Relaxations and Bounds: Applications to Knapsack Problems

M2 ORO: Advanced Integer Programming
Lecture 7
Sophie.Demassey@mines-nantes.fr

October 24, 2011

1 Introduction

1.1 Why relaxation ?

Relaxation is a key component for solving MILP. In a branch-and-bound method, it allows to reduce the size of the search tree by recognizing and pruning:

- infeasible nodes (if the relaxation is infeasible)
- solution nodes (if the solution of the relaxation is feasible for the problem)
- sub-optimal nodes (if the value of the relaxation is not better than the current incumbent)

A good relaxation:

- has a **tight bound**: the optimal value of the relaxation should be close to the optimal value of the problem in order to prove more sub-optimality and to prune more nodes
- is **solved quickly**: the relaxation must be solved many times, i.e. at each node of the branch-and-bound. The running time of a B&B method is the sum of the running times (for solving the relaxation) at each node.

These two criteria are usually orthogonal: tighter relaxations are more constrained and, consequently, more difficult to solve. We have to choose the better compromise between the quality of the relaxation and its complexity.

1.2 Relaxation of combinatorial optimization problems

Definition 1 *Let* $(P) : z = \max\{f(x) \mid x \in S\}$ *a combinatorial optimization problem. A* ***combinatorial relaxation*** *of* (P) *is any combinatorial optimization problem* $(\bar{P}) : \bar{z} = \max\{g(y) \mid y \in \bar{S}\}$ *such that:*

$$\forall x \in S, \quad \exists\, y \in \bar{S}, \quad | \quad g(y) \geqslant f(x).$$

The optimum of relaxation $(\bar{P})$ *is an* ***upper bound*** *of the optimum of* (P) *since* $z \leqslant \bar{z}$.

Example 1 *The following cases define relaxations:*

1. $S \subseteq \bar{S}$ *and* $f \equiv g$
2. $S = \{x \in \mathbb{Z}^n_+ \mid Ax \leqslant b, Ex \leqslant d\}$ *and* $\bar{S} = \{x \in \mathbb{Z}^n_+ \mid Ax \leqslant b\}$ *and* $f \equiv g$*: the constraints* $Ex \leqslant d$ *are said to be* relaxed *(i.e. removed)*
3. $S = \{x \in \mathbb{Z}^n_+ \mid Ax \leqslant b, Ex \leqslant d\}$ *and* $\bar{S} = \{x \in \mathbb{R}^n_+ \mid Ax \leqslant b, Ex \leqslant d\}$ *and* $f \equiv g$*: the integrality constraints are relaxed; it is the* ***LP relaxation*** *or* continuous relaxation $(\bar{P})$ *of the ILP* (P).

Definition 2 *Let* $(P) : z = \max\{f(x) \mid x \in S\}$ *a combinatorial optimization problem. A* ***dual relaxation*** *of* (P) *is any combinatorial optimization problem* $(D) : u = \min\{g(y) \mid y \in T\}$ *such that:*

$$\forall x \in S, \quad \forall\, y \in T, \quad g(y) \geqslant f(x).$$

The optimum of dual relaxation (D) *is an* ***upper bound*** *of the optimum of* (P) *since* $z \leqslant u$. P *and* D *form a* ***weak-dual pair****. Furthermore, if* $z = u$*, then* P *and* D *form a* ***strong-dual pair****.*

Example 2 *The following cases define dual relaxations:*

1. $(P) : z = \max\{cx \mid Ax \leqslant b, x \in \mathbb{R}^n_+\}$ *and* $(D) : u = \min\{by \mid yA \geqslant c, y \in \mathbb{R}^m_+\}$ *form a strong-dual pair,* (D) *is the* ***LP dual*** *of* (P)
2. *as a consequence,* $(P) : z = \max\{cx \mid Ax \leqslant b, x \in \mathbb{Z}^n_+\}$ *and* $(D) : u = \min\{by \mid yA \geqslant c, x \in \mathbb{Z}^m_+\}$ *form (at least) a weak-dual pair*

1.3 Relaxation of integer linear programs

Any problem formulated as an ILP has a natural relaxation: the LP relaxation. Generally, such an LP can be solved efficiently by the simplex method. For some problems, specific algorithms exist which are still more efficient. As an example, we will study in Section 2 an algorithm solving the continuous 0-1 knapsack problem in linear time at each node of a search tree (and in quadratic time at the root of the tree).

The quality of the bound obtained by any LP relaxation depends on the strength of the formulation. Strong formulations and, *a fortiori*, ideal formulations (i.e. the extreme points of the polyhedron are integer) are usually hard to find: as hard as to solve the ILP. Furthermore, they may be of exponential size. Hence, solving the LP relaxation can often be too time consuming within a branch-and-bound.

For hard ILP with weak formulations, relaxing complicating or coupling constraints other than integrity constraints is often more efficient within a branch-and-bound: the relaxation may give better bounds or it may be easier to solve. Furthermore, we can easily build relaxations, not just by removing constraints, but by penalizing their violation. These kinds of relaxations will be illustrated in Section 3 on the multi-knapsack problem.

2 LP Relaxation of the 0-1 Knapsack Problem

An instance of the 0-1 Knapsack Problem is defined by a container of capacity c and a set I of items, $|I| = n$, with size (or weight) w_i and value p_i for item $i \in I$. All values are non-negative integers. We want to select a subset of items to place into the container such that the sum of the sizes does not exceed the capacity of the container, and such that the sum of the values is maximized:

$$(P) : z = \max \sum_{i \in I} p_i x_i, \quad \text{s.t.} \quad \sum_{i \in I} w_i x_i \leqslant c, \quad x_i \in \{0,1\}\ (\forall i \in I).$$

W.l.o.g, we assume that $w_i \leqslant c$ for all $i \in I$ (any item can be selected) and that $\sum_{i \in I} w_i > c$ (all the items cannot be placed).

2.1 The 0-1 knapsack problem with unit sizes $w_i = 1$

If all items have the same size $w_i = 1$ then any subset of items of cardinality no greater than c leads to a feasible solution. We can easily compute an optimal solution of (P), by reordering the items in decreasing order of their value ($p_1 \geqslant p_2 \geqslant \cdots p_n$), then selecting the c first items to pack into the container:

$$x_i = \begin{cases} 1 & \text{if } i = 1, \dots, c \\ 0 & \text{otherwise.} \end{cases}$$

In this algorithm, sorting can be executed once for all in $O(n \ln n)$ time, then selection runs in $O(n)$. Note that this solution is optimum for (P) and for its LP relaxation $(\bar{P})$, i.e.: $z = \bar{z}$. Actually the formulation of (P) is ideal since the matrix of constraints is totally unimodular and the right hand side c is integer.

2.2 The 0-1 knapsack problem with identical sizes $w_i = w$

Now the feasible solutions are given by the set of items of cardinality no greater than c/w. Sorting the items as previously then selecting the first $\lfloor c/w \rfloor$ items[1] leads to a feasible solution of (P):

$$x_i = \begin{cases} 1 & \text{if } i = 1, \dots, s-1 \\ 0 & \text{otherwise.} \end{cases}$$

where the threshold value s is defined by $s = \lfloor c/w \rfloor + 1$. As the right hand side c/w may not be integer, the property of totally unimodularity does not apply anymore. Actually, the following solution is optimum for the LP relaxation $(\bar{P})$ (it will be proved in the general case by Theorem 2):

$$\bar{x}_i = \begin{cases} 1 & \text{if } i = 1, \dots, s-1 \\ \frac{c}{w} - \lfloor \frac{c}{w} \rfloor & \text{if } i = s \\ 0 & \text{otherwise.} \end{cases}$$

This solution gives an upper bound for (P): $\bar{z} = \sum_{i=1}^{s-1} p_i + p_s(\frac{c}{w} - \lfloor \frac{c}{w} \rfloor)$. As the optimal value of (P) is integer, then a valid upper bound is:

$$\lfloor \bar{z} \rfloor = \sum_{i=1}^{s-1} p_i + \lfloor \frac{c}{w} p_s \rfloor - \lfloor \frac{c}{w} \rfloor p_s.$$

If cp_s/w is not integer then the optimal value of $(\bar{P})$ is strictly greater than the optimum of (P):

$$z \leqslant \lfloor \bar{z} \rfloor < \bar{z}.$$

2.3 The 0-1 knapsack problem with arbitrary sizes

The principle to compute an optimal solution for the LP relaxation $(\bar{P})$ is as before: sorting the items in decreasing order of p_i/w_i then by selecting the first items consecutively until the first item, the *critical item* s, is found which does not fit into the container. Then the container is filled up with a fraction of the critical item.

