# 620-362
# Applied Operations Research

# Branch & Price


**Department of Mathematics and Statistics**
**The University of Melbourne**




Some contents of this presentation are adapted from year 2005 course notes for 620-362 Applied Operations Research, Department of Mathematics and Statistics, The University of Melbourne (compiled by Prof Natashia Boland and Dr Renata Sotirov)

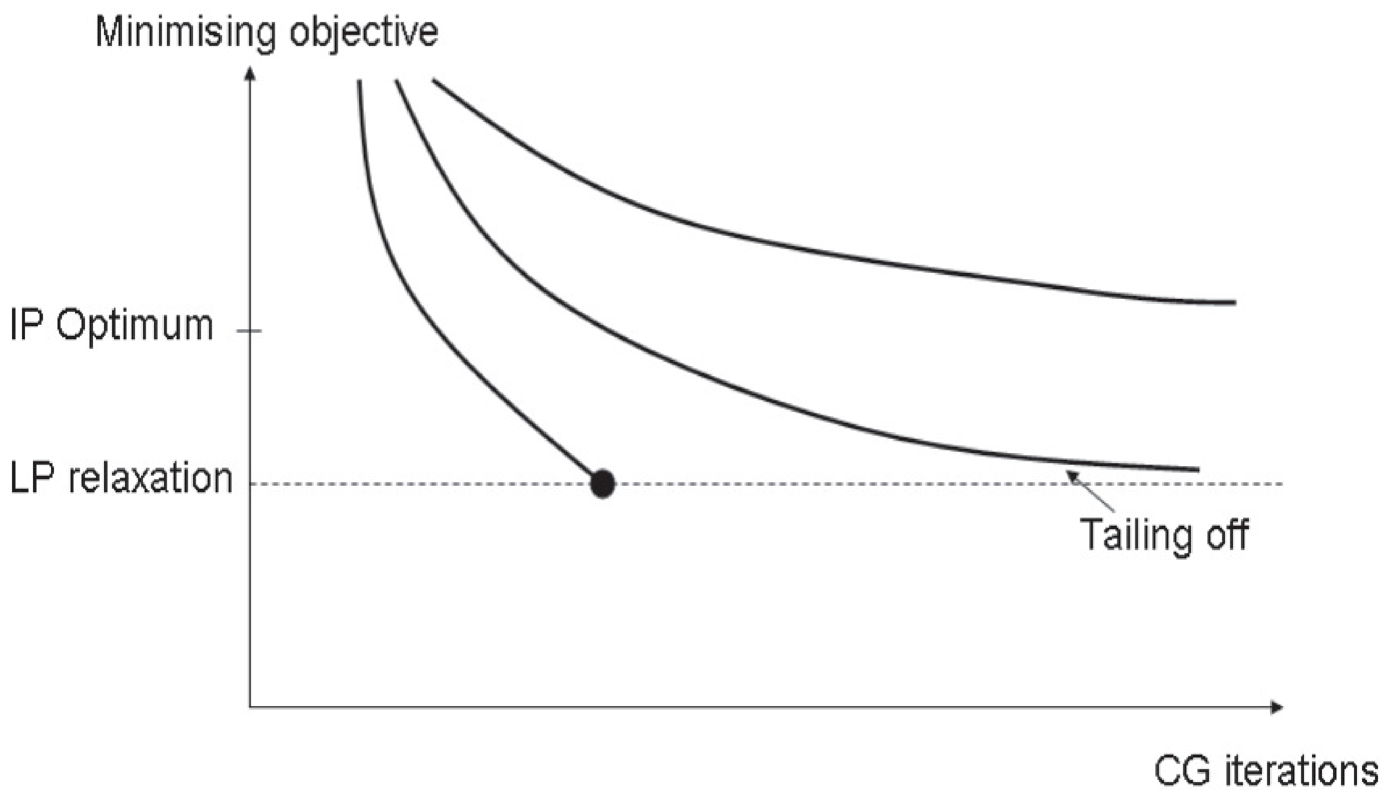
# Branch & Price

- We have previously looked at column generation at the root node.

