Branch and Price

Denis Pinha, Qipeng P. Zheng

Department of Industrial and Management Systems Engineering
West Virginia University

November 8th, 2012
Outline

1. Introduction
2. Branch and Bound Tests
3. Branch and Price Application
4. Branch and Price Numerical Example
5. Branching strategies
Branch and Price integrates Branch and Bound and Column Generation methods for solving large-scale IPs.

Branch and Price is similar to Branch and Cut, except that the procedure focuses on Column Generation rather than Row Generation.

At each node of the Branch and Bound tree, columns may be generated to tighten the LP relaxation.

In Branch and Price, sets of columns are left out of the LP relaxation of large IPs.

- There are too many columns to handle and most of them will have their associated variables equal to zero in an optimal solution.

To check optimality, a sub-problem, also called the pricing problem, is solved to identify columns to enter the basis.
A subproblem is dismissed from further consideration if:

- **Test 1:** Its bound $\leq Z^*$
- **Test 2:** Its LP-relaxation has no feasible solutions
- **Test 3:** The optimal solution for LP-relaxation is integer. (If this solution is better than the incumbent, it becomes the new incumbent, and test 1 is reapplied to all subproblems with the new larger $Z^*$).
Summary of Branch and Bound algorithm

Steps for each Iteration:

- Set $Z^* = -\infty$
- Branching: Among the remaining subproblems, select the most recent created. (Break ties according to which has the larger bound). Branch from this node to create two new subproblem by fixing the next variable at either 0 or 1.
- Bounding: Obtain its bound by applying simplex method to its LP-relaxation and rounding down the value of $Z$ for the resulting optimal solution.
- Fathoming: For each new subproblem, apply the three tests summarized before.
1. Introduction

2. Branch and Bound Tests

3. Branch and Price Application

4. Branch and Price Numerical Example

5. Branching strategies
Generalized Assignment Problem (GAP)

- In the GAP the objective is to find a maximum profit assignment of $n$ jobs to $m$ agents, such that each job is assigned to precisely one agent subject to capacity restrictions on the agents.

- Problem Formulation

\[
\text{max } \sum_{i=1}^{m} \sum_{j=1}^{n} p_{ij} x_{ij} \quad (1)
\]

subject to:

\[
\sum_{i=1}^{m} x_{ij} = 1, \quad \forall j = 1...n \quad (2)
\]

\[
\sum_{i=1}^{n} w_{ij} x_{ij} \leq c_i, \quad \forall i = 1...m \quad (3)
\]

\[
x_{ij} \in \{0, 1\}, \quad i = 1...m, \quad j = 1...n \quad (4)
\]
Problem Formulation (GAP)

\( p_{ij} \) = profit associated with assigning job \( j \) to agent \( i \)

\[
    x_{ij} = \begin{cases} 
        1, & \text{if job } j \text{ is assigned to agent } i \\
        0, & \text{otherwise.}
    \end{cases}
\]

\( w_{ij} \) = claim on the capacity of agent \( i \) by job \( j \)

\( c_i \) = capacity of agent \( i \)
Let $K_i = \{x_{i1}^i, x_{i2}^i, \ldots, x_{ik_i}^i\}$ be the set of all possible assignments of jobs to agent $i$.

$x_k^i = \{x_{1k}^i, x_{2k}^i, \ldots, x_{nk}^i\}$ is a feasible solution to $\sum_{i=1}^{n} w_{ij} x_{ij} \leq c_i$.

$x_{jk}^i \in \{0, 1\}$

$y_k^i = \begin{cases} 1 & \text{, if feasible assignment } x_k^i \text{ is selected} \\ 0 & \text{, otherwise.} \end{cases}$
Problem Reformulation : GAP

The GAP can be reformulated as follows: called Master Problem

\[
\text{max } \sum_{i=1}^{m} \sum_{k=1}^{k_i} \left( \sum_{j=1}^{n} p_{ij} x_{jk}^i \right) y_k^i
\]  

subject to:

\[
\sum_{i=1}^{m} \sum_{k=1}^{k_i} x_{jk}^i y_k^i = 1, \quad \forall j = 1 \ldots n  
\]  

\[
\sum_{k=1}^{k_i} y_k^i \leq 1, \quad \forall i = 1 \ldots m  
\]  

\[
y_k^i \in \{0, 1\}, \quad i = 1 \ldots m, \quad k \in k_i  
\]

Where the first set of constraints enforces that each job is assigned to precisely one agent and the second constraints enforces that at most one feasible assignment is selected for each agent.
Advantages of reformulating the problem