First, we assume that the ratio $\frac{p_i}{w_i}$ for all items i are all different, according to:

Lemma 1 *Let two items* i *and* j *such that* $\frac{p_i}{w_i} = \frac{p_j}{w_j}$ *then the LP relaxation* $(\bar{P})$ *is equivalent to the LP relaxation* $(\bar{P}')$ *of a second knapsack problem obtained by merging the two items into one new item* k *of size* $w_i + w_j$ *and of value* $p_i + p_j$.

Proof. Let x a solution of $(\bar{P})$ and define

$$x'_l = \begin{cases} x_l & \forall l \neq i, j, k \\ \frac{w_i x_i + w_j x_j}{w_i + w_j} & \text{if } l = k. \end{cases}$$

x' is feasible for $(\bar{P}')$ since $x'_k = \frac{w_i x_i + w_j x_j}{w_i + w_j} \in [0, 1]$ and $w_k x'_k = w_i x_i + w_j x_j$. Furthermore, its value is the same as x since:

$$p'_k = \frac{(p_i + p_j)(w_i x_i + w_j x_j)}{w_i + w_j} = \frac{(p_i w_i + p_j w_i)x_i + (p_i w_j + p_j w_j)x_j}{w_i + w_j} = p_i x_i + p_j x_j.$$

Conversely, to each solution x' for $(\bar{P}')$ corresponds a feasible solution x of $(\bar{P})$ with same cost, defined by:

$$x_l = \begin{cases} x'_l & \forall l \neq i, j, k \\ \max(1, \frac{(w_i + w_j)x'_k}{w_j}) & \text{if } l = j \\ \min(0, \frac{(w_i + w_j)x'_k - w_j}{w_i}) & \text{if } l = i. \end{cases}$$

[1] $\lfloor a \rfloor$ denotes the greatest integer lower than $a \in \mathbb{R}$.

Hence the two linear programs are equivalent. ■

Under the assumption that the items are sorted such that $\frac{p_1}{w_1} > \frac{p_2}{w_2} > \cdots > \frac{p_n}{w_n}$ then we can easily compute an optimal solution for $(\bar{P})$:

Theorem 2 *(Dantzig, 1957).*
Let $s = \min\{j \mid \sum_{i=1}^{j} w_i > c\}$, *and* $\bar{c} = c - \sum_{i=1}^{s-1} w_i$ *then the following solution is optimum for* $(\bar{P})$*:*

$$\bar{x}_i = \begin{cases} 1 & \textit{if } i = 1, \dots, s-1 \\ \frac{\bar{c}}{w_s} & \textit{if } i = s \\ 0 & \textit{otherwise.} \end{cases}$$

Proof. $\bar{x}$ is clearly feasible for $(\bar{P})$. Assume there exists x^* an optimal solution such that $x_k^* < \bar{x}_k = 1$ for some $k < s$, then there exists some $q \geqslant s$ such that $x_q^* > \bar{x}_q$ (otherwise, $\sum_{i=1}^{n} p_i x_i^* < \sum_{i=1}^{n} p_i \bar{x}_i$). Let $\epsilon = \min(\bar{x}_k - x_k^*, \frac{w_q}{w_k}(x_q^* - \bar{x}_q))$, and

$$x_i = \begin{cases} x_i^* + \epsilon & \text{if } i = k \\ x_i^* - \frac{w_k}{w_q}\epsilon & \text{if } i = q \\ x_i^* & \text{otherwise.} \end{cases}$$

x_i is a feasible solution of $(\bar{P})$ since $x_k^* \leqslant x_k \leqslant \bar{x}_k$, $\bar{x}_q \leqslant x_q \leqslant x_q^*$ and $\sum_{i=1}^{n} w_i x_i = \sum_{i=1}^{n} w_i x_i^* \leqslant c$. Furthermore, its cost is strictly greater than the cost of x_i^*: $\sum_{i=1}^{n} p_i(x_i - x_i^*) = (p_k - p_q \frac{w_k}{w_q})\epsilon > 0$. This is absurd, so $x_k^* = 1$ for all $k < s$. We can prove in the same say that $x_k^* > 0$ is impossible for any optimal solution and for any $k > s$. Hence, by maximality of $\bar{x}_s$, $\bar{x}$ is the unique optimal solution of $(\bar{P})$. ■

Again this solution gives an upper bound for (P) which, after rounding, is equal to:

$$\lfloor \bar{z} \rfloor = \sum_{i=1}^{s-1} p_i + \lfloor \frac{\bar{c}}{w} p_s \rfloor.$$

This LP bound is named the **Dantzig bound** for the 0-1 knapsack problem.

If the items are already sorted as assumed, the computation of the Danzig bound clearly requires $O(n)$ time. (If this is not the case, the computation can still be performed in $O(n)$ time by using a procedure (Balas and Zemel, 1980) to determine the critical item.)

2.4 Notes on the 0-1 Knapsack Problem

The 0-1 Knapsack Problem is NP-complete, but not in the strong sense since there exists a pseudo-polynomial time algorithm, based on dynamic programming, for solving this problem. The algorithm is based on the computation of the values $f_m(c') = \max\{\sum_{i=1}^{m} p_i x_i \mid \sum_{i=1}^{m} w_i x_i \leqslant c', x \in \{0,1\}^m\}$ at each stage m increasing from 1 to n and for each capacity c' increasing from 0 to c, according to the following recursion (Bellman, 1954):

$$f_1(c') = \begin{cases} 0 & \forall c' = 0, \dots, w_1 - 1 \\ p_1 & \forall c' = w_1, \dots, c \end{cases}$$

and

$$f_m(c') = \begin{cases} f_{m-1}(c') & \forall m = 2, \dots, n, \forall c' = 0, \dots, w_m - 1 \\ \max(f_{m-1}(c'), f_{m-1}(c' - w_m) + p_m) & \forall m = 2, \dots, n, \forall c' = w_m, \dots, c \end{cases}$$

A procedure was derived by (Toth, 1980) directly from this recursion, with a time and space complexity of $O(nc)$. (This complexity can be reduced by eliminating the dominated states, i.e. a state (m', c') such there exists a dominating state (m'', c'') with $f_{m'-1}(c') \leqslant f_{m''-1}(c'')$ and $c'' < c'$.)

3 Relaxations for the 0-1 Multi-Knapsack Problem

An instance of the 0-1 Multi-Knapsack Problem (MKP) is defined by a set J of containers, $|J| = m$, with capacity c_j for container $j \in J$, and a set I of items, $|I| = n$, with size (or weight) w_i and value p_i for item $i \in I$. All values are non-negative integers. We want to place a subset of items into the containers without exceeding the capacities of the containers, and such that the sum of the values is maximized:

$$(P) : z = \max \sum_{i \in I} \sum_{j \in J} p_i x_{ij}, \quad \text{s.t.} \quad \sum_{i \in I} w_i x_{ij} \leqslant c_j (\forall j \in J), \quad \sum_{j \in J} x_{ij} \leqslant 1 (\forall i \in I), \quad x \in \{0,1\}^{I \times J}.$$

W.l.o.g, we assume that $w_i \leqslant c_j$ for all $i \in I$ and that $\sum_{i \in I} w_i > c_j$ on each container $j \in J$.