- It is likely there could be more “useful” columns that could be added during B&B
  - since we will be solving LPs (at nodes) which are different from the root node LP

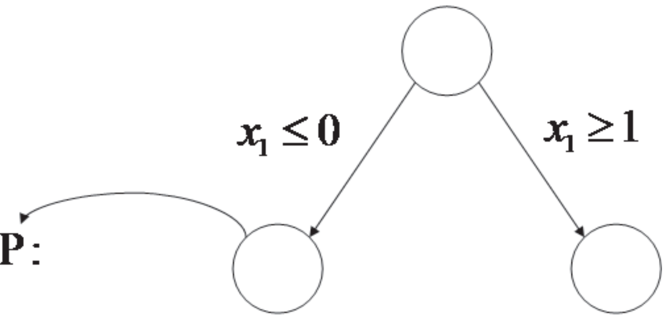
# Branch & Price

- It possible at the root node, LP objective tails off with increasing number of column generation iterations.
  – more columns have to be “priced” during the B&B search to rule out missing any useful columns.



# Branch and & Bound

- Defined by three main components
  - the bounds at each node
  - the tree search strategy (which node to branch from?)
  - **the branching rule (which variable to branch-on next?)**




Master LP:

$$\begin{cases} \min cx \\ \text{s.t. } Ax \geq b \quad (\pi) \\ \qquad x_1 \leq 0 \quad (\mu_1 \geq 0) \\ \qquad x \geq 0 \end{cases}$$

- Branching on, say $x_1 = 0$ requires that the column generation subproblem at the node to disregard $x_1$ in its solution.

# Example: A Cutting Stock Problem

Consider the modified cutting stock column generation subproblem, if we branch on $x_1 \leq 0$.

$$\min \quad 1-\sum_{i=1}^{n}\pi_i n_i$$

$$s.t.$$

$$\sum_{i=1}^{n} w_i n_i \leq W$$

$$n_i \geq 0, \quad \text{integer for all widths } i$$

$$n_i \neq a_{i1}, \quad \text{for some width i}$$

Constraint $n_i \neq a_{i1}$ is a very difficult constraint. It destroys the subproblem structure.

– it requires the subproblem to find the $2^{nd}$ best knapsack solution

# How to branch in with column generation?

- Branching rules have to make sure columns that are infeasible due to branching constraints are not generated by subproblem.

- Futhermore, subproblem must remain tractable after modification.

- Possible rules:
  - base branching rules on original or subproblem variables
  - look for a dichotomy which splits columns (Ryan-Foster)

# Example: A Binary Cutting Stock Problem

Requires demand for each width *i* equal 1.

Parameters:

- P = set of all cutting patterns
- $P_i$ = set of cutting patterns that has piece with width *i*
- $w_i$ = width of piece *i*

Variable:

- $x_j$ = 1, if pattern *j* is used; 0, otherwise.

$$\min \sum_{j \in P} x_j$$

$$s.t.$$

$$\sum_{j \in P_i} x_j = 1, \quad \text{for all widths } i$$

$$x_j \geq 0, \quad \text{integer}, \forall j \in P$$

# Example: A Binary Cutting Stock Problem

Suppose Master LP variables are fractional, say $0 < x_1 < 1$.

