15.053 Tuesday, April 9

- Branch and Bound

Handouts: Lecture Notes

Overview of Techniques for Solving Integer Programs

- **Enumeration Techniques**
  - Complete Enumeration
    - list all "solutions" and choose the best
  - Branch and Bound
    - Implicitly search all solutions, but cleverly eliminate the vast majority before they are even searched
  - Implicit Enumeration
    - Branch and Bound applied to binary variables
- **Cutting Plane Techniques**
  - Use LP to solve integer programs by adding constraints to eliminate the fractional solutions.

Capital Budgeting Example

Investment budget = $14,000

Investment	1	2	3	4	5	6
Cash Required (1000s)	$5	$7	$4	$3	$4	$6
NPV added (1000s)	$16	$22	$12	$8	$11	$19

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

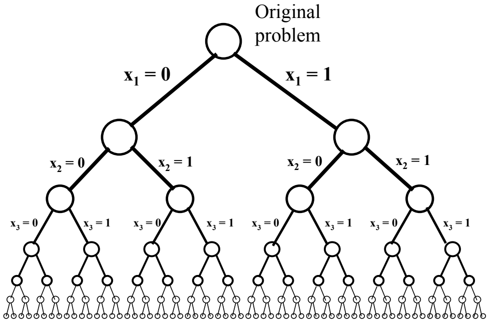
subject to $5x_1 + 7x_2 + 4x_3 + 3x_4 + 4x_5 + 6x_6 \leq 14$

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



On complete enumeration

- **Suppose that we could evaluate 1 billion solutions per second.**
- **Let n = number of binary variables**
- **Solutions times**
  - n = 30, 1 second
  - n = 40, 17 minutes
  - n = 50 11.6 days
  - n = 60 31 years

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

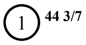
– n = 70,	1 second
– n = 80,	17 minutes
– n = 90	11.6 days
– n = 100	31 years

Branch and Bound

The essential idea: search the enumeration tree, but at each node

1. Solve the linear program at the node
2. Eliminate the subtree (fathom it) if
   1. The solution is integer (there is no need to go further) or
   2. The best solution in the subtree cannot be as good as the best available solution (the incumbent) or
   3. There is no feasible solution

Branch and Bound



Node 1 is the original LP Relaxation

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

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

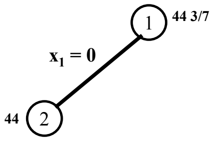
$0 \le x_j \le 1$ for j = 1 to 6

Solution at node 1:

$x_1 = 1$ $x_2 = 3/7$ $x_3 = x_4 = x_5 = 0$ $x_6 = 1$ $z = 44\ 3/7$

The IP cannot have value higher than 44 3/7.

Branch and Bound



Node 2 is the original LP Relaxation plus the constraint $x_1 = 0$.

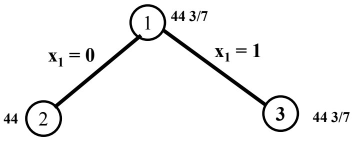
maximize $16x_1 + 22x_2 + 12x_3 + 8x_4 + 11x_5 + 19x_6$

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

$0 \le x_j \le 1$ for j = 1 to 6, $x_1 = 0$

Solution at node 2:
$x_1 = 0$ $x_2 = 1$ $x_3 = 1/4$ $x_4 = x_5 = 0$ $x_6 = 1$ $z = 44$

Branch and Bound



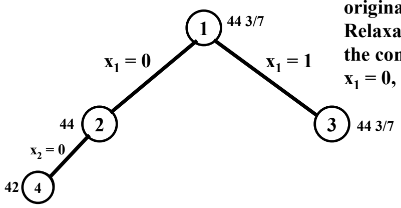
Node 3 is the original LP Relaxation plus the constraint $x_1 = 1$.

The solution at node 1 was

$x_1 = 1$ $x_2 = 3/7$ $x_3 = x_4 = x_5 = 0$ $x_6 = 1$ $z = 44\ 3/7$

Note: it was the best solution with no constraint on x_1. So, it is also the solution for node 3. (If you add a constraint, and the old optimal solution is feasible, then it is still optimal.)

