Integer Programming: Lagrangian Relaxation

J. N. Hooker

MSC2000:90C10, 90C30

Article Outline

Keywords Lagrangian relaxation; Integer programming; Lagrangian dual; Lagrange multipliers; Branch and bound

Relaxation is important in optimization because it provides bounds on the optimal value of a problem. One of the more popular forms of relaxation is Lagrangian relaxation, which is used in integer programming and elsewhere.

A problem is relaxed by making its constraints weaker, so that the feasible set is larger, or by approximating the objective function. In the case of a minimization problem, the optimal value of the relaxation is a lower bound on the optimal value of the original problem. For a maximization problem it is an upper bound. The art of relaxation is to design a relaxed problem that is easy to solve and yet provides a good bound.

J.N. Hooker
Tepper School of Business, Carnegie Mellon University, Pittsburgh, USA

Purpose of Relaxation

Relaxation bounds are useful for two reasons. First, they can indicate whether a suboptimal solution is close to the optimum. If a minimization problem, for example, is hard to solve, one might settle for a suboptimal solution whose value is close to a known lower bound. An optimal solution would not be much better.

Second, relaxation bounds are useful in accelerating a search for an optimal solution. In a solution of an integer programming problem, for example, one normally solves a relaxation of the problem at each node of the *branch and bound* tree. Suppose again that the objective is to minimize. If the value of the relaxation at some node is greater than or equal to the value of a feasible solution found earlier in the search, then there is no point in branching further at that node. Any optimal solution found by branching further will have a value no better than that of the relaxation and therefore no better than that of the solution already found. Lagrangian relaxation is often used in this context, because it may provide better bounds than the standard linear programming (LP) relaxation.

Lagrangian Relaxation

Lagrangian relaxation is named for the French mathematician J.L. Lagrange, presumably due to the occurrence of what we now call Lagrange multipliers in his calculus of variations [2]. Because this form of relaxation changes the objective function as well as enlarging the feasible set, it is necessary to broaden the concept of relaxation somewhat.

Consider the problem of minimizing a function $f(x)$ subject to $x \in S$, where x is a vector of variables and S the set of feasible solutions. The *epigraph* of the problem is the set of all points (z, x) for which $x \in S$ and $z \geq f(x)$. This is illustrated in Fig. 1. The problem of minimizing $f'(x)$ subject to $x \in S'$ is a *relaxation* of the original problem if its epigraph contains the epigraph of the original problem. That is, (a) $S \subset S'$ and (b) $f(x) \leq f'(x)$ for all $x \in S$. Relaxation is therefore conceived as enlarging the epigraph; enlarging the feasible set is a special case. It is clear that the optimal value of a relaxation still provides a lower bound on the optimal value of the original problem.

Lagrangian relaxation is available for problems in which some of the constraints are inequalities or equations. Such problems may be written as

