Column generation

Operational Research Complements

Giovanni Righini
Università degli Studi di Milano

Column generation

Column generation is a technique to solve LP models with a very large set N of columns (variables):

$$z = \min\{cx : Ax = b, x \in \Re_+^{|N|}\}.$$

The main idea of column generation is to solve a restricted LP, where only a (small) subset $\overline{N} \subseteq N$ of columns are in the tableau and can be selected as basic:

$$z_{RM} = \min\{cx : Ax = b, x \in \Re_+^{|\overline{N}|}, x_j = 0 \ \forall j \in N \backslash \overline{N}\}.$$

When we find an optimal solution x^*, we also know the optimal dual solution λ^* and we can ask the question (pricing problem):

Is there a column $j \in N \backslash \overline{N}$ not currently in the tableau, such that its reduced cost $\overline{c}_j$ in the current basic solution is negative?

Column generation

The answer depends on coefficients A and c (i.e. the "structure" of the column), because

$$\overline{c}_j = c_j - \sum_{i=1}^{m} a_{ij}\lambda_i.$$

- If "No": x^* is optimal for the original LP.
- If "Yes": insert column j into $\overline{N}$ and reoptimize (dual simplex).

The algorithm proceeds by alternating two steps:

- solution of the LP with the column subset $\overline{N}$ (restricted master problem)
- search for negative reduced cost columns (without explicitly enumerating all of them): integer optimization sub-problem (pricing sub-problem).

Why should we use CG?

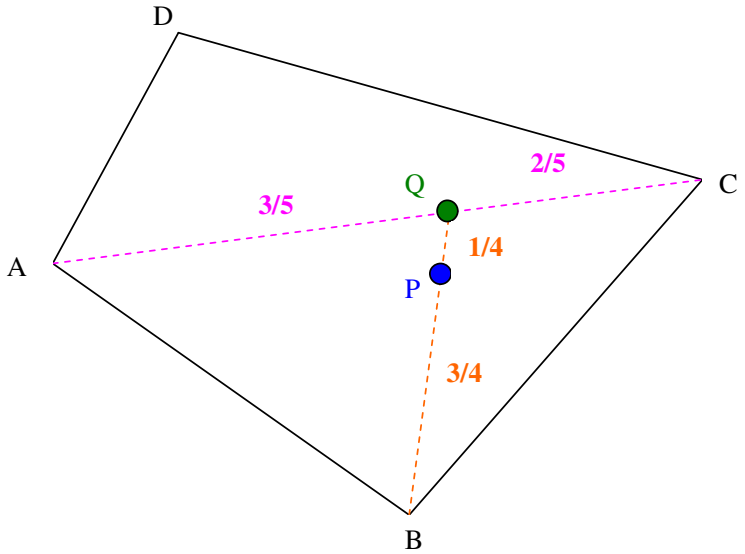
The main reasons for applying column generation to LP problems are

- the LP has a huge number of columns (variables): for instance, when variables have many indices.
- the LP has a huge number of rows (constraints): the computing time spent by the simplex algorithm usually depends more on the number of constraints than on the number of variables: so, we solve the dual.
- the LP has a block-diagonal structure, so that it can be decomposed into independent sub-problems when linking constraints are considered separately.
- the LP arise as the linear relaxations of an extended formulation of a combinatorial problem, with an exponential number of variables.

Minkowsky and Weil theorem

Theorem (Minkowsky and Weil). *Every point x in a polytope $\mathcal{P}$ can be obtained as a convex combination of its extreme points $x^{(u)}$.*

$$\forall x \in \mathcal{P}, \exists \theta \geq 0 : x = \sum_{u=1}^{p} \theta_u x^{(u)}, \sum_{u=1}^{p} \theta_u = 1.$$



$$Q = \tfrac{2}{5}A + \tfrac{3}{5}C$$
$$P = \tfrac{3}{4}Q + \tfrac{1}{4}B = \tfrac{3}{4}(\tfrac{2}{5}A + \tfrac{3}{5}C) + \tfrac{1}{4}B = \tfrac{6}{20}A + \tfrac{5}{20}B + \tfrac{9}{20}C.$$

Extended formulation

If $\mathcal{P}$ is the convex hull of a discrete set X of integer points,

$$\forall x \in X, \exists \theta \geq 0 : x = \sum_{u=1}^{p} \theta_u x^{(u)}, \sum_{u=1}^{p} \theta_u = 1.$$

In particular, this holds for the vertices of the polyhedron, i.e. the extreme points $x^{(u)}$ of $\mathcal{P} = conv(X)$.

Finding the optimal solution of an ILP

$$z^* = \min_{x \in \mathcal{Z}_+^n} \{cx : x \in X\} \text{ where } X = \{x \in \mathcal{Z}_+^n : Ax \geq b\}$$

is equivalent to solving the LP

$$z^* = \min_{x \in \Re_+^n} \{cx : x \in conv(X)\} = \min_{\theta \in \Re_+^p} \{cx : x = \sum_{u=1}^{p} \theta_u x^{(u)}, \sum_{u=1}^{p} \theta_u = 1\}.$$

Convexification

If an ILP problem

$$z^* = \min_{x \in \mathcal{Z}_+^n} \{cx : Ax \geq b\}$$

is such that the polyhedron $\{x \in \Re_+^n : Ax \geq b\}$ has integer extreme points, then the problem has got the integrality property (IP) and it can be solved as an LP problem.

When we solve an ILP without the IP to optimality, we are convexifying it, because we are finding the optimal solution of the LP whose polyhedron is the convex hull of the feasible integer solutions.

Given an ILP without the IP it often happens that

- we do not know how to convexify the whole ILP;
- we know how to convexify some sub-problem.

