A view of Lagrangian relaxation and its applications

Manlio Gaudioso

Abstract We provide an introduction to Lagrangian relaxation, a methodology which consists in moving into the objective function, by means of appropriate *multipliers*, certain *complicating* constraints of integer programming problems. We focus, in particular, on the solution of the *Lagrangian dual*, a nonsmooth optimization problem aimed at finding the best multiplier configuration. The algorithm for solving the Lagrangian dual can be equipped with heuristic procedures for finding feasible solutions of the original integer programming problem. Such an approach is usually referred to as *Lagrangian heuristic*. The core of the Chapter is the presentation of several examples of Lagrangian heuristic algorithms in areas such as assignment problems, network optimization, wireless sensor networks and machine learning.

1 Introduction

The relevance of any mathematical concept lies in its ability to generate many fruits in diverse areas and to produce long-lasting effects. This is definitely the case of the Lagrange multipliers [41], which have influenced the development of modern mathematics and are still fertile as far as mathematical programming theory and algorithms are concerned.

In this chapter we confine the discussion to the treatment of numerical optimization problems in a finite dimension setting, where the decision variables are the vectors of $\mathbb{R}^n$.

Looking at the cornerstones of the historical development of such an area, we observe that Lagrange multipliers have survived the crucial passages from equality- to inequality-constrained problems, as well as from continuous to

Manlio Gaudioso
Universitá della Calabria, 87036 Rende, Italia, e-mail: manlio.gaudioso@unical.it

discrete optimization. In addition they have shown their potential also when nonsmooth analysis has come into play [51], [52].

Originally the rationale of the introduction of Lagrange multipliers was to extend to the equality constrained case the possible reduction of an optimization problem to the solution of a system of nonlinear equations, based on the observation that an optimality condition must necessarily involve both objective function and constraints. The geometry of the constraints became more and more important as soon as inequality constrained problems were taken into consideration, and Lagrangian multipliers were crucial in the definition of the related optimality conditions.

The introduction of the duality theory highlighted the role of the Lagrangian multipliers in the game-theoretic setting, as well as in the study of value functions associated to optimization problems. More recently, the '60s of last century, it was clear [19] that a Lagrangian multiplier-based approach was promising even in dealing with optimization problems of discrete nature, showing that in some sense it was possible to reduce the gap between continuous and integer optimization.

In this chapter we focus on Lagrangian relaxation, which was introduced in [32], for dealing with integer or mixed integer optimization problems and immediately conquered the attention of many scientist as a general-purpose tools for handling hard problems [22]. Pervasiveness of Lagrangian relaxation is well evidenced in [43].

The objective of the presentation is to demonstrate the usefulness of combined use of Lagrangian relaxation and heuristic algorithms to tackle hard integer programming problems. We will introduce several examples of application in fairly diverse areas of practical optimization.

The chapter is organized as follows. Basic notions on Lagrangian relaxation are in Section 2, while solution methods for tackling the Lagrangian dual are discussed in Section 3. A basic scheme of Lagrangian heuristics, together with an example of dual ascent for the classic Set Covering problem, is in Section 4. A number of applications in areas such as Assignment, Network optimization, Sensor location, Logistics and Machine Learning are discussed in Section 5. Some conclusions are drawn in Section 6.

2 Basic concepts

We introduce first the following general definition.

Definition 1. Relaxation. Given a minimization problem (P) in the following form:

$$\begin{cases} \text{minimize} & f(\mathbf{x}) \\ \text{subject to} & \mathbf{x} \in X \subset \mathbb{R}^n, \end{cases} \tag{1}$$

with $f : \mathbb{R}^n \to R$, any problem (P_R):

$$\begin{cases} \text{minimize} & h(\mathbf{x}) \\ \text{subject to} & \mathbf{x} \in X_R \subset \mathbb{R}^n, \end{cases} \tag{2}$$

with $h : \mathbb{R}^n \to \mathbb{R}$, is a *relaxation* of (P), provided that the two following conditions hold:

$$X \subset X_R; \tag{3}$$

$$h(\mathbf{x}) \leq f(\mathbf{x}), \quad \mathbf{x} \in X. \tag{4}$$

Under the above definition, if $\mathbf{x}^*$ is any (global) minimum of (P_R), then $h(\mathbf{x}^*)$ is a lower bound on the optimal value of problem (P). Of course it is advantageous to spend some effort to obtain such information whenever (P_R) is substantially easier to solve than (P).

There exist many different ways for constructing relaxations. For example, the usual continuous relaxation of any Integer Linear Programming (ILP) problem satisfies the conditions (3) and (4).

We introduce Lagrangian relaxation starting from a Linear Program (LP) in standard form:

$$\begin{cases} \text{minimize} & \mathbf{c}^T\mathbf{x} \\ \text{subject to} & A\mathbf{x} = \mathbf{b}, \\ & \mathbf{x} \geq 0 \end{cases} \tag{5}$$

with $\mathbf{x}, \mathbf{c} \in \mathbb{R}^n$, $A \in \mathbb{R}^{m \times n}$ and $\mathbf{b} \in \mathbb{R}^m$. We assume that the problem is feasible and has optimal solution, so that the dual problem is feasible as well.

For any choice of the Lagrangian multiplier vector $\boldsymbol{\lambda} \in \mathbb{R}^m$ (or, simply, the multiplier vector), we define the Lagrangian relaxation $(LR(\boldsymbol{\lambda}))$

$$\begin{cases} \text{minimize} & \mathbf{c}^T\mathbf{x} + \boldsymbol{\lambda}^T(\mathbf{b} - A\mathbf{x}) \\ \text{subject to} & \mathbf{x} \geq 0, \end{cases} \tag{6}$$

which can be rewritten as

$$\begin{cases} \mathbf{b}^T\boldsymbol{\lambda} + \text{minimize} & (\mathbf{c} - A^T\boldsymbol{\lambda})^T\mathbf{x} \\ \qquad\quad \text{subject to} & \mathbf{x} \geq 0. \end{cases} \tag{7}$$

Letting $z_{LR}(\boldsymbol{\lambda})$ be the optimal value of problem (7), we obtain

$$z_{LR}(\boldsymbol{\lambda}) = \begin{cases} \mathbf{b}^T\boldsymbol{\lambda} & \text{if} \quad \mathbf{c} - A^T\boldsymbol{\lambda} \geq 0 \\ -\infty & \text{otherwise.} \end{cases} \tag{8}$$

From the definition of relaxation, $z_{LR}(\boldsymbol{\lambda})$ is a lower bound for problem (LP), possibly the trivial one for those values of $\boldsymbol{\lambda}$ which do not satisfy the condition

$A^T\boldsymbol{\lambda} \le \mathbf{c}$. It is quite natural to look for the *best* lower bound and this results in solving the problem

$$\text{maximize} \quad z_{LR}(\boldsymbol{\lambda}), \tag{9}$$

which, taking into account (8), is exactly the dual of (LP),

$$\begin{cases} \text{maximize} & \mathbf{b}^T\boldsymbol{\lambda} \\ \text{subject to} & A^T\boldsymbol{\lambda} \le \mathbf{c}. \end{cases} \tag{10}$$

Problem (9) will be referred to as the *Lagrangian dual* of problem (LP), and in fact the optimal solution $\boldsymbol{\lambda}^*$ of the dual is optimal for (9) as well.

The main motivation for the introduction of Lagrangian relaxation is the treatment of integer linear programming (ILP) problems. We will not consider in the sequel the mixed integer linear programming case $(MILP)$, where Lagrangian relaxation theory can be developed in a completely analogous way as in the pure case. The binary linear programming problems (BLP) can be considerd as a special case of (ILP). Thus we focus on the following problem:

$$\begin{cases} \text{minimize} & \mathbf{c}^T\mathbf{x} \\ \text{subject to} & A\mathbf{x} = \mathbf{b}, \\ & B\mathbf{x} = \mathbf{d} \\ & \mathbf{x} \ge 0, \text{ integer}, \end{cases} \tag{11}$$

with $\mathbf{x}, \mathbf{c} \in \mathbb{R}^n$, $A \in \mathbb{R}^{m\times n}$, $B \in \mathbb{R}^{p\times n}$, $\mathbf{b} \in \mathbb{R}^m$ and $\mathbf{d} \in \mathbb{R}^p$. We assume that the problem is feasible and that the set

$$X = \{\, \mathbf{x} \in \mathbb{R}^n \mid B\mathbf{x} = \mathbf{d}, \;\; \mathbf{x} \ge 0, \;\text{integer}\}$$

is finite, that is $X = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_K\}$ and we denote by $\mathcal{K} = \{1, 2, \dots, K\}$ the corresponding index set. In writing problem (11) two different families of constraints are highlighted, those defined through $A\mathbf{x} = \mathbf{b}$ being the *complicating* ones. By focusing exclusively on such set of constraints, we come out with the Lagrangian relaxation defined for $\boldsymbol{\lambda} \in \mathbb{R}^m$

$$\begin{cases} \text{minimize} & \mathbf{c}^T\mathbf{x} + \boldsymbol{\lambda}^T(\mathbf{b} - A\mathbf{x}) \\ \text{subject to} & \mathbf{x} \in X, \end{cases} \tag{12}$$

which is supposed easier to solve than the original problem. By letting $\mathbf{x}(\boldsymbol{\lambda})$ and $z_{LR}(\boldsymbol{\lambda})$ be, respectively, the optimal solution and the optimal value of (12), it is

$$z_{LR}(\boldsymbol{\lambda}) = \mathbf{c}^T\mathbf{x}(\boldsymbol{\lambda}) + \boldsymbol{\lambda}^T(\mathbf{b} - A\mathbf{x}(\boldsymbol{\lambda})) = \min_{k=1,\dots,K} \mathbf{c}^T\mathbf{x}_k + \boldsymbol{\lambda}^T(\mathbf{b} - A\mathbf{x}_k) \tag{13}$$

Remark 1. Function $z_{LR}(\boldsymbol{\lambda})$ is often referred to as the *dual* function. Note that, in case $\mathbf{x}(\boldsymbol{\lambda})$ is feasible, that is $A\mathbf{x}(\boldsymbol{\lambda}) = \mathbf{b}$, then it is also optimal for the original problem (11)

In order to look for the best among the lower bounds $z_{LR}(\boldsymbol{\lambda})$, we define also in this case the Lagrangian dual

$$z_{LD} = \max_{\boldsymbol{\lambda} \in \mathbb{R}^m} \quad z_{LR}(\boldsymbol{\lambda}), \tag{14}$$

that is, from(3)

$$z_{LD} = \max_{\boldsymbol{\lambda}} \min_{k=1,\dots,K} \mathbf{c}^T \mathbf{x}_k + \boldsymbol{\lambda}^T(\mathbf{b} - A\mathbf{x}_k). \tag{15}$$

The optimal value z_{LD} is the best lower bound obtainable through Lagrangian relaxation.

It is worth noting that problem (15), which we will discuss later in more details, consists in the maximization of a concave and piecewise affine function. It is in fact a nonsmooth optimization problems which can be tackled by means of any of the methods described in this book. On the other hand, by introducing the additional variable $v \in \mathbb{R}$, it can be rewritten in the following Linear Programming form:

$$\begin{cases} \text{maximize}_{\boldsymbol{\lambda}, v} & v \\ \text{subject to} & v \leq \mathbf{c}^T \mathbf{x}_k + \boldsymbol{\lambda}^T(\mathbf{b} - A\mathbf{x}_k), \;\; k = 1, \dots, K, \end{cases} \tag{16}$$

whose dual, in turn, is

$$\begin{cases} \text{minimize}_{\boldsymbol{\mu}} & \mathbf{c}^T (\sum_{k=1}^{K} \mu_k \mathbf{x}_k) \\ \text{subject to} & A(\sum_{k=1}^{K} \mu_k \mathbf{x}_k) = \mathbf{b} \\ & \sum_{k=1}^{K} \mu_k = 1 \\ & \mu_k \geq 0, \;\; k = 1, \dots, K, \end{cases} \tag{17}$$

or, equivalently,

$$\begin{cases} \text{minimize} & \mathbf{c}^T \mathbf{x} \\ \text{subject to} & A\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \in \operatorname{conv} X = \operatorname{conv}\{\, \mathbf{x} \in \mathbb{R}^n \mid B\mathbf{x} = \mathbf{d}, \;\; \mathbf{x} \geq 0 \text{ integer}\}. \end{cases} \tag{18}$$

The Lagrangian dual is thus a *partially convexified* version of the *ILP* (11). From the inclusion:

$$\operatorname{conv} X \subseteq \{\, \mathbf{x} \in \mathbb{R}^n \mid B\mathbf{x} = \mathbf{d},\ \mathbf{x} \geq 0\} = \bar{X}$$

it follows

$$z_{LD} \geq z_{LP}, \tag{19}$$

where z_{LP} is the optimal value of the continuous LP relaxation of (11):

$$\begin{cases} \text{minimize} & \mathbf{c}^T\mathbf{x} \\ \text{subject to} & A\mathbf{x} = \mathbf{b}, \\ & B\mathbf{x} = \mathbf{d} \\ & \mathbf{x} \geq 0. \end{cases} \tag{20}$$

Property (19) indicates that the lower bound provided by the *best* (not by any!) multiplier vector setting is not worse than the optimal value of the continuous relaxation. Moreover it is important to observe that in case the so-called *integrality property* holds, that is in case the vertices of the polyhedron $\bar{X}$ are integer, it is $z_{LD} = z_{LP}$.

Whenever the integrality property is satisfied (and this is a rather common case in Integer Programming applications) Lagrangian relaxation may appear a quite weak approach compared to classic continuous relaxation: we need to solve a nonsmooth optimization problem of the *maxmin* type just to get he same bound obtainable by solving a Linear Program! Nonetheless Lagrangian relaxation may be a useful tool also in this case for the following reasons:

- In several applications the continuous relaxation is a huge Linear Program and it may be a good idea not to tackle it;
- In Lagrangian relaxation the decision variables (think e.g. binary variables) keep the original physical meaning, which, instead, gets lost in continuous relaxation;
- The possibly infeasible solutions of the Lagrangian relaxation are often more suitable for *repairing* heuristics than those of the continuous relaxation.

Coming back to the two formulations (15) and (17), we observe that they offer two possible schemes for solving the Lagrangian dual.