$$\text{minimize } f(x) \tag{1}$$

$$\text{subject to } g(x) \leq 0 \tag{2}$$

$$x \in S. \tag{3}$$

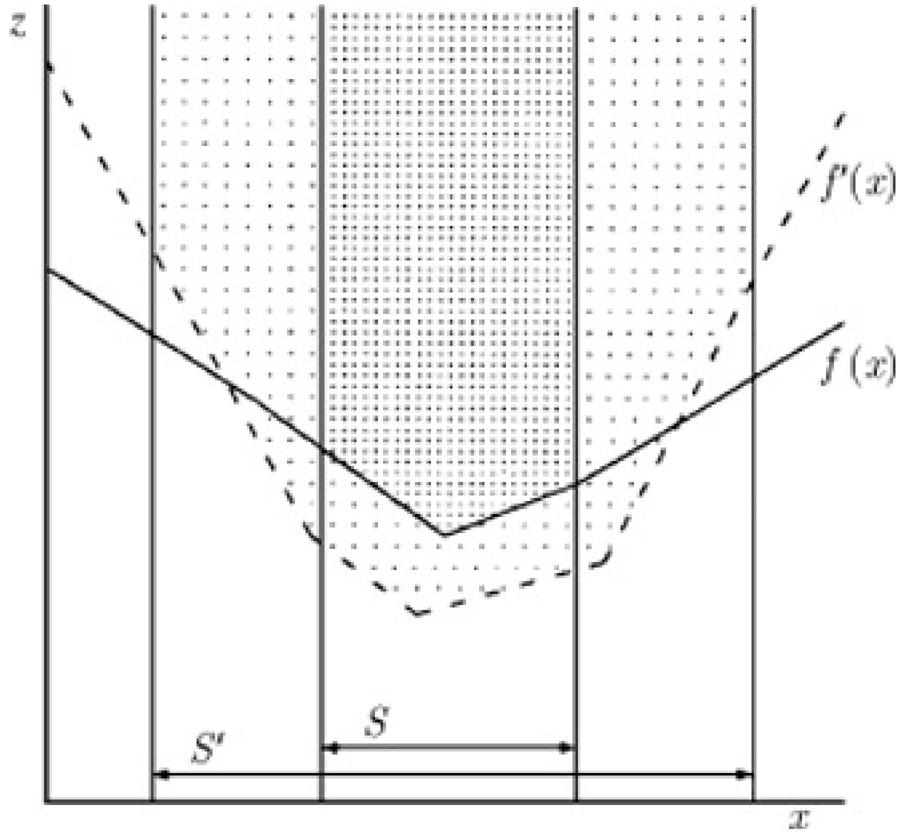

Figure 1 Epigraph of an optimization problem min $\{f(x): x \in S\}$ (darker shaded area) and of a relaxation min $\{f'(x): x \in S'\}$ (darker and lighter shaded areas)

Here, $g(x)$ is a vector of functions $(g_1(x), \ldots, g_m(x))$, and (2) is a family of m constraints $g_i(x) \leq 0$. There is no loss of generality in omitting equality constraints $h_i(x) = 0$ from this formulation, because they can be written as two inequality constraints, $h_i(x) \leq 0$ and $-h_i(x) \leq 0$. The constraints (3) may take any form, inequality or otherwise.

The *Lagrangian relaxation* is formed by 'dualizing' the constraints (2):

$$\begin{cases} \min & f(x) + \lambda g(x) \\ & x \in S. \end{cases} \tag{4}$$

Here, $\lambda = (\lambda_1, \ldots, \lambda_m)$ is a vector of nonnegative *Lagrange multipliers* that correspond to the inequality constraints. The aim of dualization is to remove the hardest constraints from the constraint set, so that the relaxed problem is relatively easy to solve.

The Lagrangian relaxation is in fact a relaxation because its epigraph contains the epigraph of the original problem (1)-(3). This can be verified by checking conditions a) and b):

(a) The feasible set of the original problem is a subset of the feasible set of the relaxation, because the relaxation omits some of the original constraints.
(b) If x is feasible in the original problem, then $f(x) \geq f(x) + \lambda g(x)$. This is because $\lambda \geq 0$ and, due to the feasibility of x, $g(x) \leq 0$.

The Lagrangian Dual

A relaxation can be constructed simply by eliminating the constraints (2) rather than dualizing them. One might ask what is the advantage of dualization. One rationale is that when the Lagrange multipliers are properly chosen, the penalties $\lambda_i g_i(x)$ in the objective function hedge against infeasibility. To the extent that constraints $g_i(x) \leq 0$ are violated and the bound thereby weakened, the objective function will be penalized, restoring the quality of the bound.

Fortunately, one can search for a proper choice of multipliers. The Lagrangian relaxation is actually a 'family' of relaxations, parameterized by the vector λ of multipliers. This provides the possibility of searching over values of λ to find a relaxation that gives a good lower bound on the optimal value.

The problem of finding the best possible relaxation bound is the *Lagrangian dual* problem. If $\theta(\lambda)$ is the optimal value of the relaxation (4), the Lagrangian dual of (1)–(3) is the problem of maximizing $\theta(\lambda)$ subject to $\lambda \geq 0$.

Under certain conditions, the best relaxation bound is equal to the optimal value of the original problem (1)–(3) [1]. Generally, however, it falls short. The amount by which it falls short is the *duality gap*.

