5 LAGRANGIAN RELAX-AND-CUT ALGORITHMS

Abilio Lucena[1]

[1]Departamento de Administração
Universidade Federal do Rio de Janeiro
Rio de Janeiro, Brazil
lucena@facc.ufrj.br

Abstract: Attempts to allow exponentially many inequalities to be candidates to Lagrangian dualization date from the early 1980s. In the literature, the term *Relax-and-Cut* is being used to denote the whole class of Lagrangian Relaxation algorithms where Lagrangian bounds are attempted to be improved by dynamically strengthening relaxations with the introduction of valid constraints (possibly selected from families with exponentially many constraints). In this chapter, Relax-and-Cut algorithms are reviewed in their two flavors. Additionally, a general framework to obtain feasible integral solutions (that benefit from Lagrangian bounds) is also presented. Finally, the use of Relax-and-Cut is demonstrated through an application to a *hard-to-solve* instance of the Knapsack Problem. For that application, Gomory cuts are used, for the first time, within a Lagrangian relaxation framework.

Keywords: Relax-and-cut, Lagrangian relaxation, cutting planes, knapsack problem.

5.1 INTRODUCTION

Attempts to allow exponentially many inequalities to be candidates to Lagrangian dualization date from the early 1980s. A short list of selected contributions in this area, in chronological order, is initiated with the *Restricted Lagrangian Approach* of Balas and Christofides (1981) for the Traveling Salesman Problem. Later on, Gavish (1985) suggested an *Augmented Lagrangian Approach* to solve a Centralized Network Design Problem. A contribution by Lucena (1992; 1993) follows with a scheme to dualize *violated* inequalities *on the fly*, as they become violated at the solution to a valid Lagrangian Relaxation of the problem being solved. Almost simultaneously, Escudero et al. (1994) proposed an algorithm to solve the Sequential Ordering Problem with Precedence Constraints. The authors called that algorithm *Relax-and-Cut*. In a more recent development, Barahona and Ladányi (2001) proposed the use of the Volume Algorithm Barahona and Anbil (2000) as an alternative to the use of the Simplex method (or Interior Point algorithms) in implementations of Branch-and-Cut algorithms (Padberg and Rinaldi, 1991). Finally, Ralphs and Galati (2003) describe a framework where various decomposition methods (including Lagrangian relaxation) and polyhedral cutting plane algorithms can be viewed from a common theoretical perspective.

Following Lucena (2004), we use the term Relax-and-Cut to refer not only to the algorithm in Escudero et al. (1994) but to a much larger class of algorithms which includes that one. According to that guideline, a Relax-and-Cut algorithm must be Lagrangian based. Furthermore, Lagrangian bounds must be attempted to be improved by dynamically strengthening relaxations with the introduction of valid constraints (possibly selected from families with exponentially many constraints). Finally, strengthening constraints may or may not be explicitly dualized.

The definition of Relax-and-Cut above is broad enough to include all algorithms in Balas and Christofides (1981), Gavish (1985), Lucena (1992), Lucena (1993), Escudero et al. (1994), and Barahona and Ladányi (2001). These algorithms, however, differ significantly among themselves. Differences are particularly pronounced in the way strengthening constraints are treated. For instance, in Gavish (1985) and Escudero et al. (1994), strengthening constraints are only used after Lagrangian Dual Problems are solved (and corresponding best possible Lagrangian bounds are obtained). Differently from that, strengthening constraints in Lucena (1992; 1993) are identified and used after every Lagrangian Relaxation Problem is solved.

As for any Lagrangian Relaxation algorithm, Relax-and-Cut is initiated with a relaxation of a given model where a set of *complicating constraints* is dualized while remaining constraints are *kept* (Guignard, 2004). Following the nomenclature introduced in Lucena (2004), *Delayed Relax-and-Cut* (DRC) algorithms (Lucena, 2004) then proceed by solving the corresponding Lagrangian Dual Problem. Valid constraints which violate the solution to that problem are then identified and may be either dualized or else *kept*. Either way, a new Lagrangian Dual Problem is formulated and solved and the procedure continues until a stopping criteria is reached. DRC was introduced in Escudero et al. (1994) and a number of DRC applications are discussed in Guignard (1998). These include the Asymmetric Traveling Salesman Problem, the Generalized Assignment Problem, and the Multiple Choice Knapsack Problem.

As mentioned above, Relax-and-Cut algorithms in Lucena (1992; 1993) do not *delay* the identification and use of violated constraints until the Lagrangian Dual Problem is solved. Differently from DRC, violated inequalities are attempted to be identified (and are dualized, in case of success) for every Lagrangian Relaxation Problem eventually solved. That variant of Relax-and-Cut is called *Non Delayed Relax-and-Cut* (NDRC) in Lucena (2004). NDRC was first proposed and successfully used for the Steiner Problem in Graphs in Lucena (1992; 1993). Later on, it was used for the Edge-Weighted Clique Problem (Hunting et al., 2001), the Quadratic Knapsack Problem (de Moraes Palmeira et al., 1999), the Traveling Salesman Problem (Belloni and Lucena, 2000), the Vehicle Routing Problem (Martinhon et al., 2004), the Linear Ordering Problem (Belloni and Lucena, 2003), the Rectangular Partition Problem (Calheiros et al., 2003), and the Capacitated Minimum Spanning Tree Problem (da Silva, 2002).

