B6.3 Integer Programming

Raphael Hauser

Mathematical Institute
University Of Oxford

Michaelmas Term 2018

Recap on Delayed Column Generation

Starting with an IP of the form

$$
\begin{aligned}
\text{(IP)} \quad z = \max_{x} & \sum_{k=1}^{K} c^{k\mathsf{T}} x^k \\
\text{s.t.} & \sum_{k=1}^{K} A^k x^k = b \\
& x^k \in X^k = \{x^k \in \mathbb{Z}_+^{n_k} : D^k x^k \leq d^k\}, \quad (k = 1, \ldots, K),
\end{aligned}
$$

the Dantzig-Wolfe reformulation yields the *IP Master Problem*

$$
\begin{aligned}
\text{(IPM)} \quad z = \max_{\lambda} & \sum_{k=1}^{K} \sum_{t=1}^{T_k} (c^{k\mathsf{T}} x^{k,t}) \lambda_{k,t} \\
\text{s.t.} & \sum_{k=1}^{K} \sum_{t=1}^{T_k} (A^k x^{k,t}) \lambda_{k,t} = b, \\
& \sum_{t=1}^{T_k} \lambda_{k,t} = 1, \quad (k = 1, \ldots, K), \\
& \lambda_{k,t} \in \{0, 1\}, \quad (t = 1, \ldots, T_k; k = 1, \ldots, K).
\end{aligned}
$$

The LP relaxation of the IP Master Problem is the *LP Master Problem*,

$$
\begin{aligned}
\text{(LPM)} \quad z^{LPM} = \max_{\lambda} & \sum_{k=1}^{K} \sum_{t=1}^{T_k} (c^{k\mathsf{T}} x^{k,t}) \lambda_{k,t} \\
\text{s.t.} & \sum_{k=1}^{K} \sum_{t=1}^{T_k} (A^k x^{k,t}) \lambda_{k,t} = b, \\
& \sum_{t=1}^{T_k} \lambda_{k,t} = 1, \quad (k = 1, \ldots, K), \\
& \lambda_{k,t} \geq 0, \quad (t = 1, \ldots, T_k; k = 1, \ldots, K).
\end{aligned}
$$

The dual of the LP Master Problem is the *Dual Master Problem*

$$
\begin{aligned}
(DM) \quad z^{DM} = \min_{\mu, \pi} & \sum_{i=1}^{m} b_i \pi_i + \sum_{k=1}^{K} \mu_k \\
\text{s.t.} & \ \pi^{\mathsf{T}} A^k x^k + \mu_k \geq c^{k\mathsf{T}} x^k, \quad (x^k \in X^k).
\end{aligned}
$$

Take a *Restricted LP Master Problem*

$$\begin{aligned} \text{(RM)} \quad \tilde{z}^{RM} = \max\ & \tilde{c}^{\mathsf{T}} \tilde{\lambda} \\ \text{s.t.}\ & \tilde{A}\tilde{\lambda} = \tilde{b}, \\ & \tilde{\lambda} \geq 0, \end{aligned}$$

obtained by setting all but a few $\lambda_{k,t} = 0$ and forcing them to be non-basic variables. Only the remaining columns of (LPM) need to be generated.

Using the simplex algorithm, find an optimal solution $\tilde{\lambda}^*$ of (RM) and, by complementary slackness, the corresponding optimal solution $(\pi, \mu) \in \mathbb{R}^m \times \mathbb{R}^K$ of the dual of (RM).

In parallel, for all $k \in [1, K]$ solve

$$\begin{aligned} (\text{CGIP}_k) \quad \tilde{x}^k = \arg\max_x\ & (c^k - \pi^{\mathsf{T}} A^k)x - \mu_k \\ \text{s.t.}\ & x \in X^k. \end{aligned}$$

- If the *reduced prices* $\zeta_k := (c^k - \pi^T A^k)\tilde{x}^k - \mu_k \leq 0$ for all k, then (π, μ) is (DM)-feasible and $\tilde{\lambda}^*$ is (LPM)-optimal.
- Otherwise, pick k such that $\zeta_k > 0$ and add the column

$$\begin{bmatrix} c^k \tilde{x}^k \\ A^k \tilde{x}^k \\ e_k \end{bmatrix}$$

to (RLPM) to obtain a new restricted LP master problem (RLPM_+). Re-optimise using warmstarting.

