Fall 2016, MATH-566

Integer Programming - Solution *Methods* - Branch and Bound

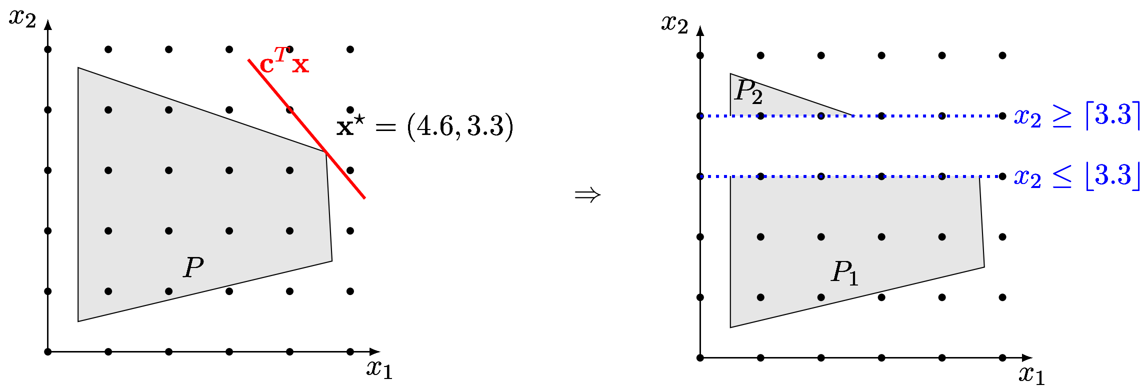
Source: `http://co-at-work.zib.de/files/Gurobi_MIP.pdf`

Problem:

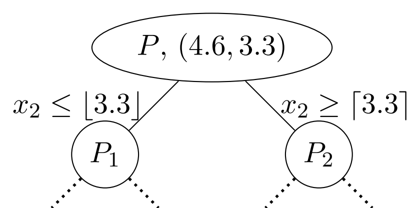
$$(IP)\begin{cases}\text{maximize} & \mathbf{c}^T\mathbf{x}\\ \text{subject to} & A\mathbf{x}\le\mathbf{b},\end{cases}$$

where $\mathbf{c}\in\mathbb{Z}^n, \mathbf{b}\in\mathbb{Z}^m, A\in\mathbb{Z}^{m\times n}$, and $\mathbf{x}\in\mathbb{Z}^n$.

Suppose we try to relax the problem and solve it as a linear programming problem. The set of feasible solutions is P. Suppose that the optimum is $\mathbf{x}^\star=(4.6,3.3)$. We know x_2 cannot be 3.3. So we create two new instances, where we add constraints $x_2\ge\lceil 3.3\rceil$ and $x_2\le\lfloor 3.3\rfloor$. Variable x_2 is a*branch variable*. We solve both instances and better of the solutions is the solution to the original problem.



The same process repeats with P_1 and P_2. Result is a *big* branch and bound tree T.



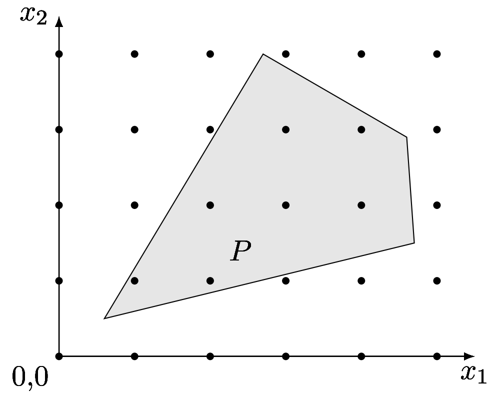
Branch and (no Bound) outline

1. Let $P=\{\mathbf{x}: A\mathbf{x}\le\mathbf{b}\}$
2. Build tree T with one node P (and mark it unexplored)
3. while T has unexplored node X
4. $\quad$ $\mathbf{x}^\star :=$ optimum for LP relaxation of X; mark X explored
5. $\quad$ If $\mathbf{x}_i^\star\notin\mathbb{Z}$ for some i
6. $\quad\quad$ $X_1 := X\cap\{\mathbf{x}:\mathbf{x}_i\le\lfloor\mathbf{x}_i^\star\rfloor\}$
7. $\quad\quad$ $X_2 := X\cap\{\mathbf{x}:\mathbf{x}_i\le\lceil\mathbf{x}_i^\star\rceil\}$
8. $\quad\quad$ Add X_1 and X_2 to T as unexplored nodes
9. Return maximum of integer solutions in T.