Assuming that the increasingly reinforced Lagrangian Dual Problems are solved to optimality, theoretical convergence of DRC algorithms is guaranteed (to a bound at least as good as that given by the Linear Programming (LP) relaxation of the original formulation, reinforced with the additional families of valid inequalities used within the Lagrangian framework). Recently, Belloni and Sagastizábal (2004) obtained a similar result, under not very restrictive conditions, for a Bundle Method implementation of NDRC.

Starting with the pioneering work of Everett III (1963) and Held and Karp (1970; 1971), a vast literature exists on Lagrangian relaxation. Among these, references such as Geoffrion (1974), Shapiro (1974; 1979), Fisher (1981), Beasley (1993), Lemaréchal (2001), and Guignard (2004) are highly relevant. On the other hand, Relax-and-Cut is a relatively recent development and surveys on it can be found in Guignard (1998), Guignard (2004), and Lucena (2004).

In this chapter, Relax-and-Cut algorithms are reviewed in their two variants, namely NDRC and DRC. In addition, the effectiveness of the approach is demonstrated in an application to a *hard-to-solve* instance of the Knapsack Problem (KP). That instance involves coefficients of the order of 10^{15}, which makes it considerably more challenging to solve than ordinary KP instances. Throughout that application, Gomory cuts (Gomory, 1963) are used, for the first time, within a Lagrangian relaxation framework.

NDRC and DRC algorithms are discussed, respectively, in Sections 5.2 and 5.3. A framework for generating primal integral solutions (which benefit from Lagrangian dual information) is presented in Section 5.4. The generation of Gomory cuts for KP follows in Section 5.5. The KP instance used as an example (for the application of NDRC and DRC) is also introduced in that section. In Section 5.6, the generic *dual* algorithms of Sections 5.2 and 5.3 and the generic *primal* algorithm of Section 5.4 are specialized to KP. The use of these algorithms is illustrated with a numerical example. Finally the chapter is closed in Section 5.7 with suggestions for future work.

5.2 NON DELAYED RELAX AND CUT

The NDRC algorithm in Lucena (1992; 1993) is based upon the use of SM and, throughout this chapter, we follow Lucena (1992; 1993; 2004) in using SM to describe and test NDRC. For the sake of clarity, the material presented in this section is focused on binary 0-1 problems. However, with the exception of Subsection 5.4, it generalizes to Mixed Integer Programming problems.

Assume that a formulation for a $\mathcal{NP}$-hard combinatorial optimization problem is given. Assume as well that exponentially many inequalities are included in it. Such a formulation can be generically described as

$$\max\{cx : \ Ax \leq b, \ \ x \in X\}, \tag{5.1}$$

where, for simplicity, x denotes binary $0-1$ variables (i.e. $x \in \mathbb{B}^n$, for positive integral values of n). Accordingly, for positive integral values of m, we have $c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$, $A \in \mathbb{R}^{m\times n}$ and $X \subseteq \mathbb{B}^n$. Assume, as it is customary in Lagrangian relaxation, that

$$\max\{cx : \ \ x \in X\} \tag{5.2}$$

is an *easy* problem to solve. On the other hand, in what is unusual for the application of Lagrangian relaxation, assume that m is an exponential function of n, i.e. (5.1) contains exponentially many inequalities. Assume as well that one dualizes

$$\{a_i x \leq b_i : \quad i = 1,2,\dots,m\} \tag{5.3}$$

in a Lagrangian fashion (regardless of the difficulties associated with the dualization of exponentially many inequalities) and let $\lambda \in \mathbb{R}^m_+$ be the corresponding vector of Lagrangian multipliers. A valid upper bound on (5.1) is given by the solution to the *Lagrangian Relaxation Problem*

$$LRP(\lambda) = \max\{(c - \lambda A)x + \lambda b : \quad x \in X\}. \tag{5.4}$$

Subgradient Optimization (SO) could then be used to solve the corresponding *Lagrangian Dual Problem*

$$LDP = \min_{\lambda \in \mathbb{R}^m_+} \{LRP(\lambda)\} \tag{5.5}$$

and obtain the best possible Lagrangian bound on (5.1). Optimization is typically conducted here in an interactive way with multipliers being updated so that (5.5) is attained. For the sake of completeness, let us briefly review SM, as implemented in Fisher (1981). That implementation is precisely the one which is adapted in this paper to produce the computational results in the following sections.

5.2.1 *A brief description of the Subgradient Method*

At iteration k of SM, for a feasible vector λ^k of Lagrangian multipliers, let $\bar{x}$ be an optimal solution to $LRP(\lambda^k)$ and z_{lb} be a known lower bound on (5.1). Additionally, let $g^k \in \mathbb{R}^m$ be a subgradient associated with the relaxed constraints at $\bar{x}$. Corresponding, entries for g^k are

$$g^k_i = (b_i - a_i\bar{x}), \quad i = 1,2,\dots,m. \tag{5.6}$$

In the literature (see Fisher (1981), for instance), to update Lagrangian multipliers, one generates first a *step size*

$$\theta^k = \frac{\alpha[LRP(\lambda^k) - z_{lb}]}{\sum\limits_{i=1,\dots,m} (g^k_i)^2}, \tag{5.7}$$

where α is a real number assuming values in $(0,2]$, and then computes

$$\lambda^{k+1}_i \equiv \max\{0; \lambda^k_i - \theta^k g^k_i\}, \quad i = 1,\dots,m. \tag{5.8}$$

After Lagrangian multipliers are updated, one moves on to iteration $k+1$ of SM.

