Column generation

Operational Research Complements

Giovanni Righini
Università degli Studi di Milano

Discrete optimization

Discrete optimization problems are in general *NP*-hard $\Rightarrow$ implicit enumeration.

Two main implicit enumeration techinques:

- branch-and-bound,
- dynamic programming.

In branch-and-bound algorithms we must associate a dual bound with every sub-problem.

The linear continuous relaxation (effectively) provides a valid dual bound.

Linear relaxation

Consider a discrete optimization problem with a linear objective function and linear constraints (ILP - integer linear programming):

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

Its linear relaxation is

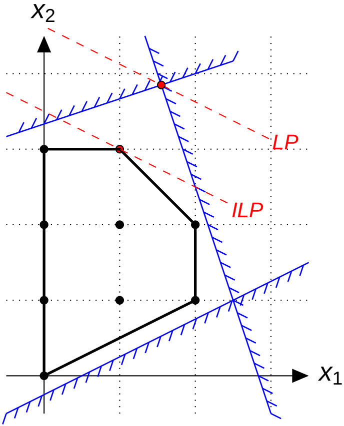
$$z_{LP}^* = \min\{cx : Ax \geq b, x \in \Re_+^n\}.$$

If the polyhedron $\{x \in \Re_+^n : Ax \geq b\}$ has integer extreme points, then the problem has the integrality property (IP) and it can be solved as an LP problem: the optimal solution of its linear relaxation is guaranteed to be integer.

Otherwise, the optimal solution of its linear relaxation is not guaranteed to be integer.

Convex hull

The ideal formulation of the constraint set of an ILP problem is the convex hull $conv(X)$ of the feasible integer solutions X.



Convexification

Finding the optimal solution of an ILP

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

is equivalent to solving the LP

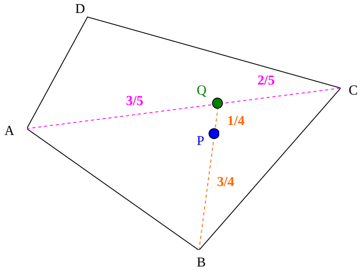
$$z^* = \min\{cx : x \in \Re^n_+, x \in conv(X)\}.$$

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.

Minkowsky and Weil theorem

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

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



Vice versa: no point out of $\mathcal{P}$ can be obtained by any convex combination of the extreme points.

Extended formulation

Consider an ILP problem.

Let X be the set of its feasible integer solutions.

Let $\mathcal{P}$ be the convex hull of X.

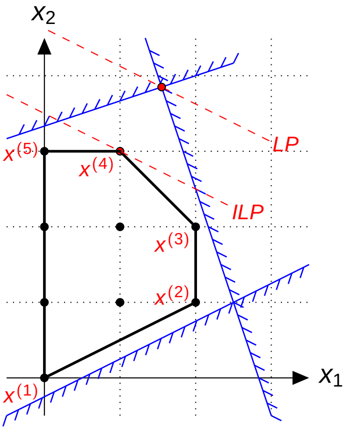
Let $x^{(1)}, x^{(2)}, \ldots, x^{(p)}$ be the extreme points of $\mathcal{P}$.

For the Minkowsky and Weil theorem,

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

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

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



Partial convexification

Given an ILP without the integrality property, it often happens that

- we do not know how to convexify the whole ILP: we do not know $conv(X)$;
- we know how to convexify some sub-problem: we know how to generate integer optimal solutions when some constraints are relaxed.

Smaller polyhedra $\Rightarrow$ tighter dual bounds from LP $\Rightarrow$ faster branch-and-bound algorithms.

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.

Extended formulation of a relaxation

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 $conv(Q)$.

Comparing formulations

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:

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

Linear relaxation of the extended formulation:

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

The linear relaxation of the extended formulation is tighter than that of the compact formulation.

Decomposition

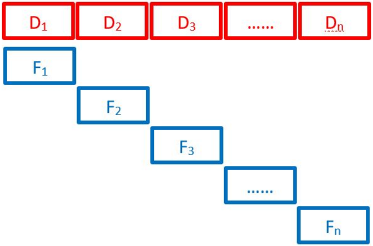
Consider an ILP whose constraint set can be partitioned into two sets of constraints:

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

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.