Linking and complicating constraints

We can rewrite an ILP

$$z^* = \min_{x \in \mathcal{Z}_+^n} \{cx : Ax \geq b\},$$

as

$$z^* = \min_{x \in \mathcal{Z}_+^n} \{cx : Dx \geq e, Fx \geq g\}.$$

There are two reasons for this:

- we are able to convexify $X = \{x \in \mathcal{Z}_+^n : Fx \geq g\}$; in this case $Dx \geq e$ are complicating constraints;
- $X = \{x \in \mathcal{Z}_+^n : Fx \geq g\}$ can be decomposed into independent sub-problems; in this case $Dx \geq e$ are linking constraints;
- both things may occur simultaneously.

Complicating constraints

After removing a set of complicating constraints from an ILP, we may be left with

- a sub-problem with the IP: nothing to convexify; the problem is defined by its ideal formulation.
  We cannot improve the dual bound given by the linear relaxation of the ILP; however, it may be fast to solve (possibly faster than by the simmplex algorithm).
- a sub-problem without the IP: if we can solve it to optimality, the result is the same as if we had solved the linear relaxation of the ideal formulation of the sub-problem.
  We can improve the dual bound given by the linear relaxation of the original ILP.

Complicating constraints

Example 1: the Capacitated Shortest Path Problem (CSPP).

Given a digraph $\mathcal{G} = (\mathcal{N}, \mathcal{A})$ with

- a cost $c_{ij} \in \Re_+ \ \forall (i,j) \in \mathcal{A}$
- a weight $w_{ij} \in \Re_+ \ \forall (i,j) \in \mathcal{A}$

find a directed path from a given node $s \in \mathcal{N}$ to a given node $t \in \mathcal{N}$, such that

- (constraint:) its weight does not exceed a given threshold W,
- (objective:) its cost is minimum.

Complicating constraints

We use arc variables $x_{ij} \in \{0, 1\}\ \forall (i, j) \in \mathcal{A}$:

$$\text{minimize } z^* = \sum_{(i,j)\in\mathcal{A}} c_{ij} x_{ij}$$

$$\text{s.t.} \sum_{j\in\mathcal{N}} x_{ij} = \sum_{j\in\mathcal{N}} x_{ji} \qquad \forall i \in \mathcal{N}\backslash\{s, t\}$$

$$\sum_{j\in\mathcal{N}} x_{sj} = \sum_{i\in\mathcal{N}} x_{it} = 1$$

$$\sum_{(i,j)\in\mathcal{A}} w_{ij} x_{ij} \leq W$$

$$x \in \mathcal{B}^{|\mathcal{A}|}$$

If the complicating constraint $\sum_{(i,j)\in\mathcal{A}} w_{ij} x_{ij} \leq W$ is relaxed, the remaining ILP has the IP (it can be solved as an LP).

Complicating constraints

Example 2: the Integer Knapsack Problem with a Demand Constraint (KPDC).

Given

- a set $\mathcal{N}$ of items,
- a value $v_j \quad \forall j \in \mathcal{N}$,
- two weights $w'_j, w''_j \quad \forall j \in \mathcal{N}$,
- two thresholds W' and W'',

select a suitable integer number of items of each type $j \in \mathcal{N}$ such that

- (capacity constraint:) their total weight according to w' does not exceed W',
- (demand constraint:) their total weight according to w'' is not smaller than W'',
- (objective:) their total value is maximum.