Under the conditions imposed here, the straightforward use of updating formulas (5.7)–(5.8) is not as simple as it might appear. The reason being the exceedingly large number of inequalities that one would have typically to deal with.

5.2.2 NDRC modifications to the Subgradient Method

Inequalities in (5.3), at iteration k of SM, may be classified into three sets. The first one contains inequalities that are violated by $\bar{x}$. The second set is for those inequalities that have nonzero multipliers currently associated with them. Notice that an inequality may be, simultaneously, in the two sets just defined. Finally, the third set consists of the remaining inequalities. Following Lucena (2004), we will refer to these sets of inequalities respectively as the *Currently Violated Active* set, the *Previously Violated Active* set, and the *Currently Inactive* set. Accordingly, they are respectively denoted by $CA(k)$, $PA(k)$, and $CI(k)$.

For the traditional use of Lagrangian relaxation, say when m is a polynomial function of n, Beasley (1993) reported *good practical convergence* of SM to (5.5), while, arbitrarily setting $g_i^k = 0$ whenever $g_i^k > 0$ and $\lambda_i^k = 0$, for $i \in \{1,\ldots,m\}$. In our context, all subgradient entries that are candidates to that modification belong to $CI(k)$.

In spite of the exponentially many inequalities one is faced with, we follow Beasley's advice, The reasoning behind the SM modifications suggested above comes from two observations. The first one is that, irrespective of the suggested changes, from (5.8), multipliers for $CI(k)$ inequalities would not change their present null values at the end of the current SM iteration. Clearly, $CI(k)$ inequalities do not directly contribute to Lagrangian costs (at the current SM iteration). On the other hand, they do play a decisive role in determining the value of θ^k and this fact brings us to the second observation. Typically, for the application being described, the number of strictly positive subgradient entries associated with $CI(k)$ inequalities, tends to be huge. If all these subgradient entries are explicitly used in (5.7), the value of θ^k would result extremely small, leaving multiplier values virtually unchanged from iteration to iteration and SM convergence problems should be expected.

By following Beasley's suggestion, we are capable of dealing adequately, within a SM framework, with the exceedingly large number inequalities in $CI(k)$. However, we may still face problems arising from a potentially *large* number inequalities in $(CA(k) \setminus PA(k))$, i.e. those inequalities that will become effectively dualized (i.e. have a nonzero multiplier associated with them) at iteration k of SM.

Assume that a *large* number of inequalities exist in $(CA(k) \setminus PA(k))$. These inequalities must therefore be violated at the solution to $LRP(\lambda^k)$ and have zero valued Lagrangian multipliers currently associated with them. Typically, such inequalities may be partitioned into subsets (associated, for instance, with a partitioning of the set of vertices in an associated graph, if that applies) and, according to some associated criteria, a maximal inequality would exist for each subset. In order to avoid repeatedly penalizing the same variables, again and again, we only dualize one maximal inequality per subset of inequalities. Remaining inequalities in $(CA(k) \setminus PA(k))$ have their subgradient entries arbitrarily set to 0, thus becoming, in effect, inequalities of $CI(k)$. Examples of applications where the situation described above prevails, are discussed in Lucena (2004). However, for the application considered in Section 5.6, only Gomory cuts are used and, at every SM iteration, there will always be at most one fractional variable to generate Gomory cuts from.

One should notice that, under the classification proposed above, inequalities may change groups from one SM iteration to another. It should also be noticed that the only multipliers that may directly contribute to Lagrangian costs $(c + \lambda^{k+1}A)$, at the end of iteration k of SM, are the ones associated with *active* inequalities, i.e. inequalities in $(CA(k) \cup PA(k))$.

An important step, in the scheme outlined above, is the identification of inequalities violated at $\bar{x}$. That problem must be solved at every iteration of SM and is equivalent to the separation problems found in Branch and Cut algorithms. However, NDRC separation problems typically involve a lower complexity than its Branch and Cut counterparts. This follows from the fact that LRP is normally chosen so that one will be separating over integral structures. For the application in Section 5.6, although one will not be working with integral structures, separation (of Gomory cuts) is quite straightforward.

According to Lucena (2004), an NDRC algorithm could be seem as a *Traditional Lagrangian Relaxation* (TLR) algorithm where exponentially many inequalities are dualized but subgradients are *projected*, at every iteration of SM, into a smaller space, i.e. the space implied by *active* inequalities (formed by inequalities in $(CA(k) \cup PA(k))$). In doing so, only a fraction of the exponentially many inequalities involved are explicitly considered, at every SM iteration, to update Lagrangian multipliers. An analog of that idea, in terms of LP based algorithms, is to find the LP relaxation of a formulation involving exponentially many inequalities. Clearly, only a *tiny* fraction of these inequalities are tight at a LP relaxation solution. Furthermore, in practice, only a *few* of these inequalities would be explicitly used in a cutting plane algorithm to attain LP bounds.

In another interpretation in Lucena (2004), NDRC may be seen as a Lagrangian relaxation algorithm where exponentially many candidate inequalities are dualized *on the fly*, as they become violated at an optimal solution to $LRP(\lambda^k)$. Since inequalities may be dualized for every LRP (and not only for LDP), an analog of the idea, in terms of LP based algorithms, generates cutting planes as the LP is being solved.

