IE 400 Principles of Engineering Management

Integer Programming: Branch And Bound

Lecture Notes 7

Geometry of IP

max x_2

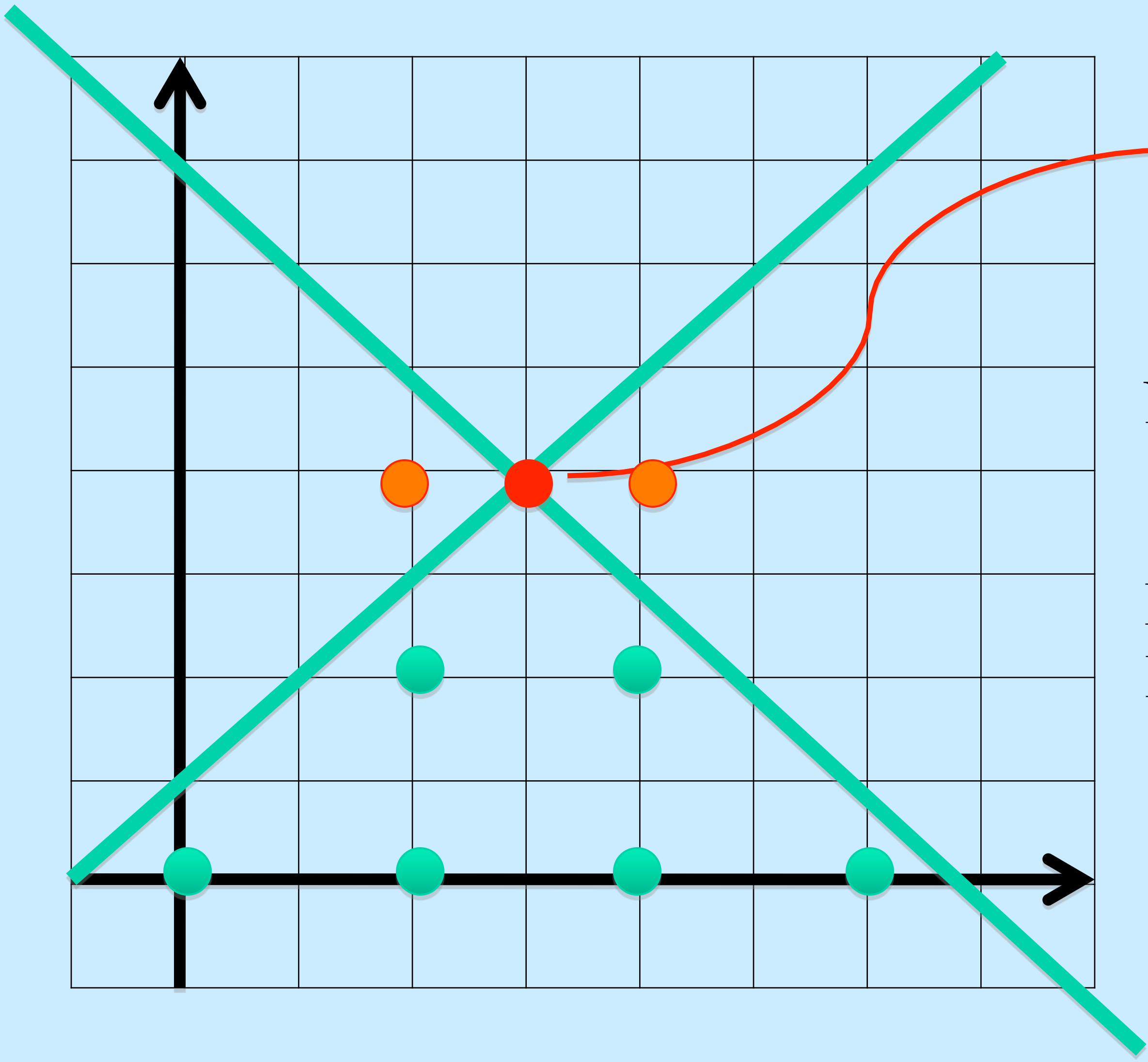
s.t.

$$-x_1 + x_2 \leq 1/2$$

$$x_1 + x_2 \leq 7/2$$

$$x_1, x_2 \geq 0, \text{ integer}$$

- By removing the integrality conditions, we get an LP problem.
- This is called the LP-relaxation and is often a good way starting IP problems.
- If we solve the LP-relaxation of the example:



$x_1 = 3/2$
$x_2 = 2$
$z = 2$

NOT INTEGER

ROUNDING
Infeasible

$x_1 = 1$, or 2
$x_2 = 2$
$z = 2$

INTEGER PROGRAMMING

- Proposition 1: For a maximization problem, the optimal value of the LP-relaxation is $\geq$ optimal value of the IP
- Proof: The LP relaxation has a larger feasible region and so more alternatives

- Proposition 2: If the optimal solution of the LP relaxation is integer valued, then that solution is also optimal for the IP.

Trading for Profit Game

Prize	1	2	3	4	5	6
Points	5	7	4	3	4	6
Utility	16	22	12	8	11	19

Budget: 14 points.

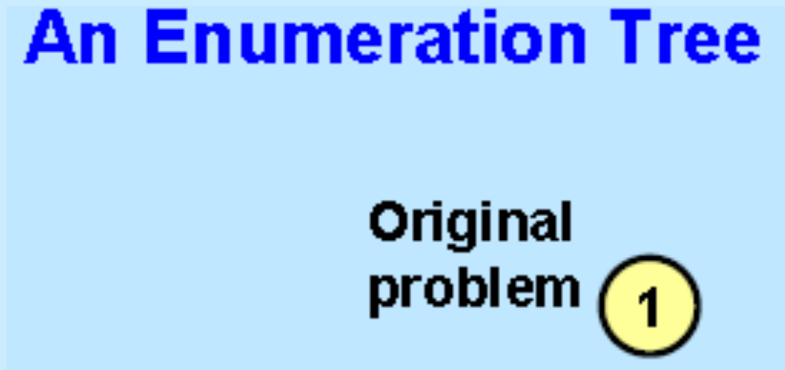
maximize $16x_1 + 22x_2 + 12x_3 + 8x_4 + 11x_5 + 19x_6$