3.1 LP Relaxation

The LP relaxation is defined by:

$$(\bar{P}) : \bar{z} = \max \sum_{i \in I} \sum_{j \in J} p_i x_{ij}, \quad \text{s.t.} \quad \sum_{i \in I} w_i x_{ij} \leqslant c_j (\forall j \in J), \quad \sum_{j \in J} x_{ij} \leqslant 1 (\forall i \in I), \quad x_{ij} \geqslant 0.$$

(Note that the upper bounds on the variables $x_{ij} \leqslant 1$ are redundant with constraints $\sum_{j \in J} x_{ij} \leqslant 1$). The dual $(\bar{D})$ of $(\bar{P})$ is:

$$(\bar{D}) : \bar{u} = \min \sum_{i \in I} \mu_i + \sum_{j \in J} c_j \pi_j \quad \text{s.t.} \quad w_i \pi_j + \mu_i \geqslant p_i (\forall i \in I, j \in J), \quad \pi_j \geqslant 0 (\forall j \in J), \mu_i \geqslant 0 (\forall i \in I).$$

The LP bound $\bar{z}$ can be computed in linear time according to the following proposition:

Proposition 3 *Assume that the items are sorted in decreasing order of ratio p_i/w_i and let s be the critical item $s = \min\{k \in I \mid \sum_{i=1}^{k} w_i > \sum_{j \in J} c_j\}$, then:*

$$\bar{z} = \sum_{i=1}^{s-1} (p_i - \frac{w_i}{w_s} p_s) + \sum_{j \in J} \frac{p_s}{w_s} c_j.$$

Proof. Consider the items in their order and place them consecutively on the first container until a first item s^1 is found wich does not fit: $s^1 = \min\{k \in I \mid \sum_{i=1}^{k} w_i > c_1\}$. Insert the maximum possible fraction $(c_1 - \sum_{i=1}^{s^1-1} w_i)$ of item s^1 in container 1. Continue by inserting the residual fraction $(\sum_{i=1}^{s^1} w_i - c_1)$ of s^1 in container 2, and so on until the last container m is full. Let s^m be the last item considered: i.e. the item placed partially into the last container m. By construction, we have $\sum_{i=1}^{s^m} w_i > \sum_{j \in J} c_j$ and $\sum_{i=1}^{s^m-1} w_i \leqslant \sum_{j \in J} c_j$, then $s^m = s$. Hence the assignment $\bar{x} \in [0,1]^{I \times J}$ associated to this solution is feasible for $(\bar{P})$ and it satisfies:

$$\sum_{j \in J} \bar{x}_{ij} = \begin{cases} 1 & \forall i = 1, \ldots, s-1 \\ (\sum_{j \in J} c_j - \sum_{i=1}^{s-1} w_i)/w_s & \text{if } i = s \\ 0 & \forall i = s+1, \ldots, n \end{cases}$$

Its cots is then equal to $\sum_{i \in I} p_i \sum_{j \in J} \bar{x}_{ij} = \sum_{i=1}^{s-1} p_i + p_s/w_s(\sum_{j \in J} c_j - \sum_{i=1}^{s-1} w_i)$. Conversely, it is easy to see that the dual solution (π, μ) defined by:

$$\begin{aligned} \pi_j &= p_s/w_s & & \forall j \in J \\ \mu_i &= p_i - \frac{w_i}{w_s} p_s & & \forall i = 1, \ldots, s-1 \\ \mu_i &= 0 & & \forall i = s, \ldots, n \end{aligned}$$

is feasible for $(\bar{D})$ and has the same cost as $\bar{x}$. Hence, by strong duality, the solutions $\bar{x}$ and (π, μ) are optimum for $(\bar{P})$ and $(\bar{D})$ respectively. ■

3.2 Simple Combinatorial Relaxation

By relaxing the knapsack constraints, we obtain a BIP that is easy to solve:

$$(P_0): z_0 = \max \sum_{i \in I} \sum_{j \in J} p_i x_{ij}, \quad \text{s.t.} \quad \sum_{j \in J} x_{ij} \leqslant 1 (\forall i \in I), \quad x_{ij} \in \{0,1\}.$$

Any assignment of items to containers leads to a feasible solution of (P_0), and if all the items are assigned then the solution is optimum. Hence, the optimum of (P_0) gives a first simple upper bound for the MKP:

$$z_0 = \sum_{i \in I} p_i.$$

It is easy to see that this bound is never better than the LP bound:

$$z \leqslant \bar{z} \leqslant z_0.$$

The knapsack constraints are said to be **complicating** constraints: relaxing them leads to an easy problem. Unfortunately, the bound obtained by relaxing these complicating constraints is clearly far from the optimum in most of the cases. Instead of totally removing these constraints, we will see two different ways to relax them by varying and controlling the degree of violation allowed.

3.3 Surrogate Relaxation

The principle of the surrogate relaxation is to replace a conjunction of constraints by one constraint obtained as a linear combination (weighted sum) of the constraints. In the context of the MKP, we can apply this principle to the conjunction of knapsack constraints. It means that we simply ensure that the global (weighted) sum of the sizes of the selected items does not exceed the (weighted) sum of the capacities of the containers. The weight (or multiplier) associated to a knapsack constraint, i.e. to a container, indicates some kind of degree of violation allowed for that constraint: if the weight is 0, then the constraint is not considered at all.

Any linear combination of constraints, i.e. any vector of non-negative multipliers, leads to a relaxation of the original problem. The surrogate relaxation is the better one (i.e. the one with the lowest optimum) among all of these relaxations. We illustrate on the example of the MKP how to determine the best relaxation without evaluating all the possible relaxations. Furthermore, we show that this relaxation can be reformulated as a 0-1 knapsack problem which can be approximated in linear time (according to the Dantzig bound) or exactly solved by dynamic programming in reasonable time when the values of the capacities are not to high.

Let $\pi \in \mathbb{R}_+^J$ a vector of multipliers, we define the following BIP:

$$(P_1^\pi): z_1^\pi = \max \sum_{i \in I} \sum_{j \in J} p_i x_{ij}, \quad \text{s.t.} \quad \sum_{j \in J} \pi_j \sum_{i \in I} w_i x_{ij} \leqslant \sum_{j \in J} \pi_j c_j, \quad \sum_{j \in J} x_{ij} \leqslant 1 (\forall i \in I), \quad x \in \{0,1\}^{I \times J}.$$

It is easy to see that any feasible solution of (P) is a feasible solution of (P_1^π), so (P_1^π) is a relaxation of (P), and we have:

$$z \leqslant z_1^\pi \leqslant z_0, \quad \forall \pi \in \mathbb{R}_+^J.$$

The surrogate relaxation is any relaxation (P_1^π), $\forall \pi \in \mathbb{R}_+^J$, that minimizes z_1^π. We note $z_1 = \min\{z_1^\pi \mid \pi \in \mathbb{R}_+^J\}$ the corresponding upper bound. Our goal is to determine a vector $\pi \in \mathbb{R}_+^J$ that minimizes z_1^π without evaluating this value (i.e. without solving (P_1^π)) for all possible vectors.

First, we can eliminate the vectors that contains a coefficient 0:

Proposition 4 $z_1 = \min\{z_1^\pi \mid \pi \in \mathbb{R}_{*+}^J\}$

Proof. We need to prove that $z_1^\pi \leqslant z_1^{\pi'}$ for all vectors π and π' such that π' contains a coefficient 0. Let $\pi' \in \mathbb{R}_+^J$ such that there exists $j' \in J$ with $\pi_{j'} = 0$. Assigning all items to container J leads to a feasible solution of $(P_1^{\pi'})$:

$$x_{ij} = \begin{cases} 1 & \forall i \in I, j = j' \\ 0 & \text{otherwise.} \end{cases}$$

The cost of this solution is $\sum_{i\in I} p_i = z_0$ so the solution is optimum and $z_1^{\pi'}$ is the trivial bound z_0. ■

From now, consider only vectors π whose minimum coefficient $\pi_{j'}$ is strictly positive. The following proposition shows that (P_1^π) is then equivalent to a 0-1 knapsack problem where all the containers are aggregated as one container of capacity $c^\pi = \lfloor \sum_{j\in J} \frac{\pi_j c_j}{\pi_{j'}} \rfloor$.