5.3 DELAYED RELAX AND CUT

Description of a DRC algorithm is quite straightforward. Assume that an initial LDP is solved and let $\bar{x}$ be the corresponding LDP optimal solution. Violated inequalities associated with $\bar{x}$ are then separated and are either kept or else dualized, thus giving rise to a new (strengthened) LDP (see Guignard (1998) for a discussion on efficient cuts in DRC). The new LDP is then solved and the procedure is repeated until some stopping criteria is met (for instance, until either optimality is proven or else the maximum number of allowed LDP solving rounds is reached). DRC could be seen as a Lagrangian relaxation analog of LP based cutting plane algorithms.

An important consideration in designing a DRC algorithm is the definition of an initial LDP to solve. In general, a trade off exists between LDP bound strength and the CPU time required to solve the problem.

An *easy to solve* LDP would typically return a *weak* bound, requiring, however, *few* SM (or Bundle Method, or Volume Algorithm) iterations to be solved. On the other hand, a *hard to solve* LDP would typically return a stronger bound requiring, however, a *large* number of SM (or Bundle Method, or Volume Algorithm) iterations to be solved. That issue is treated in more detail in Lucena (2004), where an application of

DRC to the Steiner Tree Problem is carried out. In the context of the KP application in Section 5.6, an *easy to solve* LDP is used. Such an LDP contains only one constraint, i.e. the knapsack inequality. An example of a *harder to solve* LDP would be, for instance, one containing one or more Lifted Minimum Cover Inequalities (see Wolsey (1998), for instance) associated with the LP relaxation of KP.

5.4 LAGRANGIAN HEURISTICS

Assume that a given basic heuristic, denoted here BH, is available for generating feasible solutions to (5.1). We typically call BH, for the first time, prior to initializing SM. Additional calls are made alongside some of the iterations of SM (or Bundle Method or Volume Algorithm). For the first call of BH, the original costs c are used. Remaining calls are performed either under the available Lagrangian modified costs $(c - \lambda^k A)$, or else under costs given by $(1 - \bar{x})c$ (where, $\bar{x}$, as before, is an optimal solution to $LRP(\lambda^k)$). Costs $(1 - \bar{x})c$ attempt to make it attractive to BH to select variables set to 1 in $\bar{x}$. After a feasible solution to (5.1) is generated, either under Lagrangian costs or else under costs $(1 - \bar{x})c$, the actual value for that feasible solution must be computed (under the original costs c). For the applications we have so far considered, the use of costs $(1 - \bar{x})c$ has proved, in most cases, more effective than the use of Lagrangian modified costs.

The motivation for the Lagrangian Heuristic (LH) sketched above, is the common sense belief that dual information must obviously be relevant to primal heuristics.

As it is the case for any non exact solution algorithm, feasible solutions generated by LH may also be attempted to be improved through local search procedures.

Ideally, it is desirable that BH be fast, to allow a large number of calls to be made to it. In our experience, in most cases, a simple greedy heuristic suffices to eventually return (alongside the application of SM) *good quality* feasible solutions to problem (5.1).

So far, LH have been specifically tailored to the Steiner Problem in Graphs (Lucena, 1992; 1993), the Traveling Salesman Problem (Belloni and Lucena, 2000), the Vehicle Routing Problem (Martinhon et al., 2004), the Linear Ordering Problem (Belloni and Lucena, 2003), the Rectangular Partition Problem (Calheiros et al., 2003), and the Capacitated Minimum Spanning Tree Problem (da Silva, 2002).

5.5 GOMORY CUTS FOR THE 0-1 KNAPSACK PROBLEM

The $0 - 1$ KP (see Martello and Toth (1997) and Pisinger (1997) for solution algorithms for the problem) is formulated as

$$z = \max \sum_{j=1,\ldots,n} c_j x_j \tag{5.9}$$

subject to

$$\sum_{j=1,\ldots,n} a_j x_j \leq b, \tag{5.10}$$

$$x \in \mathbb{B}^n, \tag{5.11}$$

where coefficients $\{a_j : j = 1,\dots,n\}$ and b are positive integers. The analogy normally associated with the problem is that of filling a knapsack of capacity b with items selected from among n objects with capacities $\{a_i : i = 1,\dots,n\}$. Items should be selected in order to maximize value (as expressed by the objective function) of knapsack load.

The LP relaxation of (5.9)–(5.11) is obtained by relaxing integrality enforcing constraints (5.11), thus resulting in

$$\bar{z} = \max \sum_{j=1,\dots,n} c_j x_j \tag{5.12}$$

subject to

$$\sum_{j=1,\dots,n} a_j x_j \le b, \tag{5.13}$$

$$0 \le x_j \le 1, \quad j = 1,\dots,n. \tag{5.14}$$

For the purposes of this chapter, it is more convenient to explicitly associate slack variables $\{x_{n+j} : j = 1,\dots,n\}$ to the right most inequalities in (5.14), associate slack variable x_{2n+1} to inequality (5.13) and rewrite (5.12)–(5.14) as

$$z = \max \sum_{j=1,\dots,n} c_j x_j \tag{5.15}$$

subject to

$$x_j + x_{n+j} = 1, \quad j = 1,\dots,n, \tag{5.16}$$

$$\sum_{j=1,\dots,n} a_j x_j + x_{2n+1} = b, \tag{5.17}$$

$$x_j \ge 0, \quad j = 1,\dots,n. \tag{5.18}$$

Following Dantzig (1957), an optimal solution to (5.12)–(5.14) is straightforward to compute. Assume that the coefficients $\{c_j : \ j = 1,\dots,n\}$ are ordered so that