subject to $5x_1 + 7x_2 + 4x_3 + 3x_4 + 4x_5 + 6x_6 \leq 14$

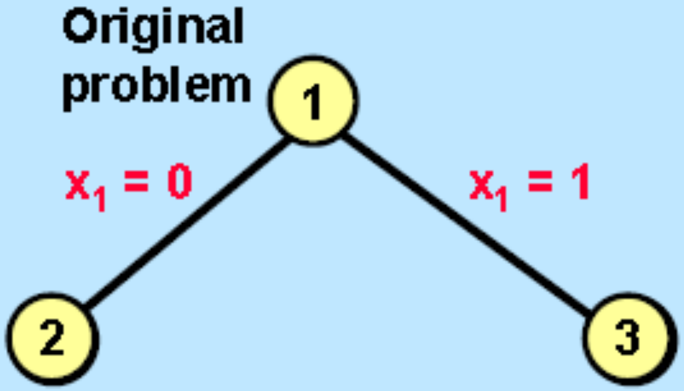
x_j binary for $j = 1$ to 6

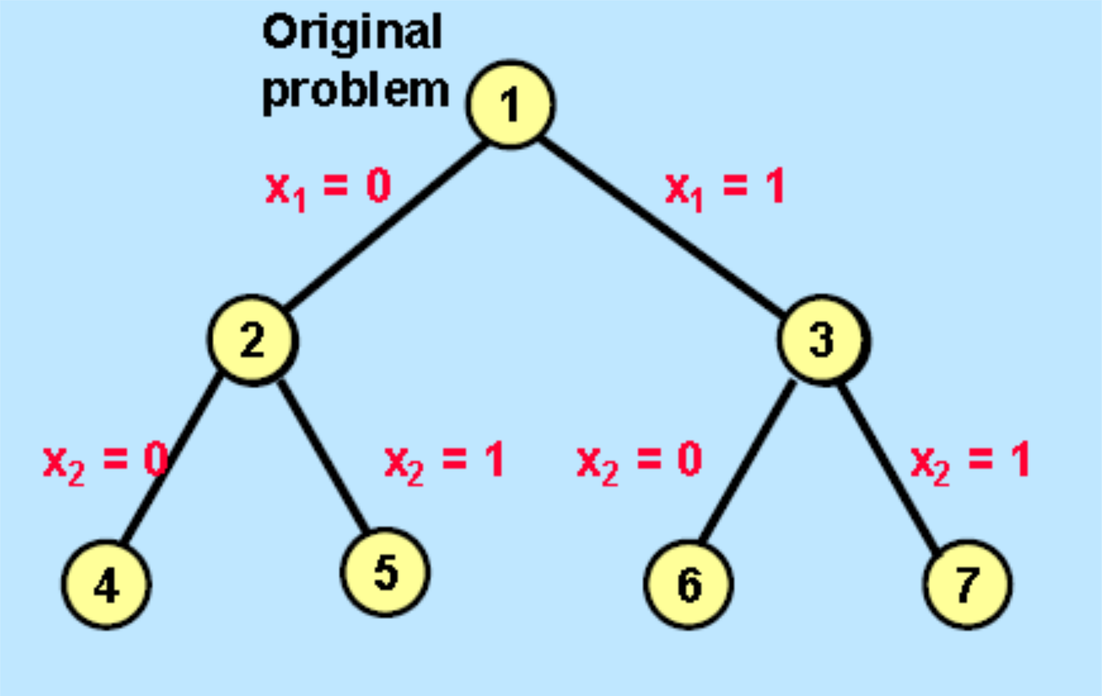
Complete Enumeration

- Systematically considers all possible values of the decision variables.
  - If there are n binary variables, there are 2^n different ways.
- Usual idea: iteratively break the problem in two. At the first iteration, we consider separately the case that $x_1 = 0$ and $x_1 = 1$.

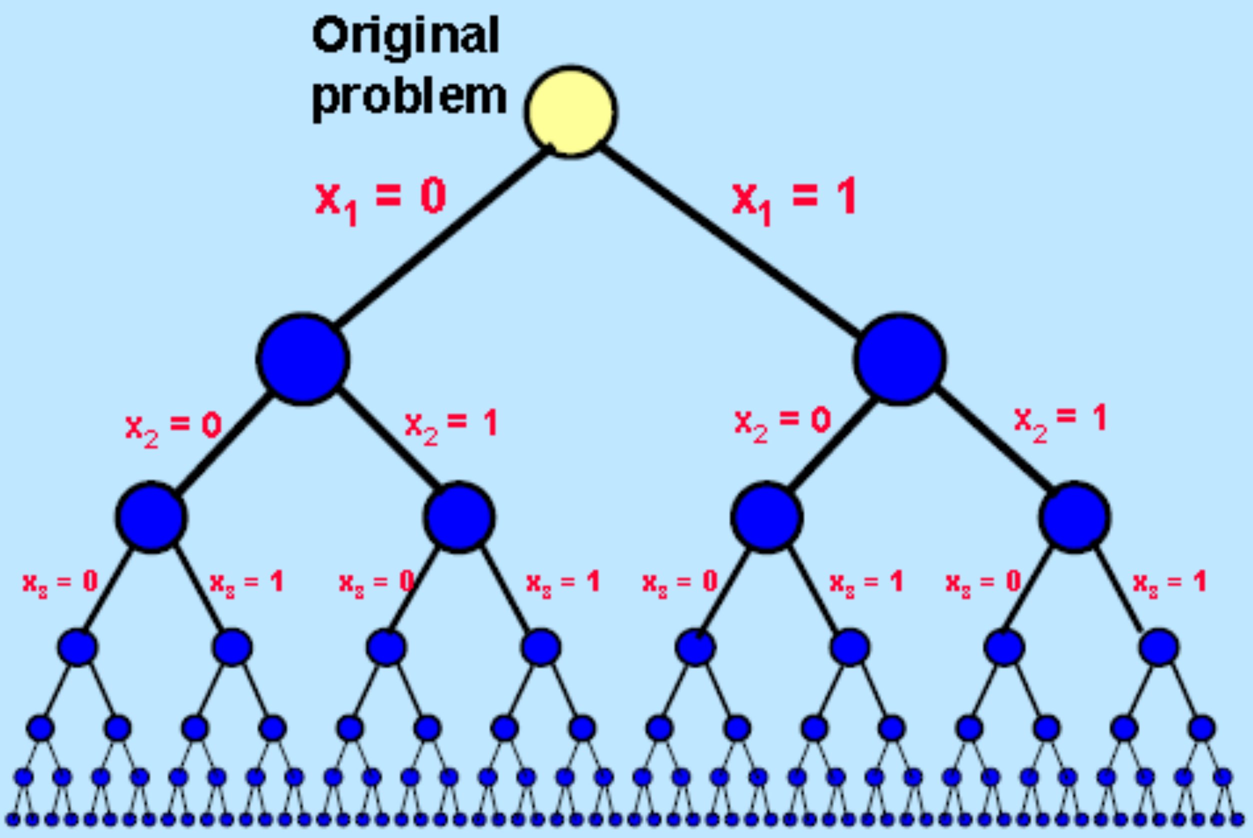

An Enumeration Tree
Original
problem
1

An Enumeration Tree




Original
problem
1
$x_1 = 0$
$x_1 = 1$
2
3
$x_2 = 0$
$x_2 = 1$
$x_2 = 0$
$x_2 = 1$
4
5
6
7