Example: the Generalized Assignment Problem (GAP)

Data:

- a set J of jobs
- a set M of machines
- an assignment cost $c : (J \times M) \mapsto \Re_+$,
- a resource consumption $a : (J \times M) \mapsto \Re_+$,
- a capacity b_m for each machine $m \in M$,

find a minimum cost assignment of the jobs to the machines such that

- (assignment constraints) all jobs are assigned,
- (capacity constraints) the total resource consumption for each machine does not exceed its capacity.

GAP: compact formulation

$$
\begin{aligned}
\text{minimize } z = & \sum_{j \in J} \sum_{m \in M} c_{jm} x_{jm} & \\
\text{s.t. } & \sum_{m \in M} x_{jm} = 1 & \forall j \in J \\
& \sum_{j \in J} a_{jm} x_{jm} \leq b_m & \forall m \in M \\
& x_{jm} \in \{0, 1\} & \forall i \in J, m \in M.
\end{aligned}
$$

With this compact formulation

- the ILP model has a polynomial number of variables x;
- the LP relaxation provides a weak lower bound.

Decomposition: the GAP

$$\begin{aligned}
\text{minimize } z = & \sum_{j \in J} \sum_{m \in M} c_{jm} x_{jm} & \\
\text{s.t. } & \sum_{m \in M} x_{jm} = 1 & \forall j \in J \\
& \sum_{j \in J} a_{jm} x_{jm} \leq b_m & \forall m \in M \\
& x_{jm} \in \{0, 1\} & \forall i \in J, m \in M.
\end{aligned}$$

Constraints $\sum_{m \in M} x_{jm} = 1 \ \ \forall j \in J$ are linking constraints.

When removed, we are left with a Binary Knapsack sub-problem for each machine.

The Binary KP is *NP*-hard, but we know how to solve it effectively $\Rightarrow$ convexification!.

GAP: extended formulation

Let P_m be the set of all feasible integer duties of machine $m \in M$, i.e. assignments of jobs to machine m.

Each feasible duty $p \in P_m$ is defined by n binary assignments y_{jp} complygin with the capacity constraints:

$$\sum_{j \in J} a_{jm} y_{jp} \leq b_m \quad \forall p \in P_m.$$

Its cost is

$$c_p = \sum_{j \in J} c_{jm} y_{jp} \quad \forall p \in P_m.$$

A solution of the GAP is defined by selecting a duty for each machine (assignment constraints).

We use a binary selection variable θ_p for each duty $p \in P_m$.

$$\sum_{p \in P_m} \theta_p = 1 \quad \forall m \in M.$$

GAP: extended formulation

$$
\begin{aligned}
\text{minimize } z' = \sum_{m \in M} \sum_{p \in P_m} c_p \theta_p & \\
\text{s.t. } \sum_{m \in M} \sum_{p \in P_m} y_{jp} \theta_p = 1 & \qquad \forall j \in J \\
\sum_{p \in P_m} \theta_p = 1 & \qquad \forall m \in M \\
\theta_p \in \{0, 1\} & \qquad \forall m \in M, p \in P_m.
\end{aligned}
$$

With this extended formulation

- the ILP model has an exponential number of variables θ;
- the LP relaxation provides a stronger lower bound.

GAP: equivalence of the two formulations

The variables in the two formulations are linked as follows:

$$x_{jm} = \sum_{p \in P_m} y_{jp}\theta_p \quad \forall j \in J, m \in M$$

with the convexity constraints

$$\sum_{p \in P_m} \theta_p = 1, \theta_p \in \{0, 1\}.$$

Assignment constraints

$$\sum_{m \in M} x_{jm} = 1 \ \ \forall j \in J$$

are equivalent to

$$\sum_{m \in M} \sum_{p \in P_m} y_{jp}\theta_p = 1 \ \ \forall j \in J.$$

GAP: equivalence of the two formulations

The objective function

$$z = \sum_{j \in J} \sum_{m \in M} c_{jm} x_{jm}$$

is equivalent to

$$z = \sum_{j \in J} \sum_{m \in M} \sum_{p \in P_m} c_{jm} y_{jp} \theta_p .$$

Defining

$$c_p = \sum_{j \in J} c_{jm} y_{jp} \quad \forall m \in M, p \in P_m ,$$

we have

$$z = \sum_{m \in M} \sum_{p \in P_m} c_p \theta_p .$$

GAP: equivalence of the two formulations

Capacity constraints

$$\sum_{j \in J} a_{jm} x_{jm} \leq b_m \quad \forall m \in M$$

are equivalent to

$$\sum_{p \in P_m} \theta_p \left(\sum_{j \in J} a_{jm} y_{jp} \right) \leq b_m \quad \forall m \in M.$$

In the extended formulation only feasible integer duties are considered, so that

$$\sum_{j \in J} a_{jm} y_{jp} \leq b_m \quad \forall m \in M, \forall p \in P_m.$$

This guarantees that their convex combination with coefficients θ_p is also feasible.