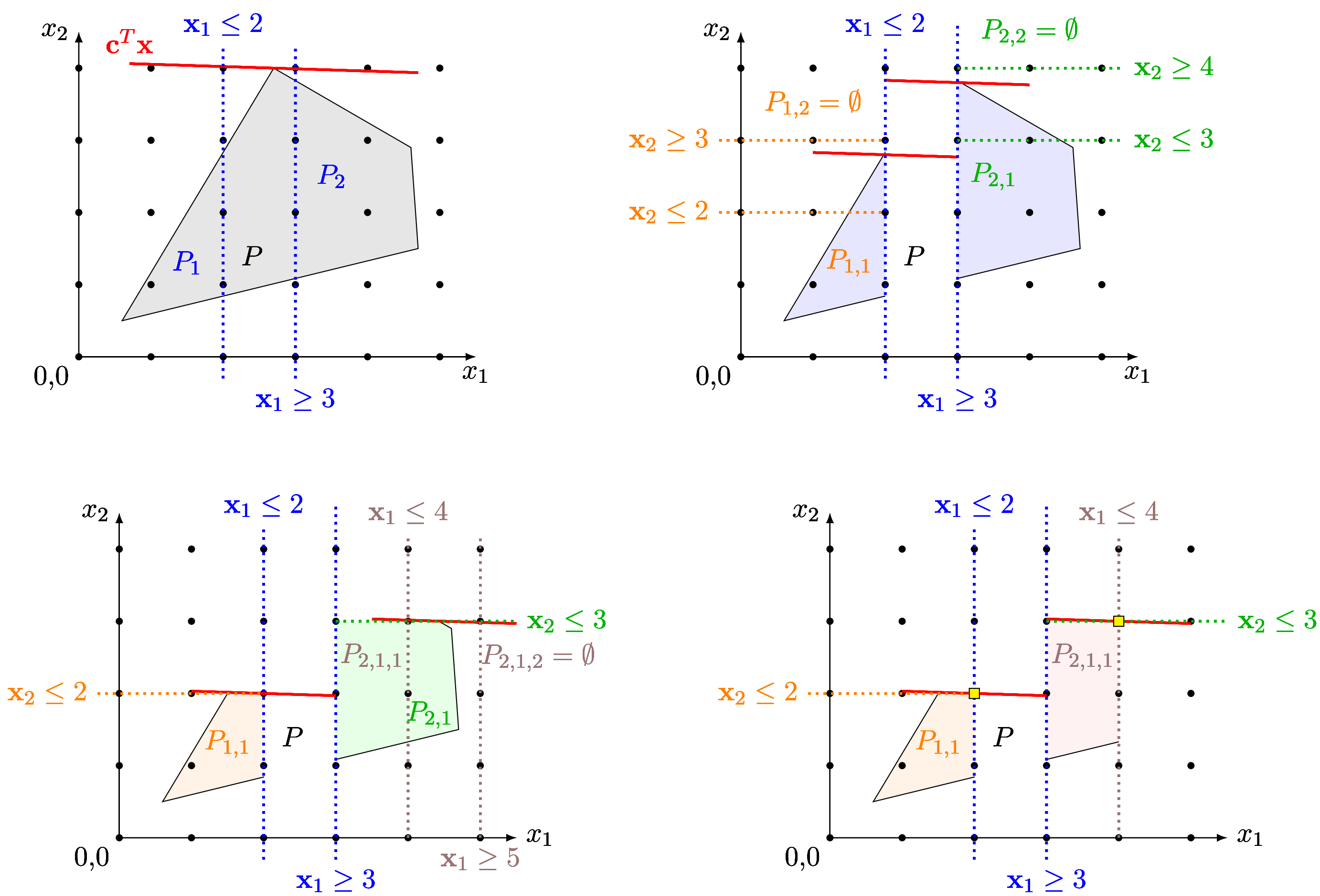
1: Consider problem

$$(IP)\begin{cases}\text{maximize} & 100x_2 + x_1\\ \text{subject to} & (x_1, x_2) \in P,\end{cases}$$