The Lagrangian dual problem has three attractive features:

- It need not be solved to optimality. Any feasible solution provides a valid lower bound.
- Its objective function $\theta(\lambda)$ is always a concave function of λ. One need only find a local maximum, which is necessarily a global maximum as well.
- Its solutions have a *complementary slackness* property. If certain λ_i's are positive in an optimal solution of the dual problem, then the corresponding constraints $g_i(x) \leq 0$ are satisfied as equations in some optimal solution of the primal problem (1)–(3).

A serious drawback of the Lagrangian dual is that simply evaluating the objective function $\theta(\lambda)$ for a given λ normally requires solution of an optimization problem. The relaxation must be carefully chosen so that this is practical. Moreover the function θ is typically nondifferentiable.

Why is the Lagrangian dual a 'dual'? One explanation is that it generalizes the LP dual, which is the Lagrangian dual of an LP problem. To see this, consider the LP problem $\min\{cx : Ax \geq a,\ x \geq 0\}$. Its Lagrangian dual maximizes

$$\begin{aligned}\theta(\lambda) &= \min_{x\geq 0}\{cx + \lambda(a - Ax)\}\\ &= \min_{x\geq 0}\{(c - \lambda A)x + \lambda a\}\end{aligned}$$

over $\lambda \geq 0$. So $\theta(\lambda)$ is $-\infty$ if some component of $c - \lambda A$ is negative and is λa otherwise. This means that maximizing $\theta(\lambda)$ over $\lambda \geq 0$ is equivalent to maximizing λa subject to $\lambda A \leq c$ and $\lambda \geq 0$, which is precisely the LP dual.

Integer Programming

The application of Lagrangian ideas to integer programming dates back at least to H. Everett [6]. In this arena the optimization problem (1)-(3) becomes,

$$\begin{cases}\min & cx\\ \text{s.t.} & Ax \leq a\\ & Bx \leq b\\ & x_j \text{ integer for all } j.\end{cases} \tag{5}$$

The 'hard' constraints $Ax \leq a$ are dualized in the Lagrangian relaxation,

$$\begin{cases}\min & cx + \lambda(Ax - a)\\ \text{s.t.} & Bx \leq b\\ & x_j \text{ integer for all } j,\end{cases} \tag{6}$$

and $\theta(\lambda)$ is the minimum value of this problem for a given λ. The optimal value z_{LD} of the Lagrangian dual is a lower bound on the optimal value z_{IP} of (5). It will be seen shortly that the bound z_{LD} is at least as good as the bound z_{LP} obtained by solving the LP relaxation of (5). (The LP relaxation is the result of dropping the integrality constraints.)

In the context of integer programming, the Lagrangian function $\theta(\lambda)$ is not only concave but piecewise linear. This is because $\theta(\lambda)$ is the maximum of a set of linear functions $cx + \lambda(Ax - a)$ over all integral values of x that satisfy $Bx \leq b$.

A fundamental property of the Lagrangian dual is that z_{LD} is equal to the optimal value z_{C} of

$$\begin{cases} \min & cx \\ \text{s.t.} & Ax \le a \\ & x \in \text{conv}(S), \end{cases} \tag{7}$$

where S is the set of integer points satisfying $Bx \le b$, and conv(S) is the convex hull of S [8]. The Lagrangian dual can therefore be written as an LP problem, if a linear description of conv(S) is available.

The reasoning behind this fact goes as follows. Because the feasible set of (7) is that of an LP problem, the optimal value of its Lagrangian dual is equal to z_{C}. To see that it is also equal to z_{LD}, thereby proving $z_{\text{C}} = z_{\text{LD}}$, it suffices to show that the Lagrangian relaxation of (7) always has the same optimal value as the Lagrangian relaxation of (5). But this is true because the former is the same problem as the latter, except that the constraints in former are $x \in \text{conv}(S)$ and in the latter are $x \in S$. This substitution has no effect on the optimal value because the objective function is linear.

It can now be seen that the bound z_{LD} is always at least as good as z_{LP}. Let C_{IP} be the problem (7) corresponding to (5), and let C_{LP} be the problem (7) corresponding to the LP relaxation of (5). C_{LP}'s feasible set contains that of C_{IP}, and its optimal value is therefore less than or equal to z_{LD}. But because C_{LP} is identical to (5)'s LP relaxation, $z_{\text{LP}} \le z_{\text{LD}}$.