Proposition 5 (P_1^π) *is equivalent to:*

$$(K^\pi) : \max \sum_{i\in I} p_i y_i, \quad s.t. \quad \sum_{i\in I} w_i y_i \leqslant c^\pi, \quad y_i \in \{0,1\} (\forall i \in I).$$

Proof. To each feasible solution x of (P_1^π), we can associate a feasible solution y of (K^π), selecting the same subset $I' \subseteq I$ of items ($y_i = \sum_{j\in J} x_{ij}$, i.e. $y_i = 1 \iff i \in I'$), hence having the same cost $\sum_{i\in I'} p_i$. Solution y is feasible since $\sum_{i\in I} w_i y_i$ is integer and:

$$\sum_{i\in I} w_i y_i = \sum_{i\in I} w_i \sum_{j\in J} x_{ij} \leqslant \sum_{i\in I} w_i \sum_{j\in J} \frac{\pi_j}{\pi_{j'}} x_{ij} \leqslant \sum_{j\in J} \frac{\pi_j}{\pi_{j'}} c_j.$$

Conversely, to each feasible solution y of (K^π), we associate a feasible solution x of (P_1^π) of same cost, by selecting the same subset $I' \subseteq I$ of items, and by assigning them to the unique container j':

$$x_{ij} = \begin{cases} y_i & \forall i \in I, j = j' \\ 0 & \text{otherwise.} \end{cases}$$

Solution x is feasible since

$$\sum_{j\in J} \pi_j \sum_{i\in I} w_i x_{ij} = \pi_{j'} \sum_{i\in I} w_i y_i \leqslant \pi_{j'} \lfloor \sum_{j\in J} \frac{\pi_j c_j}{\pi_{j'}} \rfloor \leqslant \sum_{j\in J} \pi_j c_j.$$

■

Hence, the surrogate bound z_1 is the lowest optimal value of the knapsack problems (K^π), with $\pi \in \mathbb{R}^J_{*+}$. Actually, all these knapsack problems are defined on the same set of items (size and value) but with different container capacities. The best bound is then obtained by considering the knapsack that is the more constrained (or the less relaxed), i.e. the knapsack with the lowest capacity c^π. The lowest capacity $\sum_{j\in J} c_j$ is achieved when $\pi_j = \pi_{j'}$ for all $j \in J$:

Proposition 6 z_1 *is the optimal value of the following knapsack problem:*

$$(P_1) : z_1 = \max\{\sum_{i\in I} p_i y_i \mid \sum_{i\in I} w_i y_i \leqslant \sum_{j\in J} c_j,\ y \in \{0,1\}^I\}.$$

Proof. Consider the knapsack problem $(P_1) = (K^{\mathbb{1}})$ corresponding to the vector $\mathbb{1} \equiv (1,\ldots,1) \in \mathbb{R}^J_{*+}$ then $c^{\mathbb{1}} = \sum_{j\in J} c_j \leqslant c^\pi$ for all $\pi \in \mathbb{R}^J_{*+}$. Consequently all feasible solution of $(K^{\mathbb{1}})$ is feasible in any (K^π) and then $z_1^{\mathbb{1}} \leqslant z_1^\pi$ for all $\pi \in \mathbb{R}^J_{*+}$. ■

3.4 LP Relaxation of the Surrogate Relaxation

Hence, the surrogate relaxation consists in agregating all the containers as one – of capacity $\sum_{j\in J} c_j$ – and selecting the items to pack into this big container. This 0-1 knapsack problem (P_1) may be solved quite quickly using dynamic programming (or branch-and-bound). Its LP relaxation $(\overline{P_1})$ gives also an upper bound $\overline{z_1}$ of the MKP. Interestingly, this bound is equal to $\bar{z}$.

Proposition 7

$$\overline{z_1} = \bar{z}$$

Proof. It is clear that for any non-negative vector π, the LP-relaxation $(\overline{P_1^\pi})$ of the surrogate relaxation (P_1^π) coincides with the surrogate relaxation $(\bar{P}_1^\pi)$ of the LP relaxation $(\bar{P})$, which is itself a relaxation of $(\bar{P})$. Hence:

$$\overline{z_1} = \min_\pi \overline{z_1^\pi} = \min_\pi \bar{z}_1^\pi \geqslant \bar{z}.$$

Conversely, an optimal solution $\bar{y}$ of $(\overline{P_1})$ can easily be determined by Dantzig theorem. Assume that the items are sorted in decreasing order of ratio p_i/w_i and let s be the critical item $s = \min\{k \in I \mid \sum_{i=1}^k w_i > \sum_{j\in J} c_j\}$, then we define $\bar{y}_i = 1$ for items $i < s$ and $\bar{y}_s = (\sum_{j\in J} c_j - \sum_{i=1}^{s-1} w_i)/w_s$. The cost of this solution coincides with $\bar{z}$ according to Proposition 3. ■

3.5 Lagrangian Relaxation

The principle of lagrangian relaxation consists also in merging a conjunction of constraints in a MILP as a linear expression. There, contrary to the surrogate relaxation, the linear expression is not considered as a constraint within the problem but as a violation penalty cost in the objective function.

3.5.1 Relaxing knapsack constraints.

We first apply this principle to the MKP by relaxing the knapsack constraints. Let $\pi \in \mathbb{R}_+^J$ a vector of non-negative multipliers, we define the following BIP:

$$(P_2^\pi) : z_2^\pi = \max \sum_{i\in I}\sum_{j\in J} p_i x_{ij} - \sum_{j\in J} \pi_j (\sum_{i\in I} w_i x_{ij} - c_j), \quad \text{s.t.} \quad \sum_{j\in J} x_{ij} \leqslant 1 (\forall i \in I), \quad x \in \{0,1\}^{I\times J}.$$

(P_2^π) is a relaxation of (P) since for any feasible solution x of (P), x is a feasible solution of (P_2^π) and:

$$\sum_{i\in I}\sum_{j\in J} p_i x_{ij} - \sum_{j\in J} \pi_j (\sum_{i\in I} w_i x_{ij} - c_j) \geqslant \sum_{i\in I}\sum_{j\in J} p_i x_{ij}.$$

(P_2^π) is a *lagrangian subproblem* of (P) and π is a *price* or *lagrangian multiplier*.

Again, the goal of lagrangian relaxation is to determine the best of these relaxations, i.e. to solve the following problem:

$$(P_2) : z_2 = \min\{z_2^\pi \mid \pi \in \mathbb{R}_+^J\}.$$

(P_2) and (P) form a weak-dual pair since $z(x) \leqslant z_2^\pi$ for all feasible solution x of (P) and for all feasible solution π of (P_2). Problem (P_2) is actually called the *lagrangian dual problem* of (P).

The difference $z_2 - z$ is called the *dual gap*. In some cases, the dual gap can be zero, then an optimal solution of (P_2) may leads to an optimal solution of (P):

Proposition 8 *Let $\pi \in \mathbb{R}_+^J$ and x^π an optimal solution of (P_2^π),*

$$\begin{aligned} &\textit{if} \quad \sum_{i\in I} w_i x_{ij}^\pi \leqslant c_j, && \forall j \in J, \\ &\textit{and} \quad \sum_{i\in I} w_i x_{ij}^\pi = c_j, && \forall j \in J \mid \pi_j > 0, \end{aligned}$$

then x^π is optimum for (P).

Proof. By assumption, x^π is a feasible solution of (P), and it is optimum since:

$$z \leqslant z_2^\pi = \sum_{i\in I}\sum_{j\in J} p_i x_{ij}^\pi - \sum_{j\in J} \pi_j (\sum_{i\in I} w_i x_{ij}^\pi - c_j) = \sum_{i\in I}\sum_{j\in J} p_i x_{ij}^\pi \leqslant z.$$

■

Note that the lagrangian relaxation of a given set of constraints is always dominated by the surrogate relaxation of the same set of constraints. Indeed, each lagrangian suproblem (P_2^π) is clearly a relaxation of the surrogate supbroblem (P_1^π) associated to the same multiplier $\pi \in \mathbb{R}_+^J$: $z_1^\pi \leqslant z_2^\pi$. Hence, we have $z_1 = \min_\pi z_1^\pi \leqslant \min_\pi z_2^\pi = z_2$.