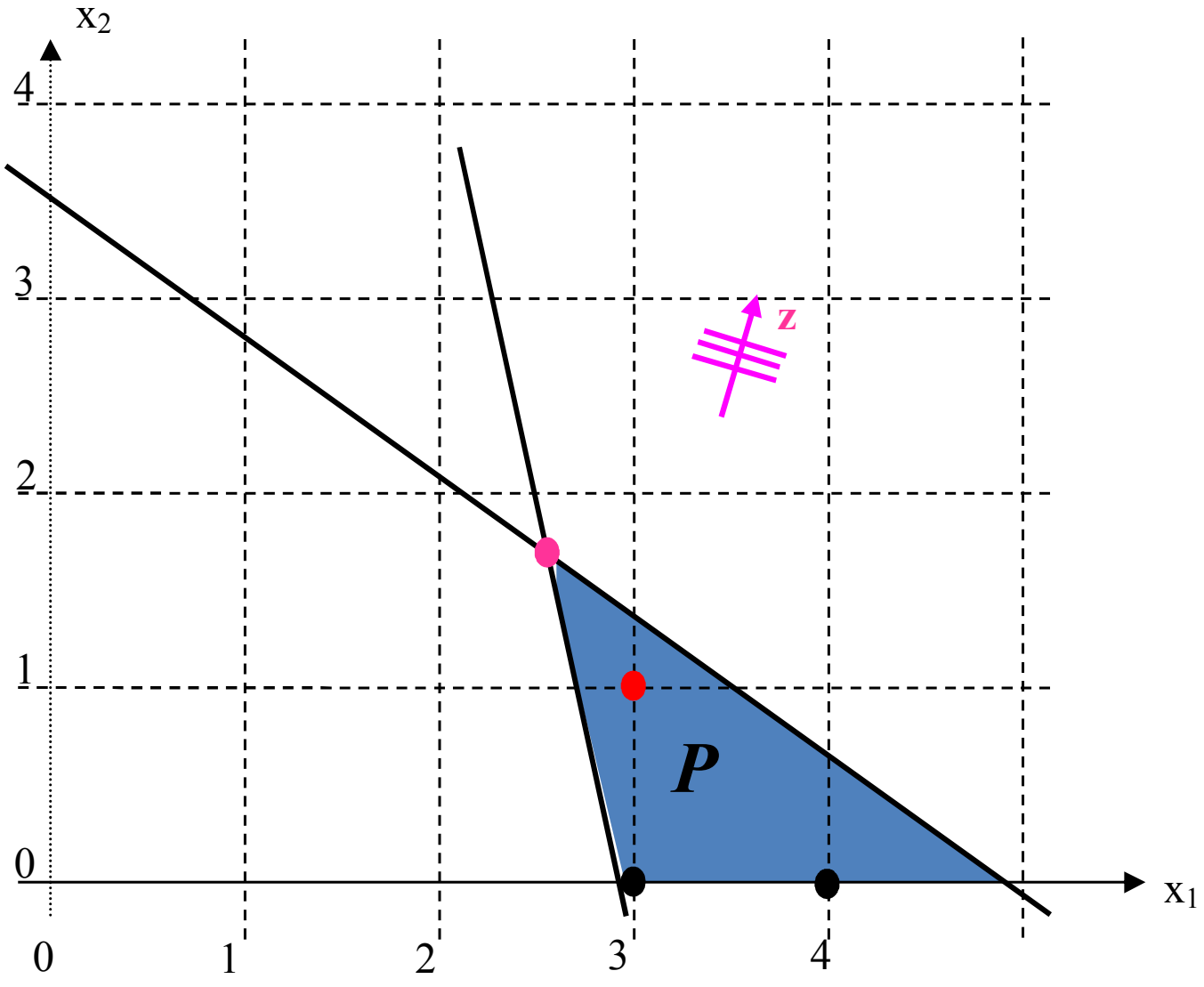
Complicating constraints

We use integer variables $x_j \quad \forall j \in \mathcal{N}$:

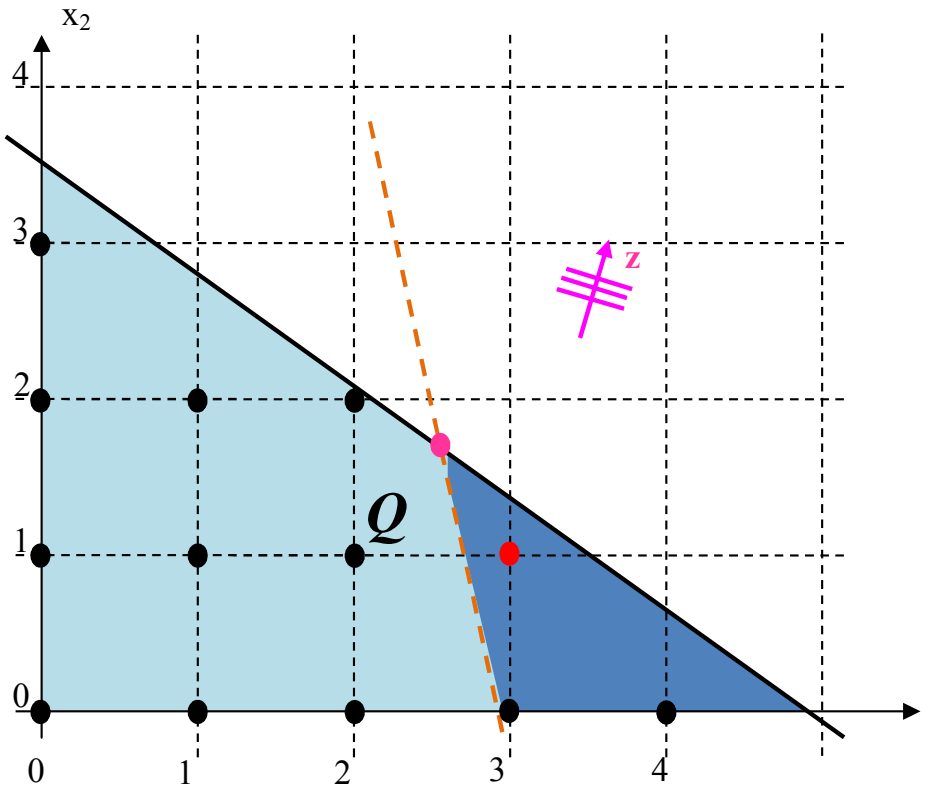
$$\begin{aligned} \text{maximize } z &= \sum_{j \in \mathcal{N}} v_j x_j \\ \text{s.t. } & \sum_{j \in \mathcal{N}} w'_j x_j \leq W' \\ & \sum_{j \in \mathcal{N}} w''_j x_j \geq W'' \\ & x \in \mathcal{Z}_+^{|\mathcal{N}|}. \end{aligned}$$

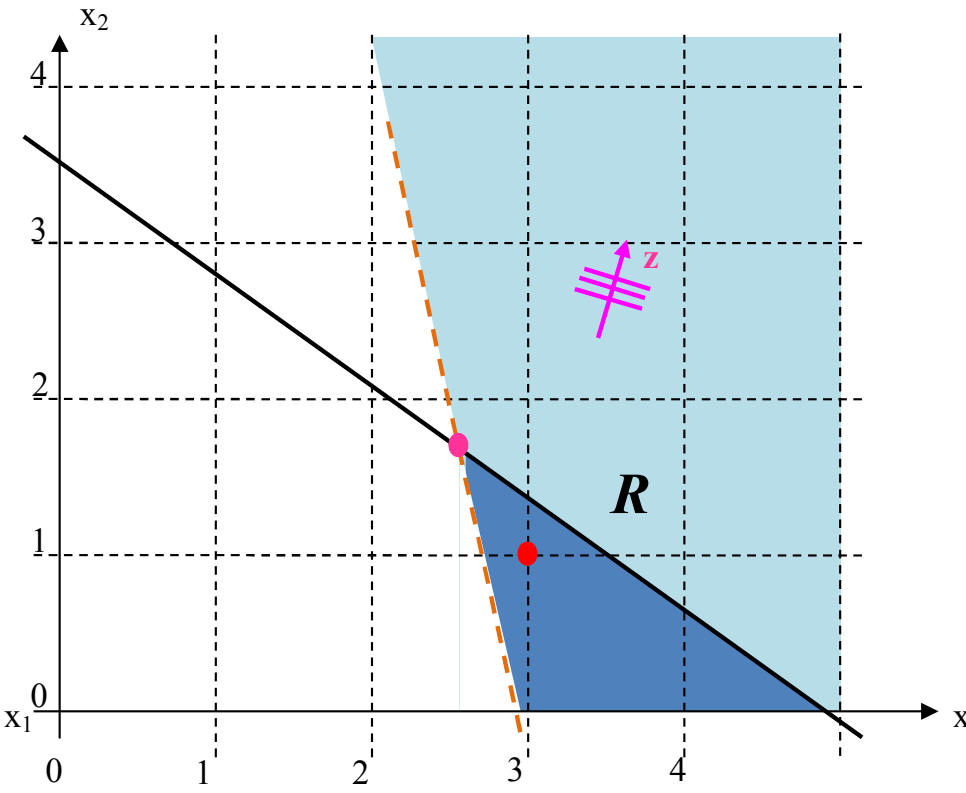
If the complicating constraint $\sum_{j \in \mathcal{N}} w''_j x_j \geq W''$ is relaxed, the remaining ILP is an Integer Knapsack Problem, that does not have the IP.