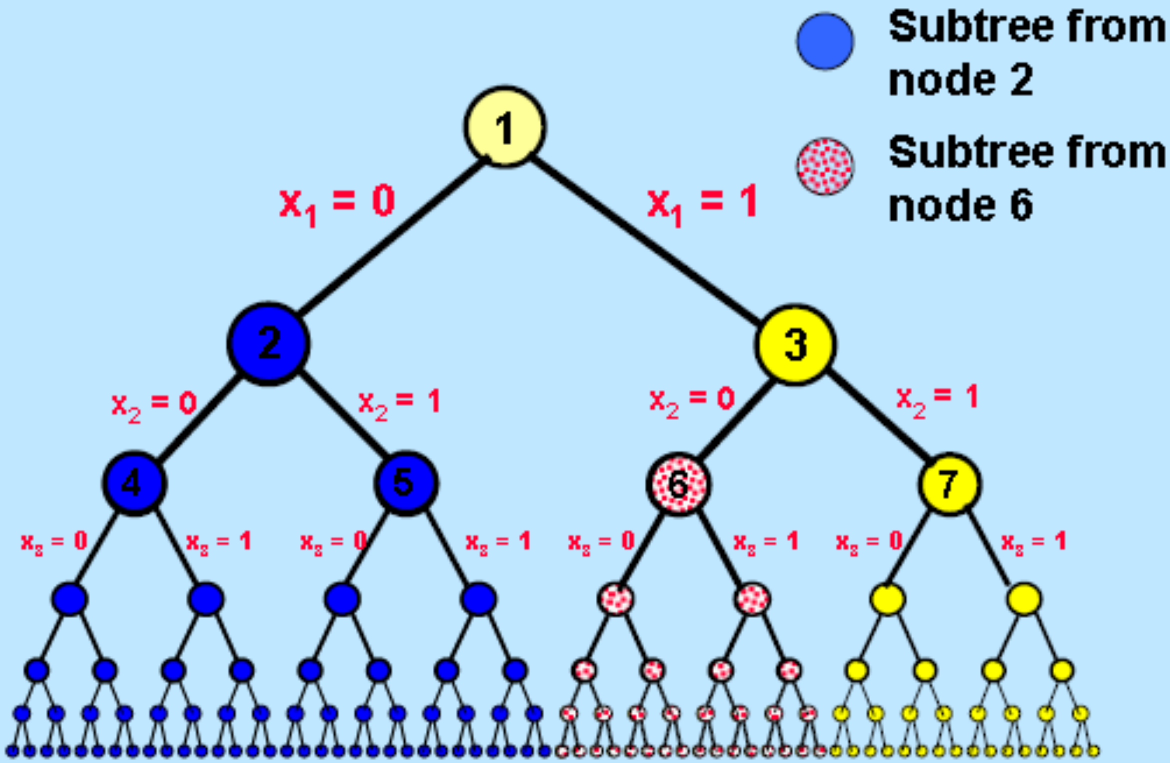
An Enumeration Tree



On complete enumeration

- Suppose that we could evaluate 1 trillion solutions per second, and instantaneously eliminate 99.9999999% of all solutions as not worth considering
- Let n = number of binary variables
- Solutions times
  - n = 70, 1 second
  - n = 80, 17 minutes
  - n = 90 11.6 days
  - n = 100 31 years
  - n = 110 31,000 years

Subtrees of an Enumeration Tree



The bottom nodes are ***leaves*** of the tree.

Something needed for Branch and Bound: The incumbent.

We need a feasible solution to the integer program. We call this the *incumbent* .

Suppose that x_I is the incumbent.

Let z_I be its objective value.

Important question: how does one find an incumbent?

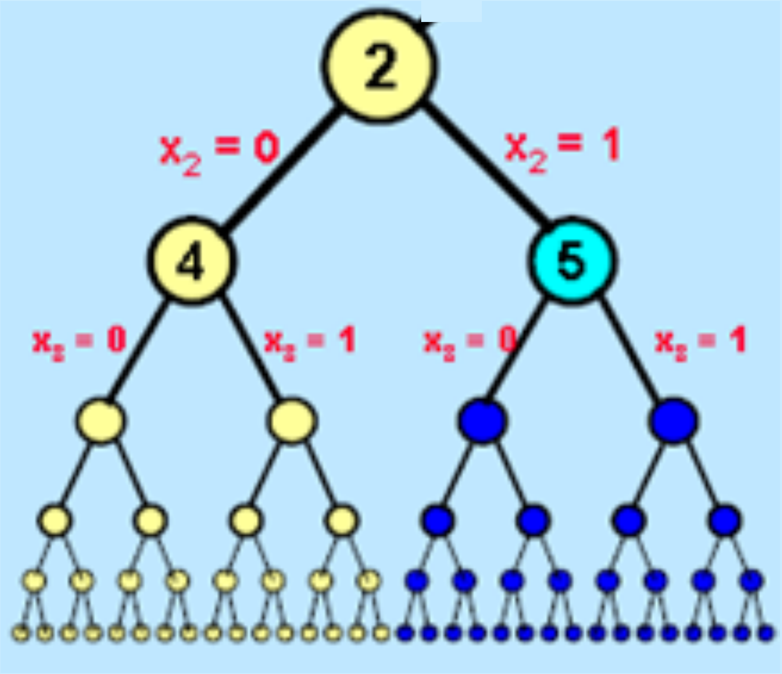
We'll deal with that later. Let us just assume we have one.

Starting incumbent (which I found by inspection.)

$x_1 = 1$; $x_2 = 1$; $x_3 = x_4 = x_5 = x_6 = 0$ $z_I = 38$;

The Essence of Branch and Bound

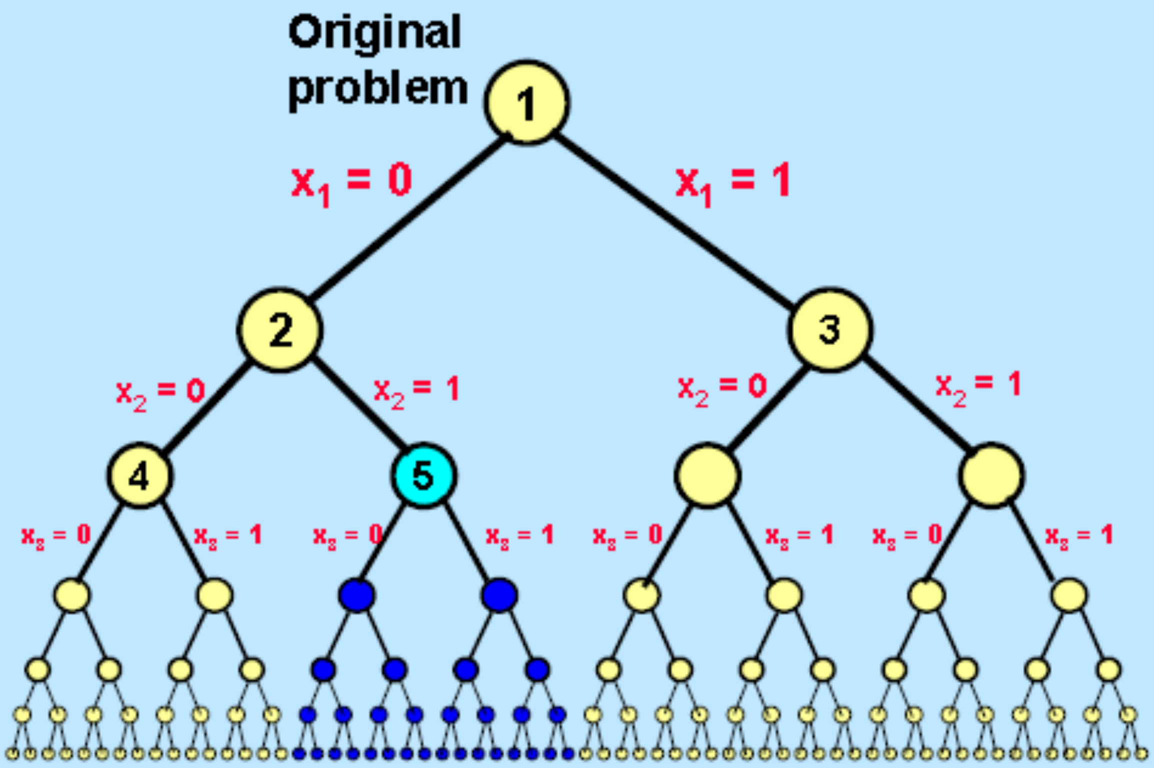
- Select nodes of the "enumeration tree" one at a time but do not branch from a node if none of its descendents can be a better solution than that of the incumbent