- The disaggregated problem is essentially obtained by applying Dantzig-Wolfe decomposition
  - Sub-divides the original problem into a Master Problem and sub-problem.
  - A column $y^i_k$ in the master problem represents a feasible assignment of jobs to agent $i$.
  - Master Problem cannot be solved directly due to exponential number of columns.
  - Restricted Master Problem (RMP): It considers only a subset of the columns.
Sub-Problem

Additional columns can be generated for the RMP by solving the following two sub-problems:

$$\max \{ z(KP_i) - v_i \}, \ i = 1, \ldots, m$$  \hspace{1cm} (9)

where $v_i$ is the optimal dual price from the solution to the RMP associated with constraint of agent $i$, $u_j$ for job $j$ and $z(KP_i)$ is the value of the optimal solution to the following problem

$$\max \sum_{j=1}^{n} (p_{ij} - u_j) x_{ij}$$  \hspace{1cm} (10)

subject to:

$$\sum_{i=1}^{n} w_{ij} x_{ij} \leq c_i$$  \hspace{1cm} (11)

$$x_{ij} \in \{0, 1\}, \ j = 1 \ldots n$$  \hspace{1cm} (12)
Example

\( p_{ij} = \) profit associated with assigning job \( j \) to agent \( i \)

<table>
<thead>
<tr>
<th>Agent</th>
<th>Job 1</th>
<th>Job 2</th>
<th>Job 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5</td>
<td>7</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>5</td>
<td>10</td>
</tr>
</tbody>
</table>

\( c_i = \) capacity of agent \( i \)

<table>
<thead>
<tr>
<th>Agent</th>
<th>Capacity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>2</td>
<td>8</td>
</tr>
</tbody>
</table>

\( w_{ij} = \) claim on the capacity of agent \( i \) by job \( j \)

<table>
<thead>
<tr>
<th>Agent</th>
<th>Job 1</th>
<th>Job 2</th>
<th>Job 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
<td>3</td>
<td>4</td>
</tr>
</tbody>
</table>
Example

\[
\text{max } 5x_{11} + 7x_{12} + 3x_{13} + 2x_{21} + 10x_{22} + 5x_{23} \tag{13}
\]

Subject to:

\[
x_{11} + x_{21} = 1 \tag{14}
\]
\[
x_{12} + x_{22} = 1 \tag{15}
\]
\[
x_{13} + x_{23} = 1 \tag{16}
\]
\[
3x_{11} + 4x_{12} + 2x_{13} \leq 5 \tag{17}
\]
\[
5x_{21} + 3x_{22} + 4x_{23} \leq 8 \tag{18}
\]
\[
x_{ij} \in \{0, 1\}, j = 1...n, i = 1...m \tag{19}
\]
Example

$k_i$ denotes the set of possible feasible assignment to agent $i$

\[
k_1 = \{(1, 0, 0), (0, 1, 0), (0, 0, 1), \left( \begin{array}{ccc}
1 & 0 & 1
\end{array} \right) \}
\]

\[
k_2 = \{(1, 0, 0), (0, 1, 0), (0, 0, 1), (1, 1, 0), (0, 1, 1) \}
\]

The Master Problem is formulated as follows:

\[
\max \sum_{i=1}^{m} \sum_{k=1}^{k_i} \left( \sum_{j=1}^{n} p_{i,j} x_{j,k}^i \right) y_k^i
\]

\[
\max z = 5y_1^1 + 7y_2^1 + 3y_3^1 + 8y_4^1 + 2y_1^2 + 10y_2^2 + 5y_3^2 + 12y_4^2 + 15y_5^2
\]
Example

\[\sum_{i=1}^{m} \sum_{k=1}^{k_i} x_{jk}^i y_k^i = 1, \quad \forall j = 1...n\]

Subject to:

\[y_1^1 + 0 + 0 + y_4^1 + y_1^2 + 0 + 0 + y_4^2 + 0 = 1 \ (u_1) \quad (23)\]

\[0 + y_2^1 + 0 + 0 + 0 + y_2^2 + 0 + y_4^2 + y_3^2 = 1 \ (u_2) \quad (24)\]

\[0 + 0 + y_3^1 + y_4^1 + 0 + 0 + y_3^2 + 0 + y_5^2 = 1 \ (u_3) \quad (25)\]

\[\sum_{k=1}^{k_i} y_k^i \leq 1, \quad \forall i = 1...m\]

\[y_1^1 + y_2^1 + y_3^1 + y_4^1 \leq 1 \ (v_1) \quad (26)\]

\[y_1^2 + y_2^2 + y_3^2 + y_4^2 + y_5^2 \leq 1 \ (v_2) \quad (27)\]
Restricted Master Problem - RMP

