END3033 Operations Research I
Integer Programming

to accompany

Operations Research: Applications and Algorithms

Fatih Cavdur

Introduction

An IP in which all variables are required to be integers is called a pure integer programming problem. For example,

$$\max z = 3x_1 + 2x_2$$

$$\begin{array}{ccccc} x_1 & + & x_2 & \leq & 6 \\ x_1 & , & x_2 & \geq & 0 \\ x_1 & , & x_2 & \in & \mathbb{Z} \end{array}$$

Introduction

An IP in which only some of the variables are required to be integers is called a mixed integer programming problem. For example,

$$\max z = 3x_1 + 2x_2$$

$$\begin{array}{ccccc} x_1 & + & x_2 & \leq & 6 \\ x_1 & , & x_2 & \geq & 0 \\ x_1 & & & \in & \mathbb{Z} \end{array}$$

Introduction

An integer programming problem in which all the variables must equal 0 or 1 is called a 0 - 1 (binary) IP.

$$\max z = x_1 - x_2$$

$$\begin{array}{rcrcc} x_1 & + & 2x_2 & \leq & 2 \\ 2x_1 & - & x_2 & \leq & 1 \\ x_1 & , & x_2 & \in & \{0,1\} \end{array}$$

Introduction

Definition: LP Relaxation

The LP obtained by omitting all integer or 0–1 constraints on variables is called the LP relaxation (LPR) of the IP.

IP	LP Relaxation
$\max z = 3x_1 + 2x_2$	$\max z = 3x_1 + 2x_2$
$x_1 + x_2 \leq 6$	$x_1 + x_2 \leq 6$
$x_1 , x_2 \geq 0$	$x_1 , x_2 \geq 0$
$x_1 , x_2 \in \mathbb{Z}$	

Introduction

Any IP may be viewed as the LP relaxation (LPR) plus additional constraints (the constraints that state which variables must be integers or be 0 or 1). Hence, the LP relaxation is a less constrained, or more relaxed, version of the IP. This means that the feasible region for any IP must be contained in the feasible region for the corresponding LPR. For any IP that is a max problem, this implies that

$$\begin{pmatrix} \text{optimal} \\ z \\ \text{for} \\ \text{LPR} \end{pmatrix} \geq \begin{pmatrix} \text{optimal} \\ z \\ \text{for} \\ \text{IP} \end{pmatrix}$$

Introduction

Example:

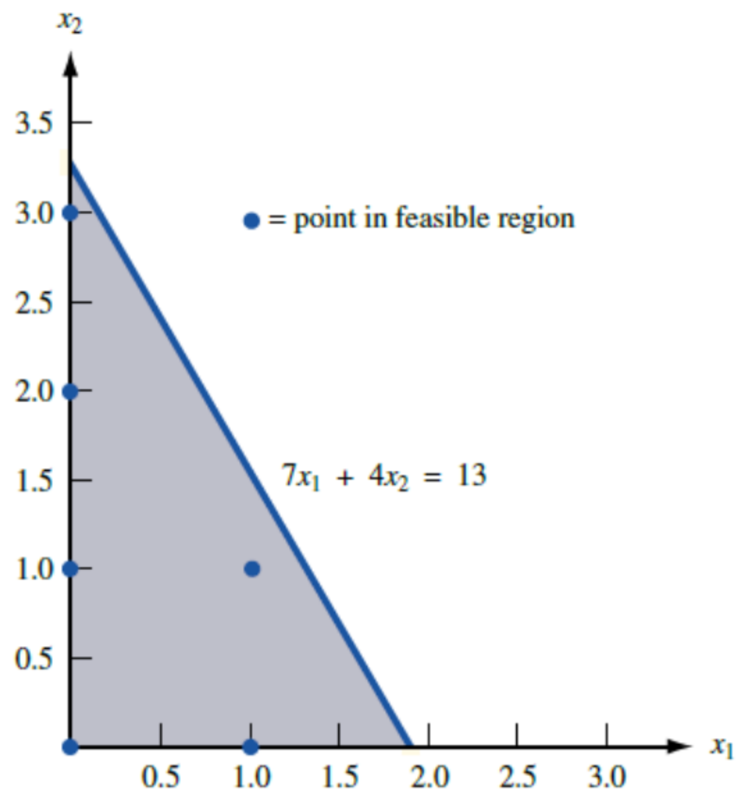
$$\max z = 21x_1 + 11x_2$$

$$\begin{array}{ccccc} 7x_1 & + & 4x_2 & \leq & 13 \\ x_1 & , & x_2 & \geq & 0 \\ x_1 & , & x_2 & \in & \mathbb{Z} \end{array}$$

The feasible set consists of the following points:

$$S = \{(0,0), (0,1), (0,2), (0,3), (1,0), (1,1)\}$$

Introduction



Introduction

If the feasible region for a pure IP's LPR is bounded, then the feasible region for the IP will consist of a finite number of points. In theory, such an IP could be solved by enumerating the objective values for each feasible point. The problem with this approach is that most actual IPs have feasible regions consisting of billions of feasible points. In such cases, a complete enumeration of all feasible points would require a large amount of computer time.

Introduction

We see that the optimal solution for the IP is $x_1 = 0$ and $x_2 = 3$, and the optimal solution for the LPR is $x_1 = 13/7$ and $x_2 = 0$. Rounding this solution, we obtain $x_1 = 2$ and $x_2 = 0$ which is not feasible, and hence, is not the optimal solution for the IP. Even if we round down x_1, we obtain $x_1 = 1$ and $x_2 = 0$ which is not optimal either.

In summary, even though the feasible region for an IP is a subset of the feasible region for the IP's LPR, the IP is usually much more difficult to solve than the IP's LPR.

IP Formulations

Knapsack Problem

Example:

Stockco is considering four investments. Investment 1 will yield a net present value (NPV) of $16,000; investment 2, an NPV of $22,000; investment 3, an NPV of $12,000; and investment 4, an NPV of $8,000. Each investment requires a certain cash outflow at the present time: investment 1, $5,000; investment 2, $7,000; investment 3, $4,000; and investment 4, $3,000. Currently, $14,000 is available for investment. Formulate an IP whose solution will tell Stockco how to maximize the NPV obtained from investments 1–4.

IP Formulations

Knapsack Problem

If we let

$$x_i = \begin{cases} 1 & \text{if we insvest in investment } i \\ 0 & \text{otherwise} \end{cases}$$

We can then write the following IP:

$$\max z = 16x_1 + 22x_2 + 12x_3 + 8x_4$$

$$\begin{array}{ccccccccc} 5x_1 & + & 7x_2 & + & 4x_3 & + & 3x_4 & \leq & 14 \\ x_1 & , & x_2 & , & x_3 & , & x_4 & \in & \{0,1\} \end{array}$$

IP Formulations

Example:

If we add the following restricitions for Stocko, modify the IP to satisfy these:

(R1) They can invest at most 2 investments.