Consider first (15), which is a finite *maxmin* problem where the function to be maximized is concave. In fact it is defined as the pointwise minimum of a finite (possibly very large) number of affine functions of variable $\boldsymbol{\lambda}$, one for each $\mathbf{x}_k \in X$. Application of the classic cutting plane model consists in generating an (upper) approximation of the *min* function, initially based on a relatively small number of affine pieces, which becomes more and more accurate as new affine pieces (*cuts*) are added. In practice, taking any subset $\mathcal{S} \subset \mathcal{K}$ (and the correpondent points $\mathbf{x}_k \in X,\ k \in \mathcal{S}$) the *cutting plane* approximation $z_{\mathcal{S}}(\boldsymbol{\lambda})$ of $z_{LR}(\boldsymbol{\lambda})$ is defined as

$$z_{\mathcal{S}}(\boldsymbol{\lambda}) = \min_{k \in \mathcal{S}} \mathbf{c}^T \mathbf{x}_k + \boldsymbol{\lambda}^T (\mathbf{b} - A\mathbf{x}_k),$$

and thus we come out with the *restricted primal* problem

$$z_{restr}(\mathcal{S}) = \max_{\boldsymbol{\lambda}} \min_{k \in \mathcal{S}} \mathbf{c}^T \mathbf{x}_k + \boldsymbol{\lambda}^T (\mathbf{b} - A\mathbf{x}_k), \tag{21}$$

which, in turn, can be put in Linear Programming form of the type (16):

$$\begin{cases} \text{maximize}_{\boldsymbol{\lambda}, v} & v \\ \text{subject to} & v \leq \mathbf{c}^T \mathbf{x}_k + \boldsymbol{\lambda}^T (\mathbf{b} - A\mathbf{x}_k), \;\; k \in \mathcal{S}. \end{cases} \tag{22}$$

Assuming problem (21) is not unbounded and letting $\boldsymbol{\lambda}_{\mathcal{S}}$ be any optimal solution, we calculate now $z_{LR}(\boldsymbol{\lambda}_{\mathcal{S}})$ (in fact we solve the Lagrangian relaxation for $\boldsymbol{\lambda} = \boldsymbol{\lambda}_{\mathcal{S}}$):

$$z_{LR}(\boldsymbol{\lambda}_{\mathcal{S}}) = \min_{k \in \mathcal{K}} \mathbf{c}^T \mathbf{x}_k + {\boldsymbol{\lambda}_{\mathcal{S}}}^T (\mathbf{b} - A\mathbf{x}_k) = \mathbf{b}^T \boldsymbol{\lambda}_{\mathcal{S}} + \min_{k \in \mathcal{K}} (\mathbf{c} - A^T \boldsymbol{\lambda}_{\mathcal{S}})^T \mathbf{x}_k. \tag{23}$$

Letting the optimal solution of the above problem be attained in correspondence to any index, say $k_{\mathcal{S}}$, and assuming $z_{LR}(\boldsymbol{\lambda}_{\mathcal{S}})$ be sufficiently smaller than $z_{restr}(\mathcal{S})$, the procedure is then iterated after augmenting the set of affine pieces in (21) by the one associated to the newly generated point $\mathbf{x}_{k_{\mathcal{S}}}$.

Consider now the formulation of the Lagrangian dual provided by (17). It is a (possibly very large) Linear Program in a form particularly suitable for column generation [56]. We observe in fact that the columns are associated to points $\mathbf{x}_k$, $k \in \mathcal{K}$, in perfect analogy with the association affine pieces–points of X in the formulation (15). To apply the simplex method it is necessary to start from any subset of columns providing a basic feasible solution and then look for a column with negative reduced cost. Such operation is not necessarily accomplished by examining the reduced costs of *all* non basic columns, instead it can be implemented by solving a *pricing* problem. In fact the constraint matrix in (17) has size $(m+1) \times K$ and has the form:

$$\begin{bmatrix} A\mathbf{x}_1, & \ldots, & A\mathbf{x}_K \\ 1 & \ldots & 1. \end{bmatrix} \tag{24}$$

Letting $\boldsymbol{\lambda}_B$ be the vector assembling the first m dual variables associated to the basis, the components of the reduced cost vector $\hat{\mathbf{c}}$ are:

$$\hat{\mathbf{c}}_k = (\mathbf{c} - A^T \boldsymbol{\lambda}_B)^T \mathbf{x}_k - \lambda_{m+1}, \;\; k \in \mathcal{K},$$

where λ_{m+1} is the last dual variable and plays the role of a constant in the definition of $\hat{\mathbf{c}}_k$, $k \in \mathcal{K}$. Consequently the pricing problem consists of solving

$$\min_{k \in \mathcal{K}} (\mathbf{c} - A^T \boldsymbol{\lambda}_B)^T \mathbf{x}_k, \tag{25}$$

which is a problem formally identical to (23).

Remark 2. It is worth noting that both (23) and (25) need not be solved at optimality, as for the former it is sufficient to generate a *cut*, that is to calculate any index k, $k \notin \mathcal{S}$ such that

$$\mathbf{c}^T\mathbf{x}_k + \boldsymbol{\lambda}_{\mathcal{S}}{}^T(\mathbf{b} - A\mathbf{x}_k) < z_{restr}(\mathcal{S}),$$

while for the latter the optimization process can be stopped as soon as a column with the negative reduced cost has been detected.

We observe, finally, that in formulation (11) the complicating constraints $A\mathbf{x} = \mathbf{b}$ are in the equality form. Lagrangian relaxation is well defined also in case they are of the type $A\mathbf{x} \geq \mathbf{b}$, the only difference being in the need of setting $\boldsymbol{\lambda} \geq 0$. This fact implies that the Lagrangian dual is now

$$\max_{\boldsymbol{\lambda} \geq \mathbf{0}} \quad z_{LR}(\boldsymbol{\lambda}). \tag{26}$$

It is worth noting that feasibility of $\mathbf{x}(\boldsymbol{\lambda})$ no longer implies optimality. In fact it can be easily proved that now $\mathbf{x}(\boldsymbol{\lambda})$ is only ϵ-optimal, for

$$\epsilon = (A\mathbf{x}(\boldsymbol{\lambda}) - \mathbf{b})^T\boldsymbol{\lambda} \geq 0.$$

3 Tackling the Lagrangian dual

Solving the Lagrangian dual problem (14) requires maximization of a concave and piecewise affine function (see (15)). Consequently all the available machinery to deal with convex nondifferentiable optimization can be put in action. We have already sketched in previous section possible use of the cutting plane method [14], [38]. On the other hand the specific features of the Lagrangian dual can be fruitfully exploited.

The basic distinction is between algorithms which do or do not use the differential properties of function z_{LR}. Observe that a subgradient (concavity would suggest the more appropriate term *supergradient*) is immediately available as soon as z_{LR} has been calculated. Letting in fact (see (3))

$$z_{LR}(\boldsymbol{\lambda}) = \mathbf{c}^T\mathbf{x}(\boldsymbol{\lambda}) + \boldsymbol{\lambda}^T(\mathbf{b} - A\mathbf{x}(\boldsymbol{\lambda})) = \min_{k=1,\dots,K} \mathbf{c}^T\mathbf{x}_k + \boldsymbol{\lambda}^T(\mathbf{b} - A\mathbf{x}_k)$$

we observe that, for any $\boldsymbol{\lambda}' \in \mathbb{R}^m$, it is

$$\begin{aligned} z_{LR}(\boldsymbol{\lambda}') &= \min_{k=1,\dots,K} \mathbf{c}^T\mathbf{x}_k + \boldsymbol{\lambda}'^T(\mathbf{b} - A\mathbf{x}_k) \leq \mathbf{c}^T\mathbf{x}(\boldsymbol{\lambda}) + \boldsymbol{\lambda}'^T(\mathbf{b} - A\mathbf{x}(\boldsymbol{\lambda})) \\ &= \mathbf{c}^T\mathbf{x}(\boldsymbol{\lambda}) + \boldsymbol{\lambda}'^T(\mathbf{b} - A\mathbf{x}(\boldsymbol{\lambda})) + \boldsymbol{\lambda}^T(\mathbf{b} - A\mathbf{x}(\boldsymbol{\lambda})) - \boldsymbol{\lambda}^T(\mathbf{b} - A\mathbf{x}(\boldsymbol{\lambda})) \\ &= z_{LR}(\boldsymbol{\lambda}) + (\boldsymbol{\lambda}' - \boldsymbol{\lambda})^T(\mathbf{b} - A\mathbf{x}(\boldsymbol{\lambda})), \end{aligned}$$

thus $(\mathbf{b} - A\mathbf{x}(\boldsymbol{\lambda}))$ is a subgradient of z_{LR} at $\boldsymbol{\lambda}$, that is $\mathbf{g}(\boldsymbol{\lambda}) = (\mathbf{b} - A\mathbf{x}(\boldsymbol{\lambda})) \in \partial z_{LR}(\boldsymbol{\lambda})$.

As for methods that use the subgradient for maximizing $z_{LR}(\boldsymbol{\lambda})$, we mention first the classic (normalized) subgradient method [49], [54], where the hth iteration is

$$\boldsymbol{\lambda}_{h+1} = \boldsymbol{\lambda}_h + t_h \frac{\mathbf{g}(\boldsymbol{\lambda}_h)}{\|\mathbf{g}(\boldsymbol{\lambda}_h)\|},$$

t_h being the *stepsize* along the normalized subgradient direction. We remark that monotonicity of the sequence of values $z_{LR}(\boldsymbol{\lambda}_h)$ is not ensured, while convergence to a maximum is guaranteed under the well known conditions on the stepsize $t_h \to 0$ and $\sum_{h=1}^{\infty} t_h \to \infty$. Very popular formulae for setting t_h are

$$t_h = \frac{C}{h}, \tag{27}$$

and the Polyak formula

$$t_h = \frac{\hat{z}_{LD}^{(h)} - z_{LR}(\boldsymbol{\lambda}_h)}{\|\mathbf{g}(\boldsymbol{\lambda}_h)\|}, \tag{28}$$

where C is any positive constant and $\hat{z}_{LD}^{(h)}$ is an improving overestimate at the iteration h of z_{LD}, the optimal value of the Lagrangian dual.

Subgradient-type were the first and the most widely used methods for dealing with the Lagrangian dual, despite their slow convergence, mainly for their implementation simplicity. In more recent years, stemming from the approach introduced in [47], the so called *fast gradient* methods have received considerable attention [27]. The approach consists in smoothing first the objective function and applying next gradient-type methods. Besides such stream, *incremental* subgradient methods, which are applicable whenever the objective function is expressed as a sum of several convex *component* functions, have been devised to deal with the Lagrangian dual [6], [39].

A careful analysis of the performance of subgradient methods for Lagrangian relaxation is in [26].

The cutting plane method previously summarized has been the building block for devising two families of algorithms, the bundle [37] and the analytic center [33] methods for convex minimization (or, equivalently, concave maximization). They can be considered as an evolution of the cutting plane, aimed at overcoming some inherent weakness in terms of stability, speed and well posedness. An update discussion on bundle methods is in Chapter FRANGIONI of this book.

The above methods have been successfully employed in several applications of Lagrangian relaxation in frameworks such as energy management [5], [9], [20], [34], [42], network design [26], train timetabling [21], telecommunication networks [44], logistics [46] etc.

As for methods which do not make explicit use of the subgradient of function z_{LR}, they are iterative algorithms of the coordinate or block-coordinate search type, according to the fact that at each iteration just one or a (small) subset of multipliers is modified. The update is aimed at increasing z_{LR}, thus such class of methods is, in general, referred to as the *dual ascent* one [35].

The choice of the component of the current multiplier vector $\boldsymbol{\lambda}$ to be updated is usually driven by the properties of the optimal solution $\mathbf{x}(\boldsymbol{\lambda})$. In general one picks up a violated constraint and modifies the corresponding multiplier trying to achieve a twofold objective:

- To increase z_{LR}.
- To ensure feasibility of the previously violated constraint.

In several cases it is possible to calculate exactly (sometimes by solving some auxiliary problem) the multiplier update which guarantees satisfaction of both the above objectives. Most of the times, however, it is necessary to introduce a line search capable to handle possible *null step* exit, which cannot be excluded as consequence of nonsmoothness of function z_{LR}.

From the computational point of view, comparison of dual ascent and subgradient methods shows that the former produce in general more rapid growth of the dual function z_{LR}. On the other hand, dual ascent algorithms are often affected by premature stop at points fairly far from the maximum, whenever no coordinate direction is actually an ascent one. In such a case it is useful to accommodate for re-initialization, by adopting any subgradient as restart direction.

4 Lagrangian heuristics

As pointed out in [43], Lagrangian relaxation is more than just a technique to calculate lower bounds. Instead, it is a general philosophy to approach problems which are difficult to tackle, because of their intrinsic complexity.

Lagrangian relaxation has been extensively used in the framework of *exact* methods for integer programming. We will not enter into the discussion on the best ways to embed Lagrangian relaxation into Branch and Bound, Branch and Cut and Branch and Price algorithms. We refer the interested reader to the surveys [25] and [36].

We will focus, instead, on the use of Lagrangian relaxation in the so called *Lagrangian heuristic* framework for solving problem (11). The computational scheme is fundamentally the following:

Algorithm 1: Lagrangian heuristic

```
Step 0  (Initialization) Choose $\boldsymbol{\lambda}^{(0)}$. Set $k = 0$ and $z_{UB} = \infty$.
Step 1  (Lagrangian relaxation ) Calculate $z_{LR}(\boldsymbol{\lambda}^{(k)})$ and the
    corresponding $\mathbf{x}(\boldsymbol{\lambda}^{(k)})$. If $\mathbf{x}(\boldsymbol{\lambda}^{(k)})$ is feasible then STOP.
Step 2  (Repairing heuristic) Implement any heuristic algorithm to
    provide, starting from $\mathbf{x}(\boldsymbol{\lambda}^{(k)})$, a feasible solution $\mathbf{x}_{eur}^{(k)}$. Set $z_{eur}^{(k)} = \mathbf{c}^T\mathbf{x}_{eur}^{(k)}$
    and possibly update $z_{UB}$.
Step 3  (Multiplier update) Calculate $\boldsymbol{\lambda}^{(k+1)}$) by applying any algorithm
    for the Lagrangian dual. Set $k = k + 1$. Perform a termination test and
    possibly return to Step 1.
```

The above scheme is just an abstract description of how a Lagrangian heuristic works and several points need to be specified.