$$c_1/a_1 \ge c_2/a_2 \ge \dots \ge c_n/a_n \tag{5.19}$$

and that j^* is the largest integer for which

$$\sum_{j=1,\dots,j^*} a_j \le b. \tag{5.20}$$

An optimal solution to (5.12)–(5.14) is then given by

$$\bar{x}_j = 1, \quad j = 1,\dots,j^*, \tag{5.21}$$

$$\bar{x}_j = 0, \quad j = j^*+2,\dots,n, \tag{5.22}$$

$$\bar{x}_{j^*+1} = (b - \sum_{j=1,\dots,j^*} a_j)/a_{j^*+1}). \tag{5.23}$$

5.5.1 *LP relaxation basis*

Let us now translate solution (5.21)–(5.23) in terms of an optimal basis to (5.15)–(5.18). For each of the first j^* rows in (5.16), the corresponding basic variable is x_j, $j = 1, \ldots, j^*$. Accordingly, $x_{(n+j)}$, for $j = (j^* + 2), \ldots, n$, is basic for the last $n - (j^* + 1)$ rows in (5.16). If $\bar{x}_{(j^*+1)} > 0$ then the basic variable associated with the $(j^* + 1)$-th row in (5.16) is $x_{(j^*+1)}$ while the basic variable associated with row (5.17) is $x_{(n+j^*+1)}$. Otherwise, if $\bar{x}_{(j^*+1)} = 0$ then $x_{(n+j^*+1)}$ is the basic variable associated with the $(j^* + 1)$-th row in (5.16) and $x_{(2n+1)}$ is the basic variable associated with row (5.17).

Whenever $\bar{x}_{(j^*+1)} > 0$, a fractional Gomory cut may be generated from the $(j^* + 1)$-th row in (5.16). Furthermore, such a cut may be obtained with very few pivoting operations. More specifically, consider the rows in (5.16) for which a variable x_j, $j = 1, \ldots, n$, is basic. Each of these rows should be multiplied by their corresponding a_j and then be subtracted from row (5.17). One should then select x_{n+j^*} as the pivot for the updated row (5.17) and from that eliminate the variable from the $(j^* + 1)$-th row in (5.16). A fractional Gomory cut is then generated from the updated row j^* in (5.16). Such a cut may clearly involve slack variables in $\{x_{(n+j)} : j = 1, \ldots, n\}$ and one should then use constraints (5.16) and (5.17) to rewrite the cut only in terms of the original variables.

It is not difficult to verify that, due to the particular structure of (5.16)-(5.18), it is always possible to generate Gomory cuts where coefficients are integral valued.

5.5.2 *Knapsack instance*

Table 5.1 describes the coefficients $\{c_j : 1, \ldots, n\}$ and $\{a_j : 1, \ldots, n\}$ associated with a particular $0 - 1$ KP instance. For that instance, the RHS coefficient in (5.10) is $b = 12107067865319564$. One should notice that such an instance involves coefficients of the order of 10^{15}. Coefficients of such a magnitude preclude the use of Dynamic Programming (DP) recursions to attain optimality since the associated state space would be out of reach of currently available computer memory. Likewise, reinforcing Linear Programming (LP) relaxation (5.12)–(5.14) with violated valid inequalities, in an LP based cutting plane approach, would probably not be an option. That applies since, after generating a few, say Gomory cuts, one would, most likely, run into numerical problems while performing matrix inversions. Finally, a 0.000001% gap between upper and lower bounds tends to be, for ordinary applications, tight enough to provide an optimality certificate. However, for our KP instance such a gap corresponds to millions of units and is by no means a guarantee of optimality.

For the numerical results in Section 5.6, coding was carried out in FORTRAN. In a 32 bits computer, like the one we use, the largest integer variable one is capable of representing, may contain only 8 decimal digits. We have thus used floating point variables to represent the coefficients of the KP instance in Table 5.1. Great care was then taken to minimize the possibilities of incurring in rounding off errors. That involved an extensive use of FORTRAN compiler intrinsic functions that round down a real valued number to the real number which represents the resulting integer. Whenever one of the coefficients of the KP instance had to be explicitly considered, it was

Table 5.1 KP instance

j	c_j	a_j	j	c_j	a_j
1	598588764667512	10437589138747	26	796893775463105	459233075380326
2	838245868682862	20572869107128	27	270887374877930	492233395576478
3	12796364724637	38212530314923	28	945141553878785	502445280551911
4	427655428647996	49889404326678	29	648177325725556	525000452995301
5	740525364875794	54476898163558	30	145332604646683	535258352756501
6	578473687171937	62906369566918	31	445022046566010	595017075538636
7	157311424612999	81033669412137	32	275523483753205	605881750583649
8	313272386789322	198077023029328	33	636593878269196	647979378700257
9	368257552385331	200995206832886	34	267737418413163	666921436786652
10	877835929393769	217693403363228	35	141873389482499	670142650604249
11	885035455226899	271198809146882	36	196074530482293	677149176597596
12	217146083712578	281886577606202	37	150113984942437	694714903831482
13	272873789072037	299474477767945	38	204226523637772	704419076442719
14	522032976150513	300442874431611	39	559784591197968	723579525947571
15	673082530498505	314126789569855	40	442578673362732	728680551052094
16	556192934513093	327096879482270	41	195826128125191	744018435478211
17	994231045246125	340778887271882	42	692465543746949	749621152877808
18	235551774501801	354439347982407	43	567489504814149	781238436698914
19	864161729812623	376186668872834	44	942473530769349	805823385715485
20	464697897434235	377783030271531	45	444463163614274	887850999832154
21	453489422798157	392820507287980	46	518855929374695	893622457981110
22	193823680281640	431183815002442	47	460040390491486	924746572971345
23	748618841171265	440109968185425	48	475286751985550	935340881347657
24	180304899811745	444615721702576	49	649530351161957	937711596488953
25	468244463205338	450100183486939	50	594328463077546	988966226577759