In the context of MKP, we see that this lagrangian bound z_2 is not even better than the LP bound $\bar{z}$:

Proposition 9

$$z_2 = \bar{z}$$

Proof. Observe that the formulation of (P_2^π) is ideal (the matrix is totally unimodular and the right hand side is integer) then $z_2^\pi = \overline{z_2^\pi} \geqslant \bar{z}$, and then $z_2 \geqslant \bar{z}$. Conversely, it is clear that an optimal solution of (P_2^π), with $\pi \in \mathbb{R}^J_{*+}$, can be obtained by determining a container j' with the smallest $\pi_j > 0$, then by placing in this container alone, all the items i such that $p_i - \pi_{j'} w_i > 0$. Consider the problem with $\pi_j = p_s/w_s$ for all $j \in J$, where s is the critical item of Proposition 3, then the optimum of (P_2^π) is $z_2^\pi = \sum_{i\in I}\sum_{j\in J}(p_i - \frac{p_s}{w_s}w_i)x_{ij} - \sum_{j\in J}\frac{p_s}{w_s}c_j = \sum_{i=1}^{s-1}(p_i - \frac{p_s}{w_s}w_i) + \sum_{j\in J}\frac{p_s}{w_s}c_j$. Hence, $\bar{z} \leqslant z_2 \leqslant z_2^\pi = \bar{z}$ according to Proposition 3. ■

Note that in this specific case, we were able to determine the optimal solution $\pi \equiv p_s/w_s$ of the lagrangian dual problem. This is not always the case, as we will see in the next section. Incidently, the vector $\pi \equiv p_s/w_s$ corresponds to the optimal dual variables associated to the knapsack constraints in the LP relaxation $(\bar{P})$ (see Proposition 3).

3.5.2 Relaxing assignment constraints.

The principle of lagrangian relaxation is useful in order to relax complicating constraints or coupling constraints. A constraint is said **coupling** if the problem obtained by relaxing it can naturally be decomposed into a series of independant problems.

In the context of MKP, the assignment constraints are coupling since, when they are relaxed, the problem can be decomposed into a series of m independant 0-1 knapsack problems.

We want to solve the lagrangian dual problem:

$$(P_3) : z_3 = \min\{z_3^\mu \mid \mu \in \mathbb{R}^I_+\}$$

where the lagrangian subproblem for a multiplier $\mu \in \mathbb{R}^I_+$ is defined by:

$$(P_3^\mu) : z_3^\mu = \max \sum_{i\in I}\sum_{j\in J} p_i x_{ij} - \sum_{i\in I}\mu_i(\sum_{j\in J} x_{ij} - 1), \quad \text{s.t.} \quad \sum_{i\in I} w_i x_{ij} \leqslant c_j (\forall j \in J), \quad x \in \{0,1\}^{I\times J}.$$

Note that in this relaxation, one exemplar of an item is allowed to be placed in each container. Actually (P_3^μ) can be decomposed as a series of m independant 0-1 knapsack problems, one for each container $j \in J$:

$$(K_j^\mu) : k_j^\mu = \max \sum_{i\in I}(p_i - \mu_i)y_i, \quad \text{s.t.} \quad \sum_{i\in I} w_i y_i \leqslant c_j, \quad y \in \{0,1\}^I.$$

Proposition 10

$$z_3^\mu = \sum_{j\in J} k_j^\mu + \sum_{i\in I}\mu_i.$$

Proof. Let $y_i^j = x_{ij}$ for all item $i \in I$ and for all container $j \in J$ then x is a feasible solution of (P_3^μ) if and only if each y^j is a feasible solution of (K_j^μ), for all container j. Furthermore, the cost of x is the sum of the costs of y^j plus $\sum_{i\in I}\mu_i$:

$$\begin{aligned} z_3^\mu &= \max(\sum_{i\in I}\sum_{j\in J} p_i x_{ij} - \sum_{i\in I}\mu_i(\sum_{j\in J} x_{ij} - 1)) = \max \sum_{i\in I}\sum_{j\in J}(p_i - \mu_i)x_{ij} + \sum_{i\in I}\mu_i \\ &= \sum_{j\in J}\max(\sum_{i\in I}(p_i - \mu_i)y_i^j) + \sum_{i\in I}\mu_i = \sum_{j\in J} k_j^\mu + \sum_{i\in I}\mu_i. \end{aligned}$$

■

Note that all knapsack problems (K_j^μ) are defined on the same set of items (with value $p_i - \mu_i$ and with weight w_i) but on containers of different capacities. This can be exploited in order to solve efficiently the series of knapsacks for a given μ. The bound z_3^{μ} can be computed in pseudo-polynomial time or approximated in polynomial time by the sum of the Dantzig bounds for the k_j^{μ}.

Contrary to (P_1) and (P_2), it is not known how to determine analytically the optimum multiplier μ for (P_3). An approximation of this optimum can be obtained through *subgradient optimization* or *cutting-plane generation* which are, however, generally time consuming. For saving computational time, one may be interested by solving only one lagrangian subproblem (P_3^{μ}). The optimal dual variables $\bar{\mu}$ associated with the assignment constraints in $(\bar{P})$ can be a good choice for the multiplier (see Proposition 3):

$$\bar{\mu}_i = \begin{cases} p_i - \frac{w_i}{w_s} p_s & \forall i \in I, i < s \\ 0 & \forall i \in I, i \geqslant s. \end{cases}$$

Furthermore, considering this multiplier allows us to prove that the lagrangian bound z_3 is always as good as the LP bound $\bar{z}$:

Proposition 11

$$z_3 \leqslant z_3^{\bar{\mu}} \leqslant \overline{z_3^{\bar{\mu}}} = \bar{z}.$$

Proof. Following the previous proposition, we have that $\overline{z_3^{\bar{\mu}}} = \sum_{j \in J} \overline{k_j^{\bar{\mu}}} + \sum_{i \in I} \bar{\mu}_i$ where $\overline{k_j^{\bar{\mu}}}$ is the optimum of $(\overline{K_j^{\bar{\mu}}})$ the LP relaxation of $(K_j^{\bar{\mu}})$. Furthermore, for each knapsack problem $(K_j^{\bar{\mu}})$, we have: $\frac{p_i - \bar{\mu}_i}{w_i} = \frac{p_s}{w_s}$ for all $i < s$ and $\frac{p_i - \bar{\mu}_i}{w_i} = \frac{p_i}{w_i} \leqslant \frac{p_s}{w_s}$ for all $i \geqslant s$. It means that the items are ordered for applying Dantzig theorem. Furthermore, as $c_j \leqslant \sum_{j \in J} c_j$, then only items $i \leqslant s$, i.e. items with the same maximal ratio $(p_i - \bar{\mu}_i)/w_i = p_s/w_s$, can be placed on the container j. Let s^j be the critical item for container j then

$$\overline{k_j^{\bar{\mu}}} = \sum_{i=1}^{s^j - 1} (p_i - \bar{\mu}_i) + (p_{s^j} - \bar{\mu}_{s^j}) \frac{c_j - \sum_{i=1}^{s^j - 1} w_i}{w_{s^j}} = \sum_{i=1}^{s^j - 1} w_i \frac{p_s}{w_s} + \frac{p_s}{w_s}(c_j - \sum_{i=1}^{s^j - 1} w_i) = \frac{p_s}{w_s} c_j.$$

Hence, $\overline{z_3^{\bar{\mu}}} = \sum_{j \in J} \frac{p_s}{w_s} c_j + \sum_{i=1}^{s-1} (p_i - \frac{w_i}{w_s} p_s) = \bar{z}$ from Proposition 3. ■

3.6 Conclusion

The upper bounds that we have considered for MKP can be sorted as follows:

$$\begin{array}{lcl} z \leqslant & z_3 \leqslant z_3^{\bar{\mu}} & \leqslant z_2 = \overline{z_1} = \bar{z} \leqslant z_0 \\ z \leqslant & z_1 & \leqslant z_2 = \overline{z_1} = \bar{z} \leqslant z_0 \end{array}$$

No general dominance exists between the surrogate bound z_1 and the lagrangian bound z_3. The surrogate bound z_1 can be computed in pseudo-polynomial time, by solving the 0-1 knapsack problem of Proposition 6, while the LP bound $\bar{z}$ can be computed in linear time, by solving the LP relaxation of this knaspack problem, i.e by computing formula of Proposition 3. The lagrangian bound z_3 can be computed or at least approximated by solving iteratively several instances of (P_3^{μ}). Each instance can be solved in pseudo-polynomial time by solving a series of similar knapsack problems.

Note that it is easy to check whether a solution of a given instance (P_3^{μ}) is feasible for (P): one has to ensure that each item appears in at most one sub-solution of the knapsack problems (K_j^{μ}), $j \in J$. Furthermore, if the solution is optimum for (P_3^{μ}) and if any item i such that $\mu_i > 0$ is selected (in exactly one knapsack) then the solution is optimum for (P). On the contrary, checking feasibility for the surrogate relaxation (P_1) is NP-hard, since it consists in checking whether the subset of selected items – which is globally compatible with the set of containers – can be decomposed into m subsets, compatible each with one of the containers. This problem is a generalization of the Bin-Packing Problem where bins (containers) are of different capacities.