When $Bx \le b$ happens to describe a polyhedron whose vertices have integral coordinates, C_{IP} and C_{LP} are the same problem. In this case $z_{\text{LD}} = z_{\text{LP}}$. To sum up,

$$z_{LP} \le z_{LD} = z_C \le z_{IP},$$

where the first inequality is an equation when $Bx \le b$ describes an integral polyhedron.

As an example consider the integer programming problem (Fig. 2):

$$\begin{cases} \min & -2x_1 - x_2 \\ \text{s.t.} & 4x_1 + 5x_2 \le 10 \\ & 0 \le x_j \le 3 \\ & x_j \text{ integer}, \ j = 1, 2. \end{cases} \tag{8}$$

The optimal solution is $x = (2, 0)$, with value $z_{\text{IP}} = -4$. Dualizing the first constraint decouples the variables:

$$\begin{aligned} \theta(\lambda) &= \min_{\substack{0 \le x_j \le 3 \\ x_j \text{ integer}}} \{-2x_1 - x_2 + \lambda(4x_1 + 5x_2 - 10)\} \\ &= \min_{\substack{0 \le x_j \le 3 \\ x_j \text{ integer}}} \{(4\lambda - 1)x_1 + (5\lambda + 1)x_2 - 10\lambda\}. \end{aligned}$$

Because of the decoupling, $\theta(\lambda)$ is easily computed:

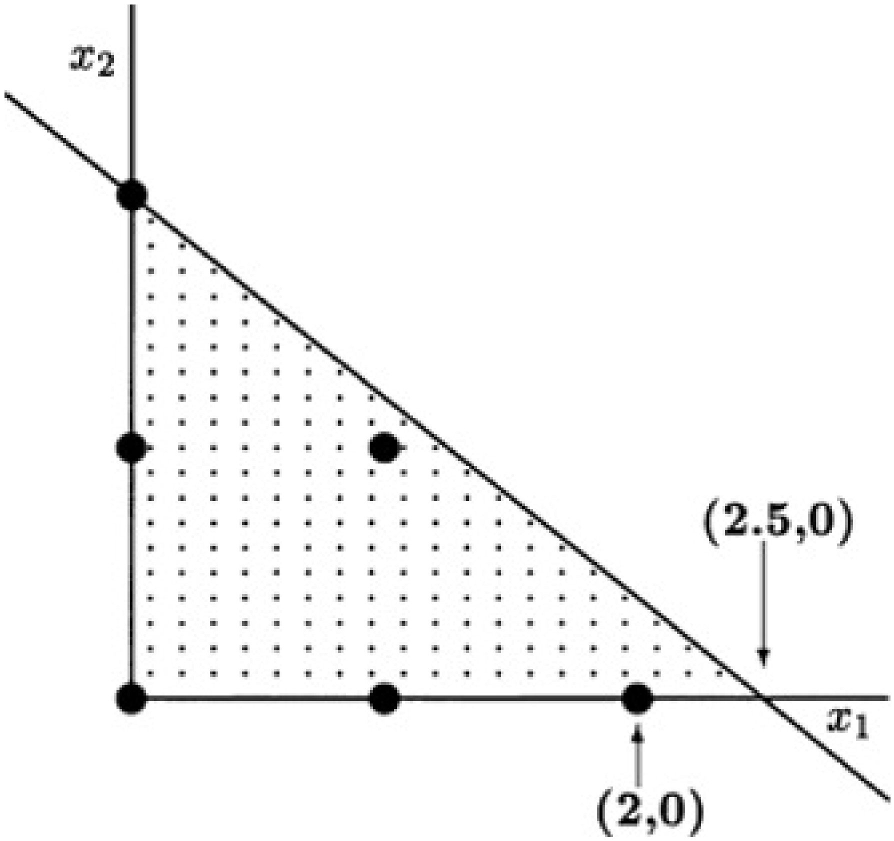

Figure 2 Feasible set of an integer programming problem (large dots) and its linear programming relaxation (area shaded by small dots). The point (2, 0) is the optimal solution, and (2.5, 0) is the solution of the LP relaxation

$$\theta(\lambda) = \begin{cases} 17\lambda - 9 & \text{if } 0 \leq \lambda \leq \frac{1}{5}, \\ 2\lambda - 6 & \text{if } \frac{1}{5} \leq \lambda \leq \frac{1}{2}, \\ -10\lambda & \text{if } \lambda \geq \frac{1}{2}. \end{cases}$$