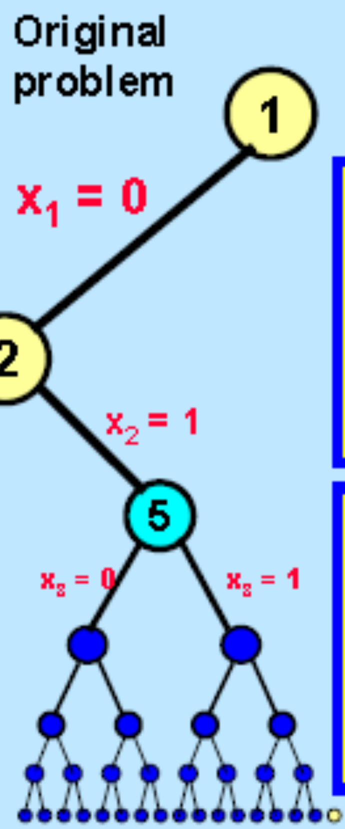
Eliminate subtree 2 if

- Incumbent z^I
- Bound $z^{LP}(2)$
- $z^{LP}(2) \leq z^I$

How do we find an upper bound on all descendent solutions from node 5?



Finding an upper bound for descendents of node 5 (or any other node)



To find the <u>optimum descendent</u> of node 5, we can solve the following IP called <u>*Subproblem 5*</u>.

Subroblem (5)

max $\quad 16x_1 + 22x_2 + 12x_3 + 8x_4 + 11x_5 + 19x_6$

s.t. $\quad 5x_1 + 7x_2 + 4x_3 + 3x_4 + 4x_5 + 6x_6 \leq 14$

$\quad x_1 = 0,\ x_2 = 1 \qquad x_j$ binary for $j = 3$ to 6

The LP relaxation:

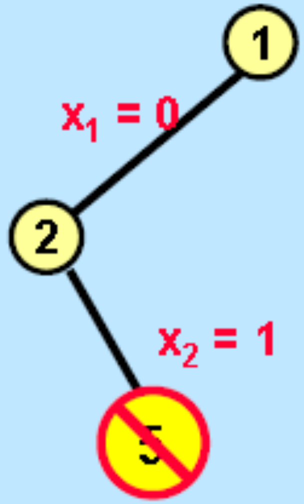
max $\quad z_{LP}(5) = 16x_1 + 22x_2 + 12x_3 + 8x_4 + 11x_5 + 19x_6$

s.t. $\quad 5x_1 + 7x_2 + 4x_3 + 3x_4 + 4x_5 + 6x_6 \leq 14$

$\quad x_1 = 0,\ x_2 = 1 \qquad 0 \leq x_j \leq 1$ for $j = 3$ to 6.

$Z_{LP}(5) = 44$. Found by solving the LP.

Can we eliminate node 5?



The incumbent solution has value z_I = 38

$z_{LP}(5) = 44$.

Possibly, some descendent of node 5 has a better solution value than 38.

<u>Conclusion</u>: we cannot stop enumerating solutions from node 5. We need to branch from node 5.

There would be no further branching from node 5 if z_I = 45.

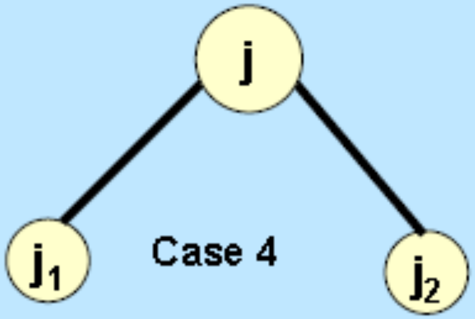
But suppose that we had an incumbent with z_I = 45.

Then no descendent of node 5 can be better than x_I. We can ***fathom*** node 5.

Branch and Bound overview

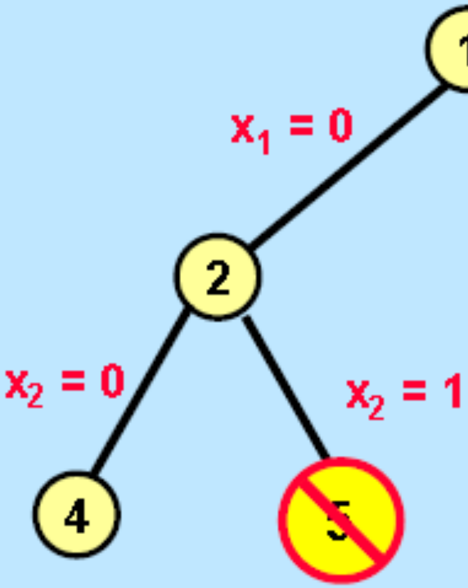
- Branch and bound creates the enumeration tree, one node at a time, and one branch at a time.
- Before branching on a node j, it solves LP(j). Depending on the solution to LP(j), Branch and Bound either fathoms node j or it branches on node j and creates two children.

Cases 1, 2, 3



Case 4

Branch and Bound: Case 1.



The incumbent solution has value $z_I = 38$

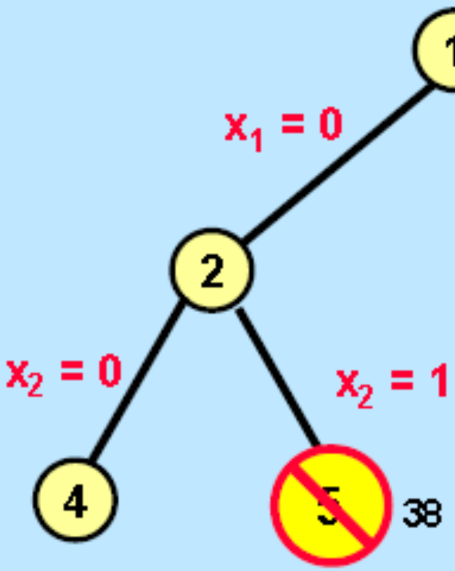
CASE 1. $z_{LP}(j) = -\infty$ **That is, the LP for Subproblem j is infeasible.**

Fathom Node j.