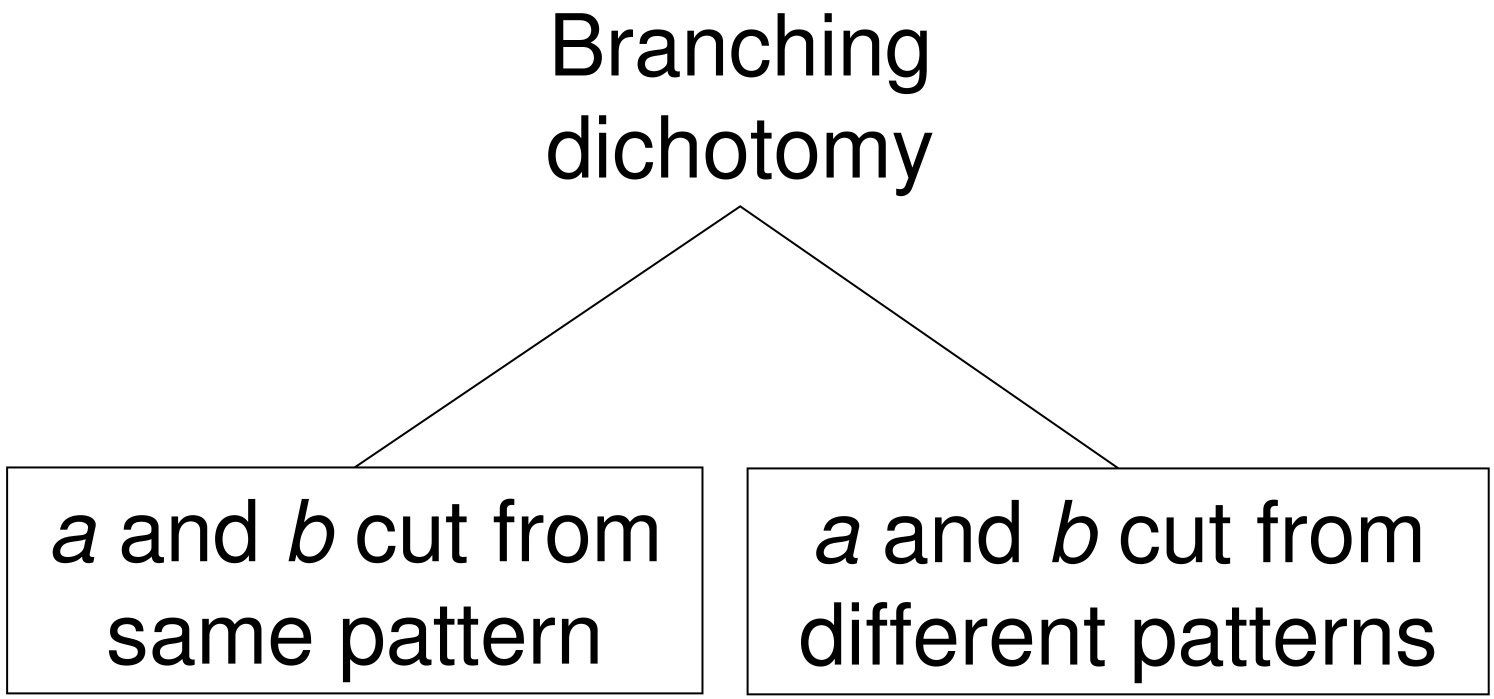
Consider width *a* with Pattern $1 \in P_a$.

Now suppose there is another pattern, say Pattern $2 \in P_a$ and has $0 < x_2 < 1$.

Patterns 1 and 2 must be different:

so for some b, Pattern $1 \in P_b$, Pattern $2 \notin P_b$

# Example: A Binary Cutting Stock Problem



# Example: A Binary Cutting Stock Problem

On the left branch:

$$\min \quad 1-\sum_{i=1}^{n}\pi_i n_i$$

$$s.t.$$

$$\sum_{i=1}^{n} w_i n_i \le W$$

$$n_a = n_b$$

$$n_i \text{ binary}, \quad \text{for some width i}$$

This is standard Binary Knapsack Problem – easy to solve

# Example: A Binary Cutting Stock Problem

On the right branch:

$$\min \quad 1-\sum_{i=1}^{n}\pi_i n_i$$

$$s.t.$$

$$\sum_{i=1}^{n} w_i n_i \leq W$$

$$n_a \neq n_b$$

$$n_i \text{ binary}, \quad \text{for some width i}$$

This is more difficult, but solvable.

We can represent $n_a \neq n_b$ using $n_a + n_b \leq 1$.