As for the initialization, a possible choice, whenever the LP continuous relaxation (20) is not too hard to solve, is to set $\boldsymbol{\lambda}^{(0)} = \boldsymbol{\lambda}^*_{LP}$, where $\boldsymbol{\lambda}^*_{LP}$ is the dual optimal vector associated to constraints $A\mathbf{x} = \mathbf{b}$.

At Step 2 the Lagrangian relaxation (12) is solved. As previously mentioned, it is expected to be substantially easier than the original problem (11), and in fact it is solvable, in many applications, in polynomial or pseudo-polynomial time. This is not always the case, as (12) may be still a hard combinatorial problem. In such cases it is possible to substitute an approximate calculation of z_{LR} to the exact one, which amounts to adopt a heuristic algorithm to tackle the Lagrangian relaxation (12) at Step 1.

Inexact calculation of z_{LR} has a strong impact on the way in which the Lagrangian dual (14) is tackled at Step 3. Here all machinery of convex optimization with inexact calculation of the objective function enters into play. For an extensive treatment of the subject see [17], [40] and Chapter DE OLIVEIRA SOLODOV of this book. An example of the use of inexact calculation of z_{LR} in a Lagrangian heuristic framework is in [29].

The repairing heuristic at Step 2 is, of course, problem dependent and it is very often of the greedy type.

The termination test depends on the type of algorithm used for solving the Lagrangian dual. Classic termination tests based on approximate satisfaction of the condition $\mathbf{0} \in \partial z_{LR}(\boldsymbol{\lambda}^{(k)})$ can be used whenever bundle type algorithms are adopted. In case subgradient or dual ascent algorithms are at work, stopping tests based on variation of z_{LR} can be adopted, together with a bound on the maximum number of iterations.

An important feature of Lagrangian heuristics is that they provide both a lower and an upper bound and, consequently, significant indications on the upper–lower bound gap are often at hand.

It is worth remarking, however, that, differently from the case where Lagrangian relaxation is used within an exact algorithm, here the main aim is to produce good quality feasible solutions rather than to obtain tight lower bounds. In fact it happens in several applications that poor lower bounds are accompanied by a fairly good upper bound. This is probably due to the fact that, as the algorithm proceeds, many feasible solutions are explored, through a search mechanism where multiplier update is finalized to get feasibility. From this point of view, adopting a not too fast nonsmooth optimization algorithm is not always a disadvantage!

Consequence of the above observations is that Lagrangian heuristics can be satisfactorily applied even to problems which exhibit the integrality property, that is also in case one knows in advance that Lagrangian dual is unable to produce anything better than the continuous LP relaxation lower bound.

4.1 An example of dual ascent

The following example shows how to get, by tackling the Lagrangian dual, both a lower and an upper bound for a classic combinatorial problem. The Lagrangian dual is approached by means of a dual ascent algorithm, which works as a co-ordinate search method and modifies just one multiplier at a time.

Consider the standard *Set Covering* problem (SCP), which is known to be NP-hard. Suppose we are given a ground-set $\mathcal{I} = \{1, \dots, m\}$, a family of n subsets $\mathcal{S}_j \subseteq \mathcal{I}$, $j \in \mathcal{J} = \{1, \dots n\}$ and a real cost vector $\mathbf{c} \in \mathbb{R}^n$, $\mathbf{c} > 0$. The problem is to select a minimum cost cover, that is an index set $\mathcal{J}^* \subseteq \mathcal{J}$ such that $\cup_{j \in \mathcal{J}^*} \mathcal{S}_j = \mathcal{I}$ with minimal associated cost $\sum_{j \in \mathcal{J}^*} \mathbf{c}_j$. By defining the $m \times n$ binary incidence matrix A of the subsets $\mathcal{S}_j$ and letting $\mathbf{e}$ be a vector of m ones, (SCP) reads as follows

$$\begin{cases} \text{minimize} & \mathbf{c}^T \mathbf{x} \\ \text{subject to} & A\mathbf{x} \geq \mathbf{e}, \\ & \mathbf{x} \text{ binary}, \end{cases} \tag{29}$$

where $\mathbf{x}$ is a binary decision variable vector with $\mathbf{x}_j = 1$ if j is taken into the cover and $\mathbf{x}_j = 0$ otherwise, $j \in \mathcal{J}$.

By relaxing the covering constraints $A\mathbf{x} \geq \mathbf{e}$ by means of the multiplier vector $\boldsymbol{\lambda} \geq 0,\ \boldsymbol{\lambda} \in \mathbb{R}^m$, we obtain

$$\begin{cases} z_{LR}(\boldsymbol{\lambda}) = \min & \mathbf{c}^T \mathbf{x} + \boldsymbol{\lambda}^T (\mathbf{e} - A\mathbf{x}) \\ \text{subject to} & \mathbf{x} \text{ binary}, \end{cases} \tag{30}$$

which can be in turn rewritten as

$$\begin{cases} z_{LR}(\boldsymbol{\lambda}) = \mathbf{e}^T \boldsymbol{\lambda} + & \min (\mathbf{c} - A^T \boldsymbol{\lambda})^T \mathbf{x} \\ \text{subject to} & \mathbf{x} \text{ binary}. \end{cases} \tag{31}$$

An optimal solution $\mathbf{x}(\boldsymbol{\lambda})$ to problem (31) can be obtained by simple inspection of the (reduced) cost vector $\mathbf{c}(\boldsymbol{\lambda}) = \mathbf{c} - A^T\boldsymbol{\lambda}$, by setting

$$x_j(\boldsymbol{\lambda}) = \begin{cases} 1 & \text{if } c_j(\boldsymbol{\lambda}) = c_j - \mathbf{a}_j^T\boldsymbol{\lambda} \le 0 \\ 0 & \text{otherwise,} \end{cases} \tag{32}$$

where $\mathbf{a}_j$ is the column j of matrix A. Observe that in this case the integrality property holds, thus $z_{LD} = z_{LP}$.

We introduce now a rule for updating the multiplier vector $\boldsymbol{\lambda}$, in case the corresponding $\mathbf{x}(\boldsymbol{\lambda})$ is infeasible, so that in the new multiplier setting:

- The number of satisfied constraints is increased;
- Function z_{LR} increases as well.

We proceed by updating just one component of $\boldsymbol{\lambda}$. Since we assume that $\mathbf{x}(\boldsymbol{\lambda})$ is infeasible, there exists at least one row index, say h, such that:

$$\sum_{j=1}^{n} a_{hj} x_j(\boldsymbol{\lambda}) = 0, \tag{33}$$

which implies, taking into account (32),

$$c_j(\boldsymbol{\lambda}) = c_j - \mathbf{a}_j^T\boldsymbol{\lambda} > 0, \quad \forall j \in \mathcal{J}^{(h)} = \{j \mid a_{hj} = 1\}.$$

Now, defining a new multiplier setting in the form $\boldsymbol{\lambda}^+ = \boldsymbol{\lambda} + \delta\mathbf{e}_h$ for some $\delta > 0$, where $\mathbf{e}_h$ is the hth unit vector, the updated reduced cost is $\mathbf{c}(\boldsymbol{\lambda}^+) = \mathbf{c}(\boldsymbol{\lambda}) - \delta A^T\mathbf{e}_h$, that is

$$c_j(\boldsymbol{\lambda}^+) = \begin{cases} c_j(\boldsymbol{\lambda}) - \delta & \text{if } j \in \mathcal{J}^{(h)} \\ c_j(\boldsymbol{\lambda}) & \text{otherwise.} \end{cases} \tag{34}$$

In particular, by setting $\delta = \min_{j \in \mathcal{J}^{(h)}} c_j(\boldsymbol{\lambda}) = c_{j^*}(\boldsymbol{\lambda})$, it is $c_{j^*}(\boldsymbol{\lambda}^+) = 0$ and thus, from (32), $x_{j^*}(\boldsymbol{\lambda}^+) = 1$. Summing up it is

$$x_j(\boldsymbol{\lambda}^+) = \begin{cases} x_j(\boldsymbol{\lambda}) & \text{if } j \ne j^* \\ 1 & \text{if } j = j^*. \end{cases} \tag{35}$$

Corresponding to the new solution $\mathbf{x}(\boldsymbol{\lambda}^+)$, the constraint h is no longer violated and it is $z_{LR}(\boldsymbol{\lambda}^+) = z_{LR}(\boldsymbol{\lambda}) + \delta$. Summing up, dual ascent has been achieved and at least one of the constraints previously violated is satisfied, while the satisfied ones remain such. By iterating the multiplier update, we obtain, in at most m steps, both a feasible solution and a lower bound, produced by a sequence of ascent steps.

Algorithm 2: Dual ascent for Set Covering

```
Step 0  (Initialization) Set $\boldsymbol{\lambda}^{(0)} = 0$, $\mathbf{c}(\boldsymbol{\lambda}^{(0)}) = \mathbf{c}$, $z_{LR}(\boldsymbol{\lambda}^{(0)}) = 0$,
    $\mathbf{x}(\boldsymbol{\lambda}^{(0)}) = 0$ and $k = 0$.
Step 1  (Termination test) If $A\mathbf{x}(\boldsymbol{\lambda}^{(k)}) \geq \mathbf{e}$ then calculate $z_{UB} = \mathbf{c}^T\mathbf{x}(\boldsymbol{\lambda}^{(k)})$
    and STOP. Else select $h$ such that $\sum_{j=1}^{n} a_{hj}x_j(\boldsymbol{\lambda}) = 0$, calculate
    $0 < \delta = \min_{j \in \mathcal{J}^{(h)}} c_j(\boldsymbol{\lambda}^{(k)})$ and $j^* = \text{argmin}_{j \in \mathcal{J}^{(h)}} c_j(\boldsymbol{\lambda}^{(k)})$, with
    $\mathcal{J}^{(h)} = \{j \mid a_{hj} = 1\}$.
Step 2  (Multiplier and reduced cost update) Put $\boldsymbol{\lambda}^{(k+1)} = \boldsymbol{\lambda}^{(k)} + \delta\mathbf{e}_h$,
    $\mathbf{c}(\boldsymbol{\lambda}^{(k+1)}) = \mathbf{c}(\boldsymbol{\lambda}^{(k)}) - \delta A^T\mathbf{e}_h$,
        $x_j(\boldsymbol{\lambda}^{(k+1)}) = \begin{cases} x_j(\boldsymbol{\lambda}^{(k)}) & \text{if } j \neq j^* \\ 1 & \text{if } j = j^*, \end{cases}$
    $z_{LR}(\boldsymbol{\lambda}^{(k+1)}) = z_{LR}(\boldsymbol{\lambda}^{(k)}) + \delta$. Set $k = k + 1$ and return to Step 1.
```

At stop in correspondence to any iteration index k, we obtain both a feasible solution $\mathbf{x}(\boldsymbol{\lambda}^{(k)})$, with associate objective function value z_{UB}, together with the lower bound $z_{LR}(\boldsymbol{\lambda}^{(k)})$.

Remark 3. For Set Covering problem the integrality property holds true, thus we cannot expect from the above procedure anything better than the (LP) lower bound. Moreover, since it is not at all guaranteed that the optimum of the Lagrangian is achieved when the algorithm stops, the lower bound provided might be definitely worse than the (LP) one. On the other hand the interplay between quest for feasibility and dual function improvement is a typical aspect of the applications we are going to describe in next section.

5 Applications

In this section we describe a number of applications of Lagrangian relaxation to integer programming problems coming from fairly diverse areas. In almost all cases a Lagrangian heuristic based on the abstract scheme sketched in Section 4 is designed. The Lagrangian dual is dealt with either via dual ascent or via subgradient algorithms.

In particular the following problems will be treated:

- Generalized assignment [23].
- Spanning tree with minimum branch vertices [11].
- Directional sensors location [3].
- Cross Docking scheduling [15].
- Feature selection in Support Vector Machines (SVM) [30].
- Multiple instance learning [4].

The presentation is necessarily synthetic and no numerical results are presented. The interested reader is referred to the original papers and to the references therein.

5.1 Generalized assignment

A classic application of a dual ascent procedure based on updating one multiplier at a time is the method for solving the Generalized Assignment Problem (GAP) described in [23]. (GAP) can be seen as the problem of assigning jobs to machines with limited amount of a resource (e.g. time or space), with the objective of maximizing the value of the assignment.

The sets $\mathcal{I}$ and $\mathcal{J}$ of machine and job indices, respectively, are given, together with the following data:

- a_{ij}, the resource required by job j when processed on machine i, $i \in \mathcal{I}$ and $j \in \mathcal{J}$;
- b_i, resource availability of machine i, $i \in \mathcal{I}$;
- c_{ij}, value of assigning job j to machine i, $i \in \mathcal{I}$ and $j \in \mathcal{J}$.