Do not search any of the subtree hanging from node j because none of these subproblems has a feasible solution.

Branch and Bound: Case 2.

$z_i = 38$



Assume all feasible integer solutions have integer objective value.

CASE 2. $-\infty < \lfloor z_{LP}(j) \rfloor \le z_i$

Then $z_{IP}(j) \le \lfloor z_{LP}(j) \rfloor \le z_i$

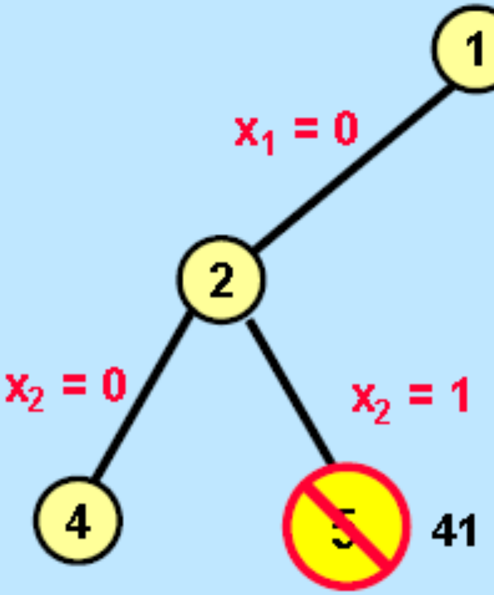
Fathom node j

No subproblem in the subtree hanging from node 5 has a solution that is better than the incumbent.

Bound(5) = 38

e.g. Suppose $z_{LP}(5) = 38.7$

Branch and Bound: Case 3.



$z_i = 38$

CASE 3. $z_{LP}(j) > z_i$ and the optimal solution for LP(j) is feasible for the IP.

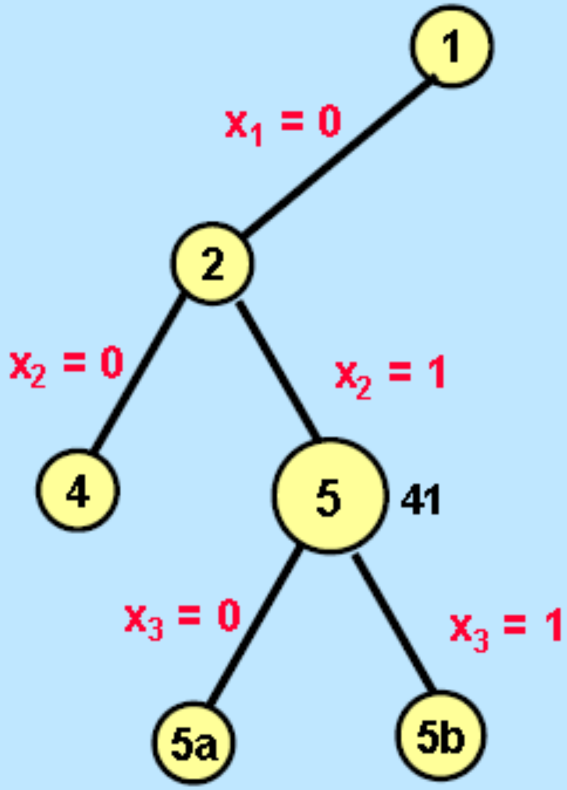
In this case, we first replace the incumbent by the integral for LP(j), which is feasible for the IP. No descendent of node 5 can be better. So we can ***fathom node j***.

e.g. Suppose the opt solution for LP(5) was

$x_1 = 0$, $x_2 = 1$, $x_3 = 0$, $x_4 = 1$, $x_5 = 1$, $x_6 = 0$, $z = 41$

$z_i = 41$

Branch and Bound: Case 4.



Bound(5) = 41

$z_i = 38$

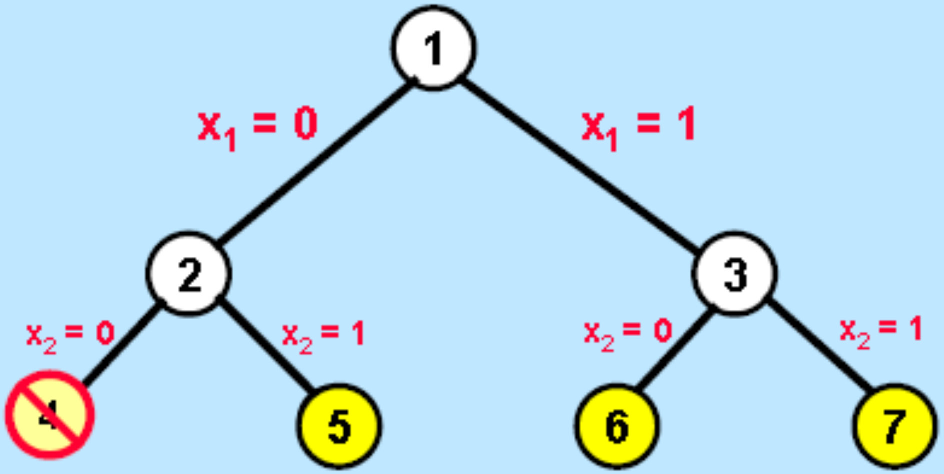
CASE 4. $\lfloor z_{LP}(j) \rfloor > z_i$ **and the optimal solution for LP(j) is not feasible for the IP.**

In this case, we cannot fathom node 5. Instead we add its two "children" to our (growing) tree and continue the branch and bound algorithm.

e.g. Suppose $z_{LP}(5) = 41$, but the solution for the relaxation is not feasible for the IP.

3

Active nodes



A node is called *active* if it has no children and it has not yet been fathomed. The active nodes are 5, 6, 7.

Initially, the only active node is node 1. The algorithm ends when there are no active nodes.

Branch and Bound for 0-1 Integer Programs

For the application of branch and bound algorithm to this problem, a document named “Branch And Bound Example for 0-1 Problems” will be sent out.

$$\begin{aligned} &\text{maximize} \quad 16x_1 + 22x_2 + 12x_3 + 8x_4 + 11x_5 + 19x_6 \\ &\text{subject to} \quad 5x_1 + 7x_2 + 4x_3 + 3x_4 + 4x_5 + 6x_6 \le 14 \\ &\qquad\qquad x_j \text{ binary for } j = 1 \text{ to } 6 \end{aligned}$$

Branch and Bound Algorithm

- <u>INITIALIZE</u> Active = {1} -- node 1 is the original problem
  Incumbent: = Ø (or some heuristic finds an incumbent)
- <u>SELECT:</u>
  - If Active = Ø, then the Incumbent is optimal if it exists, and the problem is infeasible if no incumbent exists;
  - else, let j be a node from Active. Remove j from Active.