It is evident in Fig. 3 that θ is a concave, piecewise linear function. The optimal value of the Lagrangian dual is $z_{\text{LD}} = \theta(\frac{1}{2}) = -5$, resulting in a duality gap of $z_{\text{IP}} - z_{\text{LD}} = 1$. The optimal value of the LP relaxation is likewise -5, so that in the present case $z_{\text{LP}} = z_{\text{LD}}$. This is predictable because $Bx \leq b$ consists of the bounds $0 \leq x_j \leq 3$, which define an integral polyhedron.

In practical applications, the Lagrangian relaxation is generally constructed so that it can be solved in polynomial time. It might be a problem in which the variables can be decoupled, as in the above example, or whose feasible set is an integral polyhedron. Popular relaxations include assignment or transportation problems, which can be solved quickly.

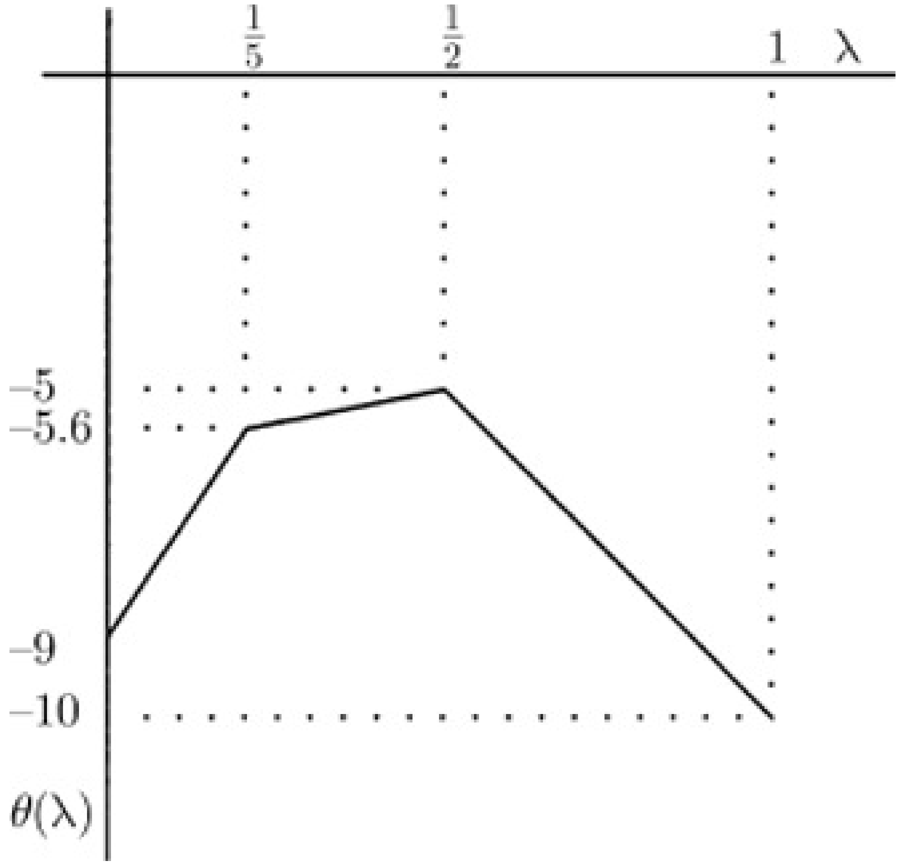

Figure 3 The Lagrangian function $\theta(\lambda)$ for an integer programming problem. The optimal value of the Lagrangian dual problem is $\theta(1/2) = -5$

A notable example is the *asymmetric traveling salesman problem* on n cities:

$$\text{minimize} \sum_{ij} c_{ij} x_{ij} \tag{9}$$

subject to

$$\sum_j x_{ij} = 1, \quad \text{all } i, \tag{10}$$

$$\sum_i x_{ij} = 1, \quad \text{all } j, \tag{11}$$

$$\sum_{i \notin V} \sum_{j \in V} x_{ij} \geq 1, \quad \text{all nonempty } V \subset \{2, \ldots, n\}, \tag{12}$$

$$x_{ij} \geq 0, \;\; x_{ij} \text{ integral}, \quad \text{all } i, j. \tag{13}$$