Defining, for $i \in \mathcal{I}$ and $j \in \mathcal{J}$, the decision variable $x_{ij} = 1$ if job j is assigned to machine i and $x_{ij} = 0$ otherwise, (GAP) is then formulated:

$$\begin{cases} \text{maximize} & \displaystyle\sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}} c_{ij}x_{ij} \\ \text{subject to} & \displaystyle\sum_{i\in\mathcal{I}} x_{ij} = 1, \;\; j \in \mathcal{J} \\ & \displaystyle\sum_{j\in\mathcal{J}} a_{ij}x_{ij} \le b_i, \;\; i \in \mathcal{I} \\ & x_{ij} \text{ binary}, \;\; i \in \mathcal{I}, \; j \in \mathcal{J}. \end{cases} \tag{36}$$

A possible relaxation is obtained by acting on the semi-assignment constraints $\sum_{i\in\mathcal{I}} x_{ij} = 1$, $j \in \mathcal{J}$, thus obtaining, for each choice of the multipliers λ_j, $j \in \mathcal{J}$ (they are grouped into vector $\boldsymbol{\lambda}$), the following upper bound $z_{LR}(\boldsymbol{\lambda})$ (note that (36) is a maximization problem, hence the Lagrangian dual is a minimization problem):

$$\begin{cases} z_{LR}(\boldsymbol{\lambda}) = \displaystyle\sum_{j\in\mathcal{J}} \lambda_j + & \max \displaystyle\sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}} (c_{ij} - \lambda_j)x_{ij} \\ & \text{subject to} \displaystyle\sum_{j\in\mathcal{J}} a_{ij}x_{ij} \le b_i, \;\; i \in \mathcal{I} \\ & \qquad x_{ij} \text{ binary}, \;\; i \in \mathcal{I}, \; j \in \mathcal{J}. \end{cases} \tag{37}$$

It is easy to verify that problem (37) decomposes into $|\mathcal{I}|$ binary knapsack subproblems, that is

$$z_{LR}(\boldsymbol{\lambda}) = \sum_{j \in \mathcal{J}} \lambda_j + \sum_{i \in \mathcal{I}} z_{LR}^{(i)}(\boldsymbol{\lambda})$$

where

$$\begin{cases} z_{LR}^{(i)}(\boldsymbol{\lambda}) = & \max \sum_{j \in \mathcal{J}} (c_{ij} - \lambda_j) x_{ij} \\ & \text{subject to } \sum_{j \in \mathcal{J}} a_{ij} x_{ij} \le b_i \\ & \qquad x_{ij} \text{ binary}, \;\; j \in \mathcal{J}. \end{cases} \tag{38}$$

If for any $\boldsymbol{\lambda}$ the solution $\mathbf{x}(\boldsymbol{\lambda})$ of the Lagrangian relaxation satisfies the relaxed constraints, then it is optimal for (GAP) as well. In [23] a judicious selection of the initial values of the multipliers by setting $\lambda_j = \max^{(2)}_{i \in \mathcal{I}} c_{ij}$, where $\max^{(2)}$ indicates the second largest number in a set, makes $\mathbf{x}(\boldsymbol{\lambda})$ satisfy the condition $\sum_{i \in \mathcal{I}} x_{ij}(\boldsymbol{\lambda}) \le 1$, $j \in \mathcal{J}$. Thus the only possible infeasibilities of such solution are of the type $\sum_{i \in \mathcal{I}} x_{ij}(\boldsymbol{\lambda}) = 0$, for one or more jobs.

The core of the algorithm is the multiplier vector update which is driven by such kind of infeasibility. In fact, consider any job h such that $\sum_{i \in \mathcal{I}} x_{ih}(\boldsymbol{\lambda}) = 0$. Any decrease in multiplier λ_h makes all reduced costs $(c_{ih} - \lambda_h)$ of such unassigned job increase. As consequence, job h becomes more competitive in view of possible assignment. The key point of the algorithm is the possibility of calculating exactly the minimum reduction Δ_h of multiplier λ_h which allows, under the new setting, assignment of the previously unassigned job to at least one machine.

This calculation can be performed as follows. It is first calculated Δ_{ih}, the minimum reduction in λ_h which allows assignment of job h to machine i, $i \in \mathcal{I}$. To this aim, the following auxiliary knapsack problem is solved:

$$\begin{cases} z_{LR}^{(i,h)}(\boldsymbol{\lambda}) = & \max \sum_{j \in \mathcal{J},\, j \ne h} (c_{ij} - \lambda_j) x_{ij} \\ & \text{subject to } \sum_{j \in \mathcal{J},\, j \ne h} a_{ij} x_{ij} \le b_i - a_{ih} \\ & \qquad x_{ij} \text{ binary}, \;\; j \in \mathcal{J},\, j \ne h, \end{cases} \tag{39}$$

and then it is

$$\Delta_{ih} = z_{LR}^{(i)}(\boldsymbol{\lambda}) - \big(c_{ih} - \lambda_h + z_{LR}^{(i,h)}(\boldsymbol{\lambda})\big) \ge 0.$$

Finally we have:

$$\Delta_h = \min_{i \in \mathcal{I}} \Delta_{ih},$$

and it is easy to verify that, in case $\Delta_h > 0$ and letting $\boldsymbol{\lambda}^+ = \boldsymbol{\lambda} - \Delta_h \mathbf{e}_h$, function z_{LR} reduces of exactly Δ_h.

We skip here some details of the algorithm, e.g. how to enforce condition $\sum_{i \in \mathcal{I}} x_{ij}(\boldsymbol{\lambda}) \le 1$, $j \in J$ to hold throughout the execution. We wish to emphasize, instead, that at each iteration just one multiplier is updated (thus the algorithm is a co-ordinate descent one). Moreover, unlike the algorithm for Set Covering discussed in previous section, it is not ensured that the number of satisfied constraints increases monotonically. However, termination at a feasible (and hence optimal) solution is guaranteed.

An approach inspired by this method was introduced in [12] to deal with a classic location–allocation problem known as Terminal Location.

5.2 Spanning tree with minimum branch vertices

In previous subsection it has been presented a dual ascent procedure, based on modification of one multiplier at a time, with *exact* calculation of the stepsize along the corresponding coordinate axis. Here we discuss a dual ascent algorithm for the Spanning Tree with Minimum Branch Vertices ($ST-MBV$) problem which still works modifying just one multiplier at a time, but it is equipped with a *line search* along the coordinate axis.

Another application of Lagrangian relaxation to a variant of the Steiner tree problem is in [18].

($ST - MBV$) problem arises in the design of optical networks, where it is necessary to guarantee connection to all nodes of a given network. Thus a spanning tree is to be found. Since at least one switch must be installed at each node of the tree whose degree is greater than two (branch vertices), the problem is to find a Spanning Tree with the Minimum number of Branch Vertices (($ST - MBV$) problem).

This problem admits several formulations. We focus on the Integer Programming one described in [11], where a Lagrangian heuristic is presented in details.

We consider an undirected network $G = (V, E)$, where V denotes the set of n vertices and E the set of m edges. The decision variables are the following:

- x_e, $e \in E$, binary; $x_e = 1$ if edge e is selected and $x_e = 0$ otherwise;
- y_v, $v \in V$, binary; $y_v = 1$ if vertex v is of the branch type (that is its degree, as vertex of the tree, is greater than two), and $y_v = 0$ otherwise.

Then, the (IP) formulation of MBV is the following:

$$\begin{cases} \text{minimize} & \sum_{v \in V} y_v \\ \text{subject to} & \sum_{e \in E} x_e = n-1 \\ & \sum_{e \in E(S)} x_e \leq |S|-1, \ S \subseteq V \\ & \sum_{e \in A(v)} x_e - 2 \leq \delta_v y_v, \ v \ \in V \\ & y_v \text{ binary}, \ \ v \in V; \quad x_e \text{ binary } \ e \in E \end{cases} \tag{40}$$

where for any given subset of vertices S we denote by $E(S)$ the set of edges having both the endpoints in S. Moreover we denote by $A(v)$ the set of incident edges to vertex v and by δ_v its size, i.e. $\delta_v = |A(v)|$. The objective function to be minimized is the total number of branch vertices. Constraints $\sum_{e \in E} x_e = n-1$ and $\sum_{e \in E(S)} x_e \leq |S|-1$, $S \subseteq V$, ensure that a spanning tree is actually detected, while the complicating constraints are $\sum_{e \in A(v)} x_e - 2 \leq \delta_v y_v$, $v \in V$. They guarantee that variable y_v is set to 1 whenever v has more than two incident edges in the selected tree.

By introducing the multipliers $\lambda_v \geq 0$, $v \in V$ (grouped, as usual, in vector $\boldsymbol{\lambda}$), we obtain the following Lagrangian relaxation:

$$\begin{cases} z_{LR}(\boldsymbol{\lambda}) = & \min \sum_{v \in V} y_v + \sum_{v \in V} \lambda_v (\sum_{e \in A(v)} x_e - 2 - \delta_v y_v) \\ & \text{subject to } \sum_{e \in E} x_e = n-1 \\ & \sum_{e \in E(S)} x_e \leq |S|-1, \ S \subseteq V \\ & y_v \text{ binary}, \ \ v \in V; \quad x_e \text{ binary } \ e \in E \end{cases} \tag{41}$$

By simple manipulations, $z_{LR}(\boldsymbol{\lambda})$ may be rewritten as:

$$z_{LR}(\boldsymbol{\lambda}) = -2 \sum_{v \in} \lambda_v + z_{LR}^{(1)}(\boldsymbol{\lambda}) + z_{LR}^{(2)}(\boldsymbol{\lambda}) \tag{42}$$

where $z_{LR}^{(1)}(\boldsymbol{\lambda})$ and $z_{LR}^{(2)}(\boldsymbol{\lambda})$ are defined as follows:

$$\begin{cases} z_{LR}^{(1)}(\boldsymbol{\lambda}) = & \min \sum_{v \in V} y_v (1 - \delta_v \lambda_v) \\ & \text{subject to } y_v \text{ binary}, \ \ v \in V \end{cases} \tag{43}$$

and

$$\begin{cases} z_{LR}^{(2)}(\boldsymbol{\lambda}) = & \min \sum_{v \in V} \sum_{e \in A(v)} \lambda_v x_e \\ & \text{subject to } \sum_{e \in E} x_e = n-1 \\ & \qquad \sum_{e \in E(S)} x_e \leq |S| - 1, \; S \subseteq V \\ & \qquad x_e \text{ binary}, \;\; e \in E. \end{cases} \tag{44}$$

Note that $z_{LR}(\boldsymbol{\lambda})$ is rather easy to calculate. In fact problem (43) is solved by inspection of the cost coefficients, by setting

$$y_v = 1 \text{ if } 1 - \delta_v \lambda_v \leq 0; \; y_v = 0 \text{ otherwise}, \tag{45}$$

while $z_{LR}^{(1)}(\boldsymbol{\lambda})$ is the optimal value of the Minimum Spanning Tree problem where the weight of edge $e = (u, v)$ is $\lambda_u + \lambda_v$.

As for the Lagrangian dual, it is possible to prove [11] that the optimal multiplier vector $\boldsymbol{\lambda}^*$ satisfies the condition

$$\lambda_v^* \leq \frac{1}{\delta_v}, \; v \in V. \tag{46}$$

On the basis of such property it is possible to devise an ascent strategy which modifies one multiplier at a time. Suppose that $(\mathbf{x}(\boldsymbol{\lambda}), \mathbf{y}(\boldsymbol{\lambda}))$ is an optimal solution to the Lagrangian relaxation for a given $\boldsymbol{\lambda}$, then such strategy is at hand whenever one of the two following cases occurs for some $v \in V$:

- Case 1: $\sum_{e \in A(v)} x_e(\boldsymbol{\lambda}) > 2$ and $y_v(\boldsymbol{\lambda}) = 0$;
- Case 2: $\sum_{e \in A(v)} x_e(\boldsymbol{\lambda}) \leq 2$ and $y_v(\boldsymbol{\lambda}) = 1$.

Consider first Case 1 and observe (see the objective function of problem (41)) that the v component $\sum_{e \in A(v)} x_e(\boldsymbol{\lambda}) - 2 - \delta_v y_v(\boldsymbol{\lambda}))$ of a subgradient of z_{LR} is strictly positive, thus one can expect an increase in the objective if λ_v is increased. Observe, in addition, that property (46) suggests to adopt an Armijo-type line search along the co-ordinate direction $\mathbf{e}_v$ with $\frac{1}{\delta_v}$ as initial stepsize. On the other hand, a consequence of nonsmoothnes of z_{LR} is that direction $\mathbf{e}_v$ is not necessarily an ascent one, thus the backtracking line search must accommodate for possible failure (the so-called *null step*, to use the terminology of bundle methods).

As for Case 2, the v subgradient component $\sum_{e \in A(v)} x_e(\boldsymbol{\lambda}) - 2 - \delta_v y_v(\boldsymbol{\lambda}))$ is negative and, in addition (see (45)) it is $\lambda_v > 0$. Thus it is worth to consider a possible reduction on λ_v, that is a move along the co-ordinate direction

$-\mathbf{e}_v$, adopting, also in this case, an Armijo-type line search equipped with possible null step declaration.

The use of the co-ordinate search approach has both advantages and drawbacks. As pointed out in [11], co-ordinate search is definitely faster than standard subgradient but the lower bound is slightly worse due to possible premature stop (see Section 3).

We remark that for this application a feasible solution of problem (40) is available with no additional computational cost at each iteration of the dual ascent procedure. After all, the Lagrangian relaxation provides a spanning tree, thus, to implement a Lagrangian heuristic, it is only needed to calculate the corresponding cost in terms of the objective function of (40).

5.3 Directional sensors location

In Wireless Sensor Networks (WSN) [57] a sensor is a device capable to receive (and possibly store and forward) information coming from a sufficiently close area. In general, such an area is a circle of given radius, whose centre is the sensor location itself. Points inside the area are deemed *covered* by the sensor. In case the area, instead of being a circle, is an adjustable circular sector, the sensor is defined *directional* [13].

The Directional Sensors Continuous Coverage Problem ($DSCCP$) is about covering several *targets*, distributed in a plane, by a set of directional sensors whose locations are known. Each sensor is adjustable, being characterized by a discrete set of possible radii and aperture angles. The sensor orientation is a decision variable too. The model accommodates for possible sensor switch off. Since different power consumption is associated to the sensor adjustments, the objective is to minimize the total power cost of coverage.

We report here the formulation given in [3] as a Mixed Integer Nonlinear Program (MINLP). Note that in most of the literature only a discrete number of sensor orientations are considered (see [53], Lemma 1 and Corollary 1). The motivation for defining, instead, the orientation as a continuous variable is in the choice of Lagrangian relaxation as the attack strategy. It will be clear in the sequel, that solving the relaxed problem is easy once the sensor orientation is assumed to be a continuous variable.

Suppose that a given set $\mathcal{I}$ of sensors is located in a certain area and $\mathbf{s}_i \in \mathbb{R}^2$ is the known position of sensor i, $i \in \mathcal{I}$. The location $\mathbf{t}_j \in \mathbb{R}^2$, $j \in \mathcal{J}$, of the targets is also known, together with the sensor-target distance parameters $\mathbf{d}_{ij} = \mathbf{t}_j - \mathbf{s}_i$ and $c_{ij} = \|\mathbf{t}_j - \mathbf{s}_i\|$, $i \in \mathcal{I}$ and $j \in \mathcal{J}$. A set of $K + 1$ different power levels k, $k = 0, \ldots, K$ can be considered for sensors, each of them corresponding to a couple of values of the radius r_k and of the half-aperture angle α_k of the circular sector. In particular, the level $k = 0$ is associated to an inactive sensor ($r_0 = 0$ and $\alpha_0 = 0$). We also introduce the parameter $q_k = \cos \alpha_k$, $q_k \in [-1, 1]$.