3.7 Example

Consider the MKP with 6 items of values $p = (110, 150, 70, 80, 30, 5)$ and sizes $w = (40, 60, 30, 40, 20, 5)$, and 2 containers of capacities $c = (65, 85)$. First the items are already sorted in decreasing order of ratio $\frac{p_i}{w_i}$:

$$\frac{11}{4} > \frac{5}{2} > \frac{7}{3} > 2 > \frac{3}{2} > 1.$$

The critical item is $s = 4$ since $40 + 60 + 30 + 40 = 170 > 150 = 65 + 85$ and the residual capacity is $\bar{c} = 150 - (40 + 60 + 30) = 20$

The trivial bound is $z_0 = 110 + 150 + 70 + 80 + 30 + 5 = 445$.

For the LP bound, we need to compute the Dantzig bound of the 0-1 knapsack problem (P_1). $(1, 1, 1, \frac{1}{2}, 0, 0)$ is optimum and its cost is $\bar{z} = 370$.

For the surrogate bound, we need to solve at optimality the 0-1 knapsack problem (P_1) with capacity 150. $(1, 1, 1, 0, 1, 0)$ is optimum and its cost is $z_1 = 110 + 150 + 70 + 30 = 360$. In this specific case, we have $z_1 < \bar{z}$.

For the lagrangian bound approximation $z_3^{\bar{\mu}}$, we need to solve at optimality the 0-1 knapsack problems $(K_1^{\bar{\mu}})$ and $(K_2^{\bar{\mu}})$ with $\bar{\mu} = (110 - 40 * 2, 150 - 60 * 2, 70 - 30 * 2, 0, 0, 0) = (30, 30, 10, 0, 0, 0)$. $(0, 1, 0, 0, 0, 1)$ is optimum for $(K_1^{\bar{\mu}})$ and its cost is $k_1^{\bar{\mu}} = 125$. $(1, 0, 0, 1, 0, 1)$ is optimum for $(K_2^{\bar{\mu}})$ and its cost is $k_2^{\bar{\mu}} = 165$. then $z_3^{\bar{\mu}} = 125 + 165 + 70 = 360$. In this specific case, we have $z_1 = z_3^{\bar{\mu}} < \bar{z}$. Note that the optimum solution of $(P_3^{\bar{\mu}})$ is not feasible for (P) since item 6 is selected in both knapsacks.

Relaxations and Bounds (II): Lagrangian Relaxation

M2 ORO: Advanced Integer Programming
Lecture 8
Sophie.Demassey@mines-nantes.fr

November 7, 2011

1 Lagrangian Relaxation

Consider the optimization problem $(P) : z = \max\{cx \mid Ex \leqslant d, x \in X\}$, where X is a set of vectors in $\mathbb{R}^n_+$, $c \in \mathbb{R}^n$, $E \in \mathbb{R}^{|J| \times n}$, $d \in \mathbb{R}^{|J|}$, and assume that constraints $Ex \leqslant d$ are complicating, in the sense that problem (P) without these constraints is easy to solve.

Definition 1 *For any non-negative vector $\mu \in \mathbb{R}^J_+$, the **lagrangian subproblem** associated to the **multipliers** μ by **dualizing** constraints $Ex \leqslant d$ in (P) is defined by:*

$$(P^\mu) : z^\mu = \max\{cx - \mu(Ex - d) \mid x \in X\}.$$

*The **lagrangian dual** of (P) relative to constraints $Ex \leqslant d$ is defined by:*

$$(L) : u = \min\{z^\mu \mid \mu \in \mathbb{R}^J_+\}.$$

*u is called the **lagrangian bound** of (P).*

Clearly, any lagrangian subproblem is a relaxation of (P) and the lagrangian dual problem is to determine the best relaxation:

Proposition 1 $z \leqslant u \leqslant z^\mu, \quad \forall \mu \in \mathbb{R}^J_+$

Proof. Any feasible solution x of (P) is feasible for (P^μ) for any $\mu \in \mathbb{R}^J_+$ and $cx \leqslant cx - \mu(Ex - d)$. ■

Conversely, a solution x^μ of a lagrangian subproblem (P^μ) is *almost* feasible for (P). It satisfies all constraints of (P) except, possibly, some constraints in $Ex \leqslant d$. Furthermore, any such violated constraint is penalized by a negative cost $-\mu(E_j x^\mu - d)$ within the objective function of (P^μ). Note however, that an optimum solution of a lagrangian subproblem that is feasible for (P) is not necessary optimum for (P). Indeed, according to the following proposition, (μ, x^μ) must also satisfy the complementary slackness condition $\mu(Ex - d) = 0$:

Proposition 2 *If x^μ is an optimum solution of (P^μ) such that:*

$$E_j x^\mu \leqslant d_j \qquad \forall j \in J \tag{1}$$

$$E_j x^\mu = d_j \qquad \forall j \in J \mid \mu_j > 0 \tag{2}$$

then x^μ is optimum for (P).

Proof. By assumption (1), x^μ is feasible for (P) and, by assumption (2), it is optimum since $z \leqslant z^\mu = cx^\mu - \mu(Ex^\mu - d) = cx^\mu \leqslant z$. ■

Note that when the constraints to dualize are equality constraints $Ex = d$ then the multiplier are not any longer restricted to be non-negative, $(L) : u = \min\{z^\mu \mid \mu \in \mathbb{R}^J\}$. Furthermore, any optimum solution of (P^μ) that is feasible for (P) is necessary optimum for (P), since $z \leqslant z^\mu = cx^\mu - \mu(Ex^\mu - d) = cx^\mu \leqslant z$.

1.1 Strength of the Lagrangian Bound

The following theorem interprets the dual relaxation (L), defined on the dual space $\mu \in \mathbb{R}^J_+$, as a primal relaxation, defined on the primal space $x \in \mathbb{R}^n_+$.

Theorem 3

$$u = \max\{cx \mid Ex \leqslant d,\ x \in conv(X)\}$$

Proof. If X is empty then all problems (P) and (P^μ) are infeasible, then $u = -\infty$. Otherwise, assume that X contains a finite set of points $X = \{x^1, \dots, x^T\}$, then:

$$u = \min\{\max_{t=1..T}(cx^t - \mu(Ex^t - d)) \mid \mu \in \mathbb{R}^J_+\} = \min\{y \mid y \geqslant cx^t - \mu(Ex^t - d)(\forall t = 1, \dots, T),\ \mu \in \mathbb{R}^J_+, y \in \mathbb{R}\}$$

This is a LP then, by strong duality, we have:

$$u = \max\{\sum_{t=1}^{T}(cx^t)\lambda_t \mid \sum_{t=1}^{T}\lambda_t = 1, \sum_{t=1}^{T}(E_j x^t - d_j)\lambda_t \leqslant 0(\forall j \in J)\ ,\ \lambda \in \mathbb{R}^T_+\ \}$$

As $conv(X) = \{\sum_{t=1}^T x^t\lambda_t \mid \sum_{t=1}^T \lambda_t = 1\}$, the result follows. It holds also when X is not a finite set. ■

Corollary 4 *if* (P) *is an LP then* (L) *is equivalent to* (P). *Furthermore the dual optimal values* μ^* *associated to constraints* $Ex \leqslant d$ *are the optimum multiplier of* (L).

Proof. $(\mathrm{L}) \equiv (\mathrm{P})$ is a direct consequence of the theorem: when $X = \{x \in \mathbb{R}^n_+ \mid Ax \leqslant b\ \}$ then $conv(X) = X$. It can also be proved by considering the LP duals (D) of (P) and (D^μ) of (P^μ). Actually, if (P) is an LP then any lagrangian subproblem is an LP $(\mathrm{P}^\mu) : z^\mu = \max\{(c - \mu E)x + \mu d \mid Ax \leqslant b, x \in \mathbb{R}^n_+\}$, whose LP dual is defined by $(\mathrm{D}^\mu) : w^\mu = \min\{\lambda b + \mu d \mid \lambda A \geqslant (c - \mu E),\ \lambda \geqslant 0\}$.
As the LP dual of (P) is defined by:

$$(\mathrm{D}) : w = \min\{\lambda b + \mu d \mid \lambda A + \mu E \geqslant c,\ \lambda, \mu \geqslant 0\} = \min\{w^\mu \mid \mu \geqslant 0\},$$

then, by strong duality, we have that

$$(\mathrm{L}) \equiv \min_{\mu \geqslant 0}(\mathrm{P}^\mu) \equiv \min_{\mu \geqslant 0}(\mathrm{D}^\mu) \equiv (\mathrm{D}) \equiv (\mathrm{P}).$$

Furthermore, if (λ^*, μ^*) is optimum for (D) then μ^* is optimum for (L). ■

Corollary 5 *if* (P) *is an ILP then the lagrangian relaxation* (L) *is at least as good as the LP relaxation* $(\bar{\mathrm{P}})$.