- At all every iteration, we monitor our progress toward solving (LPM) by storing the primal and dual bounds

$$\tilde{c}^T \tilde{\lambda}^* \leq z^{LPM} \leq \pi b + \sum_{k=1}^{K} \mu_k + \sum_{k=1}^{K} \zeta_k.$$

The Branch & Price Algorithm

The application of column generation in the context of branch-and-bound for *binary* IPs is called *branch-and-price*.

- The master problem is in the format

$$
\begin{aligned}
z = \max \quad & \sum_{k=1}^{K} \sum_{t=1}^{T_k} (c^{k\mathsf{T}} x^{k,t}) \lambda_{k,t} \\
\text{s.t.} \quad & \sum_{k=1}^{K} \sum_{t=1}^{T_k} (A^k x^{k,t}) \lambda_{k,t} = b, \\
& \sum_{t=1}^{T_k} \lambda_{k,t} = 1, \quad (k = 1, \ldots, K), \\
& \lambda_{k,t} \in \{0,1\}, \quad (t = 1, \ldots, T_k;\ k = 1, \ldots, K),
\end{aligned}
$$

  and the vectors $x^{k,t}$ are binary.
- Subproblems will also be in this format. To guarantee this, we need to discuss a tailored branching method.
- We already know how delayed column generation is able to generate primal and dual bounds at each node (subproblem primal and dual bounds that can be used to update the global primal and dual bound of the root problem in the usual branch & bound fashion).

Since the points $x^{k,t} \in X^k$ are all distinct 0-1 vectors,

$$\tilde{x}^k = \sum_{t=1}^{T_k} \tilde{\lambda}_{k,t} x^{k,t}$$

is a 0-1 vector if and only if $\tilde{\lambda}$ is integer valued.

If the optimal solution $\tilde{\lambda}$ of (LPM) (found by delayed column generation) is not integer, there exists therefore κ, j such that $\tilde{x}_j^\kappa$, the j-th component of $\tilde{x}^\kappa$, is fractional.

We would like to branch by splitting the feasible set $S = S_0 \cup S_1$ into

$$S_0 = S \cap \{x : x_j^\kappa = 0\},$$
$$S_1 = S \cap \{x : x_j^\kappa = 1\}.$$

But can we identify the master problems corresponding to S_0 and S_1?

For $\delta = 0, 1$, the requirement that

$$\delta = x_j^\kappa = \sum_{t=1}^{T_\kappa} \lambda_{\kappa,t} x_j^{\kappa,t}$$

implies that $x_j^{\kappa,t} = \delta$ for all t with $\lambda_{\kappa,t} > 0$. Therefore, the master problem for S_δ is

$$z(S_\delta) = \max \sum_{k \neq \kappa} \sum_{t=1}^{T_k} (c^k x^{k,t}) \lambda_{k,t} + \sum_{t:\, x_j^{\kappa,t} = \delta} (c^\kappa x^{\kappa,t}) \lambda_{\kappa,t}$$

$$\text{s.t.} \sum_{k \neq \kappa} \sum_{t=1}^{T_k} (A^k x^{k,t}) \lambda_{k,t} + \sum_{t:\, x_j^{\kappa,t} = \delta} (A^\kappa x^{\kappa,t}) \lambda_{\kappa,t} = b,$$

$$\sum_{t=1}^{T_k} \lambda_{k,t} = 1, \quad (k \neq \kappa),$$

$$\sum_{t:\, x_j^{\kappa,t} = \delta} \lambda_{\kappa,t} = 1$$

$$\lambda_{k,t} \in \{0, 1\}, \quad (t = 1, \ldots, T_k;\ k = 1, \ldots, K).$$

Thus, the problem has the same structure as the master problem for S, but some of the columns are permanently excluded. This has the beneficial effect that the deeper the node in the branch-and-bound tree, the fewer patterns $x^{k,t}$ need to be considered.

The column generation subproblems are unchanged for $k \neq \kappa$,

$$\tilde{x}^{k} = \arg\max (c^{k} - \pi A^{k})x - \mu_{k}$$
$$\text{s.t. } x \in X^{k},$$

but take the new form

$$\tilde{x}^{\kappa}(S_{\delta}) = \arg\max (c^{\kappa} - \pi A^{\kappa})x - \mu_{\kappa}$$
$$\text{s.t. } x \in X^{k},$$
$$x_{j} = \delta$$

for $k = \kappa$.