(R2) If they invest in investment 2, they must also invest in investment 1.

(R3) If they invest in investment 2, they cannot invest in investment 4.

IP Formulations

We can then write the following IP:

$$\max z = 16x_1 + 22x_2 + 12x_3 + 8x_4$$

$$\begin{array}{rcrcrcrcll}
5x_1 & + & 7x_2 & + & 4x_3 & + & 3x_4 & \leq & 14 & \\
x_1 & + & x_2 & + & x_3 & + & x_4 & \leq & 2 & (R1) \\
-x_1 & + & x_2 & & & & & \leq & 0 & (R2) \\
 & & x_2 & & & + & x_4 & \leq & 1 & (R3) \\
x_1 & , & x_2 & , & x_3 & , & x_4 & \in & \{0,1\} &
\end{array}$$

IP Formulations

Fixed-Charge Problems

Example:

Gandhi Cloth Company is capable of manufacturing three types of clothing: shirts, shorts, and pants. The manufacture of each type of clothing requires that Gandhi have the appropriate type of machinery available. The machinery needed to manufacture each type of clothing must be rented at the following rates: shirt machinery, $200 per week; shorts machinery, $150 per week; pants machinery, $100 per week. The manufacture of each type of clothing also requires the amounts of cloth and labor shown in below table. Each week, 150 hours of labor and 160 sq. yd. of cloth are available. The variable unit cost and selling price for each type of clothing are shown in below table.

IP Formulations

Fixed-Charge Problems

Type	Labor (hours)	Cloth (sq. yd.)	Price ($)	Variable Cost ($)
Shirt	3	4	12	6
Shorts	2	3	8	4
Pants	6	4	15	8

IP Formulations

Fixed-Charge Problems

x_1 = number of shirts produced weekly
x_2 = number of shorts produced weekly
x_3 = number of pants produced weekly

$$y_1 = \begin{cases} 1, & \text{if any shirts are produced} \\ 0, & \text{otherwise} \end{cases}$$

$$y_2 = \begin{cases} 1, & \text{if any shorts are produced} \\ 0, & \text{otherwise} \end{cases}$$

$$y_3 = \begin{cases} 1, & \text{if any pants are produced} \\ 0, & \text{otherwise} \end{cases}$$

IP Formulations

Fixed-Charge Problems

$$\max z = 6x_1 + 4x_2 + 7x_3 - 200y_1 - 150y_2 - 100y_3$$

$$\begin{array}{ccccccccccccc}
3x_1 & + & 2x_2 & + & 6x_3 & & & & & & & \leq & 150 \\
4x_1 & + & 3x_2 & + & 4x_3 & & & & & & & \leq & 160 \\
x_1 & , & x_2 & , & x_3 & & & & & & & \geq & 0 \\
x_1 & , & x_2 & , & x_3 & & & & & & & \in & \mathbb{Z} \\
 & & & & & , & y_1 & , & y_2 & , & y_3 & \in & \{0,1\}
\end{array}$$

Is the above formulation correct?

IP Formulations

Fixed-Charge Problems

$$\max z = 6x_1 + 4x_2 + 7x_3 - 200y_1 - 150y_2 - 100y_3$$

$$\begin{array}{rcrcrcrcrcrcl}
3x_1 & + & 2x_2 & + & 6x_3 & & & & & & & \leq & 150 \\
4x_1 & + & 3x_2 & + & 4x_3 & & & & & & & \leq & 160 \\
x_1 & & & & & - & M_1 y_1 & & & & & \leq & 0 \\
 & & & & & & & - & M_2 y_2 & & & \leq & 0 \\
 & & & & & & & & & - & M_3 y_3 & \leq & 0 \\
x_1 & , & x_2 & , & x_3 & & & & & & & \geq & 0 \\
x_1 & , & x_2 & , & x_3 & & & & & & & \in & \mathbb{Z} \\
 & & & & & , & y_1 & , & y_2 & , & y_3 & \in & \{0,1\}
\end{array}$$

What about this one?

IP Formulations

Set Covering Problems

Example:

There are six cities (cities 1–6) in Kilroy County. The county must determine where to build fire stations. The county wants to build the minimum number of fire stations needed to ensure that at least one fire station is within 15 minutes (driving time) of each city. The times (in minutes) required to drive between the cities in Kilroy County are shown below. Formulate an IP that will tell Kilroy how many fire stations should be built and where they should be located.

IP Formulations

Set Covering Problems

From/To	C1	C2	C3	C4	C5	C6	Cities within 15 minutes
City 1	-	10	20	30	30	20	1,2
City 2	-	-	25	35	20	10	1,2,6
City 3	-	-	-	15	30	20	3,4
City 4	-	-	-	-	15	25	3,4,5
City 5	-	-	-	-	-	14	4,5,6
City 6	-	-	-	-	-	-	4,5,6

IP Formulations

Set Covering Problems

$$x_i = \begin{cases} 1, & \text{if a fire station is built in city } i \\ 0, & \text{otherwise} \end{cases}$$

$$\min z = x_1 + x_2 + x_3 + x_4 + x_5 + x_6$$

$$\begin{array}{ccccccccccccc}
x_1 & + & x_2 & & & & & & & & & \geq & 1 \\
x_1 & + & x_2 & & & & & & & + & x_6 & \geq & 1 \\
 & & & & x_3 & + & x_4 & & & & & \geq & 1 \\
 & & & & x_3 & + & x_4 & + & x_5 & & & \geq & 1 \\
 & & & & & & x_4 & + & x_5 & + & x_6 & \geq & 1 \\
 & & x_2 & & & & & + & x_5 & + & x_6 & \geq & 1 \\
x_1 & , & x_2 & , & x_3 & , & x_4 & , & x_5 & , & x_6 & \in & \{0,1\}
\end{array}$$

IP Formulations

Either-Or Constraints

If we have the following constraints

$$f(x_1, x_2, \dots, x_n) \leq 0 \vee f(x_1, x_2, \dots, x_n) \leq 0$$

By letting $y \in \{0,1\}$, we can write

$$f(x_1, x_2, \dots, x_n) \leq My$$
$$g(x_1, x_2, \dots, x_n) \leq M(1 - y)$$

IP Formulations

Either-Or Constraints-Example

Dorian Auto is considering manufacturing three types of autos: ompact, midsize, and large. The resources required for, and the profits yielded by, each type of car are shown in the Table below. Currently, 6,000 tons of steel and 60,000 hours of labor are available. For production of a type of car to be economically feasible, at least 1,000 cars of that type must be produced. Formulate an IP to maximize Dorian's profit.