CASE 1. $z_{LP}(j) = -\infty$. Then fathom node j.

CASE 2. $-\infty < \lfloor z_{LP}(j) \rfloor \leq z_i$. Then fathom node j.

CASE 3. $z_{LP}(j) > z_i$ and the optimal solution for LP(j) is feasible for the IP. Then fathom node j, and replace the incumbent with this new solution.

CASE 4. $\lfloor z_{LP}(j) \rfloor > z_i$ and the optimal solution for LP(j) is not feasible for the IP. Then create two children for node j.

B & B for Pure Integer Programs

maximize $8x_1 + 5x_2$

subject to $x_1 + x_2 \leq 6$

$9x_1 + 5x_2 \leq 45$

$x_1, x_2 \geq 0, \quad x_1, x_2$ integer

The first node of the B & B tree

41 1

There is no initial incumbent.

$z_I = -\infty$

maximize $8x_1 + 5x_2$

subject to $x_1 + x_2 \le 6$

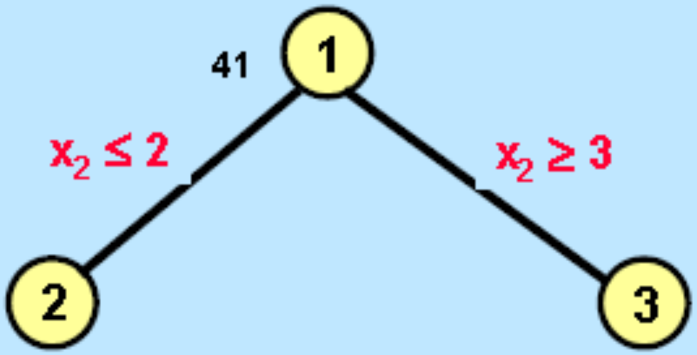
$9x_1 + 5x_2 \le 45$

$x_1, x_2 \ge 0,$

LP(1)

Optimal solution to LP(1) is
$x_1 = 3.75$, $x_2 = 2.25$, $z_{LP}(1) = 41.25$.

This is Case 4 of B&B. Create two children.

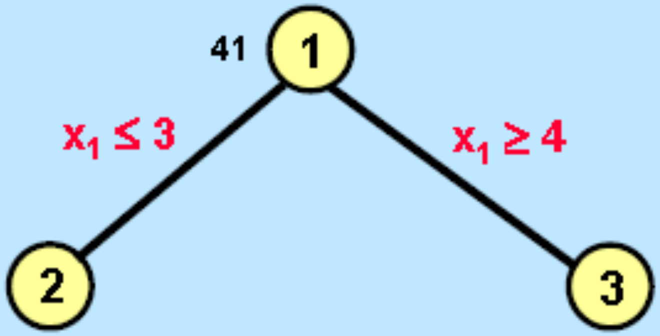


Note. can create subproblems any way that we want, so long as eventually every solution would be enumerated if we did not fathom.

That is, no feasible solution to the integer program ever gets eliminated by branching. It will be feasible for one of the branches.

Node 2 of the B&B Tree

$z_I = -\infty$



Optimal solution to LP(2) is
$x_1 = 3$, $x_2 = 3$, $z_{LP}(2) = 39$.

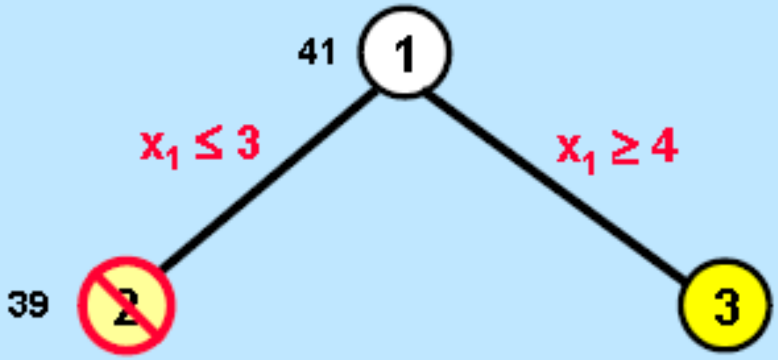
Standard branching choice: create children so that the solution for LP(1) is not feasible for either subproblem.

This gives us a first integer solution.
It is Case 3 of B&B.

We replace the incumbent, and fathom the node.

