COURSE #9
Thursday, December 14, 2023 6:03 PM

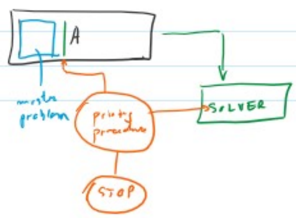
COLUMN GENERATION

CONTEXT IP $\longrightarrow$ LP relaxation
dropping integrality constraint

$$\begin{cases} \min(c^T x) \\ Ax = b \\ x \geq 0 \\ x \in \mathbb{Z} \end{cases} \qquad \rho = \frac{OPT(IP)}{OPT(LP)} \geq 1 \quad \underline{\text{integrality gap}}$$

PROBLEM — $IP_1 \longrightarrow LP_1$, $IP_2 \longrightarrow LP_2$ } prefer formulations with $\rho \simeq 1$

Some problems have good formulations using huge # of variables / equations



CUTTING STOCK PROBLEM

L = 110

l_1 d_1 boards

l_m d_m boards

l_i	d_i
20	48
45	35
50	24
55	10
75	8

GOAL Minimize the total # of boards of 110 cm. that we use.

Pattern of cutting $110 = 5 \times 20 \; \{+10\} \quad \{5,0,0,0,0\}$
$110 = 3 \times 20 + 1 \times 50 \quad \{3,0,1,0,0\}$

$c = \{ n_{c,1}, \dots n_{c,m} \} \quad \left[\sum_{i=1}^{m} l_i \cdot n_{c_i} \leq L \right]$

$n_{c,i} \in \mathbb{N}$ (# of pieces of size l_i cut in this pattern)

$\mathcal{P}$ = set of all possible patterns. (huge set in general)

$p \in \mathcal{P} \quad X_p$ = # of boards cut using pattern p

CUTTING STOCK $\longrightarrow$ (IP)
$$\begin{cases} \min \left(\sum_{c \in \mathcal{P}} X_c \right) \\ \sum_{c \in \mathcal{P}} n_{c,i} \cdot X_c \geq d_i \qquad i = \overline{1,m} \\ X_c \geq 0 \qquad X_c \in \mathbb{Z} \end{cases}$$

(IP) $\Rightarrow$ (LP) (forget condition $X_c \in \mathbb{Z}$)

$$\begin{cases} \min \left(\sum 1 \cdot X_c \right) \end{cases}$$

$$(LP)\ \begin{cases} \min \left(\sum \boxed{1} \cdot X_c \right) \\ \sum\limits_{c \in P} \underline{m_{c,i}}\, X_c \geq d_i \quad i = \overline{1,m} \\ X_c \geq 0 \end{cases}$$

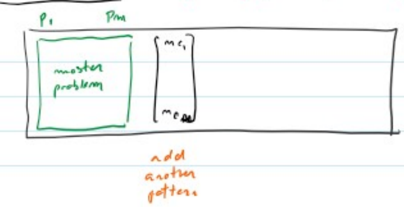
$(LP) \Rightarrow X_c^* = 3.5$

$\underset{LP}{(X_c^*)}$ optimal soln. $\Rightarrow \lceil X_c^* \rceil$ soln. for cutting stock

NOT optimal

very good solution

Problem with LP — very many possible patterns!



1. How to chose patterns in the master problem?

Example

$P_1 = (5, 0, 0, 0, 0)$ (20)
$P_2 = (0, 2, 0, 0, 0)$ (45)
$P_3 = (0, 0, 2, 0, 0)$ (55)
$P_4 = (0, 0, 0, 2, 0)$ (55)
$P_5 = (0, 0, 0, 0, 1)$ (75)

2. How do I write the pricing procedure?

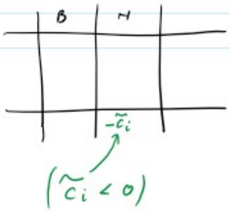
while (can improve soln.)
add new pattern ← which pattern?
resolve extended LP

FOR CUTTING STOCK

pricing → solving a knapsack problem → IP (knapsack easy)

pricing ↓ search for missing pattern that improves soln

SIMPLEX



if $\exists\, i \in N$ s.t $\underline{\tilde{c}_i < 0}$ then add variable X_i

else STOP

PRICING PROCEDURE $\rightarrow \boxed{\min\limits_{i \in N} (\tilde{c}_i)}$ → < 0 $\tilde{c}_i \rightarrow$ ADD x_i ; → ≥ 0 STOP