Similar further restrictions apply of course deeper down the branches, where the subproblems are further branched.

The Cutting Stock Problem (CSP)

The above ideas can be applied to solve the cutting-stock problem in an approach developed by Gilmore & Gomoroy.

Example (Cutting Stock Problem)

A factory has an unlimited stock of 20-inch paper rolls that it can cut into rolls of smaller widths.

They receive an order of 301 9-inch paper rolls, 401 8-inch paper rolls, 201 7-inch paper rolls and 501 6-inch paper rolls.

Assuming both trim loss and overproduction are waste, how to fill all the orders under minimal cost?

Details of Branch-and-Price for the Cutting Stock Problem:

- More generally, if the stock rolls have width W and there are m different widths w_i $(i = 1, \ldots, m)$ in the order, we can generate all patterns $a_j = [\, a_{1j} \cdots a_{mj} \,]^T$ of patterns consisting of a_{ij} rolls of width w_i that can be cut into a roll of width W, i.e., such that

$$\sum_{i=1}^{m} w_i a_{ij} \leq W.$$

- Note that $a_{ij} \in \mathbb{Z}_+ = \{0, 1, 2, 3, \ldots\}$ for all i, j.
- Conceptually, we assemble the columns a_j into a matrix A (although we never want to generate the full data).
- Decision variables: $x_j \in \mathbb{N}_0$, the number of times pattern j is used $(j = 1, \ldots, n)$. (Note: the x_j play the roles of variables $\lambda_{k,t}$ used earlier.)
- Constraints: if there are b_i orders of width w_i, filling the orders requires $Ax \geq b$, and due to the assumption that overproduction is waste, w.l.o.g.,

$$Ax = b.$$

- Objective: minimise $\sum_{j=1}^{n} x_j$, the total number of stock rolls used.

- This yields the IP model

$$\begin{aligned} \text{(ICS)} \quad \min & \sum_{j=1}^{n} x_j \\ \text{s.t.} \quad & Ax = b, \\ & x \in \mathbb{Z}_+^n. \end{aligned}$$

- The LP relaxation is given by

$$\begin{aligned} \text{(LCS)} \quad \min & \vec{1}^T x \\ \text{s.t.} \quad & Ax = b, \\ & x \geq 0, \end{aligned}$$

  which has the dual

$$\begin{aligned} \text{(DCS)} \quad \max & b^T y \\ \text{s.t.} \quad & A^T y \leq \vec{1}. \end{aligned}$$

- The number n of patterns can be huge, so apply delayed column generation to solve (LCS), e.g., starting with initial restricted pattern set

$$\tilde{A} = \begin{bmatrix} \lfloor W/w_1 \rfloor & & 0 \\ & \ddots & \\ 0 & & \lfloor W/w_m \rfloor \end{bmatrix}$$

- Here we select only m patterns in each simplex iteration, i.e., the columns corresponding to the basic variables. All other variables (the non-basic variables) are forced to zero.
- To solve the restricted subproblem, we only need to solve a linear system,

$$\tilde{x} = \tilde{A}^{-1} b.$$

- Theorem (Complementary Slackness) implies that the optimal dual variables of the restricted LPM are given by $\tilde{y} = \tilde{A}^{-T}\vec{1}$.
- If $\tilde{y}$ is dual feasible (feasible for (DCS)), $\tilde{x}$ is optimal for (LCS).
- Else there exists a pattern j (an column of the full matrix A that has not yet been generated) corresponding to a non-basic variable x_j such that

$$\sum_{i=1}^{m} a_{ij}\tilde{y}_i > 1.$$

- Although pattern j has not yet been generated, we can find out whether or not it exists by solving the knapsack problem

$$\begin{aligned} \text{(KS)} \quad p^* = \arg\max_{p} \ & \tilde{y}^{\mathsf{T}} p \\ \text{s.t.} \ & \sum_{i=1}^{m} w_i p_i \leq W, \\ & p \in \mathbb{Z}_+^m. \end{aligned}$$

- If $\tilde{y}^{\mathsf{T}} p \leq 1$, then $\tilde{y}$ is dual feasible, and we are in the first case. Else take $(a_{ij})_{i=1}^{m} := (p_i^*)_{i=1}^{m}$ as the entering pattern.
- The exiting variable is determined in the usual simplex fashion, and the dictionary/tableau is pivoted in the usual fashion.