If the assignment constraints (10) are dualized, the Lagrangian relaxation minimizes

$$\begin{aligned} &\sum_{ij} c_{ij} x_{ij} + \sum_i \lambda_i (\sum_j x_{ij} - 1) \\ &\quad = \sum_{ij} (c_{ij} + \lambda_i) x_{ij} - \sum_i \lambda_i \end{aligned}$$

subject to (11)-(13). This is equivalent to finding a minimum-cost spanning arborescence that is rooted at node 1, which can be done in polynomial time [5]. This and related methods are discussed in [12, 19]. A Lagrangian approach to the symmetric traveling salesman problem is presented in [14, 15].

Solving the Dual

Subgradient optimization is a popular method for solving the Lagrangian dual, because subgradients of θ (and gradients when they exist) can be readily calculated.

Let $X(\bar{\lambda})$ be the set of optimal solutions of the Lagrangian relaxation (4) when $\lambda = \bar{\lambda}$. If $X(\bar{\lambda})$ is a singleton $\{\bar{x}\}$, then the gradient of θ at $\bar{\lambda}$ is simply the vector $g(\bar{x})$. This is because for values of λ in a neighborhood of $\bar{\lambda}$, $\theta(\lambda)$ is the linear function $f(\bar{x}) + \lambda g(\bar{x})$.

More generally, for any $\bar{x} \in X(\bar{\lambda})$, $g(\bar{x})$ is a subgradient of θ at $\bar{\lambda}$. In fact, every subgradient of θ at $\bar{\lambda}$ is a convex combination of subgradients that correspond to the solutions in $X(\bar{\lambda})$.

In the integer programming case, the subgradients of θ at $\bar{\lambda}$ are $A\bar{x} - a$ for each $\bar{x} \in X(\bar{\lambda})$, and convex combinations thereof. Consider the example (8), where

$$X(\lambda) = \begin{cases} \{(3,3)\} & \text{if } 0 \le \lambda < \frac{1}{5}, \\ \{(3,3),(3,0)\} & \text{if } \lambda = \frac{1}{5}, \\ \{(3,0)\} & \text{if } \frac{1}{4} < \lambda < \frac{1}{2}, \\ \{(3,0),(0,0)\} & \text{if } \lambda = \frac{1}{2}, \\ \{(0,0)\} & \text{if } \lambda > \frac{1}{2}. \end{cases}$$

Thus at $\lambda = 0$, θ has the gradient (slope) of $4(3) + 5(3) - 10 = 17$. At $\lambda = \frac{1}{5}$, the subgradients of θ are 17, 2, and their convex combinations; i. e., all slopes in the interval $[2, 17]$. This can be seen in Fig. 3.

The subgradient algorithm begins with an initial estimate λ_0 of the optimal λ. Iteration $k+1$ accepts estimate λ^k from the previous iteration and defines the next iterate as $\lambda^{k+1} = \lambda^k + \sigma_k g(x^k)$ where x^k is the value of x obtained by computing $\theta(\lambda^k)$, and $g(x^k)$ is the corresponding subgradient. Any negative components of λ^{k+1} are projected to zero. The stepsize σ_k must be carefully chosen to achieve timely convergence. The simplest alternative is to let $\sigma_k = 1/(k+1)$, but this results in extremely slow convergence. A widely used approach is Polyak's method [20], in which

$$\sigma_k = \frac{\theta^* - \theta(\lambda^k)}{\|g(x^k)\|^2}\alpha_k$$

and in which θ^* is a known upper bound on $\max_{\lambda \ge 0}\{\theta(x)\}$. Initially $\alpha_0 = 1$, and $\alpha_k = \alpha_{k-1}$ unless $\theta(x^k)$ has not improved for several iterations, in which case one might set $\alpha_k = \frac{1}{2}\alpha_{k-1}$. The choice of an efficient stepsize is problem dependent and typically found by trial and error. Other versions of the subgradient algorithm include Nesterov's smoothing scheme [18] and the Kelly-Cheney-Goldstein bundle method [17].