first increased by a conveniently small tolerance and then rounded down as suggested above.

5.6 GOMORY BASED RELAX-AND-CUT ALGORITHMS FOR KP

Assuming that $p \geq 1$ Gomory cuts are currently dualized, the Lagrangian Relaxation Problem one will be faced with, either under NDRC or DRC, at any SM iteration, is given by

$$\mathrm{LRP}(\lambda) = \max \sum_{j=1,\ldots,n} c'_j x_j + \mathrm{const}(\lambda) \tag{5.24}$$

subject to

$$\sum_{j=1,\ldots,n} a_j x_j \leq b, \tag{5.25}$$

$$0 \leq x_j \leq 1, \quad j = 1,\ldots,n, \tag{5.26}$$

where $\lambda \in \mathbb{R}^p_+$ is a vector of Lagrangian multipliers associated with dualized Gomory cuts, const(λ) is a non negative constant associated with these multipliers and $\{c'_j : j = 1,\ldots,n\}$ are the corresponding Lagrangian modified objective function coefficients. Clearly, if no dualized Gomory cut exists, Lagrangian Relaxation Problem (5.24)–(5.26) corresponds to (5.12)–(5.14). One should also notice that a Lagrangian modified objective function coefficient, say c'_j, may eventually become negative throughout the application of SM. In that case, variable x_j is guaranteed to assume a value of 0 at an optimal solution $\bar{x}$ to (5.24)–(5.26).

5.6.1 A Lagrangian heuristic to KP

Following Section 5.4, a basic greedy heuristic BH was used to generate feasible solutions to the KP instance on Table 5.1. BH was called for every SM iteration and took, as an input, Lagrangian modified costs $\{c'_j : j = 1,\ldots,n\}$. Assume that an ordering

$$c'_{i_1}/a_{i_1} \geq c'_{i_2}/a_{i_2} \geq \ldots \geq c'_{i_n}/a_{i_n} \tag{5.27}$$

is obtained from these input costs. BH then considers indices in $\{i_1,\ldots,i_n\}$ (for corresponding item inclusion into the knapsack) in the order they appear in (5.27). Assume that iteration k of BH is being performed (i.e. that the item implied by index i_k is being considered for possible inclusion into the knapsack). Assume as well that b_k is the capacity left at the knapsack at iteration k of BH. The item implied by i_k should then be included into the knapsack if $a_{i_k} \leq b_k$. Iterations should be performed until either the knapsack is filled to capacity or else until it could be established that none of the items still to be investigated could be successfully introduced into the knapsack.

Once a feasible solution is returned by BH, the cost of that solution under $\{c_j : j = 1,\ldots,n\}$ must be computed (to obtain a valid KP lower bound). The solution should then be subjected to Local Search in an attempt to improve it. The search neighborhood we use is formed by those items that could feasibly replace an item currently included in the knapsack. Item replacement should be carried out for that pair of items (if any) leading to the largest KP lower bound increase. In case of success one should check if spare space still exists in the knapsack to include items currently left out of it (whenever applicable, items should be included by their cost/benefit ratios) and iterate.

Within a Lagrangian relaxation framework, repeated calls to BH followed by Local Search (as outlined above), give rise to a Lagrangian heuristic LH to KP (see Section 5.4).

5.6.2 Variable fixing tests

Whenever an optimal solution $\bar{x}$ is obtained for (5.24)–(5.26), a valid upper bound LRP(λ) is generated for KP. One may then use LP reduced costs, together with a feasible KP lower bound z_{lb}, in an attempt to price variables out of an optimal solution. Accordingly if $\bar{c}_j$ is the reduced cost associated with variable x_j in $\bar{x}$, the variable is guaranteed to be out of an optimal solution if

$$\mathrm{LRP}(\lambda) + \bar{c}_j < z_{lb}. \tag{5.28}$$

Assume that variable x_j is such that $\bar{x}_j = 1$. One may attempt to price that variable into an optimal solution by solving (5.24)–(5.26) with the additional constraint $x_j = 0$. Clearly, if the solution value thus obtained is less than z_{lb}, variable x_j is guaranteed to be in an optimal solution to KP. The variable could then be fixed to 1.

5.6.3 A NDRC algorithm for KP

Whenever applicable (i.e. when $\bar{x}$ is fractional), a NDRC algorithm for KP would separate Gomory cuts, as suggested in Section 5.5. These cuts should be dynamically dualized, as proposed in Section 5.2. In our experiments with the KP instance in Table 5.1, a total of 1000 SM iterations were performed. Parameter α, initially set to 2.0, was halved after 50 consecutive SM iterations without an overall improvement on the best upper bound so far generated. LH, as described above, was called for every SM iteration. The same applies to the proposed variable fixing tests.