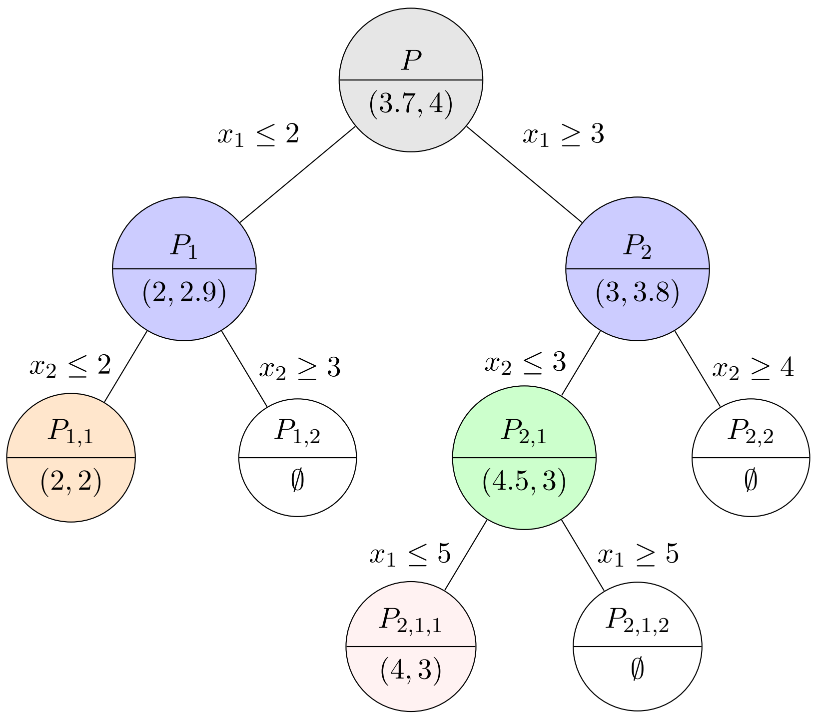
where P is depicted below. Solve (IP) using Brand and Bound. Create branch and bound tree T.



Solution: Here is the sequence of cuttings.



And this is the resulting tree T.



Notice that the leaves have either integer solution or are empty. Also notice that there is more than just one branching on x_1. And the optimum solution is $(4, 3)$, value 304.

2: Will branch and bound ALWAYS find an optimal solution if one exists?

Solution: Yes, this EVENTUALLY gets the right answer.

3: Is there a *good* bound on the size of the tree?

Solution: No - the tree may explode. It may have exponential size.

4: Is it possible to identify nodes in T that will not contain the optimal solution?

Solution: Sometimes. See the example above. Consider we computed node $P_{2,1,1}$ and get an integer solution of value 304. This tells us that the optimum integral solution has value at least 304. Now we look at node P_1 - it gives solution with value 292. In the whole subtree under P_1, all integer solutions in the subtree rooted at P_1 will have value at most 292. Hence no need to solve under P_1. **That is why the method is branch and bound** Note: good idea to try to round and get some integers solutions - helps cut the tree. This is the bound part of the name.

5: What are (dis)advantages of processing nodes deep in the search tree vs nodes close to the root?

Solution: Deep is more likely to give integer solution. But more likely to be eliminated later by some better solution. No clear winner.

6: Which if a solution in a node has more non-integer coordinates, which variable to branch on first?

Solution: Depends on problem - branch on important first. Example - decide if building factory at all before deciding how many production lines it should have.

Next time: Cutting Planes for Integer Programming.