Resource	Compact	Midsize	Large
Steel	1.5 tons	3 tons	5 tons
Labor	30 hours	25 hours	40 hours
Profit ($)	2,000	3,000	4,000

Table: Resources and Profits

IP Formulations

Either-Or Constraints-Example

We let

x_1 = # of compact cars produced

x_2 = # of midsize cars produced

x_3 = # of large cars produced

IP Formulations

Either-Or Constraints-Example

$$x_1 \leq M_1 y_1$$
$$1{,}000 - x_1 \leq M_1(1 - y_1)$$
$$x_2 \leq M_2 y_2$$
$$1{,}000 - x_2 \leq M_2(1 - y_2)$$
$$x_3 \leq M_3 y_3$$
$$1{,}000 - x_3 \leq M_3(1 - y_3)$$

$$1.5x_1 + 3x_2 + 5x_3 \leq 6{,}000$$
$$30x_1 + 25x_2 + 40x_3 \leq 60{,}000$$

IP Formulations

Either-Or Constraints-Example

$$x_i \in \mathbb{Z}, \quad \forall i$$
$$y_i \in \{0,1\}, \quad \forall i$$
$$x_i \geq 0, \quad \forall i$$

Note that we can substitute $M_1 = M_2 = 2{,}000$ and $M_2 = 1{,}200$. Why?

IP Formulations

If-Then Constraints

If we have the following two constraints f and g as

$$f(x_1, x_2, \dots, x_n) > 0 \Rightarrow g(x_1, x_2, \dots, x_n) \geq 0$$

By letting $y \in \{0,1\}$, we can write

$$\begin{aligned} f(x_1, x_2, \dots, x_n) &\leq M(1-y) \\ g(x_1, x_2, \dots, x_n) &\geq My \end{aligned}$$

IP Formulations

Piecewise Linear Functions

When we have a piecewise linear function $f(x)$, we can use the following approach to linearize it:

IP Formulations

Piecewise Linear Functions

Step 1) Replace $f(x)$ by $z_1 f(b_1) + z_2 f(b_2) + \cdots + z_n f(b_n)$

Step 2) Add the following constraints to the problem:

IP Formulations

Piecewise Linear Functions

$$
\begin{aligned}
z_1 &\leq y_1 \\
z_2 &\leq y_1 + y_2 \\
\cdots &\leq \cdots \\
z_{n-1} &\leq y_{n-2} + y_{n-1} \\
z_n &\leq y_{n-1}
\end{aligned}
$$

$$
\begin{aligned}
y_1 + y_2 + \cdots + y_{n-1} &= 1 \\
z_1 + z_2 + \cdots + z_n &= 1
\end{aligned}
$$

$$x = z_1 b_1 + z_2 b_2 + \cdots + z_n b_n$$

$$y_i \in \{0,1\}, \quad \forall i$$

$$z_i \geq 0, \quad \forall i$$

The Branch-and-Bound Method

In practice, most IPs are solved by using the technique of branch-and-bound by efficiently enumerating the points in a sub-problem's feasible region. Before explaining how branch-and-bound works, we need to make the following elementary but important observation:

If you solve the LPR of a pure IP and obtain a solution in which all variables are integers, then the optimal solution to the LPR is also the optimal solution to the IP.

The Branch-and-Bound Method

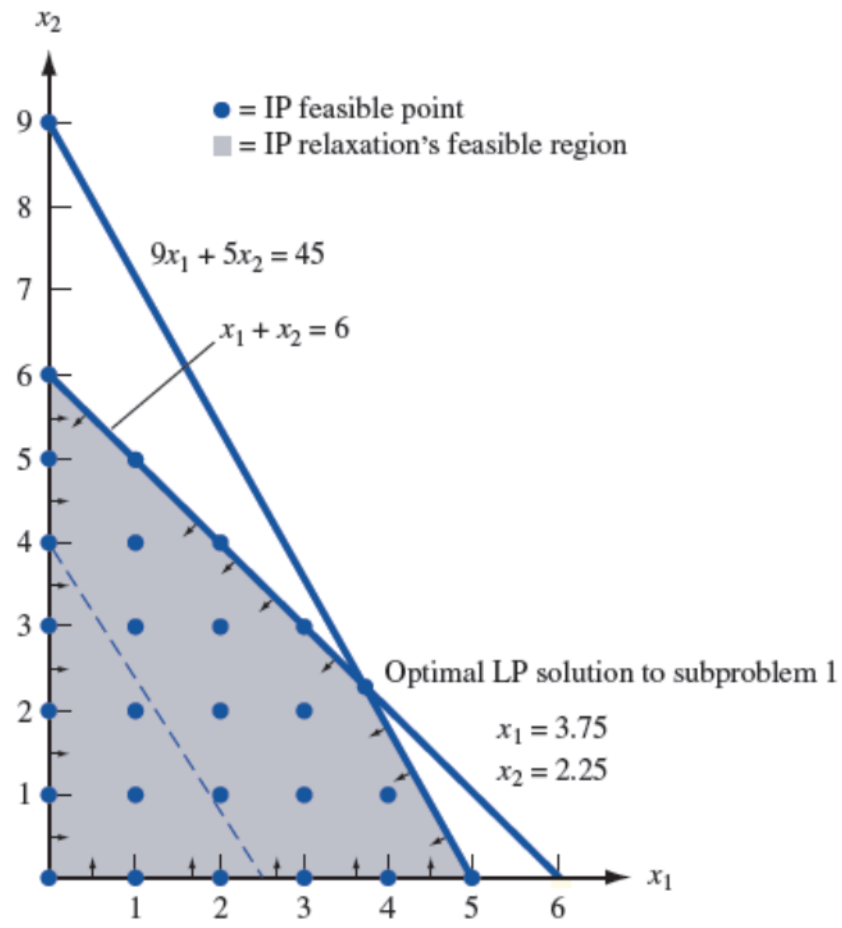
Example:

$$\max z = 8x_1 + 5x_2$$

$$\begin{array}{rcrcr} x_1 & + & x_2 & \leq & 6 \\ 9x_1 & + & 5x_2 & \leq & 45 \\ x_1 & , & x_2 & \geq & 0 \\ x_1 & , & x_2 & \in & \mathbb{Z} \end{array}$$

The optimal solution to the LPR $z = 165/4$; $x_1 = 15/4$, $x_2 = 9/4$. We know that the objective value of the LPR $z = 165/4$ sets an upper bound for the objective value of the IP.