Branch and Bound



Node 4 is the original LP Relaxation plus the constraints $x_1 = 0$, $x_2 = 0$.

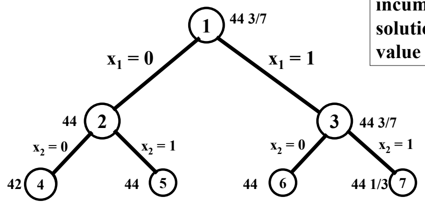
Solution at node 4: 0 0 1 0 1 1 z = 42

Our first incumbent solution!

No further searching from node 4 because there cannot be a better integer solution.

Branch and Bound

The incumbent solution has value 42

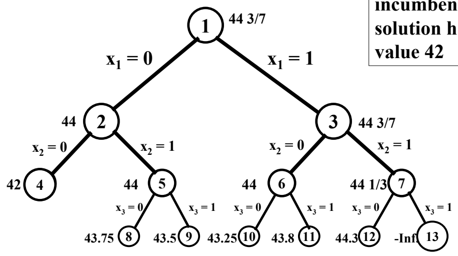


We next solved the LP's associated with nodes 5, 6, and 7

No new integer solutions were found.

Branch and Bound

The incumbent solution has value 42



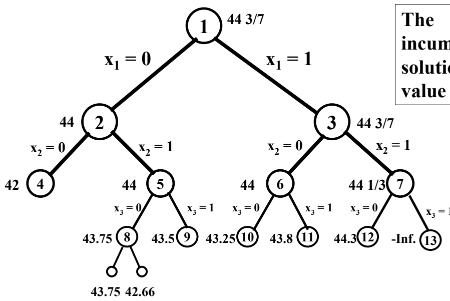
We next solved the LP's associated with nodes 8 -13

Summary so far

- We have solved 13 different linear programs so far.
  - One integer solution found
  - One subtree *fathomed* (*pruned*) because the solution was integer (node 4)
  - One subtree fathomed because the solution was infeasible (node 13)
  - No subtrees fathomed because of the bound

Branch and Bound

The incumbent solution has value 42



We next solved the LP's associated with the next nodes.

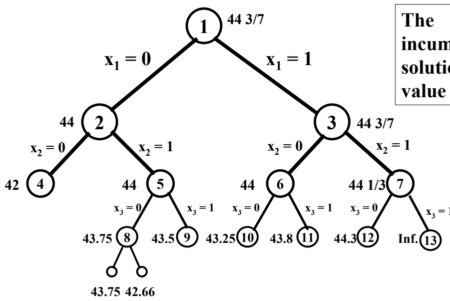
We can fathom the node with z = 42.66. Why?

Getting a better bound

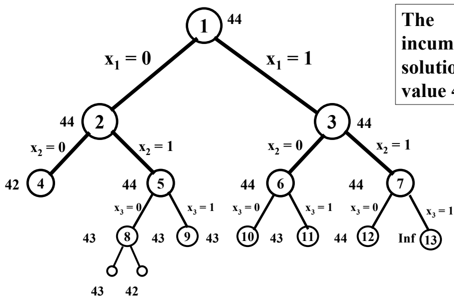
- The bound at each node is obtained by solving an LP.
- But we know the best integer solution has an integer objective value.
- If the best integer valued solution for a node is at most 42.66, then we know the best bound is at most 42.
- Other bounds can also be rounded down.

Branch and Bound

The incumbent solution has value 42

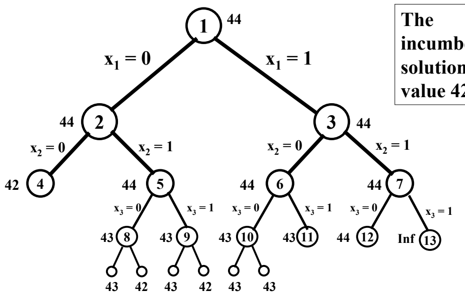


Branch and Bound



The incumbent solution has value 42

Branch and Bound

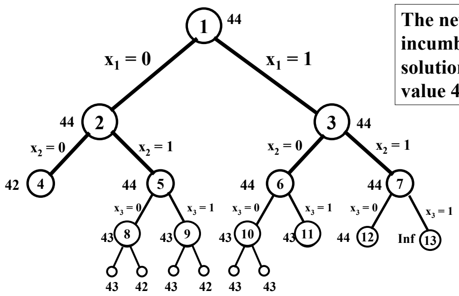


The incumbent solution has value 42

We found a new incumbent solution!