The computational results obtained are shown on Table 5.6.3. Best lower and upper bounds obtained up to the given fixed number of SM iterations are presented in the table. Twenty four variables were fixed into an optimal solution while ten variables were fixed out. Comparing the LP relaxation of the KP instance with the best upper bound generated, it is possible to see that the initial duality gap (i.e. LP relaxation value minus the best lower bound found) was closed by over 82% by the best NDRC upper bound obtained.

5.6.4 A DRC algorithm for KP

A DRC algorithm for KP would generate an initial Gomory cut from LP relaxation (5.12)–(5.14). Such a cut would then be dualized and a fixed number of SM iterations would be carried out in an attempt to solve the corresponding LDP. Assume that a few rounds of DRC Gomory cut separations have already been performed and consider the corresponding LDP. Let $\bar{x}$ be an optimal solution to the very last LRP(λ) solved while attempting to solve LDP with SM. Three outcomes are then possible. If $\bar{x}$ is integral, the DRC algorithm should be stopped since no Gomory cut could be generated from $\bar{x}$. If $\bar{x}$ is fractional but the newly separated Gomory cut is already currently

Table 5.2 Lower and upper bounds for NDRC

Iteration	Lower Bound	Upper Bound
1	18970083333551896	19110448305312768
100	18970083333551900	19068005714952528
200	19014186620712300	19049279232335140
300	19014186620712300	19042422190532812
400	19014186620712300	19041389490291912
500	19014186620712300	19038788425295852
600	19014186620712300	19036609424140228
700	19014186620712300	19035349741901788
800	19014186620712300	19033798315232872
900	19014186620712300	19032012697947240
1000	19014186620712300	19031191289895520

dualized, LDP should be, once again, stopped. Finally, if $\bar{x}$ is fractional and a Gomory cut different from the ones currently dualized is separated from $\bar{x}$, the cut should be dualized thus giving rise to a new LDP to be solved. In our experiments with the KP instance in Table 5.1, a total of 10 LDP rounds were performed. Each of these rounds involved 100 SM iterations where parameter α, initially set to 2.0, was halved after 5 consecutive SM iterations without an overall improvement on the best upper bound so far generated in the round.

LH, as described above, was called for every SM iteration. The same applies to the proposed variable fixing tests.

The computational results obtained are shown in Table 5.6.4. Best lower and upper bounds obtained up to the end of every LDP solving round are presented in the table (round 0 corresponds to the initial lower and upper bounds). Twenty four variables were fixed in an optimal solution while ten variables were fixed out. Comparing the LP relaxation of the KP instance with the best upper bound generated, it is possible to see that the initial duality gap (i.e. LP relaxation minus best lower bound) was closed by over 74% by the best upper bound obtained.

Comparing the best lower bounds generated respectively by NDRC and DRC, one should notice that a difference of only 4 units exists in favor of the NDRC bound. Such a small difference between feasible solution values involving 17 digit numbers, give a hint on the difficulty of finding proven optimal solutions to our KP instance.

The slightly better results obtained by NDRC are in accordance with results obtained in Lucena (2004) for the Steiner Tree Problem. In any case, duality gap reductions were substantial for both NDRC and DRC. One should then expect that a Branch-and-Bound algorithm based on either approach would perform well for the KP instance tested.

Table 5.3 Lower and upper bounds for DRC

Round	Lower Bound	Upper Bound
0	18970083333551896	19110448305312768
1	19014186620712292	19050199902159760
2	19014186620712292	19044749414273156
3	19014186620712292	19044745465383692
4	19014186620712292	19044744079996880
5	19014186620712300	19042060411170460
6	19014186620712300	19041048482141080
7	19014186620712300	19041026532351020
8	19014186620712304	19038583646728036
9	19014186620712304	19038583646728036
10	19014186620712304	19038583646728036

5.7 CONCLUSIONS AND SUGGESTIONS FOR FUTURE WORK

Relax-and-Cut is proving to be an attractive proposition for generating *good quality* dual bounds to Integer Programming Problems. The technique may be used on its own, as exemplified in this chapter, or be combined with LP based solution algorithms into a hybrid solution approach (see Calheiros et al. (2003), for details). Relax-and-Cut also appears very attractive for the development of Lagrangian heuristics.

Implementing NDRC under subgradient optimization methods different from SM, appears clearly relevant. One example of this appears in Belloni and Sagastizábal (2004) where NDRC was adapted to operate under a Bundle method (Bonnans et al., 1997). Investigating variants of SM to NDRC that retain *computational lightness* while improving accuracy also appears very attractive.

For the particular application focused in this chapter we plan to implement an exact solution Branch-and-Bound algorithm based on NDRC. Such an algorithm should include, in addition to the Gomory cuts studied here, Lifted Minimum Cover Inequalities.

Acknowledgments

This research was supported by CNPq (grant 300149/ 94-8) and FAPERJ (grant E26/ 71.906/ 00).

Bibliography

E. Balas and N. Christofides. A restricted Lagrangian approach to the traveling salesman problem. *Mathematical Programming*, 21:19–46, 1981.

F. Barahona and R. Anbil. The volume algorithm: producing primal solutions with the subgradient method. *Mathematical Programming*, 87:385–399, 2000.