The Branch-and-Bound Method



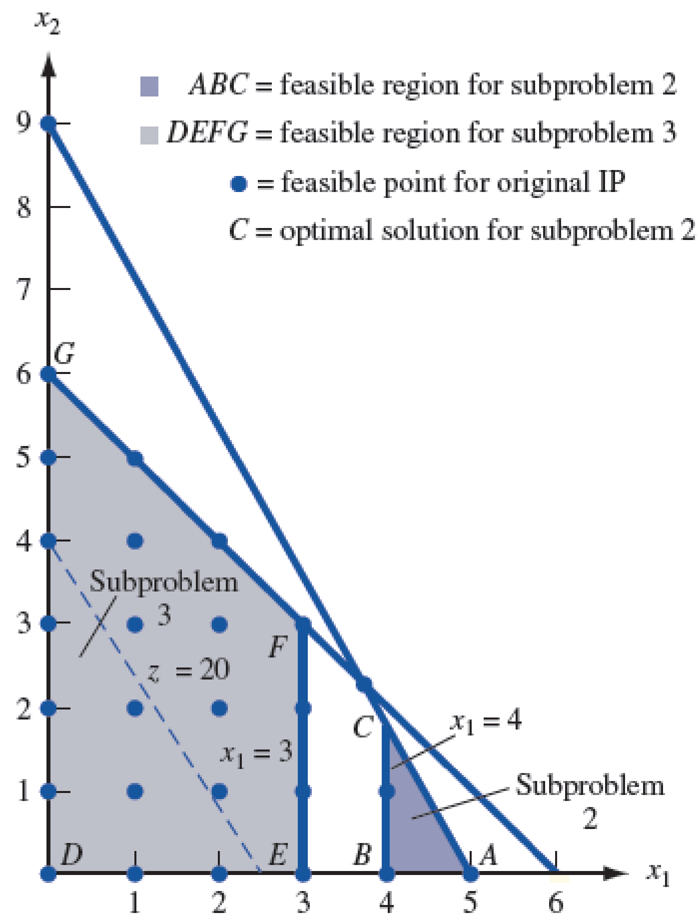
The Branch-and-Bound Method

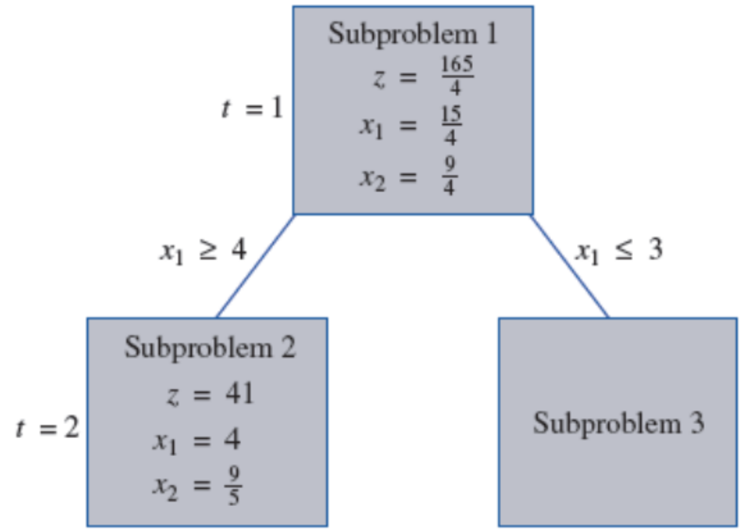
Let the original problem be the Sub-Problem 1 and construct the following sub-problems by branching on variable x_1:

Sub-Problem 2 = Sub-Problem 1 + $x_1 \geq 4$

Sub-Problem 3 = Sub-Problem 1 + $x_1 \leq 3$

The Branch-and-Bound Method





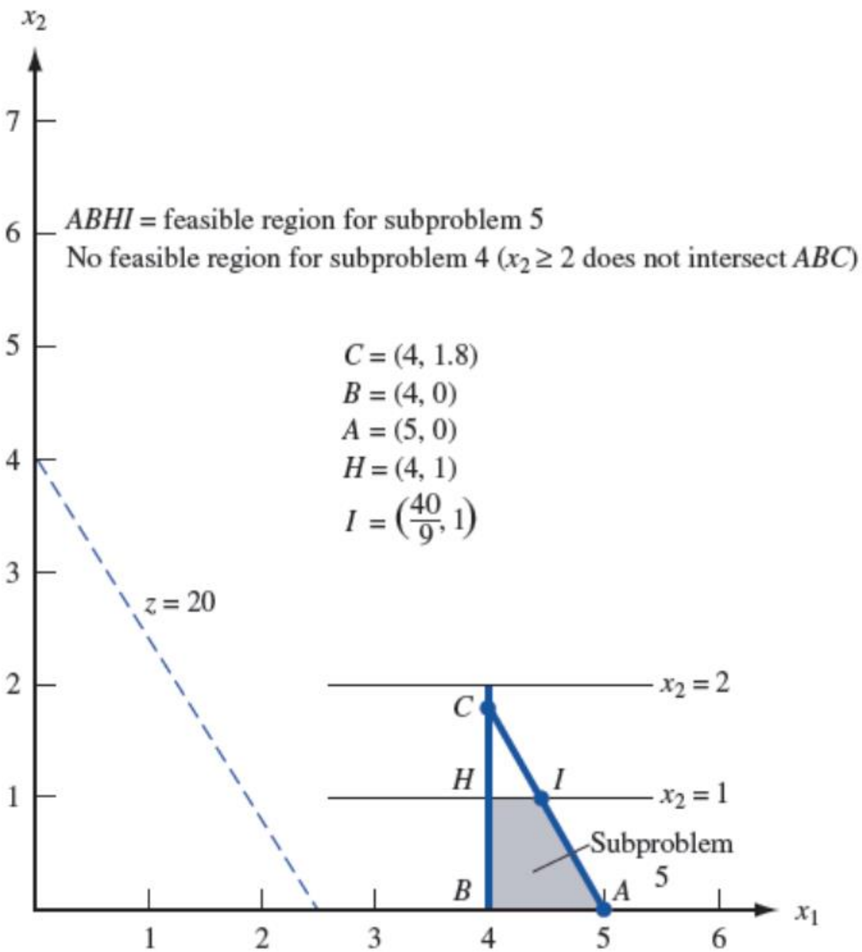
The Branch-and-Bound Method

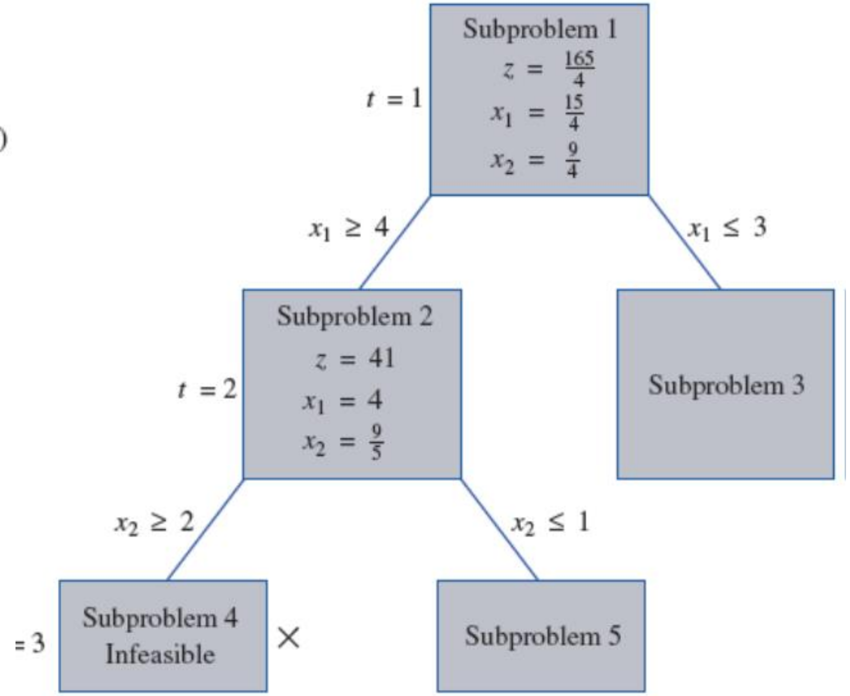
The optimal solution to sub-problem 2 did not yield an all-integer solution, so we choose to use sub-problem 2 to create two new sub-problems. We choose a fractional valued variable in the optimal solution to sub-problem 2 and then branch on that variable. Since x_2 is the only fractional variable here, we branch on x_2 as follows:

SP4 = SP1 + $x_1 \geq 4$ + $x_2 \geq 2$ = SP2 + $x_2 \geq 2$

SP5 = SP1 + $x_1 \leq 3$ + $x_2 \leq 1$ = SP2 + $x_2 \leq 1$

The Branch-and-Bound Method





The Branch-and-Bound Method

The feasible regions for sub-problems 4 and 5 are displayed below. The set of unsolved sub-problems consists of sub-problems 3, 4, and 5. We now choose a sub-problem to solve. For reasons that are discussed later, we choose to solve the most recently created sub-problem. (This is called the LIFO, or last-in-first-out, rule.) The LIFO rule implies that we should next solve sub-problem 4 or sub-problem 5. We arbitrarily choose to solve sub-problem 4.

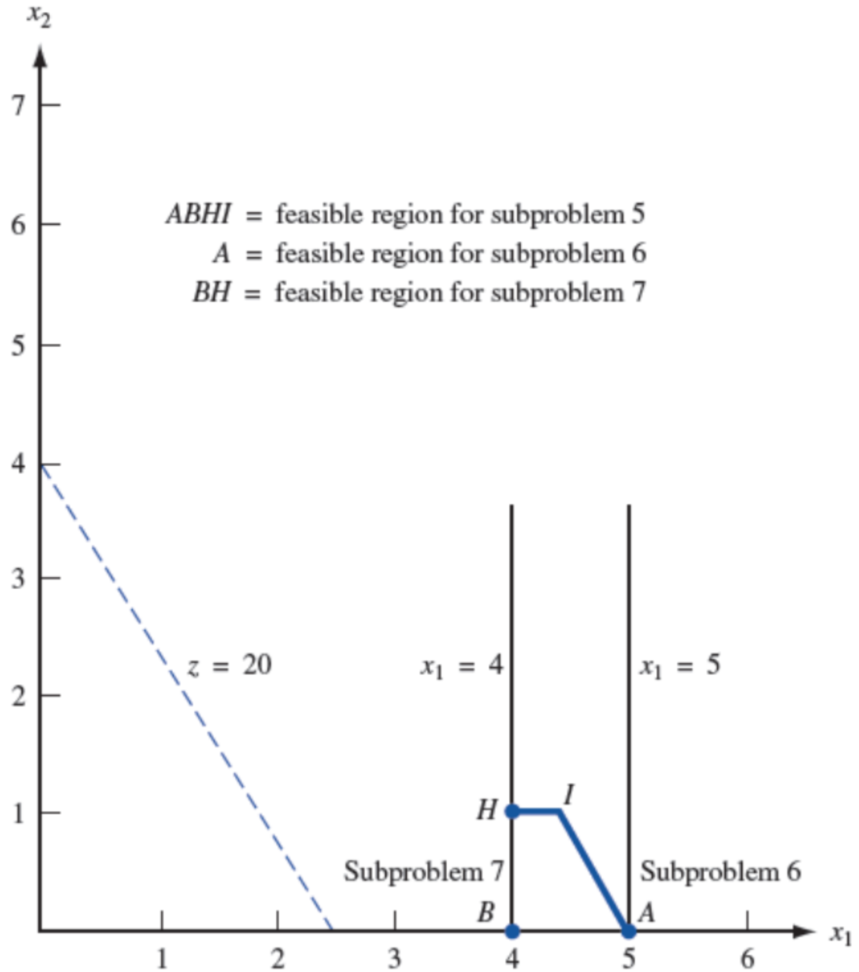
The Branch-and-Bound Method

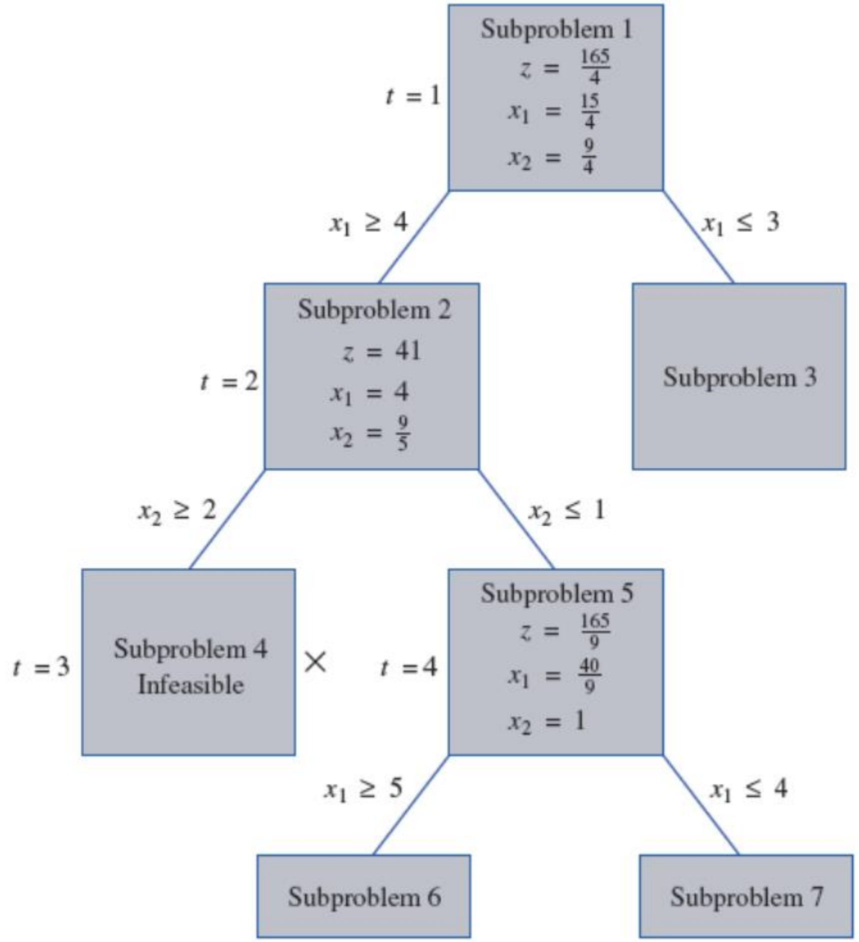
We note that sub-problem 4 is infeasible. Thus, sub-problem 4 cannot yield the optimal solution to the IP. To indicate this fact, we place an X by sub-problem 4. Because any branches emanating from sub-problem 4 will yield no useful information, it is fruitless to create them. When further branching on a sub-problem cannot yield any useful information, we say that the sub-problem (or node) is fathomed.