Node 3 of the B & B tree

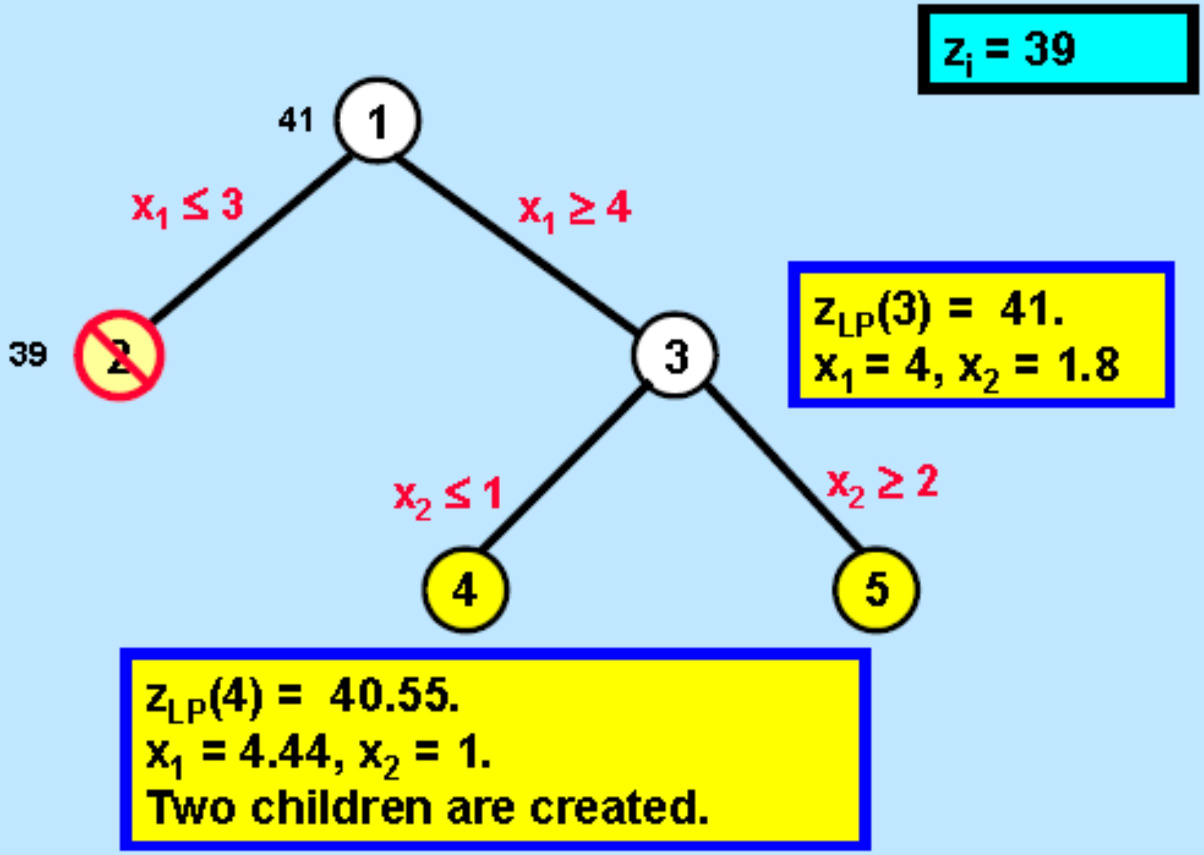
$z_I = 39$



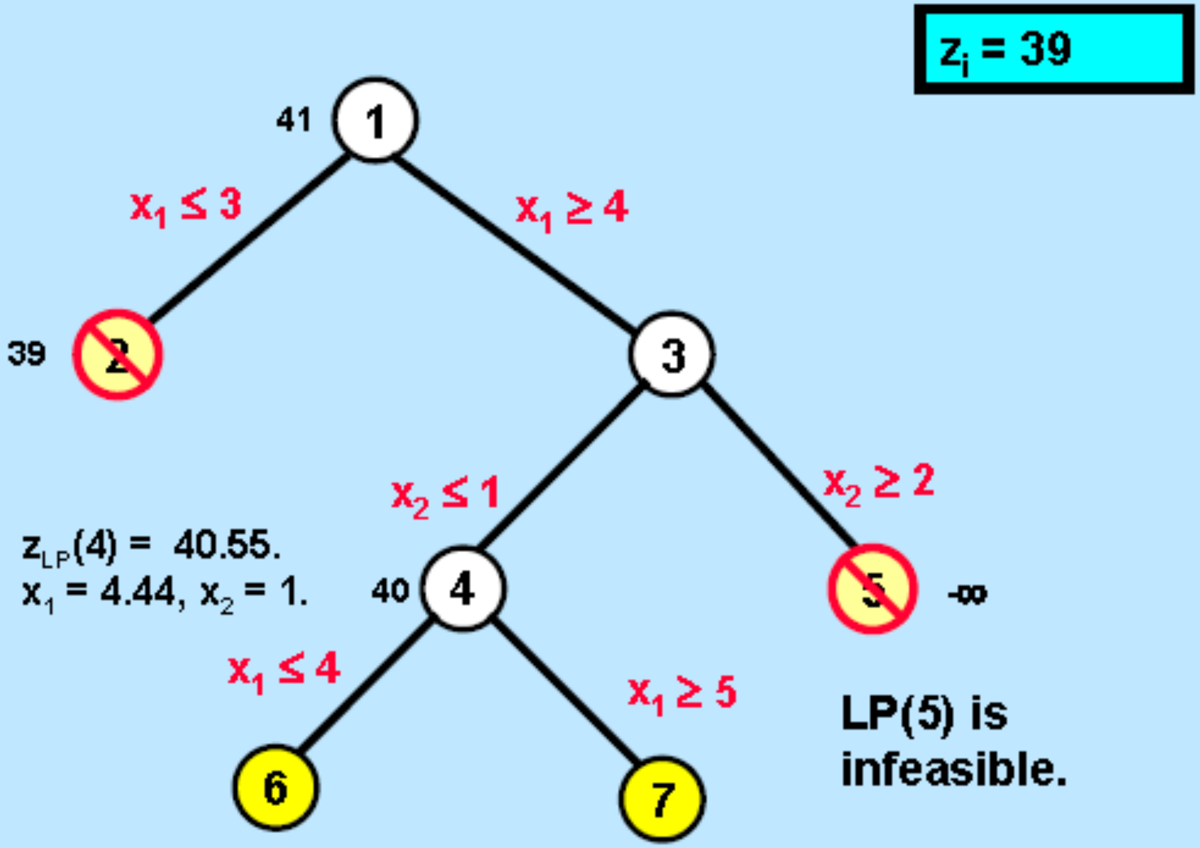
$z_{LP}(3) = 41.$ $x_1 = 4, x_2 = 1.8$

This is case 4 of B&B. We create two children.