Convexification (1)

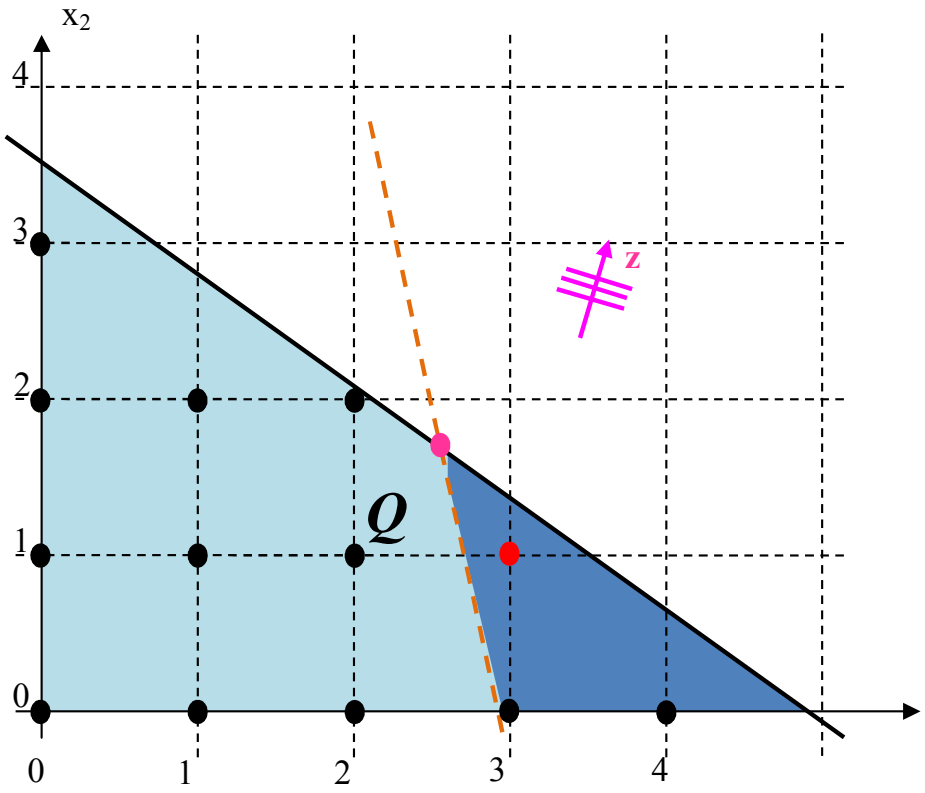


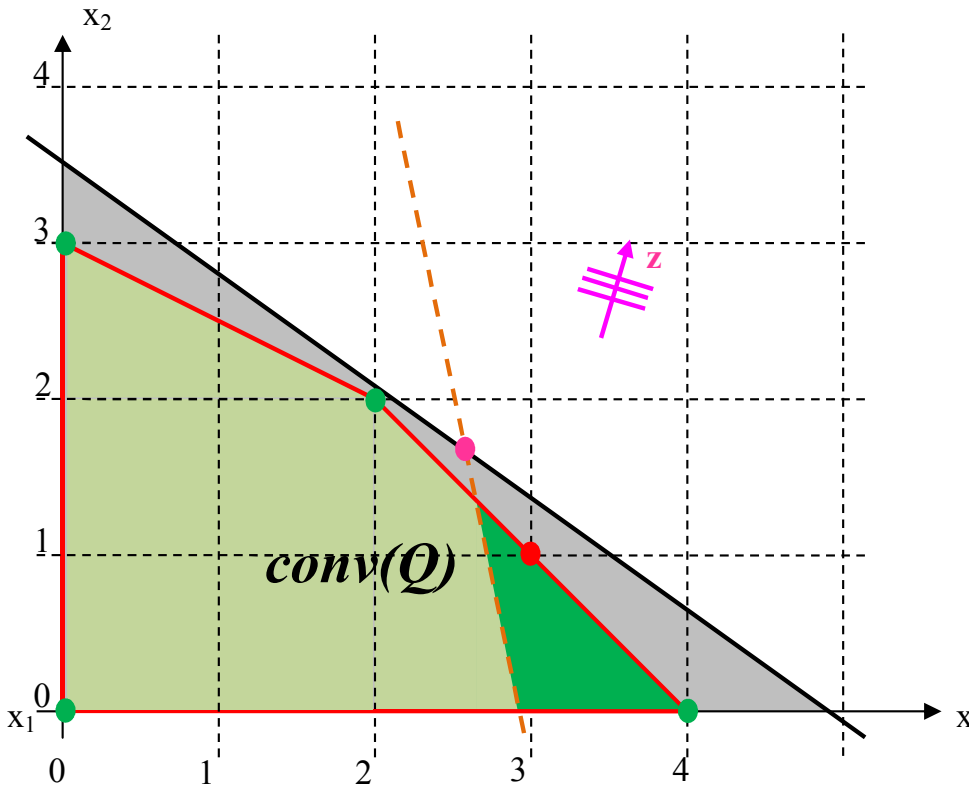
Convexification (2)