Example (CSP continued)

We follow through by applying delayed column generation to solve the LP relaxation of the ICS of our example. Since $\tilde{A}, \tilde{x}, \tilde{y}$ change in each iteration, we write $A^{(k)}, x^{(k)}, y^{(k)}$ for the corresponding data in iteration k, and we write $z^{(k)} = \sum_j x_j^{(k)}$ for the objective value.

- Using the initialisation discussed above, we have

$$A^{(0)} = \begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 3 \end{bmatrix},$$

$$x^{(0)} = (A^{(0)})^{-1} \begin{bmatrix} 301 \\ 401 \\ 201 \\ 501 \end{bmatrix} = \begin{bmatrix} 150.5 \\ 200.5 \\ 100.5 \\ 167 \end{bmatrix},$$

$$z^{(0)} = 618.5, \quad \text{(objective value)}$$

$$y^{(0)} = \left((A^{0})^{-1}\right)^{\mathsf{T}} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1/2 \\ 1/2 \\ 1/2 \\ 1/3 \end{bmatrix}.$$

Example (CSP continued)

- To find the entering variable x_j, we need to identify whether or not a corresponding column exists and generate it, i.e., solve the knapsack problem

$$\begin{aligned} \text{(KP)} \quad p^* = \arg\max_{\boldsymbol{p}} \; & \frac{1}{2}p_1 + \frac{1}{2}p_2 + \frac{1}{2}p_3 + \frac{1}{3}p_4 \\ \text{s.t.} \; & 9p_1 + 8p_2 + 7p_3 + 6p_4 \leq 20, \\ & p_i \in \mathbb{Z}_+, \quad (i = 1, \ldots, 4). \end{aligned}$$

- Solving (KP) with branch-and-bound, we find $p^* = [0, 0, 2, 1]^{\mathbf{T}}$. Since $(y^{(\mathbf{0})})^{\mathbf{T}} p^* = 4/3 > 1$, p^* will enter the basic cutting patterns as a column of $A^{(\mathbf{1})}$.
- To identify the pattern that corresponds to the leaving basic variable, we must calculate

$$x^{(\mathbf{0})}./(A^{(\mathbf{0})})^{-\mathbf{1}} p^* = [\infty, \infty, 140, 500]^{\mathbf{T}}.$$

It is x_3 that imposes the most restrictive bound, thus the third column of $A^{(\mathbf{0})}$ leaves the basis.

Example (CSP continued)

- Next iteration:

$$A^{(1)} = \begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 1 & 3 \end{bmatrix},$$

$$x^{(1)} = (A^{(1)})^{-1} \begin{bmatrix} 301 \\ 401 \\ 201 \\ 501 \end{bmatrix} = \begin{bmatrix} 150.5 \\ 200.5 \\ 100.5 \\ 133.5 \end{bmatrix},$$

$$z^{(1)} = 585.$$

- Solving another knapsack problem identifies $p^* = [0, 1, 0, 2]^T$ as entering pattern, and column 4 of $A^{(1)}$ as leaving pattern.
- Next iteration:

$$A^{(2)} = \begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & 2 & 0 & 1 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 1 & 2 \end{bmatrix},$$

$$x^{(2)} = \begin{bmatrix} 150.5 \\ 100.375 \\ 100.5 \\ 200.25 \end{bmatrix},$$

$$z^{(2)} = 551.625.$$

Example (CSP continued)

- This time the optimal solution of the knapsack problem yields $(y^{(2)})^{\mathsf{T}} p^* \leq 1$, showing that $y^{(2)}$ is dual optimal, and hence, $x^{(2)}$ is primal optimal.
- The dual bound $z^{(2)} = 551.625$ could now be used in a branch-and bound algorithm to solve the (ICS).
- Alternatively, note that rounding down the usage of each pattern leaves a shortfall of 1 roll of each of the ordered widths, which can be covered by two additional stock rolls cut into patterns $[1, 1, 0, 0]$ and $[0, 0, 1, 1]$. This shows that

$$551.625 \leq z_{IP} \leq 550 + 2,$$

  and hence, $z_{IP} = 552$ and we have found the optimal solution.
- This rounding procedure cannot be guaranteed to yield the optimal solution in general, but it produces a primal and a dual bound which often sandwich z_{IP} in a narrow interval, thus yielding an approximation guarantee.