The area covered by the sensor i, activated at level k, is then the revolution cone defined as:

$$\big\{\mathbf{x} \mid \mathbf{x} \in \mathbb{R}^2,\ \|\mathbf{x}-\mathbf{s}_i\| \leq r_k,\ q_k\|\mathbf{x}-\mathbf{s}_i\| \leq (\mathbf{x}-\mathbf{s}_i)^T\mathbf{w}_i\big\}, \tag{47}$$

where $\mathbf{w}_i \in \mathbb{R}^2$, $\|\mathbf{w}_i\| = 1$, is the orientation direction assigned to sensor i, $i \in \mathcal{I}$ ($\|.\|$ is assumed to be the Euclidean norm).

As a consequence, target j is covered by the sensor i, activated at level k, with orientation direction w_i if and only if the following two conditions hold:

$$r_k \geq c_{ij}, \quad \text{and} \quad q_k c_{ij} \leq \mathbf{d}_{ij}^T\mathbf{w}_i. \tag{48}$$

The decision variables are:

- $\mathbf{w}_i \in \mathbb{R}^2$, the orientation direction assigned to sensor i, $i \in \mathcal{I}$;
- x_{ik}, $i \in \mathcal{I}$, $k = 0, \ldots, K$, binary: $x_{ik} = 1$ if sensor i is activated at power level k and $x_{ik} = 0$ otherwise;
- σ_{ij}, $i \in \mathcal{I}$, $j \in \mathcal{J}$, binary: $\sigma_{ij} = 0$ implies that both conditions (48) are satisfied;
- u_j, $j \in \mathcal{J}$, binary: $u_j = 0$ implies that target j is covered by at least one sensor.

The model considers two types of costs:

- p_k, the activation cost for turning on any sensor at power level k, $k = 0, \ldots, K$ (with $p_0 = 0$);
- H, the penalty cost associated to an uncovered target.

Finally ($DSCCP$) can be stated as follows:

$$\begin{cases} \text{minimize} & \displaystyle\sum_{i\in\mathcal{I}}\sum_{k=0}^{K} p_k x_{ik} + H\sum_{j\in\mathcal{J}} u_j \\ \text{subject to} & \displaystyle\sum_{k=0}^{K} x_{ik} = 1,\ \ i \in \mathcal{I} \\ & \displaystyle\sum_{k=0}^{K} x_{ik} r_k \geq c_{ij} - M\sigma_{ij},\ i \in \mathcal{I};\ \ j \in \mathcal{J} \\ & \displaystyle c_{ij}\sum_{k=0}^{K} q_k x_{ik} - \mathbf{d}_{ij}^T\mathbf{w}_i \leq M\sigma_{ij},\ i \in \mathcal{I};\ j \in \mathcal{J} \\ & \displaystyle u_j \geq \sum_{i\in\mathcal{I}} \sigma_{ij} - (|I|-1),\ \ j \in \mathcal{J} \\ & \|\mathbf{w}_i\| = 1,\ \ i \in \mathcal{I} \\ & \mathbf{x}, \boldsymbol{\sigma}, \mathbf{u}\ \text{ binary},\ \mathbf{w}_i \in \mathbb{R}^2,\ \ i \in \mathcal{I}. \end{cases} \tag{49}$$

The model contains the "big M" positive input parameter, which is common in formulation of location problems. Variables x_{ik}, σ_{ij} and u_j are grouped into the vectors x, σ and u, respectively.

The objective function is the sum of activation cost and penalty for possibly uncovered targets.

The first set of constraints are classic semi-assignment, ensuring that exactly one power level (possibly the 0-level) is assigned to each sensor.

Constraints:

$$\begin{cases} \displaystyle\sum_{k=0}^{K} x_{ik} r_k \geq c_{ij} - M\sigma_{ij}, \ i \in \mathcal{I}; \quad j \in \mathcal{J} \\ c_{ij} \displaystyle\sum_{k=0}^{K} q_k x_{ik} - \mathbf{d}_{ij}^T \mathbf{w}_i \leq M\sigma_{ij}, \ i \in \mathcal{I}; \ j \in \mathcal{J}, \end{cases} \tag{50}$$

are related to target coverage. They are trivially satisfied if $\sigma_{ij} = 1$ but, whenever they are satisfied with $\sigma_{ij} = 0$, then target j is covered by sensor i (see (48)).

Constraints

$$u_j \geq \sum_{i \in \mathcal{I}} \sigma_{ij} - (|\mathcal{I}| - 1), \ \ j \in \mathcal{J} \tag{51}$$

impose that, for any target j, $u_j = 1$ in case $\sigma_{ij} = 1$ for all $i \in \mathcal{I}$, that is if target j remains uncovered. At the optimum $u_j = 1$ if and only if $\sum_{i \in \mathcal{I}} \sigma_{ij} = |\mathcal{I}|$.

Finally the (nonconvex) constraints $\|\mathbf{w}_i\| = 1$, $i \in \mathcal{I}$, aim at normalizing the orientation direction assigned to each sensor.

By introducing the nonnegative multiplier vectors $\boldsymbol{\lambda}$, $\boldsymbol{\theta}$ and $\boldsymbol{\gamma}$, the following Lagrangian relaxation is defined

$$\begin{cases} z_{LR}(\boldsymbol{\lambda}, \boldsymbol{\theta}, \boldsymbol{\gamma}) = & \min \displaystyle\sum_{i \in \mathcal{I}} \sum_{k=0}^{K} p_k x_{ik} + H \sum_{j=1}^{n} u_j + \\ & + \displaystyle\sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}} \lambda_{ij} (c_{ij} - M\sigma_{ij} - \sum_{k=0}^{K} x_{ik} r_k) + \\ & + \displaystyle\sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}} \theta_{ij} (c_{ij} \sum_{k=0}^{K} x_{ik} q_k - \mathbf{d}_{ij}^T \mathbf{w}_i - M\sigma_{ij}) + \\ & + \displaystyle\sum_{j \in \mathcal{J}} \gamma_j (\sum_{i \in \mathcal{I}} \sigma_{ij} - (|\mathcal{I}| - 1) - u_j) \\ & \text{subject to} \displaystyle\sum_{k=0}^{K} x_{ik} = 1, \quad i \in \mathcal{I} \\ & \qquad \|\mathbf{w}_i\| = 1, \ \ i \in \mathcal{I} \\ & \qquad \mathbf{x}, \boldsymbol{\sigma}, \mathbf{u} \ \text{binary}, \ \mathbf{w}_i \in \mathbb{R}^2, \ i \in \mathcal{I}, \end{cases} \tag{52}$$

which, rearranging the objective function, may be rewritten as

$$\begin{cases} z_{LR}(\boldsymbol{\lambda},\boldsymbol{\theta},\boldsymbol{\gamma}) = & \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}}\lambda_{ij}c_{ij} + (1-|\mathcal{I}|)\sum_{j\in\mathcal{J}}\gamma_j + \\ & \min\sum_{i\in\mathcal{I}}\sum_{k=0}^{K}[p_k + \sum_{j\in\mathcal{J}}(\theta_{ij}c_{ij}q_k - \lambda_{ij}r_k)]x_{ik} + \\ & \quad + \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}}[\gamma_j - M(\lambda_{ij}+\theta_{ij})]\sigma_{ij} + \\ & \quad - \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}}\theta_{ij}\mathbf{d}_{ij}^T\mathbf{w}_i + \sum_{j\in\mathcal{J}}(H-\gamma_j)u_j \\ & \text{subject to} \sum_{k=0}^{K}x_{ik} = 1, \;\; i\in\mathcal{I} \\ & \qquad \|\mathbf{w}_i\| = 1, \;\; i\in\mathcal{I} \\ & \qquad \mathbf{x},\boldsymbol{\sigma},\mathbf{u} \text{ binary}, \; \mathbf{w}_i\in\mathbb{R}^2, \; i\in\mathcal{I}. \end{cases} \tag{53}$$

The above formulation leads to the following decomposition:

$$z_{LR}(\boldsymbol{\lambda},\boldsymbol{\theta},\boldsymbol{\gamma}) = C(\boldsymbol{\lambda},\boldsymbol{\gamma}) + z_{LR}^{(1)}(\boldsymbol{\lambda},\boldsymbol{\theta}) + z_{LR}^{(2)}(\boldsymbol{\lambda},\boldsymbol{\theta},\boldsymbol{\gamma}) + z_{LR}^{(3)}(\boldsymbol{\theta}) + z_{LR}^{(4)}(\boldsymbol{\gamma}), \tag{54}$$

where

$$C(\boldsymbol{\lambda},\boldsymbol{\gamma}) = \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}}\lambda_{ij}c_{ij} + (1-|\mathcal{I}|)\sum_{j\in J}\gamma_j, \tag{55}$$

$$\begin{cases} z_{LR}^{(1)}(\boldsymbol{\lambda},\boldsymbol{\theta}) = & \min\sum_{i\in\mathcal{I}}\sum_{k=0}^{K}[p_k + \sum_{j\in\mathcal{J}}(\theta_{ij}c_{ij}q_k - \lambda_{ij}r_k)]x_{ik} \\ & \text{subject to} \sum_{k=0}^{K}x_{ik} = 1, \;\; i\in\mathcal{I} \\ & \qquad \mathbf{x} \text{ binary}, \end{cases} \tag{56}$$

$$\begin{cases} z_{LR}^{(2)}(\boldsymbol{\lambda},\boldsymbol{\theta},\boldsymbol{\gamma}) = & \min\sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}}[\gamma_j - M(\lambda_{ij}+\theta_{ij})]\sigma_{ij} \\ & \text{subject to } \boldsymbol{\sigma} \text{ binary}, \end{cases} \tag{57}$$

$$\begin{cases} z_{LR}^{(3)}(\boldsymbol{\theta}) = & -\max\sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}}\theta_{ij}\mathbf{d}_{ij}^T\mathbf{w}_i \\ & \text{subject to } \|\mathbf{w}_i\| = 1, \; i\in\mathcal{I} \\ & \qquad \mathbf{w}_i\in\mathbb{R}^2, \; i\in\mathcal{I}, \end{cases} \tag{58}$$

$$\begin{cases} z_{LR}^{(4)}(\boldsymbol{\gamma}) = & \min \sum_{j \in \mathcal{J}} (H - \gamma_j) u_j \\ & \text{subject to } \mathbf{u} \text{ binary.} \end{cases} \tag{59}$$

Calculation of constant $C(\boldsymbol{\lambda}, \boldsymbol{\gamma})$ is immediate. The solution of problems (56), (57) and (59) can be obtained by simple inspection of the corresponding cost coefficients, while problem (58) has optimal solution $\mathbf{w}_i(\boldsymbol{\theta})$, $i \in \mathcal{I}$, in the following closed form:

$$\mathbf{w}_i(\boldsymbol{\theta}) = \begin{cases} \dfrac{\sum_{j \in \mathcal{J}} \theta_{ij} \mathbf{d}_{ij}}{\| \sum_{j \in \mathcal{J}} \theta_{ij} \mathbf{d}_{ij} \|}, & \text{if } \sum_{j \in \mathcal{J}} \theta_{ij} \mathbf{d}_{ij} \neq \mathbf{0} \\ \text{any vector } \mathbf{w} \in \mathbb{R}^2, \ \|\mathbf{w}\| = 1, & \text{otherwise.} \end{cases} \tag{60}$$

In [3] the Lagrangian dual is tackled by applying a coordinate search method which modifies one multiplier at a time, in view of possible increase of function $z_{LR}(\boldsymbol{\lambda}, \boldsymbol{\theta}, \boldsymbol{\gamma})$. The choice of such multiplier is driven, as usual, by the type of infeasibility occurring at the current solution of the Lagrangian relaxation. Thus different rules are designed, according to the choice of the component either of $\boldsymbol{\lambda}$ or $\boldsymbol{\theta}$ or $\boldsymbol{\gamma}$ to be updated. For the details of such rules we refer the reader to [3].

The dual ascent procedure is particularly suitable for embedding a Lagrangian heuristic. In fact the solution of the Lagrangian relaxation can be made feasible at a quite low computational cost, hence providing an upper bound.

More specifically, whenever the solution to (52) is not feasible for (49), the set $\bar{\mathcal{J}}$ of possibly uncovered targets is considered. For each of them at least one of the following conditions holds:

$$\sum_{k=0}^{K} x_{ik} r_k < c_{ij} \quad \forall i \in \mathcal{I}, \tag{61}$$

$$c_{ij} \sum_{k=0}^{K} q_k x_{ik} > \mathbf{d}_{ij}^T \mathbf{w}_i \quad \forall i \in \mathcal{I} \tag{62}$$

at the optimum of the Lagrangian relaxation. Thus, for each target $j \in \bar{\mathcal{J}}$, possible coverage is sought by acting on the power levels (variables $\mathbf{x}$) assigned to those sensors which are eligible for covering it. In such search, sensor orientations (variables $\mathbf{w}_i$) returned by the Lagrangian relaxation are not modified.

Summing up, at each iteration of the dual ascent procedure both a lower and an upper bounds are calculated. In [3] it is highlighted that while the lower bound provided by the algorithm is extremely poor, the best feasible solution found is, in general, of good quality also for relatively large scale problems. It is worth noting that the algorithm is equipped with possible restart along a subgradient direction in case the Armijo line seach fails along all coordinate directions.

5.4 Cross Docking scheduling

A cross docking distribution centre (CD centre, in the following) is a modern logistic node where goods are unloaded from inbound trucks, consolidated with respect to the customer orders and then immediately loaded on outbound trucks, cancelling in practice the traditional and costly storing and retrieval phases. Management of a CD centre is a serious challenge since the processes of arrival and departure of goods are strongly coupled and sophisticated synchronization schemes are to be fulfilled.

Intensive research efforts have been made to devise effective models and algorithms for optimizing the combined scheduling of the inbound and outbound trucks; more recently Lagrangian relaxation has been applied in this area (see[15] and the references therein). The problem addressed is about finding the optimal inbound and outbound sequences at a CD centre characterized by only two gates (or doors), one for the inbound and the other for the outbound trucks. The objective is to minimize the total completion time (the *makespan* in scheduling theory parlance).

The rules of the game are the following:

- only one truck at a time can be handled at a door and no interruption (*preemption*) is allowed;
- the loading of an outbound truck cannot start until all goods it is expected to deliver have been unloaded;
- all trucks require the same processing time (one *slot*) and are ready at the time 0.