Proof. Let $X = \{x \in \mathbb{Z}^n_+ \mid Ax \leqslant b\ \}$ then

$$\begin{aligned}\{x \in \mathbb{Z}^n_+ \mid Ax \leqslant b, Ex \leqslant d\ \} &\subseteq conv\{x \in \mathbb{Z}^n_+ \mid Ax \leqslant b, Ex \leqslant d\ \} \\ &\subseteq conv(X) \cap \{x \in \mathbb{R}^n_+ \mid Ex \leqslant d\ \} \\ &\subseteq \{x \in \mathbb{R}^n_+ \mid Ax \leqslant b, Ex \leqslant d\ \}.\end{aligned}$$

Hence, $(\bar{\mathrm{P}})$ is a relaxation of (L) which is a relaxation of (P): $\bar{z} \geqslant u \geqslant z$. ■

In some cases, when formulation X is integral, the two relaxations are equivalent:

Corollary 6 *If* (P) *is an ILP with* $X = \{x \in \mathbb{Z}^n_+ \mid Ax \leqslant b\ \}$ *and* $conv(X) = \{x \in \mathbb{R}^n_+ \mid Ax \leqslant b\ \}$ *then* $(\mathrm{L}) \equiv (\bar{\mathrm{P}})$.

Even in the case where (L) is not better than $(\bar{\mathrm{P}})$, this corollary is of practical interest since it provides an alternative way to the simplex method, to solve – or at least to approximate – $(\bar{\mathrm{P}})$. Consider for example that $conv(X)$ is defined by an exponential number of constraints so that the LP $\max\{cx \mid Ex \leqslant d,\ x \in conv(X)\}$ cannot be solved directly, but such that any lagrangian subproblem $\max\{cx - \mu(Ex - d) \mid x \in X\}$ is a classical combinatorial problem for which there exists a polynomial time algorithm. Then the LP can be approximated by solving iteratively several lagrangian subproblems.

2 Solving the Lagrangian Dual Problem

The dual problem (L) is a min-max problem. The question is then:

$$\boxed{\text{how to solve } (L) \text{ without solving all lagrangian subproblems } (P^{\mu})\ ?}$$

When (P) is an integer linear program, then Theorem 3 indicates to ways to solve the lagrangian dual problem (L): either by solving a linear program or by minimizing the convex fonction $\mu \mapsto z^{\mu}$. In both cases, the algorithms consist in solving iteratively subproblems (P^{μ}) for different values of multiplier μ, then recording the best (i.e. the lowest) optimum z^{μ^*} found. They result only in an approximation (an upper bound) of the lagrangian dual optimum u, unless in the case where $(P) \equiv (L)$, where we find an optimum solution of subproblem (P^{μ^*}) that is optimum for (P). The two agorithms differ by the way the vector μ is updated at each iteration.

2.1 Cutting-plane generation

Theorem 3 shows that (L) can be formulated as a LP having potentially an exponential number of constraints:

$$u = \min\{y \mid y \geqslant cx^t - \mu(Ex^t - d)(\forall t),\ \mu \in \mathbb{R}^J_+, y \in \mathbb{R}\ \}$$

Solving this LP requires to use a cutting-plane generation algorithm: starting with an arbitrary multiplier $\mu^0 \in \mathbb{R}^J_+$, then at each iteration $t \geqslant 1$, a new constraint $y \geqslant cx^t - \mu(Ex^t - d)$ is added to the LP, where x^t is an optimum solution of the lagrangian subproblem $(P^{\mu_{t-1}})$ such that $(\mu_{t-1}, y_{t-1}) \in \mathbb{R}^J_+ \times \mathbb{R}$ is the optimum solution of the LP computed at the previous iteration $t-1$. The algorithm stops when the constraint to add is already satisfied by the current LP solution:

Proposition 7 *At iteration* $t > 1$, *if* $y_{t-1} \geqslant cx^t - \mu_{t-1}(Ex^t - d)$ *then* $u = y_{t-1} = z^{\mu_{t-1}}$.

Proof. Let (LP_{t-1}) denotes the LP at iteration $t-1$:

$$(LP_{t-1}) : \min\{y \mid y \geqslant cx^s - \mu(Ex^s - d)(\forall s = 1, \ldots, t-1),\ (\mu, y) \in \mathbb{R}^J_+ \times \mathbb{R}\ \}.$$

On one hand, as (μ_{t-1}, y_{t-1}) is an optimum solution of (LP_{t-1}), then there exists $s \in \{1, \ldots, t-1\}$ such that $y_{t-1} = cx^s - \mu_{t-1}(Ex^s - d)$. As x^t is an optimum solution and x^s is a feasible solution for $(P^{\mu_{t-1}})$, then we have:

$$z^{\mu_{t-1}} = cx^t - \mu_{t-1}(Ex^t - d) \geqslant cx^s - \mu_{t-1}(Ex^s - d) = y_{t-1}.$$

By assumption, we conclude that $y_{t-1} = z^{\mu_{t-1}}$. On the other hand, by definition of z^{μ}, it is easy to see that (μ, z^{μ}) is a feasible solution of (LP_{t-1}) for all $\mu \in \mathbb{R}^J_+$. Since $(\mu_{t-1}, z^{\mu_{t-1}})$ is optimum (i.e. minimum) for (LP_{t-1}), it then implies that $z^{\mu_{t-1}} \leqslant z^{\mu}$ for all $\mu \in \mathbb{R}^J_+$, and then we conclude that $u = z^{\mu_{t-1}}$. ■

This method is often time consuming because it may require an exponential number of iterations and the LP becomes bigger and harder to solve at each iteration. Obviously, the algorithm can be stopped at any moment, after a given ellapsed time or after a given number of iterations. As the value of the LP decreases at each iteration, the last LP optimum computed gives an upper bound of u.

2.2 Subgradient algorithm

It is not hard to see that the function $\mu \mapsto z^{\mu}$ is piecewise linear convex but non-differentiable: actually, for a sufficiently small change of μ, say $\mu + \epsilon$, the optimum solution of (P^{μ}) remains optimum for $(P^{\mu+\epsilon})$ and the delta of the optimum values $z^{\mu+\epsilon} - z^{\mu}$ is proportional to ϵ.

The subgradient algorithm is a simple and easy to implement approach for minimizing such a function. The algorithm starts with an arbitrary multiplier $\mu_0 \in \mathbb{R}^J_+$ then, at each iteration $t \geqslant 1$, a new multiplier is computed as $\mu_t = \max(0, \mu^{t-1} + \lambda_t(Ex^t - d))$ where x^t is an optimum solution of the lagrangian subproblem $(P^{\mu_{t-1}})$, and where $(\lambda_t)_{t \geqslant 0}$ are non-negative values called the *step lenghts*. The best bound z^{μ_t} computed so far is returned at end, after a fixed number of iterations or when no improvement is performed. The step lengths can be choosen according one of the three rules below:

Proposition 8 1. *if* $\lim_{t\to\infty}\lambda_t = 0$ *and* $\sum_{t\geqslant 0}\lambda_t \to \infty$ *then* $\lim_{t\to\infty} z^{\mu_t} = u$;

2. *if* $\lambda_t = \lambda_0\rho^t$ *for some sufficiently large values of* $\rho < 1$ *and* λ_0 *then* $\lim_{t\to\infty} z^{\mu_t} = u$;

3. *if* $\lambda_{t+1} = \epsilon_t(z^{\mu_t} - \bar{u})/||Ex^t - d||^2$ *where* $\bar{u}$ *is an upper bound of* u *and* $0 < \epsilon_t \leqslant 2$ *then either* $\lim_{t\to\infty} z^{\mu_t} = \bar{u}$ *or the algorithm finds some* μ_t *such that* $u \leqslant z^{\mu_t} \leqslant \bar{u}$.