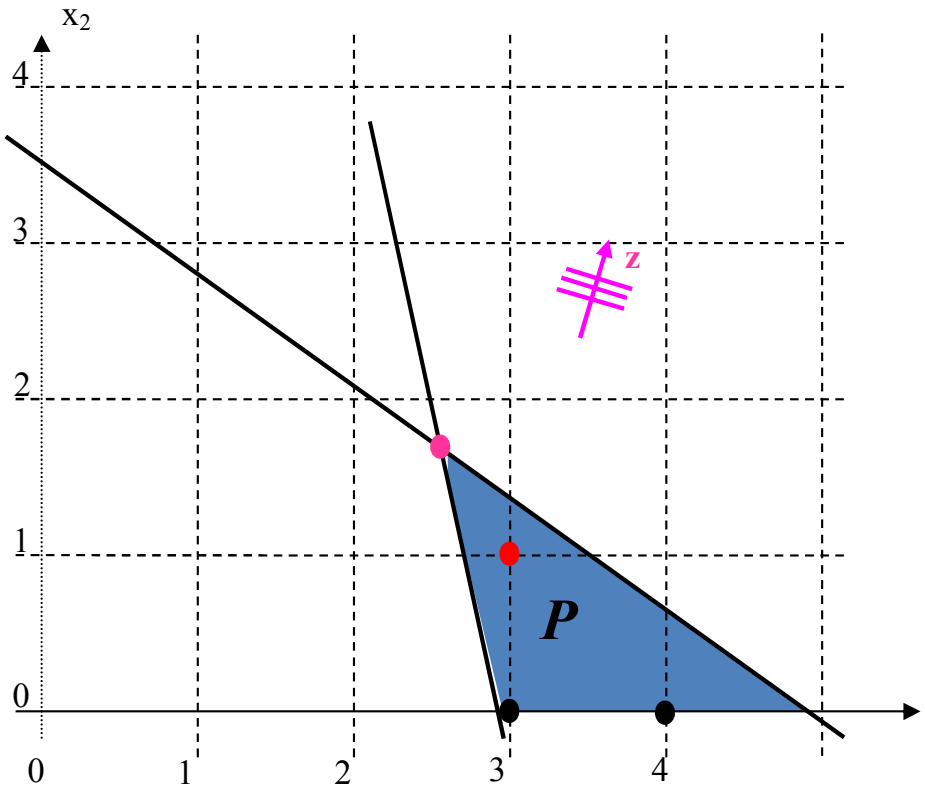


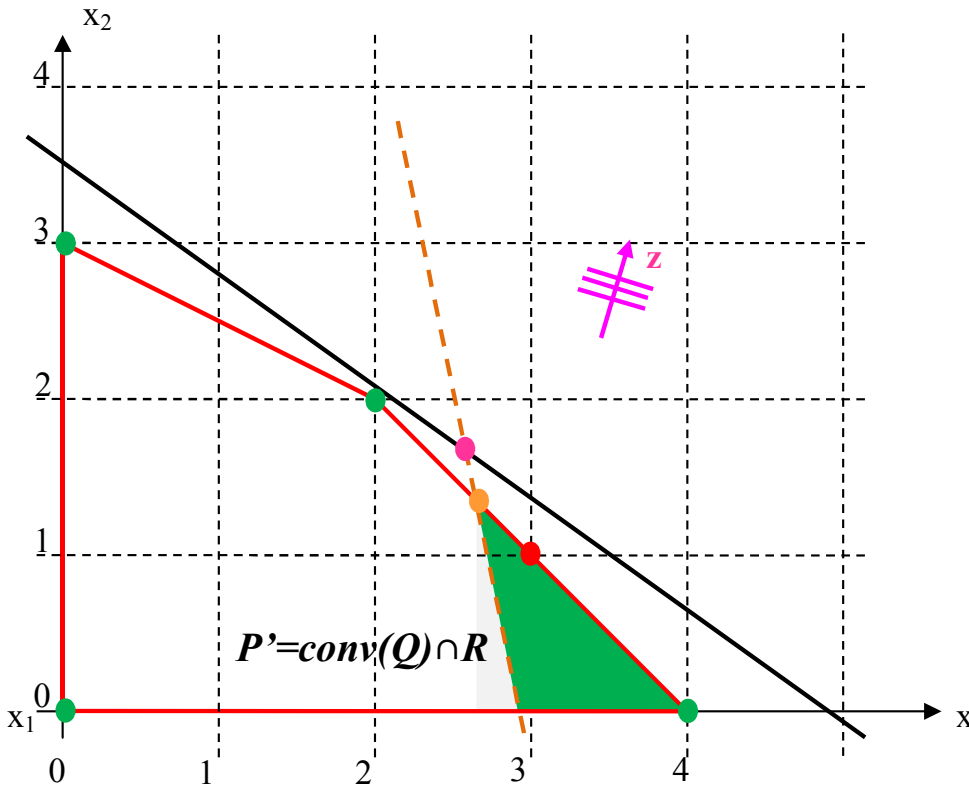
Convexification (3)





Convexification (4)





Extended formulation

Instead of solving

$$z^* = \min\{cx : Dx \geq e, Fx \geq g, x \in \mathcal{Z}^n_+\} = \min\{cx : Dx \geq e, x \in Q\}$$

where $Q = \{x \in \mathcal{Z}^n_+ : Fx \geq g\}$, we solve

$$z^* = \min\{cx : Dx \geq e, x \in conv(Q)\}.$$

To obtain this, we use the extended formulation, based on the substitution

$$x = \sum_{u=1}^{p} \theta_u x^{(u)}$$

where $x^{(u)}$ is the generic extreme point of $conv(Q)$.