As input data we consider a set of n inbound trucks ($\mathcal{I} = \{1, \dots, n\}$) to be unloaded, together with a set of m outbound trucks ($\mathcal{J} = \{1, \dots, m\}$) to be loaded. The following sets are also given:

- $\mathcal{J}_i$: the set of outbound trucks receiving goods from the inbound truck i, $i \in \mathcal{I}$;
- $\mathcal{I}_j$: the set of inbound trucks providing goods to the outbound truck j, $j \in \mathcal{J}$;

The planning horizon is discretized into time-slots, each of them being capable to accommodate for processing of one truck. Let $\mathcal{K} = \{1, \dots, n\}$ and $\L = \{1, \dots, H\}$, $\ H \geq m + n$, be the time horizon for the inbound and outbound services, respectively (note that, under the assumptions made, $n + m$ is an upper bound on the makespan).

Introducing the following binary decision variables:

- $x_{ik} = 1$ if the inbound truck i is assigned to the time-slot k and $x_{ik} = 0$ otherwise, $i \in \mathcal{I}$, $k \in \mathcal{K}$;
- $y_{jh} = 1$ if the outbound truck j is assigned to the time-slot h, $y_{jh} = 0$ otherwise, $j \in \mathcal{J}$, $h \in \text{Ł}$;

and the integer variable C_{Max}, the makespan, then the One Door Cross Docking ($ODCD$) problem is modelled as follows:

$$\left\{\begin{aligned}
Z = \quad & \min C_{Max} \\
& \text{subject to} \sum_{k \in \mathcal{K}} x_{ik} = 1, \quad i \in \mathcal{I} \\
& \qquad\qquad \sum_{i \in \mathcal{I}} x_{ik} = 1, \quad k \in \mathcal{K} \\
& \qquad\qquad \sum_{h \in \text{Ł}} y_{jh} = 1, \quad j \in \mathcal{J} \\
& \qquad\qquad \sum_{j \in \mathcal{J}} y_{jh} \leq 1, \quad h \in \text{Ł} \\
& \qquad\qquad C_{Max} \geq \sum_{h=1}^{H} h y_{jh}, \quad j \in \mathcal{J} \\
& \qquad\qquad \sum_{h=1}^{H} h y_{jh} \geq \sum_{k=1}^{n} k x_{ik} + 1, \quad j \in \mathcal{J},\ i \in \mathcal{I}_j \\
& x_{ik} \text{ binary, } i \in \mathcal{I},\ k \in \mathcal{K}; \quad y_{jh} \text{ binary, } j \in \mathcal{J},\ h \in \text{Ł}.
\end{aligned}\right. \tag{63}$$

The first four constraint sets regulate the time slot-truck assignment at both the inbound and outbound door. The constraints $C_{Max} \geq \sum_{h=1}^{H} h y_{jh}$, $j \in \mathcal{J}$ define the makespan C_{Max} as the maximum truck completion time, of course on the outbound side. The constraints

$$\sum_{h=1}^{H} h y_{jh} \geq \sum_{k=1}^{n} k x_{ik} + 1, \quad j \in J,\ i \in I_j,$$

ensure that loading of each outbound truck cannot start until the unloading of all corresponding inbound trucks has been completed. They are the complicating ones and lead to the following Lagrangian relaxation obtained via the multiplier vector $\boldsymbol{\lambda} \geq 0$.

$$\left\{\begin{array}{l} z_{LR}(\boldsymbol{\lambda}) = \min\big\{C_{Max} + \sum_{j\in\mathcal{J}}\sum_{i\in\mathcal{I}_j}\lambda_{ij}\big(\sum_{k\in\mathcal{K}} kx_{ik} - \sum_{h\in \text{Ł}} hy_{jh} + 1\big)\big\} \\ \text{subject to} \sum_{k\in\mathcal{K}} x_{ik} = 1, \quad i\in\mathcal{I} \\ \qquad\qquad \sum_{i\in\mathcal{I}} x_{ik} = 1, \quad k\in\mathcal{K} \\ \qquad\qquad \sum_{h\in \text{Ł}} y_{jh} = 1, \quad j\in\mathcal{J} \\ \qquad\qquad \sum_{j\in\mathcal{J}} y_{jh} \le 1, \quad h\in \text{Ł} \\ \qquad\qquad C_{Max} \ge \sum_{h=1}^{H} hy_{jh}, \quad j\in\mathcal{J} \\ x_{ik} \text{ binary}, \ i\in\mathcal{I},\ k\in\mathcal{K}; \quad y_{jh} \text{ binary},\ j\in\mathcal{J},\ h\in \text{Ł}. \end{array}\right. \tag{64}$$

The relaxation decomposes into two independent matching problems, related, respectively, to the inbound and outbound door. In fact, by simple manipulations based on the observation $i \in \mathcal{I}_j \Leftrightarrow j \in \mathcal{J}_i$, and setting:

$$s = \sum_{j\in\mathcal{J}}\sum_{i\in\mathcal{I}_j}\lambda_{ij} = \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}_i}\lambda_{ij}; \tag{65}$$

$$\rho_i = \sum_{j\in\mathcal{J}_i}\lambda_{ij},\ i\in\mathcal{I}; \quad \sigma_j = \sum_{i\in\mathcal{I}_j}\lambda_{ij},\ j\in\mathcal{J}, \tag{66}$$

we rewrite z_{LR} as a function of vectors $\boldsymbol{\rho}$ and $\boldsymbol{\sigma}$ (grouping the ρ_is and the σ_js, respectively) as follows

$$z_{LR}(\boldsymbol{\rho},\boldsymbol{\sigma}) = s + \min\big\{\sum_{i\in\mathcal{I}}\sum_{k\in\mathcal{K}} k\rho_i x_{ik} + C_{Max} - \sum_{j\in\mathcal{J}}\sum_{h\in \text{Ł}} h\sigma_j y_{jh}\big\}. \tag{67}$$

Thus we define:

$$z_{LR}(\boldsymbol{\rho},\boldsymbol{\sigma}) = s + z^{(1)}_{LR}(\boldsymbol{\rho}) + z^{(2)}_{LR}(\boldsymbol{\sigma}),$$

with

$$\begin{cases} z_{LR}^{(1)}(\boldsymbol{\rho}) = & \min \sum_{i\in\mathcal{I}} \sum_{k\in\mathcal{K}} k\rho_i x_{ik} \\ & \text{subject to} \sum_{k\in\mathcal{K}} x_{ik} = 1, \quad i \in \mathcal{I} \\ & \qquad \sum_{i\in\mathcal{I}} x_{ik} = 1, \quad k \in \mathcal{K} \\ & x_{ik} \text{ binary}, \ i \in \mathcal{I}, \ k \in \mathcal{K}, \end{cases} \tag{68}$$

and

$$\begin{cases} z_{LR}^{(2)}(\boldsymbol{\sigma}) = & \min C_{Max} - \sum_{j\in\mathcal{J}} \sum_{h\in\text{Ł}} h\sigma_j y_{jh} \\ & \text{subject to} \sum_{h\in\text{Ł}} y_{jh} = 1, \quad j \in \mathcal{J} \\ & \qquad \sum_{j\in\mathcal{J}} y_{jh} \le 1, \quad h \in \text{Ł} \\ & \qquad C_{Max} \ge \sum_{h=1}^{H} h y_{jh}, \quad j \in \mathcal{J} \\ & y_{jh} \text{ binary}, \ j \in \mathcal{J}, \ h \in \text{Ł}. \end{cases} \tag{69}$$

Problems (68) and (69) [15] are two simple single machine scheduling problems which can be solved by appropriate sorting of the vectors $\boldsymbol{\rho}$ and $\boldsymbol{\sigma}$.

The following holds for the Lagrangian dual.

Theorem 1. *There exists an optimal solution to the Lagrangian Dual Problem such that*

$$s = \sum_{i\in I} \sum_{j\in J_i} \lambda_{ij} = \sum_{i\in I} \rho_i = \sum_{j\in J} \sum_{i\in I_j} \lambda_{ij} = \sum_{j\in J} \sigma_j \le 1. \tag{70}$$

The above property is helpful in designing algorithms to solve the Lagrangian dual, mainly as it enables appropriate sizing of the stepsize in implementing a line search.

It is worth noting that $(ODCD)$ does not enjoy the integrality property (see, in particular, problem (69)), thus the lower bound obtained from the Lagrangian dual is more accurate than that provided by the (LP) relaxation.

In addition, any solution of a Lagrangian relaxation which is not feasible for $(ODCD)$ can be *repaired*, at low computational cost, by implementing some heuristic method based on simple forward-shifting of those outbounds trucks j for which the relaxed constraints

$$\sum_{h=1}^{H} h y_{jh} \ge \sum_{k=1}^{n} k x_{ik} + 1$$

have been violated for at least one $i \in \mathcal{I}_j$ at the optimum of (64).

In [15] the Lagrangian dual has been tackled by resorting to a standard (projected) subgradient method, which has been equipped, at each iteration, with a repairing procedure, thus allowing to possibly update both the lower and upper bounds.

The Lagrangian heuristic has provided satisfactory results on test problem characterized by a number of inbound trucks n up to 20 and a number of outbound ones m in the range $[10, 40]$.

5.5 Feature selection in Support Vector Machines (SVM)

Feature Selection is a relevant issue in Machine Learning and, in particular, in Pattern Classification. In fact classification is a kind of diagnostic process which consists in attaching a *label* (that is certifying exactly one class membership) to an individual (a *sample* or a *pattern*), on the basis of a given number of known parameters (the *features*). Most of the research work has been focussed on binary classification, where the classes are only two. A binary *classifier* then is a tool which is able to attach the appropriate label to a sample whose class membership is unknown.

The classification models we are dealing with are of the *supervised* type, since the classifier is constructed on the basis of the information provided by a certain number of samples (the *training set*) whose class membership is known.

A fundamental paradigm to construct such classifier is the Support Vector Machine (SVM) [16], [48] which consists of separating the samples belonging to the training set by means of a hyperplane, either in the feature space or in a higher dimension one, upon an appropriate kernel transformation. Once the hyperplane has been calculated, it is used to classify newly incoming patterns whose class membership is unknown.

In standard SVM approach, the training set is formed by two given point-sets $A = \{\mathbf{a}_i, i \in \mathcal{I}\}$ and $B = \{\mathbf{b}_j, j \in \mathcal{J}\}$ in $\mathbb{R}^n$ (the feature space). The problem is about finding a hyperplane defined by a couple $(\mathbf{w} \in \mathbb{R}^n, \gamma \in \mathbb{R})$ that strictly separates A and B. Thus one would require

$$\mathbf{w}^T\mathbf{a}_i + \gamma \leq -1, \;\; i \in \mathcal{I} \tag{71}$$

and

$$\mathbf{w}^T\mathbf{b}_j + \gamma \geq 1, \;\; j \in \mathcal{J}. \tag{72}$$

Since the necessary and sufficient condition for existence of such strictly separating hyperplane

$$\text{conv}\{A\} \cap \text{conv}\{B\} = \emptyset$$

is, in general, not known to hold in advance, we define the *point classification error functions* $\xi_i(\mathbf{w},\gamma)$, $i \in \mathcal{I}$ and $\zeta_j(\mathbf{w},\gamma)$, $j \in \mathcal{J}$ as follows:

$$\xi_i(w,\gamma) = \max\left(0, 1 + (\mathbf{w}^T\mathbf{a}_i + \gamma)\right), \;\; i \in \mathcal{I}, \tag{73}$$

and

$$\zeta_j(w,\gamma) = \max\left(0, 1 - (\mathbf{w}^T\mathbf{b}_j + \gamma)\right), \;\; j \in \mathcal{J}. \tag{74}$$

Note that ξ_i and ζ_j are positive if and only if (71) and (72) are violated, respectively. Consequently they can be considered as a measure of the classification error related to point $\mathbf{a}_i \in A$ and $\mathbf{b}_j \in B$.

The convex and nonsmooth error function $E(\mathbf{w},\gamma)$ is then defined as:

$$\begin{aligned} E(\mathbf{w},\gamma) =& \sum_{i\in\mathcal{I}} \max\left(0, 1 + (\mathbf{w}^T\mathbf{a}_i + \gamma)\right) + \sum_{j\in\mathcal{J}} \max\left(0, 1 - (\mathbf{w}^T\mathbf{b}_j + \gamma)\right) = \\ =& \sum_{i\in\mathcal{I}} \xi_i(w,\gamma) + \sum_{j\in\mathcal{J}} \zeta_j(w,\gamma). \end{aligned}$$

Finally we come out with the standard SVM model:

$$\left\{ \begin{array}{l} \min\limits_{\mathbf{w},\gamma,\boldsymbol{\xi},\boldsymbol{\zeta}} \|\mathbf{w}\|^2 + C(\sum\limits_{i\in\mathcal{I}} \xi_i + \sum\limits_{j\in\mathcal{J}} \zeta_j) \\ \text{subject to} \\ \mathbf{a}_i^\top \mathbf{w} + \gamma \le \xi_i - 1, \quad i \in \mathcal{I} \\ -\mathbf{b}_l^\top \mathbf{w} - \gamma \le \zeta_j - 1 \quad j \in \mathcal{J} \\ \xi_i \ge 0, \quad i \in \mathcal{I} \\ \;\zeta_j \ge 0, \quad j \in J. \end{array} \right. \tag{75}$$

The objective function is the sum of the error function E weighted by the parameter $C > 0$ and the square of the norm of $\mathbf{w}$. The latter term is aimed at maximising the *separation margin* between the two sets A and B. In fact (see [16], Ch.6) $\frac{2}{\|w\|}$ is the distance (the separation margin) between the hyperplanes $H^- = \{\mathbf{x} | \mathbf{w}^T\mathbf{x} + b = -1\}$ and $H^+ = \{\mathbf{x} | \mathbf{w}^T\mathbf{x} + b = 1\}$, thus minimization of $\|w\|$ leads to maximization of the margin. In practical applications the squared norm $\|\mathbf{w}\|^2$ replaces $\|w\|$ in the objective function.

In the SVM framework the role of Feature Selection (FS) is to detect those features that are really relevant for classification purposes. In other words a classifier embedding a FS mechanism is expected to guarantee classification correctness (effective separation of A and B) and, also, to be *parsimonious*, that is to provide a vector $\mathbf{w}$ (the normal to the separating hyperplane) with as few as possible non-zero components.

Several different optimization-based approaches for Feature Selection are available in literature; we cite here, among the others, [10] and [50]. We focus on treatment of FS problem via mixed integer programming (see [45], [8]).