$x_1 = 1, x_2 = x_3 = 0, x_4 = 1, x_5 = 0, x_6 = 1 \quad z = 43$

Branch and Bound

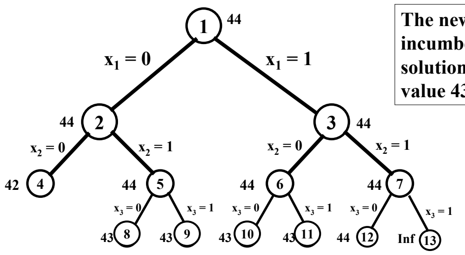


The new incumbent solution has value 43

We found a new incumbent solution!

$x_1 = 1, x_2 = x_3 = 0, x_4 = 1, x_5 = 0, x_6 = 1 \quad z = 43$

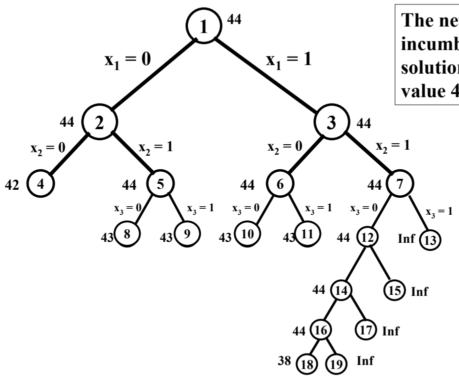
Branch and Bound



The new incumbent solution has value 43

If we had found this incumbent earlier, we could have saved some searching.

Finishing Up



The new incumbent solution has value 43

Lessons Learned

- Branch and Bound can speed up the search
  - Only 25 nodes (linear programs) were evaluated
  - Other nodes were fathomed
- Obtaining a good incumbent earlier can be valuable
  - only 19 nodes would have been evaluated.
- Solve linear programs faster, because we start with an excellent or optimal solution
  - uses a technique called the dual simplex method
- Obtaining better bounds can be valuable.
  - We sometimes use properties that are obvious to us, such as the fact that integer solutions have integer solution values

Branch and Bound

Notation:

- z^* = optimal integer solution value
- ***Subdivision***: a node of the B&B Tree
- ***Incumbent***: the best solution on hand
- z^I: value of the incumbent
- z^{LP}: value of the LP relaxation of the current node
- LIST: the collection of ***active*** (not fathomed) nodes
- ***Children of a node***: the two problems created for a node, e.g., by saying $x_j = 1$ or $x_j = 0$.

Initialization: LIST = {original problem}
Incumbent: = ∅
$z^I = -\infty$

Branch and Bound Algorithm

INITIALIZE

SELECT:

If LIST = ∅, then the Incumbent is optimal if it exists, and the problem is infeasible if no incumbent exists;
else, let S be a subdivision from LIST.
Let x^{LP} be the optimal solution to S
Let z^{LP} = its objective value

CASE 1. $z^{LP} = -\infty$ (the LP is infeasible)
Remove S from LIST (fathom it)
Return to SELECT

Branch and Bound Algorithm

INITIALIZE

SELECT:

If LIST = ∅, then the Incumbent is optimal (if it exists), and the problem is infeasible if no incumbent exists;
else, let S be a subdivision from LIST.
Let x^{LP} be the optimal solution to S
Let z^{LP} = its objective value

CASE 2. $-\infty < z^{LP} <= z^I$.
That is, the LP is dominated by the incumbent)

Then remove S from LIST (fathom it)
Return to SELECT

Branch and Bound Algorithm

INITIALIZE

SELECT:

If LIST = ∅, then the Incumbent is optimal (if it exists), and the problem is infeasible if no incumbent exists;
else, let S be a subdivision from LIST.
Let x^{LP} be the optimal solution to S
Let z^{LP} = its objective value

CASE 2. $-\infty < z^{LP} <= z^I$.
That is, the LP is dominated by the incumbent)

Then remove S from LIST (fathom it)
Return to SELECT

Branch and Bound Algorithm

INITIALIZE

SELECT:

If LIST = ∅, then the Incumbent is optimal (if it exists), and the problem is infeasible if no incumbent exists;
else, let S be a subdivision from LIST.
Let x^{LP} be the optimal solution to S
Let z^{LP} = its objective value

CASE 3. $z^I < z^{LP}$ and x^{LP} is integral.
That is, the LP solution is integral and dominates the incumbent.

Then Incumbent := x^{LP};
$z^I := z^{LP}$
Remove S from LIST (fathomed by integrality)
Return to SELECT

Branch and Bound Algorithm

INITIALIZE

SELECT:

If LIST = ∅, then the Incumbent is optimal (if it exists), and the problem is infeasible if no incumbent exists;
else, let S be a subdivision from LIST.
Let x^{LP} be the optimal solution to S
Let z^{LP} = its objective value

CASE 4. $z^I < z^{LP}$ and x^{LP} is not integral.
There is not enough information to fathom S

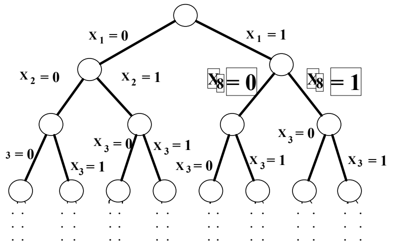
Remove S from LIST
Add the children of S to LIST
Return to SELECT

Different Selection Rules are Possible

- Rule of Thumb 1: Don’t let LIST get too big (the solutions must be stored). So, prefer nodes that are further down in the tree.
- Rule of Thumb 2: Pick a node of LIST that is likely to lead to an improved incumbent. Sometimes special heuristics are used to come up with a good incumbent.

Branching

One does not have to have the B&B tree be symmetric, and one does not select subtrees by considering variables in order.



Choosing how to branch so as to reduce running time is largely “art” and based on experience.

Different Branching Rules are Possible

- Branching: determining children for a node. There are many choices.
- Rule of thumb 1: if it appears clear that $x_j = 1$ in an optimal solution, it is often good to branch on $x_j = 0$ vs $x_j = 1$.
  - The hope is that a subdivision with $x_j = 0$ can be pruned.
- Rule of thumb 2: branching on important variables is worthwhile
  - e.g., in the location problem, branch on the plant location variables first

Different Bounding Techniques are Possible

- We use the bound obtained by dropping the integrality constraints (LP relaxation). There are other choices.
- Key tradeoff for bounds: time to obtain a bound vs quality of the bound.
- If one can obtain a bound much quicker, sometimes we would be willing to get a bound that is worse
- It usually is worthwhile to get a bound that is better, so long as it doesn’t take too long (see next lecture)

What if the variables are general integer variables?

- One can choose children as follows:
  - child 1: $x_1 \leq 3$ (or $x_j \leq k$)
  - child 2 $x_1 \geq 4$ (or $x_j \geq k+1$)
- How would one choose the variable j and the value k
  - A common choice would be to take a fractional value from x^{LP}. e.g., if $x_7 = 5.62$, then we may branch on $x_7 \leq 5$ and $x_7 \geq 6$.
  - Other choices are also possible.

Summary

- Branch and Bound is the standard way of solving IPs to optimality.
- There is art to making it work well in practice.
- Much of the art is built into state-of-the-art solvers such as CPLEX.

A bad example for implicit enumeration

maximize $2x_1 + 2x_2 + 2x_3 + \ldots + 2x_{100}$

subject to $2x_1 + 2x_2 + 2x_3 + \ldots + 2x_{100} \leq 101$

$x_i \in \{0,1\}$ for i = 1 to 100.

Why is this a bad example? What would happen if we used branch and bound, as described earlier?