Further Reading and Extensions

A lucid geometrical exposition of Lagrangian relaxation may be found in Chapter 6 of [1]. A classic treatment of its application to integer programming is [7]. A tutorial can be found in [13], a more recent exposition in Section 8.1 of [4], and a recent literature survey in [3]. There is a vast literature on applications and enhancements.

The Lagrangian dual can be viewed as a special case of a *relaxation dual*, which is any dual defined over a parameterized family of relaxations [16]. The *surrogate dual*, for example, is a relaxation dual in which relaxations are obtained by replacing the original inequality constraints with a nonnegative linear combination of those constraints [9, 10, 11]. In this case, the relaxations are parameterized by the vector of multipliers used to obtain the linear combination.

See also

Branch and Price: Integer Programming with Column Generation
Decomposition Techniques for MILP: Lagrangian Relaxation
Integer Linear Complementary Problem
Integer Programming
Integer Programming: Algebraic Methods
Integer Programming: Branch and Bound Methods
Integer Programming: Branch and Cut Algorithms
Integer Programming: Cutting Plane Algorithms
Integer Programming Duality
Lagrange, Joseph-Louis
Lagrangian Multipliers Methods for Convex Programming
LCP: Pardalos-Rosen Mixed Integer Formulation
Mixed Integer Classification Problems
Multi-objective Integer Linear Programming
Multi-objective Mixed Integer Programming
Multi-objective Optimization: Lagrange Duality
Multiparametric Mixed Integer Linear Programming
Parametric Mixed Integer Nonlinear Optimization
Set Covering, Packing and Partitioning Problems
Simplicial Pivoting Algorithms for Integer Programming
Stochastic Integer Programming: Continuity, Stability, Rates of Convergence
Stochastic Integer Programs
Time-dependent Traveling Salesman Problem

References

1 Bazaraa MS, Sherali HD, Shetty CM (2006) Nonlinear Programming: Theory and Algorithms, 3rd edn. Wiley, New York
2 Boyer CB (1985) A history of mathematics. Princeton Univ. Press, Princeton
3 Bragin MA (2023) Survey in Lagrangian relaxation for MILP: Importance, challenges, historical review, recent advancements, and opportunities (2023) Ann Oper Res, published online
4 Conforti M, Cornuéjols G, Zambelli G (2014) Integer Programming. Springer, Berlin
5 Edmonds J (1967) Optimum branchings. J Res Nat Bureau Standards (B) 71:233–240
6 Everett III H (1963) Generalized Lagrange multiplier method for solving problems of optimum allocation of resources. Oper Res 11:399–417

7 Fisher ML (1981) The Lagrangean relaxation method for solving integer programs. Managem Sci 27:1–18
8 Geoffrion AM (1974) Lagrangean relaxation for integer programming. Math Program Stud 2:82–114
9 Glover F (1968) Surrogate constraints. Oper Res 16:741–749
10 Glover F (1975) Surrogate constraint duality in mathematical programming. Oper Res 23:434–451
11 Greenberg H, Pierskalla WP (1970) Surrogate mathematical programming. Oper Res 18:924–939
12 Grötschel M, Padberg MW (1985) Polyhedral theory. In: Lawler EL et al (eds) The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization. Wiley, New York, pp 251–305
13 Guinard M (1995) Lagrangean relaxation: A short course. JORBEL 35:5–21
14 Held M, Karp RM (1970) The traveling-salesman problem and minimum spanning trees. Oper Res 18:1138–1162
15 Held M, Karp RM (1971) The traveling-salesman problem and minimum spanning trees: Part II. Math Prog 1:6–25
16 Hooker JN (2000) Logic-based methods for optimization: Combining optimization and constraint satisfaction. Wiley, New York
17 Lemaréchal C (2001) Lagrangian relaxation. In: Jünger M, Naddef D (eds) Computational Combinatorial Optimization. Springer, Berlin, pp 112–156
18 Nesterov YE (2005) Smooth minimization of non-smooth functions. Math Prog A 103, 127–152
19 Padberg MW, Grötschel M (1985) Polyhedral computations. In: Lawler EL et al (eds) The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization. Wiley, New York, pp 307–360
20 Polyak BT (1987) Introduction to Optimization (translated from Russian). Optimization Software Inc, New York