SP6 = SP5 + $x_1 \geq 5$

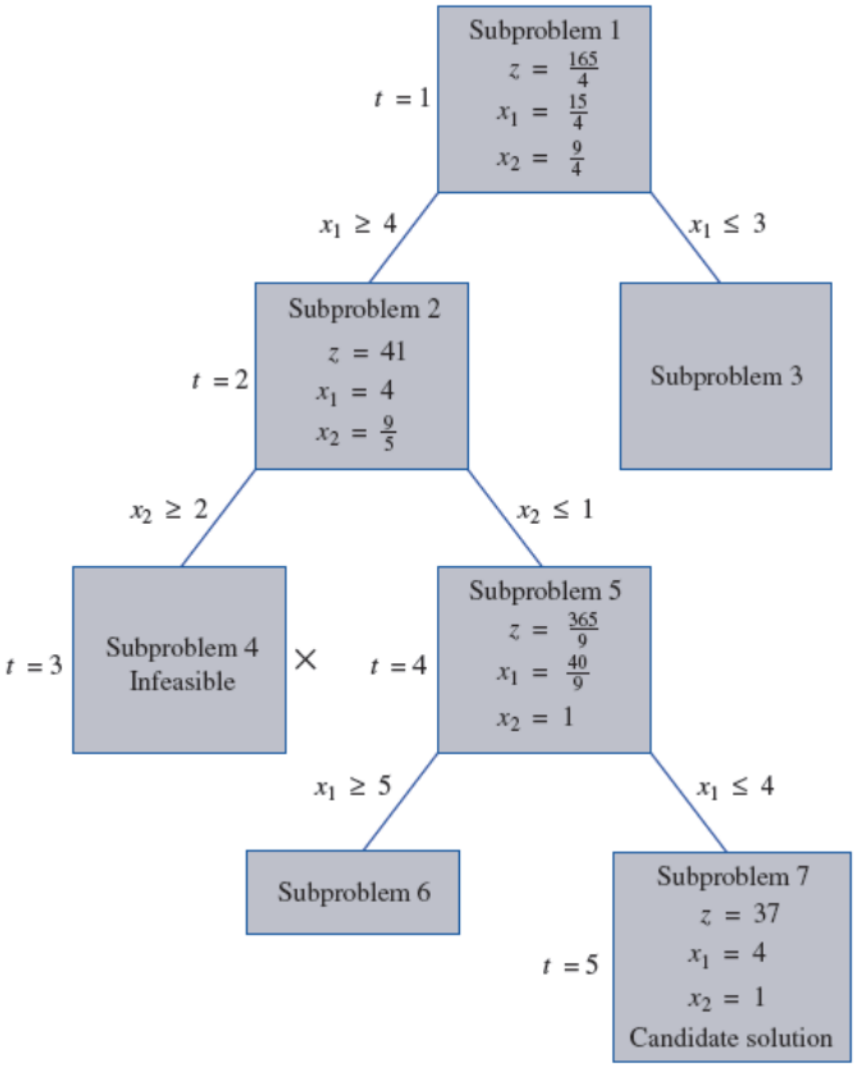
SP7 = SP5 + $x_1 \leq 4$

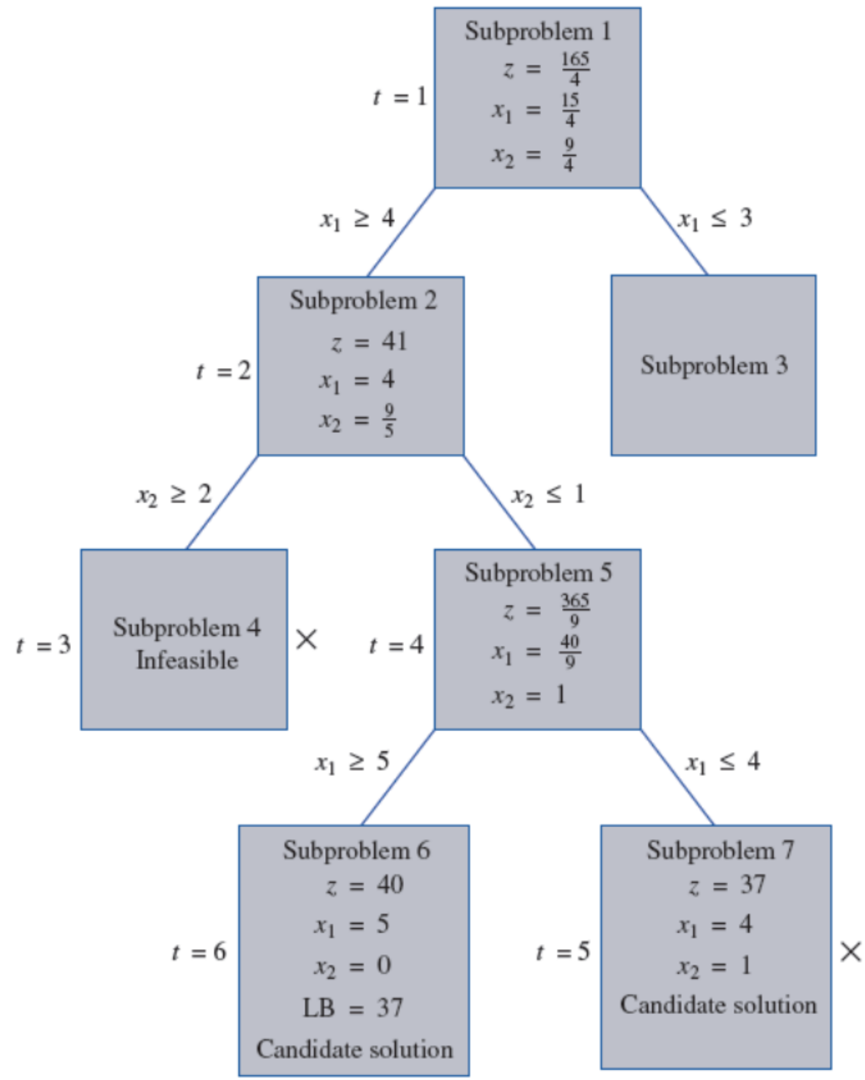
The Branch-and-Bound Method



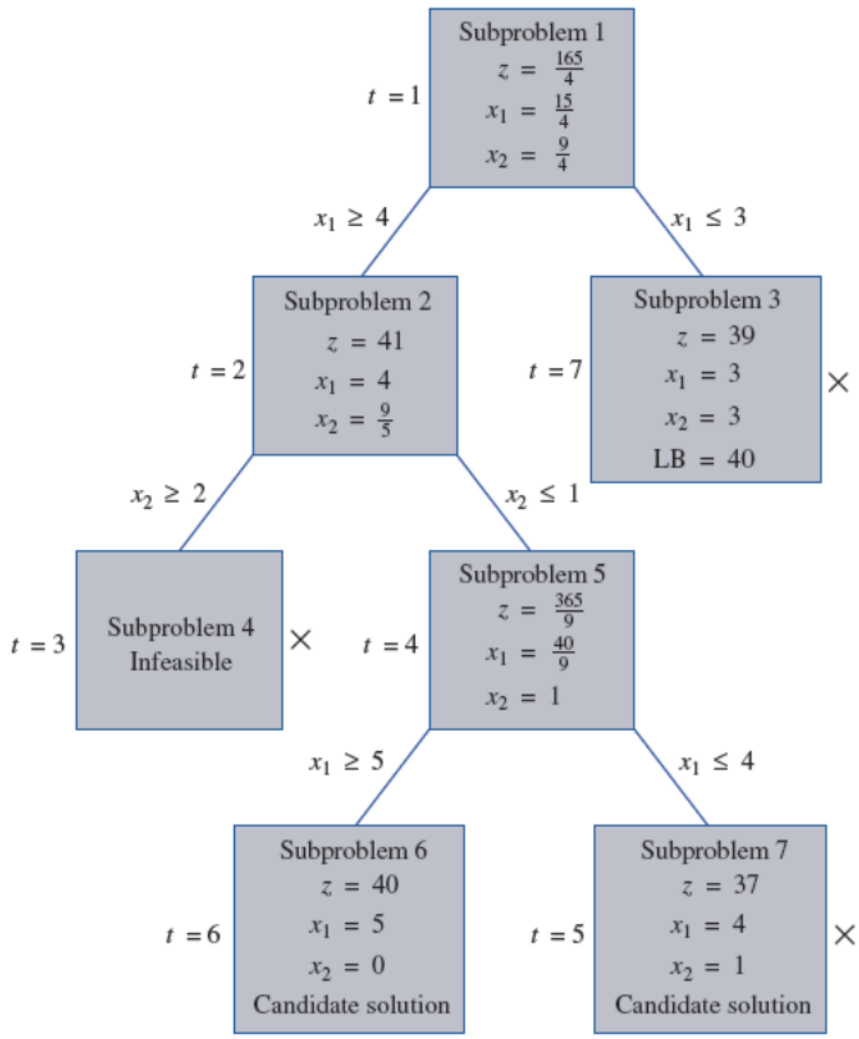


The Branch-and-Bound Method





The Branch-and-Bound Method



The Branch-and-Bound Method

Recall that in solving the Telfa problem by the branch-and-bound procedure, many seemingly arbitrary choices were made. Two general approaches are commonly used to determine which sub-problems should be solved next. The most widely used is the LIFO rule, which chooses to solve the most recently created sub-problem (back-tracking or depth-first). The second commonly used method is jump-tracking (breadth-first). When branching on a node, the jump-tracking approach solves all the problems created by the branching.

The Branch-and-Bound Method

Knapsack Problem

Let c_i be the benefit obtained if item i is chosen, b is the amount of available resource and a_i is the amount of available resource used by item i. We can then write

$$\max z = c_1 x_1 + \cdots + c_n x_n$$