F. Barahona and L. Ladányi. Branch and cut based on the volume algorithm: Steiner trees in graphs and Max-Cut. Technical report, IBM Watson Research Center, 2001.

J.E. Beasley. Lagrangean relaxation. In Collin Reeves, editor, *Modern Heuristic Techniques*, page Oxford. Blackwell Scientific Press, 1993.

A. Belloni and A. Lucena. A relax and cut algorithm for the traveling salesman problem, 2000. Talk given at the 17th International Symposium on Mathematical Programming.

A. Belloni and A. Lucena. A Lagrangian heuristic for the linear ordering problem. In M.G.C. Resende and J. Pinho de Sousa, editors, *Metaheuristics: Computer Decision-Making*, pages 123–151. Kluwer Academic Publishers, 2003.

A. Belloni and C. Sagastizábal. Dynamic Bundle Methods. Technical Report A 2004/296, Instituto de Matemática Pura e Aplicada, 2004.

J.F. Bonnans, J.Ch. Gilbert, C. Lemaréchal, and C. Sagastizábal. *Optimisation numérique: Aspects théoriques et pratiques*. Springer Verlag, 1997.

F. Calheiros, A. Lucena, and C. de Souza. Optimal Rectangular Partitions. *Networks*, 41:51–67, 2003.

J.B.C. da Silva. Uma heuristica lagrangeana para o problema da árvore capacitada de custo mínimo. Master's thesis, Programa de Engenharia de Sistemas e Computação, COPPE, Universidade Federal do Rio de Janeiro, 2002.

G.B. Dantzig. Discrete variable extremum problems. *Operations Research*, 4:266–277, 1957.

M. de Moraes Palmeira, A. Lucena, and O. Porto. A relax and cut algorithm for quadratic knapsack problem. Technical report, Departamento de Administração, Universidade Federal do Rio de Janeiro, 1999.

L. Escudero, M. Guignard, and K. Malik. A Lagrangian relax and cut approach for the sequential ordering with precedence constraints. *Annals of Operations Research*, 50:219–237, 1994.

H. Everett III. Generalized Lagrange multiplier method for solving problems of optimum allocation of resources. *Operations Research*, 11:399–417, 1963.

M.L. Fisher. The Lagrangian relaxation method for solving integer programming problems. *Management Science*, 27:1–18, 1981.

B. Gavish. Augmented Lagrangean based algorithms for centralized network design. *IEEE Trans. on Communications*, 33:1247–1257, 1985.

A.M. Geoffrion. Lagrangian relaxation for integer programming. *Mathematical Programming Study*, 2:82–114, 1974.

R.E. Gomory. An algorithm for integer solutions to linear programs. In R. Graves and P. Wolfe, editors, *Recent advances in mathematical programming*, pages 269–302. McGraw-Hill, 1963.

M. Guignard. Efficient cuts in Lagrangean relax-and-cut schemes. *European Journal of Operational Research*, 105:216–223, 1998.

M. Guignard. Lagrangean relaxation. *TOP*, 11:151–199, 2004.

M. Held and R.M. Karp. The traveling salesman problem and minimum spanning trees. *Operations Research*, 18:1138–1162, 1970.

M. Held and R.M. Karp. The traveling salesman problem and minimum spanning trees: Part II. *Mathematical Programming*, 1:6–25, 1971.

M. Hunting, U. Faigle, and W. Kern. A Lagrangian relaxation approach to the edge-weighted clique problem. *European Journal of Operational Research*, 131:119–131, 2001.

C. Lemaréchal. Lagrangian relaxation. In M. Jünger and D. Naddef, editors, *Computational Combinatorial Optimization*, pages 115–160. Springer Verlag, 2001.

A. Lucena. Steiner problem in graphs: Lagrangean relaxation and cutting planes. *COAL Bulletin*, 21:2–8, 1992.

A. Lucena. Steiner problem in graphs: Lagrangean relaxation and cutting planes. In *Proceedings of NETFLOW93*, pages 147–154. Univesitá degli Studi di Pisa, Dipartimento di Informatica, 1993. Technical report TR-21/93.

A. Lucena. Non Delayed Relax-and-Cut Algorithms. *Annals of Operations Research*, 2004. In press.

S. Martello and P. Toth. Upper bounds and algorithms for hard $0-1$ knapsack problems. *Operations Research*, 45:768–778, 1997.

C. Martinhon, A. Lucena, and N. Maculan. Stronger K-Tree relaxations for the vehicle routing problem. *European Journal of Operational Research*, 158:56–71, 2004.

M. Padberg and G. Rinaldi. A branch-and-cut algorithm for the resolution of large-scale symmetric traveling salesman problems. *SIAM Review*, 33:60–100, 1991.

D. Pisinger. A minimal algorithm for the $0-1$ knapsack problem. *Operations Research*, 45:758–767, 1997.

T.K. Ralphs and M.V. Galati. Decomposition and dynamic cut generation in integer linear programming. Technical report, Department of Industrial and Systems Engineering, Lehigh University, 2003.

J.F. Shapiro. A survey of Lagrangean techniques for discrete optimization. *Annals of Discrete Mathematics*, 5:113–138, 1974.

J.F. Shapiro. *Mathematical programming: Structures and algorithms*. John Wiley and Sons, 1979.

L.A. Wolsey. *Integer Programming*. John Wiley and Sons, 1998.