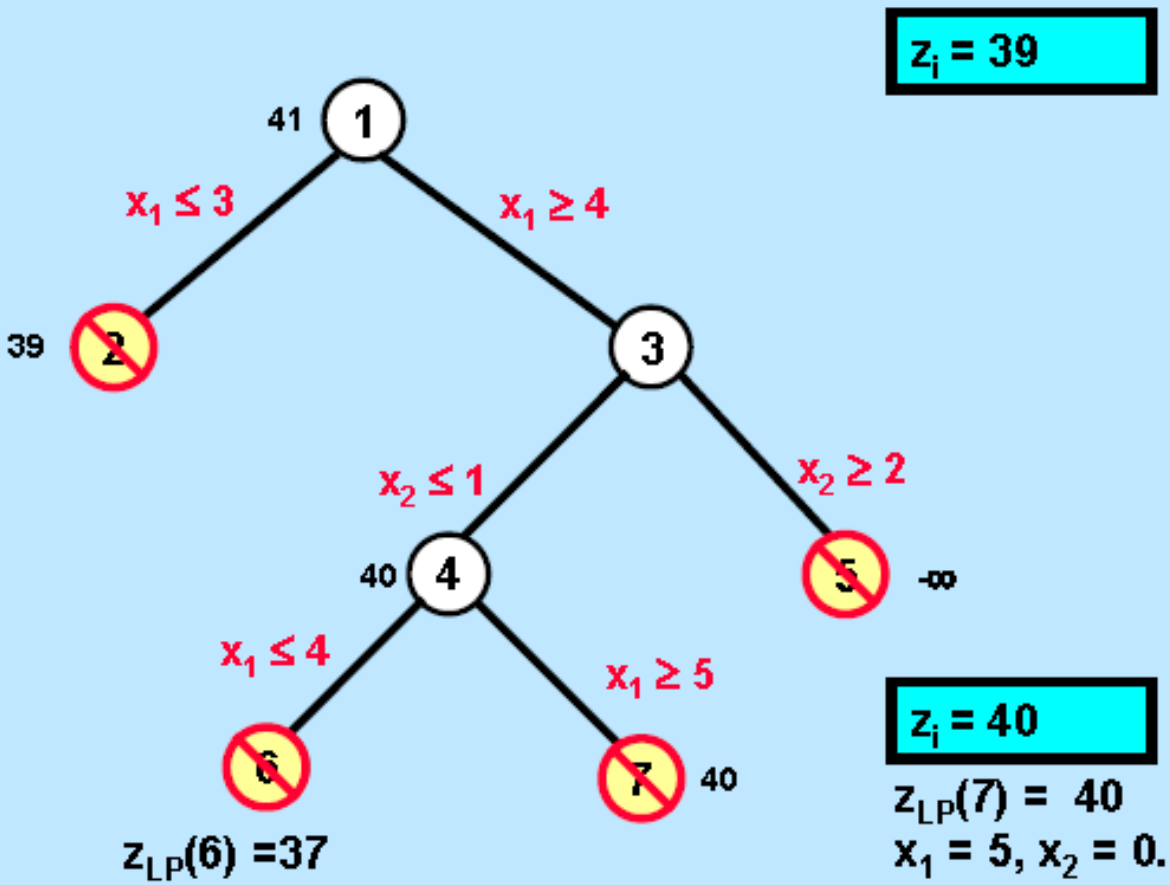
Node 4 of the B & B tree



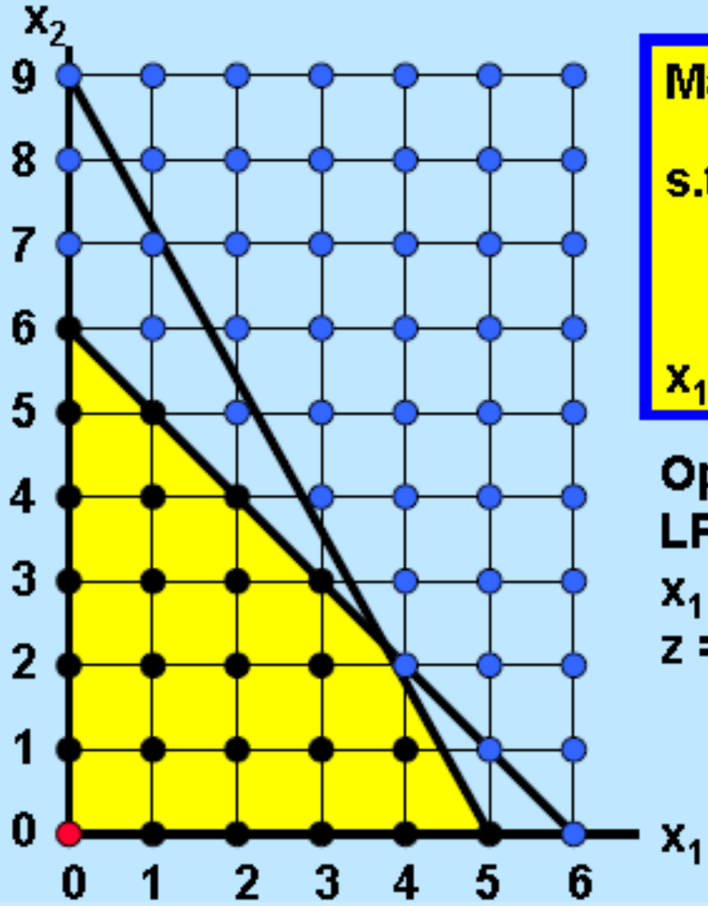
Node 5 of the B & B tree



Node 6 and 7 of the B & B tree



The Graphical Representation



Max $8x_1 + 5x_2$

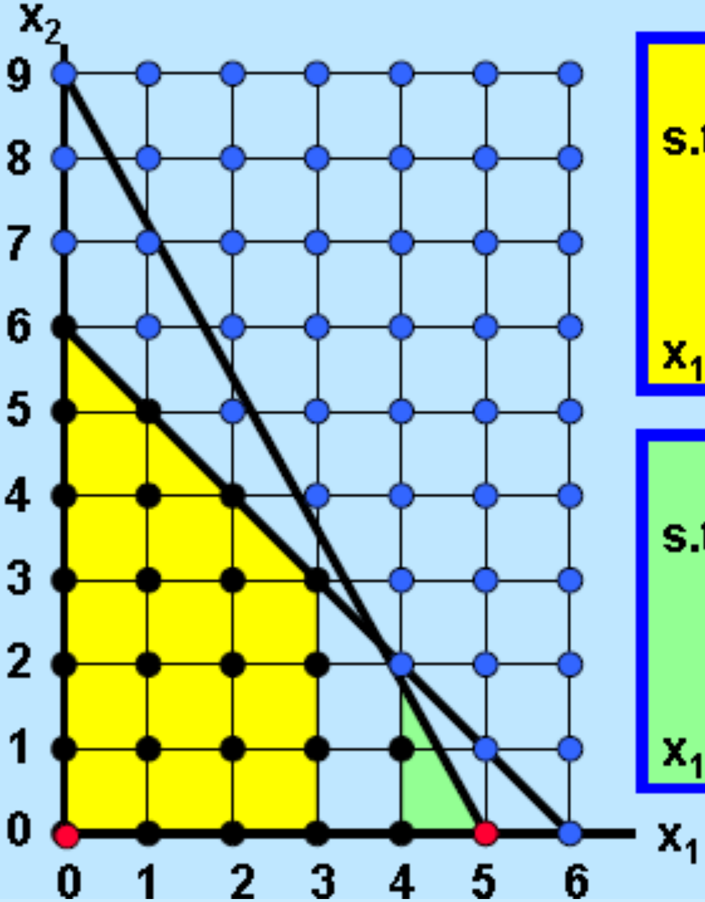
s.t $\quad x_1 + x_2 \le 6$

$9x_1 + 5x_2 \le 45$

$x_1, x_2 \ge 0, \quad x_1, x_2$ integer

Optimal solution to the LP relaxation is
$x_1 = 3.75, \; x_2 = 2.25$
$z = 41.25$

Subproblems 1 and 2



Max $8x_1 + 5x_2$

s.t $x_1 + x_2 \le 6$

$9x_1 + 5x_2 \le 45$

$x_1 \le 3$

$x_1, x_2 \ge 0$, x_1, x_2 integer

Max $8x_1 + 5x_2$

s.t $x_1 + x_2 \le 6$

$9x_1 + 5x_2 \le 45$

$x_1 \ge 4$

$x_1, x_2 \ge 0$, x_1, x_2 integer

Branch and Bound in General

Notation:

- *Incumbent*: the best solution on hand
  - Its value is z_I
- $z_{IP}(j)$ the optimum value for subproblem j
- $z_{LP}(j)$: value of the LP relaxation of node j
- Bound(j). In cases in which we are maximizing an din which integer solutions have integer objective values, we let Bound(j) = $\lfloor z_{LP}(j) \rfloor$
- *Children of a node*: the two problems created for a node that such that every solution that is a descendent of the original node is also a descendent of one of its children.
- Active: the collection of *active* (not fathomed) subproblems.

On Branch and Bound

- Branch and Bound is a standard way of solving integer programs.
- Lots of art and engineering can go into making it work effectively
- Next: a couple of issues that arise.

Different Branching and Bounding Rules are Possible

- The more accurate the bound, the quicker the fathoming.
  - The better the bounding technique, the more accurate the bound and the quicker the fathoming.
  - The quicker that the important decisions are resolved, the quicker the fathoming
- The better the incumbent, the quicker the fathoming.