Extended formulation

Compact formulation:

$$z_P^* = \min_{x \in \mathcal{Z}_+^n} \{cx : Dx \geq e, x \in Q\} \text{ where } Q = \{x \in \mathcal{Z}_+^n : Fx \geq g\}$$

Extended formulation (master problem):

$$z_{MP}^* = \min_{\theta \in \mathcal{B}^p} \left\{ \sum_{u=1}^{p} cx^{(u)}\theta_u : \sum_{u=1}^{p} Dx^{(u)}\theta_u \geq e, \sum_{u=1}^{p} \theta_u = 1 \right\}.$$

Linear relaxation of the extended formulation:

$$z_{LMP}^* = \min_{\theta \in \Re_+^p} \left\{ \sum_{u=1}^{p} cx^{(u)}\theta_u : \sum_{u=1}^{p} Dx^{(u)}\theta_u \geq e, \sum_{u=1}^{p} \theta_u = 1 \right\}.$$

Extended formulation

The extended formulation of a (sub-)problem

$$z^*_{MP} = \min_{\theta \in \mathcal{B}^p_+} \left\{ \sum_{u=1}^{p} c x^{(u)} \theta_u : \sum_{u=1}^{p} D x^{(u)} \theta_u \geq e, \sum_{u=1}^{p} \theta_u = 1 \right\}$$

is counter-intuitive:

- it uses a combinatorial number of variables θ, instead of a polynomial number of variables x;
- it is still an ILP problem, as hard as the original ILP.

However, its linear relaxation

$$z^*_{LMP} = \min_{\theta \in \Re^p_+} \left\{ \sum_{u=1}^{p} c x^{(u)} \theta_u : \sum_{u=1}^{p} D x^{(u)} \theta_u \geq e, \sum_{u=1}^{p} \theta_u = 1 \right\}$$

can be tighter than the linear relaxation of the original problem, because $\mathcal{Q}$ is replaced by $conv(\mathcal{Q})$.

Column generation

Extended formulation $\Rightarrow$ too many variables $\Rightarrow$ column generation.

Linear Restricted Master Problem (LRMP). Find an optimal convex combination of the available columns:

$$z^*_{LRMP} = \min_{\theta \in \Re^{p'}_+} \left\{ \sum_{u=1}^{p'} c x^{(u)} \theta_u : \sum_{u=1}^{p'} D x^{(u)} \theta_u \geq e, \sum_{u=1}^{p'} \theta_u = 1 \right\}.$$

Pricing sub-problem. Find a column with minimum reduced cost:

$$\overline{c}^* = \min_{x \in \mathcal{Z}^n_+} \{(c - \lambda D)x - \mu : Fx \geq g\}.$$

where

- λ is the vector of dual variables of the complicating constraints;
- μ is the dual variable of the convexity constraint.

Example (IKPDC)

Linear Restricted Master Problem (LRMP):

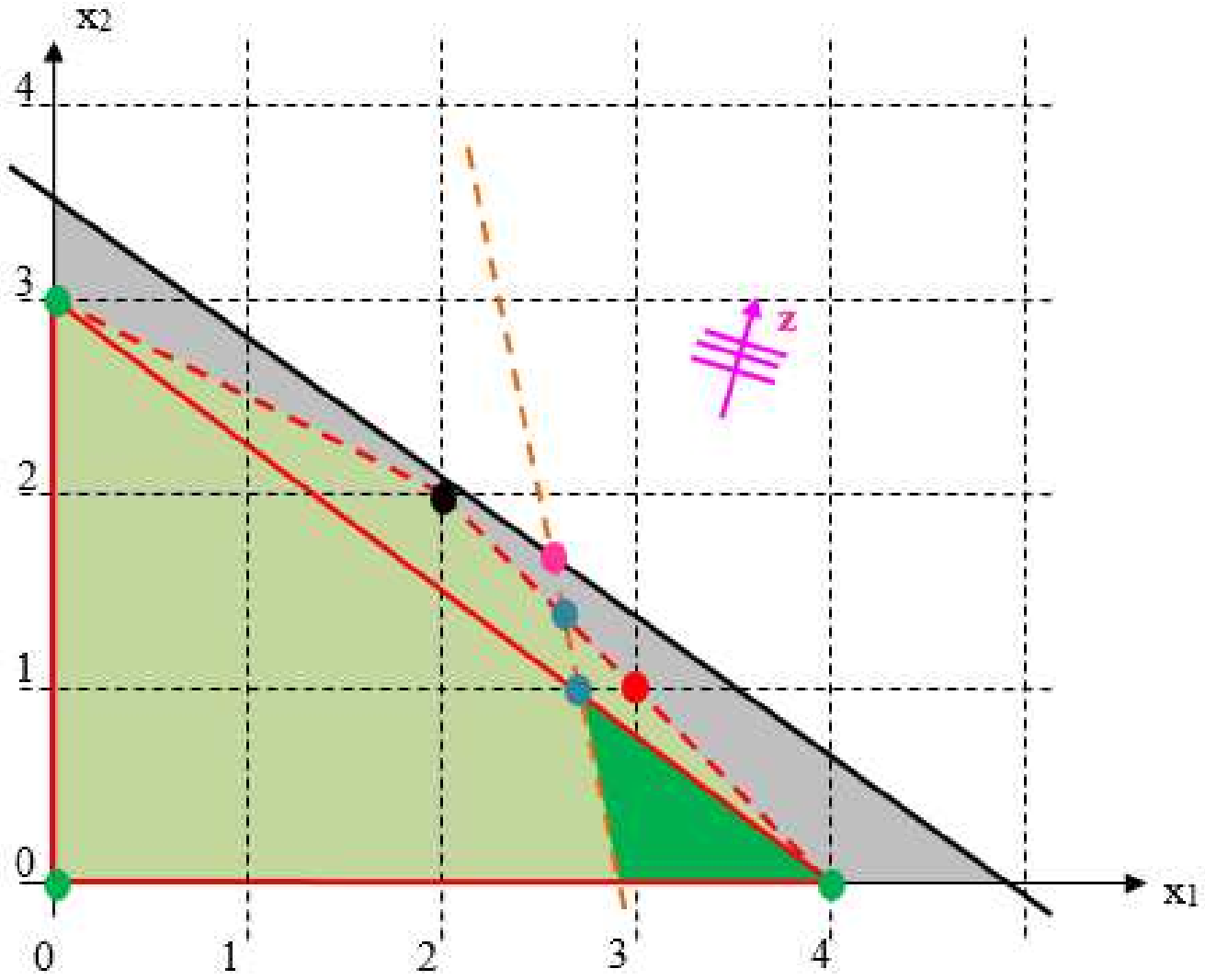
$$z^*_{LRMP} = \max_{\theta \in \Re^{p'}_+} \left\{ \sum_{u=1}^{p'} \sum_{j \in \mathcal{N}} v_j x_j^{(u)} \theta_u : \sum_{u=1}^{p'} \sum_{j \in \mathcal{N}} w''_j x_j^{(u)} \theta_u \geq W'', \sum_{u=1}^{p'} \theta_u = 1 \right\}.$$