In particular, we refer to the model described in [30], where a Lagrangian relaxation approach has been implemented.

Aiming at enforcing a feature selection mechanism, that is at reducing the number of the non-zero components of $\mathbf{w}$, the binary *feature variable* vector $\mathbf{y} \in \mathbb{R}^n$ is embedded into the model (75), with y_k indicating whether or not feature k is active. The following Mixed Binary formulation of the SVM-feature-selection problem is stated:

$$\begin{cases} \min\limits_{\mathbf{w},\gamma,\boldsymbol{\xi},\boldsymbol{\zeta}} \|\mathbf{w}\| + C(\sum\limits_{i\in\mathcal{I}} \xi_i + \sum\limits_{j\in\mathcal{J}} \zeta_j) + D\sum\limits_{k=1}^{n} y_k \\ \text{subject to} \\ \mathbf{a}_i^\top \mathbf{w} + \gamma \le \xi_i - 1, \quad i \in \mathcal{I} \\ -\mathbf{b}_l^\top \mathbf{w} - \gamma \le \zeta_j - 1, \quad j \in \mathcal{J} \\ -u_k y_k \le w_k \le u_k y_k, \quad k = 1,\dots,n \\ -u_k \le w_k \le u_k, \qquad k = 1,\dots,n \\ \xi_i \ge 0, \quad i \in \mathcal{I} \\ \zeta_j \ge 0, \quad j \in J \\ y_k \text{ binary}, \;\; k = 1,\dots,n. \end{cases} \tag{76}$$

The objective function of problem (76) is the sum of three terms. The norm $\|w\|$, as usual in SVM-type models, is intended to maximise the separation margin (we note in passing that in [30] the L_1 norm, instead of the more commonly used L_2, is adopted). The second term is the error function $E(w,\gamma)$ and, finally, the third one represents the number of non-zero components of w. Note also the presence of the positive weights C and D in the objective function and of the upper bounds $u_k > 0$ on the modulus of each component w_k of $\mathbf{w}$.

In [30] problem above has been tackled by resorting to Lagrangian relaxation of the constraints linking the variables $\mathbf{w}$ and $\mathbf{y}$, by means of the multiplier vectors $\boldsymbol{\lambda} \ge 0$ and $\boldsymbol{\mu} \ge 0$. Thus we obtain:

$$\left\{\begin{array}{ll} z_{LR}(\boldsymbol{\lambda},\boldsymbol{\mu}) = & \displaystyle\min_{\mathbf{w},\gamma,\boldsymbol{\xi},\boldsymbol{\zeta},\mathbf{y}} \|\mathbf{w}\| + C\left(\sum_{i=1}^{m_1}\xi_i + \sum_{l=1}^{m_2}\zeta_l\right) + D\sum_{k=1}^{n} y_k + \\ & \displaystyle\sum_{k=1}^{n}\lambda_k(w_k - u_k y_k) - \sum_{k=1}^{n}\mu_k(w_k + u_k y_k) \\ & \text{subject to} \\ & \mathbf{a}_i^\top \mathbf{w} + \gamma \le \xi_i - 1, \quad i \in \mathcal{I} \\ & -\mathbf{b}_l^\top \mathbf{w} - \gamma \le \zeta_j - 1, \quad j \in \mathcal{J} \\ & -u_k \le w_k \le u_k, \qquad k = 1,\dots,n \\ & \xi_i \ge 0, \quad i \in \mathcal{I} \\ & \zeta_j \ge 0, \quad j \in J \\ & y_k \text{ binary}, \ \ k = 1,\dots,n. \end{array}\right. \tag{77}$$

By rearranging the objective function, we get to the following decomposed formulation

$$z_{LR}(\boldsymbol{\lambda},\boldsymbol{\mu}) = z_{LR}^{(1)}(\boldsymbol{\lambda},\boldsymbol{\mu}) + z_{LR}^{(2)}(\boldsymbol{\lambda},\boldsymbol{\mu})$$

with

$$\left\{\begin{array}{ll} z_{LR}^{(1)}(\boldsymbol{\lambda},\boldsymbol{\mu}) = & \displaystyle\min_{\mathbf{w},\gamma,\boldsymbol{\xi},\boldsymbol{\zeta}} \|\mathbf{w}\|^2 + C\left(\sum_{i=1}^{m_1}\xi_i + \sum_{l=1}^{m_2}\zeta_l\right) + \sum_{k=1}^{n}(\lambda_k - \mu_k)w_k \\ & \text{subject to} \\ & \mathbf{a}_i^\top \mathbf{w} + \gamma \le \xi_i - 1, \quad i \in \mathcal{I} \\ & -\mathbf{b}_l^\top \mathbf{w} - \gamma \le \zeta_j - 1, \quad j \in \mathcal{J} \\ & -u_k \le w_k \le u_k, \qquad k = 1,\dots,n \\ & \xi_i \ge 0, \quad i \in \mathcal{I} \\ & \zeta_j \ge 0, \quad j \in J \end{array}\right. \tag{78}$$

and

$$\left\{\begin{array}{ll} z_{LR}^{(2)}(\boldsymbol{\lambda},\boldsymbol{\mu}) = & \displaystyle\min_{\mathbf{y}} \sum_{k=1}^{n}(D - u_k(\lambda_k + \mu_k))y_k \\ & \text{subject to} \\ & y_k \text{ binary}, \ \ k = 1,\dots,n. \end{array}\right. \tag{79}$$

Note that calculation of $z_{LR}^{(1)}(\boldsymbol{\lambda},\boldsymbol{\mu})$ requires solution of a problem SVM-like, the only difference being the presence of the linear term $\sum_{k=1}^{n}(\lambda_k - \mu_k)w_k$ into the objective function. As for the second problem, $z_{LR}^{(2)}(\boldsymbol{\lambda},\boldsymbol{\mu})$ can be simply calculated by sign inspection of the cost coefficients $(D - u_k(\lambda_k + \mu_k))$, $k = 1, \dots, n$.

As for the Lagrangian dual

$$z_{LD} = \max_{(\boldsymbol{\lambda}, \boldsymbol{\mu}) \geq 0} z_{LR}(\boldsymbol{\lambda}, \boldsymbol{\mu}),$$

the following proposition [30] holds:

Proposition 1. *There exists an optimal solution to the Lagrangian dual satisfying the condition*

$$u_k(\lambda_k + \mu_k) = D, \quad k = 1, \dots, n. \tag{80}$$

In [30] the above property is utilized to eliminate the variables μ_k, $k = 1, \dots, n$ in the Lagrangian dual, which is then tackled by resorting to the general purpose C++ bundle code developed in [24].

Solution of the Lagrangian dual is embedded into a Lagrangian heuristic algorithm. Note in fact that a feasible solution (and consequently an upper bound) for problem (76) can be easily obtained starting from the optimal $\mathbf{w}(\boldsymbol{\lambda})$ obtained by solving the Lagrangian relaxation for any feasible choice of the multiplier vector $\boldsymbol{\lambda}$. It is in fact sufficient to set $y_k = 1$ whenever $|w_k(\boldsymbol{\lambda})| > 0$ and $y_k = 0$ otherwise.

Also in this application the Lagrangian heuristic approach has proved to work well. The numerical results [30] are quite satisfactory, particularly in terms of trade-off between the classification quality and the number of active features.

5.6 Multiple instance learning

Multiple Instance Learning (MIL) [1] is a classification paradigm, connected to SVM [16], which is capable to handle complex problems, mainly in medical image analysis and in text categorization. While the objective of SVM-like methods is to classify samples in a given feature space, MIL deals with classification of *sets* of samples, bags in Machine Learning parlance. We discuss here a specific MIL problem where binary classification of bags is considered. The approach has been introduced in [2] and Lagrangian relaxation has been applied in [4], giving rise to a Lagrangian heuristic algorithm.

To provide an intuitive explanation of the problem, we describe an example driven from image classification where the objects to be classified are images, each of them representing a certain mix of geometric figures (e.g. triangles, squares, circles and stars) of different size. Each image is a bag which is segmented according to some appropriate rule [58] and in turn a feature vector in $\mathbb{R}^n$ is associated to each image segment, where aggregate information about segment luminosity, texture, geometry etc. are reported. Thus each image is represented by a set of points (instances), each of them being a vector in the feature spaces representing one of its segments. In our example the problem at hand is to discriminate between images containing at least one star (positive images) and those which contain no star (negative ones). In the supervised classification framework, the class membership of the images is known, thus each of them is labelled either as positive or negative.

A classic SVM scheme, aimed at separating by means of a hyperplane the instances of positive bags from those of the negative ones, does not seem profitable in this case. In fact the similarity degree between positive and negative images is high (after all triangles, square and circles may definitely appear in images from both classes) and, consequently, the expected separation quality is poor.

MIL introduces a different paradigm. It is assumed in fact that a hyperplane in the feature space correctly classifies the images if *all* instances of each negative bag are in the same halfspace, while *at least* one instance of each positive bag is on the other side.

The mixed integer nonlinear programming (MINLP) formulation of the problem proposed in [2] follows.

Assume m positive bags are given and let $J^+ = \{\mathcal{J}_1^+, \dots, \mathcal{J}_m^+\}$ be the family of the index sets of the instances of each bag. Analogously let $J^- = \{\mathcal{J}_1^-, \dots, \mathcal{J}_k^-\}$ be the family of the index sets for k given negative bags. We indicate by $\mathbf{x}_j \in \mathbb{R}^n$ the j-th instance belonging to a positive or negative bag.

In classic *instance based* SVM, we would look for a hyperplane defined by a couple $(\mathbf{w} \in \mathbb{R}^n, \gamma \in \mathbb{R})$ separating the instances belonging to the negative bags from those belonging to the positive ones.

Instead, according to [2], one looks for a hyperplane $H(\mathbf{w}, \gamma) = \{\mathbf{x} | \mathbf{w}^T\mathbf{x} + \gamma = 0\}$ such that:

i) All negative bags are contained in the set $S^- = \{\mathbf{x} | \mathbf{w}^T\mathbf{x} + \gamma \leq -1\}$;
ii) At least one instance of each positive bag belongs to the set $S^+ = \{\mathbf{x} | \mathbf{w}^T\mathbf{x} + \gamma \geq 1\}$.

The following optimization model, aimed at finding such a hyperplane, is then introduced. The decision variables, apart the couple $(\mathbf{w} \in \mathbb{R}^n, \gamma \in R)$, are the labels $y_j \in \{-1, 1\}$ to be assigned to all instances of the positive bags. The twofold objective consists of minimizing the classification error (which is equal to zero in case a separating hyperplane is actually found) and of maximizing the separation margin, defined as the distance between the shifted hyperplanes $H^- = \{\mathbf{x} | \mathbf{w}^T\mathbf{x} + \gamma = -1\}$ and $H^+ = \{\mathbf{x} | \mathbf{w}^T\mathbf{x} + \gamma = 1\}$ (see Subsection 5.5).

$$\begin{cases} z^* = \min\limits_{\mathbf{w},\gamma,\mathbf{y}} f(\mathbf{w}, \gamma, \mathbf{y}) \\ \qquad \displaystyle\sum_{j \in \mathcal{J}_i^+} \frac{y_j + 1}{2} \geq 1, \quad i = 1, \dots, m \\ \qquad y_j \in \{-1, 1\}, \quad j \in \mathcal{J}_i^+, \quad i = 1, \dots, m, \end{cases} \tag{81}$$

where

$$\begin{aligned} f(\mathbf{w},\gamma,\mathbf{y}) = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{k}\sum_{j\in\mathcal{J}_i^-} \max\{0, 1 + (\mathbf{w}^T\mathbf{x}_j + \gamma)\} \\ +C\sum_{i=1}^{m}\sum_{j\in\mathcal{J}_i^+} \max\{0, 1 - y_j(\mathbf{w}^T\mathbf{x}_j + \gamma)\}, \end{aligned}$$

with $\|\cdot\|$ being the Euclidean norm in $\mathbb{R}^n$ and $C > 0$ the trade-off parameter. Note that constraints

$$\sum_{j\in\mathcal{J}_i^+} \frac{y_j + 1}{2} \geq 1, \quad i = 1, \dots m$$

impose that, for each positive bag, at least one of its samples must be labelled as a positive one.

Function f is the sum of three terms:

i) $\frac{1}{2}\|w\|^2$. As previously mentioned, minimization of $\|w\|$ leads to maximization of the margin.

ii) $\sum_{i=1}^{k}\sum_{j\in\mathcal{J}_i^-} \max\{0, 1+(\mathbf{w}^T\mathbf{x}_j+\gamma)\}$. This term is the total classification error relatively to the negative bags;

iii) $\sum_{i=1}^{m}\sum_{j\in\mathcal{J}_i^+} \max\{0, 1 - y_j(\mathbf{w}^T\mathbf{x}_j + \gamma)\}$. This term represents the total classification error of the instances belonging to positive bags. Notice that such an error is zero if and only if for each positive bag $\mathcal{J}_i^+$, $i = 1, \dots, m$, there exists at least one instance $j \in \mathcal{J}_i^+$ such that $\mathbf{w}^T\mathbf{x}_j + \gamma \geq 1$. Note that, by letting the corresponding label $y_j = 1$, feasibility with respect to constraint $\sum_{j\in\mathcal{J}_i^+} \frac{y_j + 1}{2} \geq 1$ is achieved and, in addition, the classification error associated to such a bag is driven to zero, provided no instance of such bag falls into the "no man land", that is in the area where $|\mathbf{w}^T\mathbf{x} + \gamma| < 1$.

Summing up the classification error is equal to zero if and only if all negative bags are contained in the set S^-, at least one instance of each positive bag belongs to the set S^+ and no instance $\mathbf{x}_j$ of any positive bag satisfies the condition $|\mathbf{w}^T\mathbf{x}_j + \gamma| < 1$.

The Lagrangian heuristic introduced in [4] is based on relaxation of the linear constraints $\sum_{j\in\mathcal{J}_i^+} \frac{y_j + 1}{2} \geq 1$. The following problem is then obtained:

$$\begin{cases} z_{LR}(\boldsymbol{\lambda}) = \min\limits_{\mathbf{w},\gamma,\mathbf{y}} f(\mathbf{w},\gamma,\mathbf{y}) + \sum\limits_{i=1}^{m} \lambda_i \left(1 - \sum\limits_{j\in\mathcal{J}_i^+} \frac{y_j+1}{2}\right) \\ \qquad\qquad y_j \in \{-1,1\},\ \ j \in \mathcal{J}_i^+,\ \ i=1,\dots,m, \end{cases} \tag{82}$$

where $\boldsymbol{\lambda} \geq 0$ is a vector of multipliers in $\mathbb{R}^m$.