Let us arbitrarily choose columns $y_1$ and $y_5$ which are feasible solutions. RMP is then:

$$\max z = 5y_1 + 15y_5$$

Subject to:

$$y_1 + 0 = 1 \ (u_1) \quad (28)$$

$$0 + y_5 = 1 \ (u_2) \quad (29)$$

$$0 + y_5 = 1 \ (u_3) \quad (30)$$

$$y_1 + 0 \leq 1 \quad (31)$$

$$0 + y_5 \leq 1 \quad (32)$$
Example

\[ B = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \]

\[ B^{-1} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \]

\[ C_B B^{-1} = \begin{bmatrix} 5 & 15 \end{bmatrix} \]
Example

\[ u_1 = 5, \quad u_2 = 15, \quad v_1 = 0, \quad v_2 = 0 \]

\[
\max \sum_{j=1}^{n} (p_{ij} - u_j)x_j^i
\]  \hspace{1cm} (33)

subject to:

\[
\sum_{i=1}^{n} w_{ij}x_j^i \leq c_i
\]  \hspace{1cm} (34)

- Subproblem for Agent 1

\[
\max(5 - 5)x_1^1 + (7 - 15)x_2^1 + (3 - 0)x_3^1
\]

Subject to

\[
3x_1^1 + 4x_2^1 + 2x_3^1 \leq 5
\]

Optimal solution are (1,0,0) and (0,0,1) and \( z = 3 \). Hence

\[ z(KP1) - v_1 = 3 - 0 = 3 \]
Example

- Subproblem for Agent 2

\[
\max (2 - 5)x_1^2 + (10 - 15)x_2^2 + (5 - 0)x_3^2
\]

Subject to

\[5x_1^1 + 3x_2^1 + 4x_3^1 \leq 8\]

Optimal solution are \((0,0,1)\) and \(z = 5\). Hence

\[z(KP2) - v_2 = 5 - 0 = 5\]

Since the reduced cost for agent 2 is greater than in agent 1, sequence 3 in agent 2 \(y_3^2\) is the column generated.
Example

The new RMP is:

$$\max z = 5y_1^1 + 15y_2^2 + 5y_3^2$$

Subject to:

$$y_1^1 + 0 + 0 = 1 \ (u_1) \quad (35)$$

$$0 + y_5^2 + 0 = 1 \ (u_2) \quad (36)$$

$$0 + y_5^2 + y_3^2 = 1 \ (u_3) \quad (37)$$

$$y_1^1 + 0 + 0 \leq 1 \quad (38)$$

$$0 + y_5^2 + y_3^2 \leq 1 \quad (39)$$
Example

\[ C_B B^{-1} = \begin{bmatrix} 5 & 15 & 5 \end{bmatrix} \]

\[ u_1 = 5, \ u_2 = 15, \ u_3 = 5, \ v_1 = 0, \ v_2 = 0 \]

- Subproblem for Agent 1

\[ \max (5 - 5)x_1 + (7 - 15)x_2 + (3 - 5)x_3 \]

Subject to \[ 3x_1 + 4x_2 + 2x_3 \leq 5 \]

Optimal solution are \((1,0,0)\) and \((0,0,0)\) and \(z = 0\). Hence \(z(KP1) - v_1 = 0 - 0 = 0\)
Example

Subproblem for Agent 2

\[
\begin{align*}
\max(2 - 5)x_1^2 + (10 - 15)x_2^2 + (5 - 5)x_3^2 \\
\text{Subject to} & \quad 5x_1^1 + 3x_2^1 + 4x_3^1 \leq 8
\end{align*}
\]

Optimal solution are \((0,0,1)\), \((0,0,1)\) and \(z = 0\). Hence

\[z(KP2) - v_2 = 0 - 0 = 0\]

Reduced cost for all columns are 0. Hence the solution \(y_1^1 = 1\), \(y_3^2 = 1\) and \(y_3^2 = 0\) is optimal.
Example

- Optimal Assignment
  - (1,0,0) for agent 1. Job 1 is assigned to agent 1.
  - (0,1,1) for agent 2. Jobs 2 and 3 are assigned to agent 2.
  - $Z^* = 5 + 10 + 5$. 
Outline

1. Introduction
2. Branch and Bound Tests
3. Branch and Price Application
4. Branch and Price Numerical Example
5. Branching strategies
Branching strategies

Since there is no more column with positive reduce cost to be added and the optimal solution for the RMP is integer, the optimal solution was obtained.