(exp many patterns in N)

FOR CUTTING STOCK

$(*)\ \boxed{\min\limits_{i \in N} (\tilde{c}_i)} \rightarrow$ KNAPSACK

REDUCED COSTS

$$\begin{cases} \min(c^T x) \\ Ax = b \\ x \geq 0 \end{cases} \quad \underline{\text{solve}} \to \text{basis } B$$

(feasible soln for LP cutting stock)

$$(A_B^{-1}) \Big| \quad A^B x^B + A^N x^N = b$$

$\det \neq 0$

$$x^B + A_B^{-1} A^N x^N = \underbrace{(A_B^B)^{-1} b}_{\bar{b}}$$

$$\boxed{x^B = \bar{b} - A_B^{-1} A_i^N x^N}$$

$$c^T x = c^B x^B + c^N x^N = c^B\left(\bar{b} - A_B^{-1} A^N x^N\right) + c^N x^N$$

$$= \underbrace{(c^B)\bar{b}}_{\text{value}} + \Big(\underbrace{c^N - \overbrace{(c^B)(A^B)^{-1}}^{\pi} A^N}_{\tilde{c}^N = (\tilde{c}_i)_{i \in N}}\Big) X^N$$

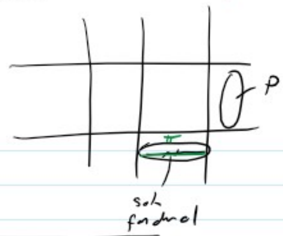
$$\tilde{c}_i = c_i - (\pi \cdot A)_i \qquad \min_{i \in N}(\tilde{c}_i)$$

soln for the dual!

Optimum $\tilde{c}_i \geq 0 \quad \forall i \in N$

$$(D)\begin{cases} \max(b^T y) \\ yA \leq c \\ y \geq 0 \end{cases}$$

($y = \pi$ feasible for dual)



Pricing procedure $\to \boxed{\min_{i \in N} \left(c_i - (\pi \underline{A})_i\right)}$

Let's apply this to cutting stock

$$P_i = \begin{bmatrix} n_{i1} \\ \vdots \\ n_{im} \end{bmatrix}$$

$c_i = 1 \quad \forall i \in P$

$$(**) \quad \boxed{\min_{c_i \in P} \left(1 - \sum_{k=1}^{m} \pi_k n_{ik}\right)}$$

exp large set

??

Comes from the soln of dual problem (SIMPLEX TABLEAU)

IDEA Turn unknown pattern P_i into vector of variables

$$(**) \implies \boxed{\begin{aligned} &\min\left(1 - \sum_{k=1}^{m} \pi_k X_k\right) \\ &\sum_{k=1}^{m} \ell_k \cdot X_k \leq L \\ &X_k \geq 0 \end{aligned}} \quad (***)$$

pricing procedure solves

$$\sum_{k=1} \ldots$$

$$X_k \geq 0$$

<u>Case 1</u> min < 0 for P_i

$\Downarrow$

add P_i

min $\geq 0 \Rightarrow$ STOP OPTIMAL

(* * '*) $\Longleftrightarrow$ KNAPSACK problem.

(** **)

$$\begin{cases} \max \left(\sum_{k=1}^{m} \pi_k y_k \right) \\ \sum_{k=1}^{m} l_k y_k \leq L \\ y_k \geq 0 \ , \ y_k \text{ integral} \end{cases}$$

profit per piece.

capacity

weights of objects

$\lfloor L / l_1 \rfloor$ identical objects of weight l_1, profit π_1 $\quad X_1$