The first rule is not of real practical interest as the convergence is ensured but slow (because the series $\sum_{t\geqslant 0}\lambda_t$ must be divergent). The two other rules lead to much faster convergence but have disadvantages too. For the second rule, the geometric series $\sum_{t\geqslant 0}\lambda_t$ may tend to zero too rapidly, and then the sequence (μ_t) may converge before reaching an optimal point. The difficulty when applying the last rule is due to the requirement to know an upper bound $\bar{u}$, but such a bound can be obtained by applying a heuristic to (P). This rule is often applied in practice by setting $\epsilon_0 = 2$ and halving ϵ_t whenever z^{μ_t} has failed to increase in some fixed number of iterations.

3 Embedding Lagrangian Relaxation within Branch-and-Bound

Solving the lagrangian dual problem (L) is generally not enough to solve the initial ILP (P), since, in mosts cases, there exists a duality gap $u - z > 0$. Furthermore, the iterative algorithms above terminate generally before the optimum u is attained. Hence, a branch-and-bound approach is required in order to solve (P) at optimality. The lagrangian relaxation can be used at each node of the search tree to provide an upper bound. Furthermore, it is also possible to derive from the lagrangian relaxation both a heuristic feasible solution and fixed variables.

3.1 Lagrangian Heuristic

The best lagrangian solution x^{μ^*} computed is close to be feasible for (P). It is often easy to devise an heuristic that converts x^{μ^*} into a feasible solution without greatly decreasing its cost. Such a problem specific method might be called *lagrangian heuristic*. A well-known example of lagrangian heuristic applies to the Generalized Assignment Problem, where n items must be packed into m containers:

$$(GAC) : z = \max\{\sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij} \mid \sum_{j=1}^{n} x_{ij} \leqslant 1\ (\forall i = 1,\dots,m), \sum_{i=1}^{m} a_{ij}x_{ij} \leqslant b_{ij}\ (\forall j = 1,\dots,n), x \in \{0,1\}^{m\times n}\}$$

By dualizing the first set of constraints, requiring that each item be used at most once, we get:

$$(GAC_\mu) : z_\mu = \max\{\sum_{i=1}^{m}\mu_i + \sum_{i=1}^{m}\sum_{j=1}^{n}(c_{ij} - \mu_i)x_{ij} \mid \sum_{i=1}^{m} a_{ij}x_{ij} \leqslant b_{ij}\ (\forall j = 1,\dots,n), x \in \{0,1\}^{m\times n}\}$$

An optimum solution x^μ of (GAC_μ) defines two types of items: the ones that are packed in at most one container ($\sum_{j=1}^{n} x_{ij} \leqslant 1$) and the other ones ($\sum_{j=1}^{n} x_{ij} > 1$). x^μ can easily be made feasible by removing the items of the second type from all but one container each. The best solution consists to choose, for each item i, a container j such that $x_{ij} = 1$ and that maximizes c_{ij}.

Consider the Set Covering Problem as a second example, where items $j = 1,\dots,n$ must be covered each by at least one selected set $i = 1,\dots,m$:

$$(SCP) : z = \min\{\sum_{i=1}^{m} c_i x_i \mid \sum_{i=1}^{m} a_{ij}x_i \geqslant 1\ (\forall j = 1,\dots,n), x \in \{0,1\}^m\}$$

and let dualize all the cover constraints. An optimum solution x^μ of lagrangian subproblem (SCP_μ) is easy to find: $x_i^\mu = 1$ if $c_i - \sum_{j=1}^{m}\mu_j a_{ij} \leqslant 0$ and $x_i^\mu = 0$ otherwise. Then a feasible solution of (SCP) can be computed by solving (heuristically) a much smaller instance of the set-covering problem obtained by removing from (SCP) all sets i selected in x^μ (i.e. $x_i^\mu = 1$) and all items j covered by x^μ (i.e. $\sum_{i=1}^{m} a_{ij}x_i^\mu \geqslant 1$). A feasible solution of (SCP) is given by $x^\mu + x'$ where x' is a feasible solution of the restricted set-covering problem.

Note that if the lagrangian heuristic runs fast, then it can be applied at each iteration of the lagrangian dual solution algorithm (e.g. subgradient method).

3.2 Variable fixing

Variable fixing is a very useful technique to accelerate a branch-and-bound by identifying the variables that can be set to their bounds. In this section, we consider that (P) is a Binary Integer Program, i.e. $X \subseteq \{0,1\}^I$. The technique tries to find variables which must necessary be set to 0 or to 1 in order to improve the current incumbent.

Given an incumbent value, i.e. a lower bound $\underline{z}$ of z, then any better feasible solution x satisfies:

$$\sum_{i \in I}(c_i - \mu E_i)x_i + \mu d = cx - \mu(Ex + d) \geqslant cx > \underline{z}.$$

Let partition the variables as follows: $I_1 = \{i \in I \mid c_i - \mu E_i > 0\}$ and $I_0 = \{i \in I \mid c_i - \mu E_i < 0\}$. If we do not consider the constraints of feasibility, then the maximum value for $\sum_{i \in I}(c_i - \mu E_i)x_i + \mu d$ can be attained by setting all variables $x_i = 1$ if $i \in I_1$ and $x_i = 0$ if $i \in I_0$.

Proposition 9 *Let $j \in I$ and x be a feasible solution of value cx strictly greater than $\underline{z}$, then:*

$$x_j = \begin{cases} 0 & \textit{if } j \in I_0 \textit{ and } \sum_{i \in I_1 \cup \{j\}}(c_i - \mu E_i) + \mu d \leqslant \underline{z} \\ 1 & \textit{if } j \in I_1 \textit{ and } \sum_{i \in I_1 \setminus \{j\}}(c_i - \mu E_i) + \mu d \leqslant \underline{z} \end{cases}$$

As lagrangian heuristics, variable fixing can also be applied at each iteration of the subgradient algorithm. Actually, it has more chance to be active in the last iterations, when z^μ is close to z.

3.3 Branching Strategy

Within a branch-and-bound, it could be helpful to exploit the structure of the complicating constraints within the branching strategy, in such a way that each branching decision will "solve" a complicating constraint (the constraint will be entailed in the subtree). Furthermore, one may design the branching strategies in order that the lagrangian subproblems at each node will have the same structure and will not be harder to solve than the lagrangian subproblems at the root node. Lastly, the last lagrangian subproblem solved at a current node can be used in the child nodes: the current solution helps to determine the next variable to branch on, while the current multiplier can be set as the initial multiplier value for the computation of the lagrangian bound in the child nodes.

For example, in the case of the (GAC) problem, a natural branching strategy is to choose at each node an item i and create $n+1$ branches: the n first branches correspond to a decision $x_{ij} = 1$ for $j = 1, \dots, n$ (item i is assigned once to container j) and the last branch corresponds to decision $\sum_{j=1}^{n} x_{ij} = 0$ (item i is not selected). A second strategy (called *GUB branching*) consists to create only two branches by selecting a subset of containers $J' \subsetneq J$ and to post $\sum_{j \in J'} x_{ij} = 0$ in the first branch and $\sum_{j \in J \setminus J'} x_{ij} = 0$ in the second branch. Clearly these two branching strategies satisfies the desired properties. For the choice of item i to branch on, we might consider the lagrangian subproblem solution x^{μ^*} at the current node in order to strenghten the bounds in the child nodes as much as possible, and choose item i that has the largest penalty cost $\mu_i^*(\sum_{j=1}^{n} x_{ij}^{\mu^*} - 1) > 0$: in any child node, as constraint $\sum_{j=1}^{n} x_{ij} \leqslant 1$ is satisfied, this penalty cost is eliminated when solving the lagrangian subproblem (GAC_μ^*) at the first iteration of the subgradient algorithm.

References

- The Lagrangian Relaxation Method for Solving Integer Programming Problems, Marshall L. FISCHER, *Management Science* 27(1):1-18, 1982.
- *Integer Programming*, Laurence A. WOLSEY, Wiley 1998.
- Lagrangean relaxation, Monique GUIGNARD, *TOP: An Official Journal of the Spanish Society of Statistics and Operations Research* 11(2):151-200, 2003.