The Lagrangian dual problem is, as usual,

$$z_{LD} = \max_{\boldsymbol{\lambda}\geq \mathbf{0}} z_{LR}(\boldsymbol{\lambda}) \tag{83}$$

It is of course $z_{LR}(\boldsymbol{\lambda}) \leq z^*$, for any choice of the multiplier $\boldsymbol{\lambda} \geq 0$, and $z_{LD} \leq z^*$. Tackling problem (83) within a Lagrangian heuristic scheme requires at each iteration calculation of function $z_{LR}(\boldsymbol{\lambda})$ by solving (82), a mixed integer nonlinear problem which is particularly suitable for application of a Block Coordinate Descent method (see [55]). In fact the algorithm adopted in [4] works by alternately fixing, at each iteration, the values of $\mathbf{y}$ and of the couple $(\mathbf{w},\gamma)$, according to the following scheme.

Algorithm 3: Calculation of $z_{LR}(\boldsymbol{\lambda})$

```
Step 0   Choose a feasible point $\mathbf{y}^{(0)}$. Set $l = 0$.
Step 1   For the current $\mathbf{y}^{(l)}$ solve the convex problem

           $\min_{\mathbf{w},\gamma} f(\mathbf{w},\gamma,\mathbf{y}^{(l)}) + \sum_{i=1}^{m} \lambda_i \left(1 - \sum_{j\in J_i^+} \frac{y_j^{(l)}+1}{2}\right)$

         and obtain the couple $(\mathbf{w}^{(l+1)},\gamma^{(l+1)})$.
Step 2   For the current couple $(\mathbf{w}^{(l+1)},\gamma^{(l+1)})$ solve the problem

           $\min_{\mathbf{y}_j\in\{-1,1\}} f(\mathbf{w}^{(l+1)},\gamma^{(l+1)},\mathbf{y}) + \sum_{i=1}^{m} \lambda_i \left(1 - \sum_{j\in J_i^+} \frac{y_j+1}{2}\right)$

         and obtain $\mathbf{y}^{(l+1)}$. Set $l = l + 1$ and go to Step 1.
```

We remark that the minimization problem at Step 1 is a standard SVM-like problem, while solution of problem at Step 2 can be easily obtained by inspection of the values $h_j^{(l+1)} = \mathbf{w}^{(l+1)T}\mathbf{x}_j + \gamma^{(l+1)}$.

In [4] update of the multiplier vector $\boldsymbol{\lambda}$ is performed by standard subgradient method adopting the Polyak stepsize (28). A projection mechanism to take into account the nonnegativity of $\boldsymbol{\lambda}$ is also embedded.

To complete the bird's-eye view of the Lagrangian heuristic, we observe that any solution $(\mathbf{w}(\boldsymbol{\lambda}), \gamma(\boldsymbol{\lambda}), \mathbf{y}(\boldsymbol{\lambda}))$ of the Lagrangian relaxation which violates the relaxed constraints can be easily "repaired" to get feasibility by means of greedy modification of one or more label variables y_j.

It is worth noting that the Lagrangian dual (83) enjoys the relevant property that the duality gap is equal to zero, that is $z_{LD} = z^*$. Moreover, by solving the Lagrangian dual, one gets, in fact, also an optimal solution for problem (81). These results are provided by the following theorem [4].

Theorem 2. *Let $\boldsymbol{\lambda}^*$ be any optimal solution to the Lagrangian dual (83) and let (w^*, γ^*, y^*) be any optimal solution to the Lagrangian relaxation (82) for $\boldsymbol{\lambda} = \boldsymbol{\lambda}^*$. Then (w^*, γ^*, y^*) is optimal for the original problem (81) and $z_{LD} = z^*$.*

The implementation of the Lagrangian heuristic has provided satisfactory results on a number of benchmark test problems. In particular the zero duality gap has been fully confirmed.

6 Conclusions

We have provided some basic notions on Lagrangian relaxations, focusing on its application in designing heuristic algorithms of the so called Lagrangian heuristic type. Although the number of applications available in the literature is practically uncountable, we are convinced that the potential for future application is still enormous.

References

1. Amores, J.: Multiple instance classification: review, taxonomy and comparative study. Artificial Intelligence, **201**, 81–105 (2013).
2. Andrews, S., Tsochantaridis, I., Hofmann, T.: Support vector machines for multiple-instance learning. In: Becker, S., Thrun, S., Obermayer, K. (eds.) Advances in Neural Information Processing Systems, pp. 561-568, MIT Press, Cambridge (2003).
3. Astorino, A., Gaudioso, M., Miglionico, G.: Lagrangian relaxation for the directional sensor coverage problem with continuous orientation. Omega, **75**, 1339–1351 (2018).
4. Astorino, A., Fuduli, A., Gaudioso, M.: A Lagrangian relaxation approach for binary Multiple Instance Classification. IEEE Transactions on Neural Networks and Learning Systems, DOI: 10.1109/TNNLS.2018.2885852, (2019).
5. Bacaud, L., Lemaréchal, C., Renaud, A., Sagastizábal, C.: Bundle methods in stochastic optimal power management: a disaggregated approach using preconditioners. Computational Optimization and Applications, **20**, 227–244 (2001).
6. Bertsekas, D.P., Nedic, A.: Incremental subgradient methods for nondifferentiable optimization. SIAM Journal on Optimization, **12**, 109-138 (2001).
7. Bertsekas, D.P.: Convex Optimization Algorithms. Athena Scientific, Belmont (2015).

8. Bertolazzi, P., Felici, G., Festa, P., Fiscon, G., Weitschek, E.: Integer programming models for feature selection: new extensions and a randomized solution algorithm. Europen Journal of Operational Research **250**, 389-399 (2016).
9. Borghetti, A., Frangioni, A., Lacalandra, F., Nucci, C.A.: Lagrangian heuristics based on disaggregated bundle methods for hydrothermal unit commitment. IEEE Transactions on Power Systems, **18**, 313–323 (2003).
10. Bradley, P.S., Mangasarian, O.L., Street, W.N.: Feature selection via mathematical programming. INFORMS Journal on Computing **10**, 209-217 (1998).
11. Carrabs, F., Cerulli, R., Gaudioso, M., Gentili, M.: Lower and upper bounds for the spanning tree with minimum branch vertices. Computational optimization and applications, **56**, 405–438 (2013).
12. Celani, M., Cerulli, R., Gaudioso, M., Sergeyev, Ya.D.: A multiplier adjustment technique for the capacitated concentrator location problem. Optimization Methods and Software, **10**, 87–102 (1998).
13. Cerulli, R., De Donato, R., Raiconi, A.: Exact and heuristic methods to maximize network lifetime in wireless sensor networks with adjustable sensing ranges. European Journal of Operational Research **220**, 58-66 (2012).
14. Cheney, E. W., Goldstein, A. A.: Newton's method for convex programming and Tchebycheff approximation. Numerische Mathematik, **1**, 253-268 (1959).
15. Chiarello, A., Gaudioso, M., Sammarra, M.: Truck synchronization at single door cross-docking terminals. OR Spectrum, **40**, 395–447 (2018).
16. Cristianini, N., Shawe–Taylor, J.: An Introduction to Support Vector Machines and Other Kernel–based Learning Methods. Cambridge University Press. Cambridge (2000).
17. de Oliveira, W., Sagastizábal, C., Lemaréchal, C.: Convex proximal bundle methods in depth: a unified analysis for inexact oracles. Mathematical Programming, **148**, 241–277 (2014).
18. Di Puglia Pugliese, L., Gaudioso, M., Guerriero, F., Miglionico, G.: A Lagrangean-based decomposition approach for the link constrained Steiner tree problem. Optimization Methods and Software, **33**, 650–670 (2018).
19. Everett, H. III: Generalized Lagrange multiplier method for solving problems of optimum allocation of resources. Operations Research **11**, 399–417 (1963).
20. Finardi, E., da Silva, E., Sagastizábal C.: Solving the unit commitment problem of hydropower plants via Lagrangian relaxation and sequential quadratic Programming. Computational and Applied Mathematics **24**, 317–342 (2005).
21. Fischer, F., Helmberg C., Janßen, J., Krostitz, B.: Towards solving very large scale train tTimetabling problems by Lagrangian relaxation. In 8th Workshop on Algorithmic Approaches for Transportation Modeling, Optimization, and Systems (ATMOS'08), Fischetti, M., Widmayer, P. eds. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, Dagstuhl (2008).
22. Fischer, M.L.: The Lagrangian relaxation method for solving integer programming problems. Management Science, **27**, 1–18 (1981).
23. Fisher, M.L., Jaikumar, R., Van Wassenhove, L.N.: A multiplier adjustment method for the generalized assignment problem. Management Science, **32**, 1095–1103 (1986).
24. Frangioni, A.: Generalized bundle methods. SIAM Journal on Optimization, **13**, 117-156 (2002).
25. Frangioni, A.: About Lagrangian Methods in Integer Optimization. Annals of Operations Research, **139**, 163–193 (2005).
26. Frangioni, A., Gorgone, E., Gendron, B.: On the computational efficiency of subgradient methods: a case study with Lagrangian bounds. Mathematical Programming Computation, **9**, 573-604 (2017).
27. Frangioni, A., Gorgone, E., Gendron, B.: Dynamic smoothness parameter for fast gradient methods. Optimization Letters, **12**, 43-53 (2018).

28. Gaudioso, M., Giallombardo, G., Miglionico, G.: An incremental method for solving convex finite min-max problems. Mathematics of Operations Research, **31**, 173–187 (2006).
29. Gaudioso, M., Giallombardo, G., Miglionico, G.: On solving the Lagrangian dual of integer programs via an incremental approach. Computational Optimization and Applications, **44**, 117–138 (2009).
30. Gaudioso, M., Gorgone, E., Labbé M., Rodríguez–Chía, A.: Lagrangian relaxation for SVM feature selection. Computers and Operations Research, **87**, 137–145 (2017).
31. Gaudioso, M., Giallombardo, G., Miglionico, G., Bagirov, A.: Minimizing nonsmooth DC functions via successive DC piecewise-affine approximations. Journal of Global Optimization, **71**, 37–55 (2018).
32. Geoffrion A.M.: Lagrangean relaxation for integer programming, Mathematical Programming Study, **2**, 82–114 (1974).
33. Goffin J.-L. G, Haurie, A., Vial, J.-P.: On the computation of weighted analytic centers and dual ellipsoids with the projective algorithm. Mathematical Programming, **60**, 81-92, (1993).
34. Goffin J.-L., Gondzio, J., Sarkissian, R., Vial, J.-P.: Solving nonlinear multicommodity flow problems by the analytic center cutting plane method. Mathematical Programming, **76**, 131–154 (1996).
35. Guignard, M., Rosenwein, B.M.: An application-oriented guide for designing Lagrangean dual ascent algorithms. European Journal of Operational Research, **43**, 197–205 (1989).
36. Guignard, M.: Lagrangean relaxation. Top, **11**, 151–228 (2003).
37. Hiriart-Urruty, J.-B., Lemaréchal, C.: Convex Analysis and Minimization Algorithms Vol. I-II. Springer-Verlag, Heidelberg (1993).
38. Kelley, J.E.: The cutting-plane method for solving convex programs. Journal of the SIAM **8**, 703-712 (1960).
39. Kiwiel, K.: Convergence of approximate and incremental subgradient methods for convex optimization. SIAM Journal on Optimization, **14**, 807-840 (2003)
40. Kiwiel, K.C., Lemaréchal, C.: An inexact bundle variant suited to column generation. Mathematical Programming, **118**, 177–206 (2009).
41. Lagrange, J.L.: Mécanique analytique. Courcier, Paris (1811).
42. Lemarchal, C., Pellegrino, F., Renaud, A., Sagastizábal C.: Bundle methods applied to the unit-commitment problem. In: System Modelling and Optimization, 395–402. Chapman and Hall, London (1996).
43. Lemaréchal, C.: The onnipresence of Lagrange. Annals of Operations Research, **153**, 9–27 (2007).
44. Lemaréchal, C., Ouorou, A., Petrou, G.: A bundle-type algorithm for routing in telecommunication data networks. Computational Optimization and Applications, **44**, 385–409 (2009).
45. Maldonado, S., Pérez, J., Weber, R., Labbé, M.: Feature selection for support vector machines via mixed integer linear programming. Information Sciences, **279**, 163-175 (2014).
46. Monaco, M.F., Sammarra, M.: The berth allocation problem: a strong formulation solved by a Lagrangean approach. Transportation Science, **41**, 265–280 (2007).
47. Nesterov, Yu.: Smooth minimization of non-smooth functions. Mathematical Programming, **103**, 127–152 (2005).
48. Piccialli, V., Sciandrone, M.: Nonlinear optimization and Support Vector Machines. 4OR, **16**, 111–149 (2018).
49. Polyak, B.T.: Introduction to Optimization. Optimization Software Inc., New York (1987).
50. Rinaldi, F., Sciandrone, M.: Feature selection combining linear support vector machines and concave optimization. Optimization Methods and Software, **10**, 117-128 (2010).

51. Rockafellar, R.T.: Convex Analysis. Princeton University Press, Princeton (1970).
52. Rockafellar, R.T.: Lagrange multipliers and optimality. SIAM Review **35**, 183–238 (1993).
53. Rossi, A., Singh, A., Sevaux, M.: Lifetime maximization in wireless directional sensor network. European Journal of Operational Research, **231**, 229-241 (2013).
54. Shor, N.Z., Minimization Methods for Non-differentiable Functions. Springer-Verlag, New York (1985).
55. Tseng, P.: Convergence of a block coordinate descent method for nondifferentiable minimization. Journal of Optimization Theory and Applications, **109**, 475-494 (2001).
56. Wolsey, L.A.: Integer Programming. Wiley-Interscience, New York (1998).
57. Yick, J., Mukherjee, B., Ghosal, D.: Wireless sensor network survey. Computer Networks, **52**, 2292-2330 (2008).
58. Zhang, H., Fritts, J.E., Goldman, S.A.: Image segmentation evaluation: A survey of unsupervised methods. Computer Vision and Image Understanding, **110**, 260-280 (2008).