- However, suppose that the variables in the optimal solution are fractional for the RMP:
  - What to do?
    - Branching!
Branching strategies

Instead of branching on the $y^i_k$-variables in the master problem we use a branching rule that corresponds to branching on the original variables $x_{ij}$.

Standard branching on the $y^i_k$-variables creates a problem along a branch where a variable has been set to zero.

- $x^i_k$ represents a particular solution to the jth knapsack problem.
- Thus $y^i_k = 0$ means that this solution is excluded.
- However, it is possible and quite likely that the next time the jth knapsack problem is solved, the optimal solution is precisely the one represented by $x^i_k$.
- It happens because the sub-problem is solved regardless the variables $y^i_k$. In other words, in case that you set $y^i_k = 0$, there is no restriction in the sub-problem to ensure that $x^i_k$ cannot be the optimal solution.
Branching strategies

Recall the Formulation:

$$\max \sum_{i=1}^{m} \sum_{j=1}^{n} p_{ij} x_{ij}$$

Reformulation:

$$\max \sum_{i=1}^{m} \sum_{k=1}^{k_i} \left( \sum_{j=1}^{n} p_{ij} x_{jk} \right) y_k^i$$

Hence:

$$x_{ij} = x_{jk}^i y_k^i$$

Example

$$x_{21} = x_{13}^2 y_3^2$$

Suppose $y_3^2 = 0.4$ then $x_{21} = 0$, $x_{22} = 0$ and $x_{23} = 0.4$
Branch and Price-Tree

Each node - Columns are added as long as possible. If the reduced cost is zero or negative and the optimal solution for the RMP is fractional. Branching!
Branching strategies

- Pick the variable $x_{ij}$ as follows:
  - The variable that has closest value to 0.5
  - Or the variable that has the closest value to the bounds 0 or 1.
  - There is no restrict rule to pick the variable. Depending on the problem picking the variable closest to 0.5 might work better (converge faster) than the variable close to the bound 0 or 1.

- When $x_{ij} = 1$, all existing columns in the master that don't assign job j to agent i are deleted and job j is permanently assigned to agent i. Variable $y^i_k$ is fixed to 1 in the jth knapsack problem.

- When $x_{ij} = 0$, all existing columns in the master that assign job j to agent i are deleted and job j cannot be assigned to agent i. Variable $y^i_k$ is removed from the jth knapsack problem.
Branching strategies

Branching using the standard formulation while working with the disaggregated formulation.

- Fixing a single variable or a set of variable in the standard formulation (SF) has an equivalent in the disaggregated formulation.
  - In the SF fixing $x_{ij}$ to zero forbids job $j$ to be assigned to agent $i$.
  - In the SF fixing $x_{ij}$ to one requires job $j$ to be assigned to agent $i$. 
Branching strategies

- To forbid a job $j$ to be assigned to agent $i$ in the disaggregated formulation:
  - If $x_{jk}^i = 1$ then $y_k^i = 0$.

- To require a job $j$ to be assigned to agent $i$ in the disaggregated formulation:
  - If $x_{jk}^i = 0$ then $y_k^i = 0$. (To ensure that others jobs don’t be assigned to agent $i$)
  - If $x_{jk}^l = 1$ then $y_k^i = 0$. (To ensure that job $j$ doesn’t be assigned to agent $l$). $1 \leq l \neq i \leq m$
Resulting branching scheme is compatible with the pricing problem

- Forbidding job \( j \) to be assigned to agent \( i \) is accomplished by not considering job \( j \) to agent \( i \) in the subproblem presented before:

\[
\max \sum_{j=1}^{n} (p_{ij} - u_j)x^i_j
\]

subject to:

\[
\sum_{i=1}^{n} w_{ij}x^i_j \leq c_i
\]
Resulting branching scheme is compatible with the pricing problem

- Requiring job $j$ to be assigned to agent $i$ is accomplished by not considering job $j$ to agent $i$ in the subproblem and reducing the capacity of agent $i$ by the claim on its capacity of job $j$:

$$\max \sum_{j=1}^{n} (p_{ij} - u_j)x_j^i$$

subject to:

$$\sum_{i=1}^{n} w_{ij}x_j^i \leq c_i$$

- For different problems some modifications like above should be done in order to have the pricing problem consistent with the branching.
Reference

- Barnhart et. al, "Branch and Price: Column Generation for Solving Huge Integer Programs" OR Vol 46. No.3, 1998
- Akella et. al, "Branch and Price: Column Generation for Solving Huge Integer Programs"
Thank You.