Pricing sub-problem:

$$\overline{c}^* = \max_{x \in \mathcal{Z}_+^{|\mathcal{N}|}} \left\{ \sum_{j \in \mathcal{N}} (v_j - \lambda w''_j) x_j - \mu : \sum_{j \in \mathcal{N}} w'_j x_j \leq W' \right\}.$$

where λ is the dual variable of the complicating constraint and μ is the dual variable of the convexity constraint.

CG iterations: geometric interpretation



Decomposition

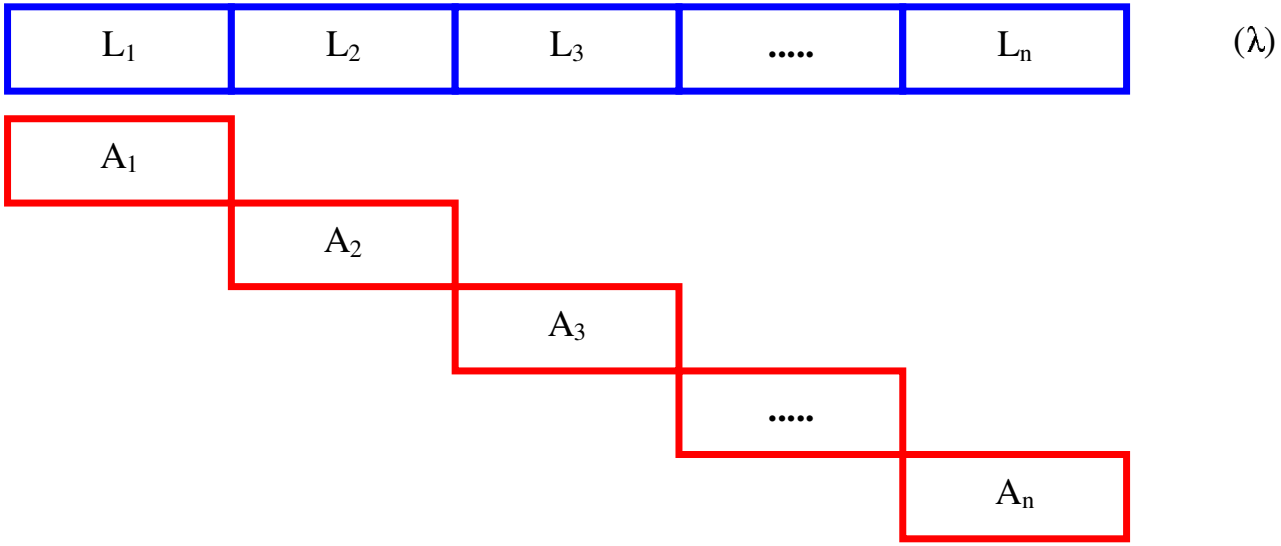
Consider an ILP whose constraint set can be partitioned into two sets of constraints:

$$z^* = \min_{x \in \mathcal{Z}_+^n} \{cx : Dx \geq e, Fx \geq g\}$$

such that one of them has got a block-diagonal structure:

$$Fx \geq g \Leftrightarrow F^{(k)}x^{(k)} \geq g^{(k)} \; \forall k \in K,$$

where $x^{(k)}$ are the variables corresponding to the columns of block $k \in K$.



Decomposition

Now the ILP can be rewritten as:

$$z^* = \min_{x \in \mathcal{Z}_+^n} \left\{ cx : Dx \geq e, F^{(k)} x^{(k)} \geq g^{(k)} \ \forall k \in K \right\}.$$

Constraints $Dx \geq e$ are linking constraints.

Assume we have an algorithm $\mathcal{A}$ able to optimize the sub-problem $\forall k \in K$:

$$z_k^* = \min_{x^{(k)} \in \mathcal{Z}_+^{|K|}} \left\{ cx^{(k)} : F^{(k)} x^{(k)} \geq g^{(k)} \right\}.$$

Then $\mathcal{A}$ allows us to convexify each instance of the sub-problem.

Decomposition

We define the set of the integer solutions

$$X_k = \left\{ x^{(k)} \in \mathcal{Z}_+^{|K|} : F^{(k)} x^{(k)} \geq g^{(k)} \right\} \quad \forall k \in K$$

and we can convexify the sub-problem:

$$\begin{aligned} z_k^* &= \min_{x^{(k)} \in \mathcal{Z}_+^{|K|}} \left\{ c x^{(k)} : F^{(k)} x^{(k)} \geq g^{(k)} \right\} = \\ &= \min_{x^{(k)} \in \mathcal{Z}_+^{|K|}} \left\{ c x^{(k)} : x \in X_k \right\} = \\ &= \min_{x^{(k)} \in \Re_+^{|K|}} \left\{ c x^{(k)} : x \in conv(X_k) \right\}. \end{aligned}$$

Decomposition: the GAP

Example: the Generalized Assignment Problem (GAP).

Given

- a set $\mathcal{J}$ of jobs with weights $w_j \; \forall j \in \mathcal{J}$,
- a set $\mathcal{M}$ of machines with capacities $b_m \; \forall m \in \mathcal{M}$,
- an assignment cost c_{jm} for each pair,

find an assignment of jobs to machines such that

- (assignment constraint:) each job is assigned to a machine,
- (capacity constraint:) the total weight of the jobs assigned to the same machine does not exceed the capacity of the machine,
- (objective:) the total assignment cost is minimized.