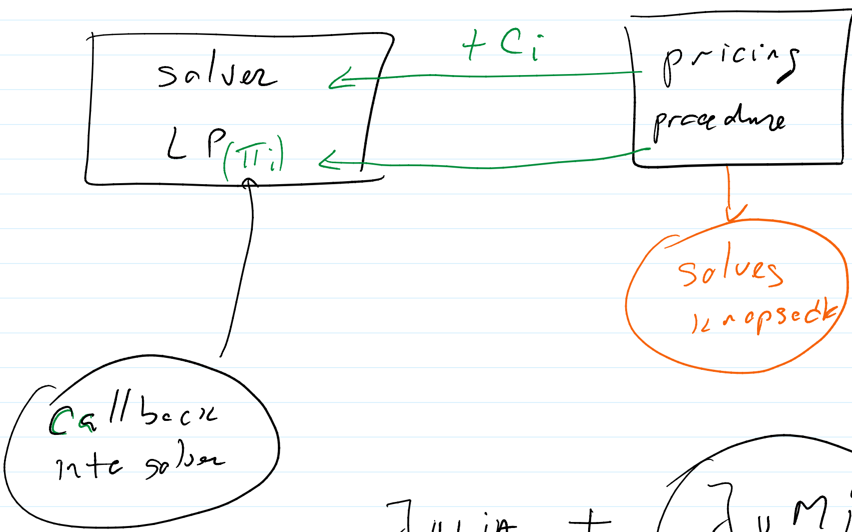
$\vdots$

$\lfloor L / l_m \rfloor$ identical objects of weight l_m, profit π_m $\quad X_m$

(** **) $\Longleftrightarrow$ KNAPSACK instance

OPT profit $> 1 \Rightarrow$ <u>pattern</u>

OPT profit $\leq 1 \Rightarrow$ <u>STOP</u>



JULIA + JUMP

{PYTHON + PULP}

solver independent interface for callbacks

HOW DO WE OBTAIN GOOD FORMULATION TO SOLVE USING COL. GEN ?

DANTZIG - WOLFE DECOMPOSITION (REFORMULATION)

$\Downarrow$

good large formulation solvable by COL. GEN.

SOLVING INTEGER LINEAR PROGRAMS

profit

knapsack as ILP

$$\begin{cases} \max(p_1 x_1 + p_2 x_2 + \dots + p_n x_n) \\ w_1 x_1 + w_2 x_2 + \dots + w_n x_n \le c \end{cases}$$

capacity

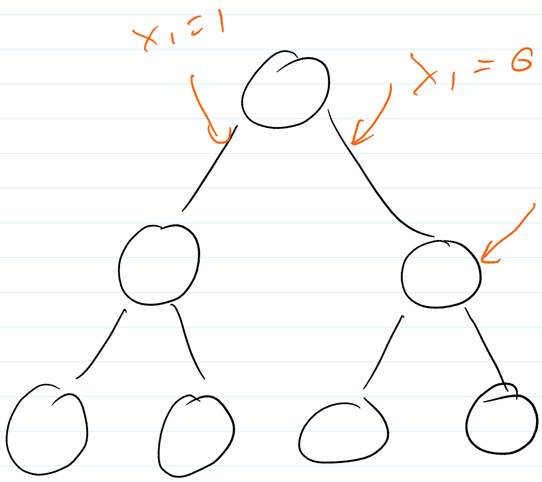
Knapsack as ILP

$$\begin{cases} w_1 x_1 + w_2 x_2 + \dots + \underbrace{w_n}_{\text{weight}} x_n \le t \\ x_i \in \{0, 1\} \end{cases}$$

$$x_i = \begin{cases} 1 & \text{if I take object } i \\ 0 & \text{otherwise} \end{cases}$$

$\boxed{\text{NP-complete!}}$

$$\begin{cases} \max\,(45 x_1 + 48 x_2 + 35 x_3) \\ 5 x_1 + 8 x_2 + 3 x_3 \le 10 \\ x_1, x_2, x_3 \in \{0,1\} \end{cases}$$



estimate for optimum

$\{ OPT \le \text{estimate} \}$

easy to compute

Solve LP relaxation

$\boxed{\text{BRANCH \& BOUND ALG}}$

$$\Big(\max\,(45 x_1 + 48 x_2 + 35 x_3)$$

$$
\begin{cases}
\max \; |45 x_1 + 40 x_2 + 35 x_3| \\
5 x_1 + 8 x_2 + 3 x_3 \le 10 \\
x_1, x_2, x_3 \in \underline{\underline{\{0, 1\}}}
\end{cases}
$$

fractional

knapsack

p3

<u>solve using</u>

<u>greedy</u>