Example: GAP

The linear relaxations of the two formulations of the GAP are

$$\begin{aligned}
\text{minimize } z = \sum_{j \in J} \sum_{m \in M} c_{jm} x_{jm} & \\
\text{s.t.} \sum_{m \in M} x_{jm} = 1 & \quad \forall j \in J \\
\sum_{j \in J} a_{jm} x_{jm} \leq b_m & \quad \forall m \in M \\
0 \leq x_{jm} \leq 1 & \quad \forall i \in J, m \in M.
\end{aligned}$$

$$\begin{aligned}
\text{minimize } z = \sum_{m \in M} \sum_{p \in P_m} c_p \theta_p & \\
\text{s.t.} \sum_{m \in M} \sum_{p \in P_m} y_{jp} \theta_p = 1 & \quad \forall j \in J \\
\sum_{p \in P_m} \theta_p = 1 & \quad \forall m \in M \\
0 \leq \theta_p \leq 1 & \quad \forall m \in M, p \in P_m.
\end{aligned}$$

Linear relaxations

In the linear relaxation of the compact formulation, the capacity constraints

$$\sum_{j \in J} a_{jm} x_{jm} \leq b_m \quad \forall m \in M$$

can be satisfied by fractional values of the variables x.

In the linear relaxation of the extended formulation, the capacity constraints

$$\sum_{j \in J} a_{jm} y_{jp} \leq b_m$$

is satisfied by integer values of the variables y_{jp}.

Extended formulation: pros and cons

The extended formulation of a (sub-)problem 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 can be tighter than the linear relaxation of the original problem, because some constraints have been convexified.

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 (Restricted Master Problem), 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 an optimal primal solution x^* is found, an optimal dual solution λ^* is also found.

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

From LP we know that the reduced cost of a column is

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

Hence, the answer depends on coefficients a and c.

- If “No”: x^* is optimal for the original LP.
- If “Yes”: insert column j into $\overline{N}$ and reoptimize.

The column generation algorithm proceeds by alternating two steps:

- solution of the LP with the column subset $\overline{N}$ (restricted master problem);
- search for a minimum reduced cost column (without explicitly enumerating all of them): discrete optimization sub-problem (pricing sub-problem), where the coefficients a_i are decision variables.

GAP: pricing

$$\text{minimize } z = \sum_{m \in M} \sum_{p \in P_m} c_p \theta_p$$

$$\text{s.t. } \sum_{m \in M} \sum_{p \in P_m} y_{jp} \theta_p = 1 \qquad \forall j \in J \qquad [\lambda_j]$$

$$\sum_{p \in P_m} \theta_p = 1 \qquad \forall m \in M \qquad [\mu_m]$$

$$0 \leq \theta_p \leq 1 \qquad \forall m \in M, p \in P_m.$$

Pricing sub-problem: $\forall m \in M$ find a min reduced cost duty $p \in P_m$:

$$\text{minimize } \overline{c}_p = c_p - \sum_{j \in J} \lambda_j y_{jp} - \mu_m$$

$$\text{s.t. } \sum_{j \in J} a_{jm} y_{jp} \leq b_m$$

$$y_{jp} \in \{0, 1\} \qquad \forall j \in J.$$

It is a Binary Knapsack Problem.

Solving the master

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

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

At every restart, the initial basic solution is feasible: primal simplex or interior point methods.

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

The master problem tends to become highly degenerate: row aggregation, stabilization techniques.

Pricing

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 long as possible. Only when heuristic pricing fails, resort to exact pricing.

Termination

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: LR vs. CG

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 the same dual bound, but there may be a "tailing-off" effect: many CG iterations are needed for very small improvement of the bound.

It is common practice to stop CG when the gap between the upper and the lower bounds is "small enough".

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 recommended 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.