Decomposition: the GAP

We use assignment variables $x_{jm} \in \{0,1\}\ \forall (j,m) \in \mathcal{J} \times \mathcal{M}$:

$$\begin{aligned}
\text{minimize } z_P^* = & \sum_{j \in \mathcal{J}} \sum_{m \in \mathcal{M}} c_{jm} x_{jm} & \\
\text{s.t. } & \sum_{m \in \mathcal{M}} x_{jm} = 1 & \forall j \in \mathcal{J} \\
& \sum_{j \in \mathcal{J}} w_j x_{jm} \leq b_m & \forall m \in \mathcal{M} \\
& x \in \mathcal{B}^{|\mathcal{J}| \times |\mathcal{M}|} &
\end{aligned}$$

We identify $\sum_{m \in \mathcal{M}} x_{jm} = 1 \ \ \forall j \in \mathcal{J}$ as linking constraints and we convexify the Binary Knapsack sub-problem for each machine

$$z_m^* = \min_{x \in \mathcal{B}^{|\mathcal{J}|}} \left\{ \sum_{j \in \mathcal{J}} c_{jm} x_{jm} : \sum_{j \in \mathcal{J}} w_j x_{jm} \leq b_m \right\}.$$

Example (GAP)

Let $x^{(um)}$ the generic extreme point of $conv(X_m)$,
where $X_m = \left\{ x \in \mathcal{B}^{|\mathcal{J}|} : \sum_{j \in \mathcal{J}} w_j x_{jm} \leq b_m \right\}$
and let p_m be the number of such extreme points.

The extended formulation of the GAP is:

$$z^*_{MP} = \min_{\theta^{(m)} \mathcal{B}^{p_m} \ \forall m \in \mathcal{M}} \{ \sum_{m \in \mathcal{M}} \sum_{u=1}^{p_m} c^{(um)} \theta_u^{(m)} : \sum_{m \in \mathcal{M}} \sum_{u=1}^{p_m} x_j^{(um)} \theta_u^{(m)} = 1 \ \forall j \in \mathcal{J},$$

$$\sum_{u=1}^{p_m} \theta_u^{(m)} = 1 \ \ \forall m \in \mathcal{M} \}.$$

where

- $x_j^{(um)}$ indicates whether job j is assigned to machine m in the extreme point u of machine m.
- $c^{(um)}$ is the cost of the extreme point u of machine m.

Example (GAP)

Let λ_j be the dual variable of each assignment constraint $\sum_{m\in\mathcal{M}}\sum_{u=1}^{p_m} x_j^{(um)}\theta_u^{(m)} = 1\ \forall j \in \mathcal{J}$.

Let μ_m be the dual variable of each convexity constraint $\sum_{u=1}^{p_m}\theta_u^{(m)} = 1\ \ \forall m \in \mathcal{M}$.

Then the pricing sub-problem for each machine $m \in \mathcal{M}$ is:

$$z_m^* = \min_{x\in\mathcal{B}^{|\mathcal{J}|}} \left\{ \sum_{j\in\mathcal{J}} (c_{jm} - \lambda_j)x_{jm} - \mu : \sum_{j\in\mathcal{J}} w_j x_{jm} \leq b_m \right\}.$$

The master problem

Primal feasibility of the RLMP must be guaranteed: dummy columns with very high cost ensuring feasibility.

Ensuring primal boundedness is not needed: if the RLMP is unbounded, the whole LMP also is.

As negative reduced cost columns are inserted into the RLMP, some non-basic columns with “large” positive reduced cost can be deleted from it.

The master problem can be further strenghened by cutting planes: branch-and-cut-and-price.

The pricing problem

The pricing sub-problem is the optimization problem we are able and willing to convexify: we need an exact optimization algorithm for it.

Keep a “pool” of known columns, where negative reduced cost columns can be found before pricing new ones.

Generate several columns with negative reduced cost for every pricing iteration (multiple pricing).

Generate columns with negative reduced cost in a heuristic way, as far as possible. Only when heuristic pricing fails, resort to exact pricing.

Bounding in branch-and-price

When

- the LRMP has been solved to optimality,
- the pricing algorithm states that no columns with negative reduced cost can be produced for any sub-problem,

we have the optimality guarantee of the solution of the LMP.

Its value z^*_{LMP} is a valid lower bound, but it is achieved only at the end of column generation.

However, one can use the dual values associated with the constraints of the master problem as Lagrangean multipliers to compute a valid lower bound at any point during column generation.

Linear programming can be seen as an alternative to the sub-gradient algorithm to provide dual values in a Lagrangean relaxation algorithm.

Bounding in branch-and-price

The value of z^*_{RLMP} (column generation) is monotone and $z^*_{RLMP} \geq z^*_{LMP}$.

The value of z^*_{LR} (Lagrangean relaxation) is not monotone in general and $z^*_{LR} \leq z^*_{LMP}$.

The two values tend to coincide as far as CG goes on, but there may be a "tailing-off" effect: many CG iterations are needed for very small improvements of the two bounds.

It is common practice to stop CG when the gap between the two bounds is "small".

It is also possible to use stabilization techniques, when the master problem is degenerate.

Branching in branch-and-price

The optimal solution of the LMP can be fractional.

To achieve integrality, we must branch.

Cutting planes algorithms (row generation) allow producing an integer optimal solution without branching.

It is recommended not to branch on the binary θ variables.
It is more recommendable to branch on the original variables x.

Ad hoc branching strategies can be devised for each particular problem. If possible, they should not change the structure of the pricing sub-problem (robust branching).

Heuristic branch-and-price

The columns in the RLMP can also be used as building blocks for a *math-heuristic*.

Master-problem-based heuristic n.1 (one shot):

- generate columns for some time;
- impose binary restrictions on the θ variables and solve the resulting discrete RMP.

Master-problem-based heuristic n.2 (iterative):

- generate columns for some time;
- solve the LRMP, obtaining a fractional optimal solution;
- round up to 1 the θ variable with the largest value;
- update the right-hand-sides of the constraints and repeat until an integer and feasible solution is obtained.

Both are easy to implement with ILP solvers.