$$\begin{array}{ccccccc} a_1 x_1 & + & \ldots & + & a_n x_n & \leq & b \\ x_1 & , & \ldots & , & x_n & \in & \{0,1\} \end{array}$$

Observe that c_i / a_i might be interpreted as the benefit by a unit of resource i. We can then say that the best and worst items to have are the ones with the largest and smallest c_i / a_i, respectively.

The Branch-and-Bound Method

Knapsack Problem

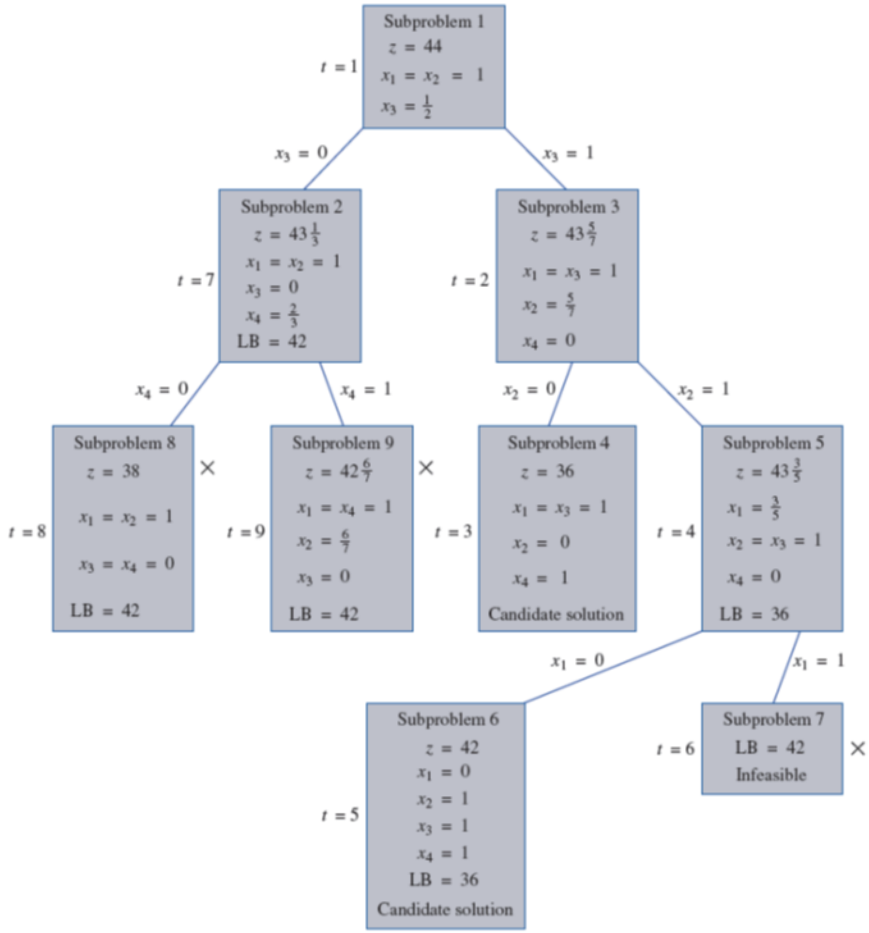
Example:

$$\max z = 16x_1 + 22x_2 + 12x_3 + 8x_4$$

$$\begin{array}{ccccccccc} 5x_1 & + & 7x_2 & + & 4x_3 & + & 3x_4 & \leq & 14 \\ x_1 & , & x_2 & , & x_3 & , & x_4 & \in & \{0,1\} \end{array}$$

The Branch-and-Bound Method

Knapsack Problem



The Branch-and-Bound Method

Knapsack Problem

Four jobs must be processed on a single machine. The time required to process each job and the date the job is due are shown below. The delay of a job is the number of days after the due date that a job is completed (if a job is completed on time or early, the job's delay is zero). In what order should the jobs be processed to minimize the total delay of the four jobs?

