution (x) is feasible for problem P (i.e., the k constraints (LC) are satisfied), (x) is also optimal for P.**

Lagrangian Relaxation of Equality Constraints (4)

$$(R(u)) \quad UB(u) = \max \left(\sum_{j=1,n} a_j(u)\, x_j \right) - \sum_{h=1,k} u_h e_h$$

$$\sum_{j=1,n} b_{ij}\, x_j \leq c_i \qquad (i = 1, \ldots, m)$$

$$x_j \in \{0, 1\} \qquad (j = 1, \ldots, n)$$

- with $a_j(u) = a_j + \sum_{h=1,k} u_h\, d_{hj} \quad (j = 1, \ldots, n)$
- $UB(u)$ is a valid Upper Bound for any array (u_h).

* Find (u^*_h) $(h = 1, 2, \ldots, k)$ so as to have:

$$UB(u^*) = \text{Min}\, \{UB(u) : \text{for any } u_h \text{ with } h = 1, 2, \ldots, k\}$$

* Lagrangian Dual Problem (exact or heuristic procedures)

Lagrangian Relaxation of Inequality Constraints (1)

Consider an array (v_i) of m non-negative elements (Lagrangian multipliers) associated with the "inequality" constraints, and modify the objective function as follows:

$$(T(v)) \quad z(v) = \max \sum_{j=1,n} a_j x_j + \sum_{i=1,m} v_i \Big(c_i - \sum_{j=1,n} b_{ij} x_j\Big)$$

$$\sum_{j=1,n} b_{ij} x_j \le c_i \qquad (i = 1, \ldots, m) \qquad (LC)$$

$$\sum_{j=1,n} d_{hj} x_j = e_h \qquad (h = 1, \ldots, k)$$

$$x_j \in \{0, 1\} \qquad (j = 1, \ldots, n)$$

*** Note that:** $z(v) \ge z(P)$ **for any non-negative array** (v_i).

Lagrangian Relaxation of Inequality Constraints (2)

• **Eliminate the inequality constraints (*LC*):**

$$(R(v)) \quad UB(v) = \max \sum_{j=1,n} a_j\, x_j + \sum_{i=1,m} v_i \,(c_i - \sum_{j=1,n} b_{ij}\, x_j\,)$$

$$\sum_{j=1,n} d_{hj}\, x_j = e_h \qquad (h = 1, \ldots, k)$$

$$x_j \in \{0, 1\} \qquad (j = 1, \ldots, n)$$

Note that: $UB(v) \geq z(v) \geq z(P)$ **for any non-negative array** $(v_i\,)$.

$$UB(v) = \max \sum_{j=1,n} a_j(v)\; x_j + \sum_{i=1,m} v_i\, c_i$$

with $a_j(v) = a_j - \sum_{i=1,m} v_i\, b_{ij} \qquad (j = 1, \ldots, n)$

Lagrangian Relaxation of Inequality Constraints (3)

$$(R(v)) \quad UB(v) = \max \sum_{j=1,n} a_j(v)\, x_j + \sum_{i=1,m} v_i\, c_i$$

$$\sum_{j=1,n} d_{hj}\, x_j = e_h \qquad (h = 1, \ldots, k)$$

$$x_j \in \{0, 1\} \qquad (j = 1, \ldots, n)$$

with $a_j(v) = a_j - \sum_{i=1,m} v_i\, b_{ij} \qquad (j = 1, \ldots, n)$

*** Well-structured Relaxed Problem**

*** If the optimal solution (x) is feasible for problem P (i.e., the m constraints (LC) are satisfied), (x) is also optimal for P if and only if** $UB(v) = \sum_{j=1,n} a_j\, x_j$

Lagrangian Relaxation of Inequality Constraints (4)

$$(R(v)) \quad UB(v) = \max \sum_{j=1,n} a_j(v)\, x_j + \sum_{i=1,m} v_i\, c_i$$

$$\sum_{j=1,n} d_{hj}\, x_j = e_h \qquad (h = 1, \ldots, k)$$

$$x_j \in \{0, 1\} \qquad (j = 1, \ldots, n)$$

with $a_j(v) = a_j - \sum_{i=1,m} v_i\, b_{ij} \qquad (j = 1, \ldots, n)$

* $UB(v)$ is a valid Upper Bound for any non-negative array (v_i).

* Find (v^*_i) $(i = 1, 2, \ldots, m)$ so as to have:

$$UB(v^*) = \text{Min}\,\{UB(v) : v_i \geq 0 \;\text{ for } i = 1, 2, \ldots, m\}$$

* Lagrangian Dual Problem (exact or heuristic procedures)

Branching Strategies (1)

Start from the "root node" (level $h = 0$) and "examine" it.

a) Depth-First Strategy:

* ***Forward Step*: from each examined node at level h, generate the corresponding descendent nodes at level $(h + 1)$, and examine them in sequence;**

* ***Backtracking Step*: when all the descendent nodes at level $(h + 1)$ have been examined, consider the next not yet examined node at level h and examine it.**

Stop when level $h = 0$ is reached.

Generally: large number of nodes;

good feasible solutions found soon.

Branching Strategies (2)

Start from the “root node” (level $h = 0$) and “examine” it.

b) Highest-First Strategy:

* **from each “examined” node, generate the corresponding descendent nodes, and compute the associated Upper Bounds;**

* **examine the not yet examined node of the tree having the highest Upper Bound;**

* **stop when all the generated nodes have been examined.**

Generally: small number of nodes;

many generated nodes not yet examined.