The Branch-and-Bound Method

Knapsack Problem

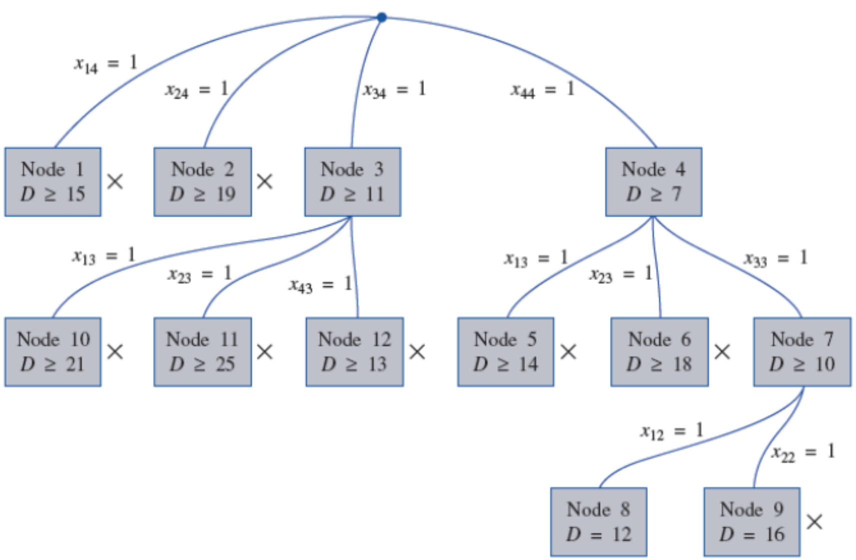
Job	Days to Complete	Due Date (end of day)
1	6	8
2	4	4
3	5	12
4	8	16

The Branch-and-Bound Method

Knapsack Problem

The branch-and-bound tree for the example is shown below if we let

$$x_{ij} = \begin{cases} 1, & \text{if job } i \text{ is the } j\text{th job to be processed} \\ 0, & \text{otherwise} \end{cases}$$



The Branch-and-Bound Method

TSP

Joe State lives in Gary, Indiana. He owns insurance agencies in Gary, Fort Wayne, Evansville, Terre Haute, and South Bend. Each December, he visits each of his insurance agencies. The distance between each agency (in miles) is shown below. What order of visiting his agencies will minimize the total distance traveled?

The Branch-and-Bound Method

TSP

	City 1	City 2	City 3	City 4	City 5
City 1	0	132	217	164	58
City 2		0	290	201	79
City 3			0	113	303
City 4				0	196
City 5					0

The Branch-and-Bound Method

We let

$$x_{ij} = \begin{cases} 1, & \text{if Joe travels from city } i \text{ to city } j \\ 0, & \text{otherwise} \end{cases}$$

	City 1	City 2	City 3	City 4	City 5
City 1	M	132	217	164	58
City 2	132	M	290	201	79
City 3	217	290	M	113	303
City 4	164	201	113	M	196
City 5	58	79	303	196	M

Cost Matrix for Sub-Problem 1

The Branch-and-Bound Method

We can find the solution of sub-problem 2 using Hungarian method as follows:

	city 1	city 2	city 3	city 4	city 5	min
city 1	M	132	217	164	58	58
city 2	132	M	290	201	79	79
city 3	217	290	M	M	303	217
city 4	164	201	113	M	196	113
city 5	58	79	303	196	M	58

The Branch-and-Bound Method

TSP

	city 1	city 2	city 3	city 4	city 5
city 1	M	74	159	106	0
city 2	53	M	211	122	0
city 3	0	73	M	M	86
city 4	51	88	0	M	83
city 5	0	21	245	138	M
min	0	21	0	106	0

The Branch-and-Bound Method

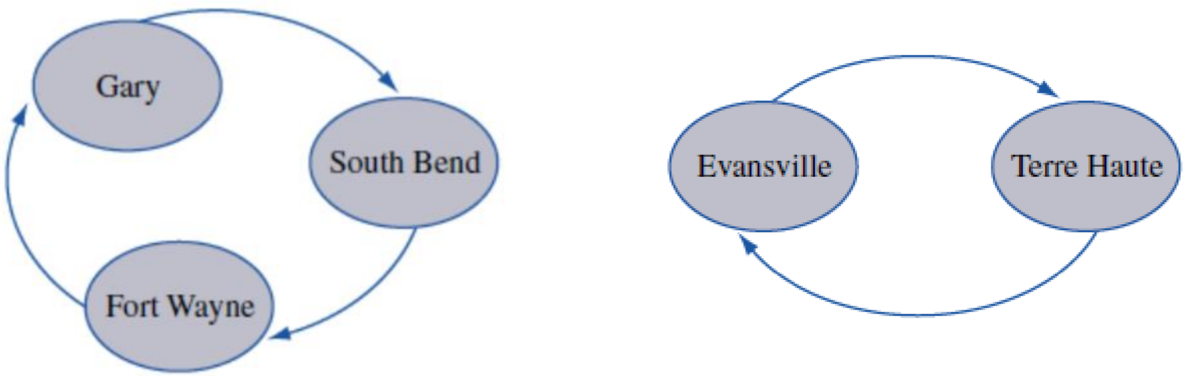
TSP

	city 1	city 2	city 3	city 4	city 5
city 1	M	53	159	0	0
city 2	53	M	211	16	0
city 3	0	52	M	M	86
city 4	51	67	0	M	83
city 5	0	0	245	32	M

The Branch-and-Bound Method

TSP

The optimal solution of the problem can be obtained using the Hungarian algorithm as $x_{15} = x_{21} = x_{34} = x_{43} = x_{52} = 1$. This solution includes two sub-tours as follows (1-5-2-1)-(3-4-3).



We then define the following sub-problems:

Sub-Problem 2 = Sub-Problem 1 + ($x_{34} = 0$) or ($c_{34} = M$)
Sub-Problem 3 = Sub-Problem 1 + ($x_{43} = 0$) or ($c_{43} = M$)

The Branch-and-Bound Method

$u_i - u_j + Nx_{ij} \leq N - 1$ expression in the IP formulation eliminates sub-tours. Consider, for instance, the above solution. Consider the sub-tour 2-5-2. If we write down the above expression for this sub-tour, we obtain

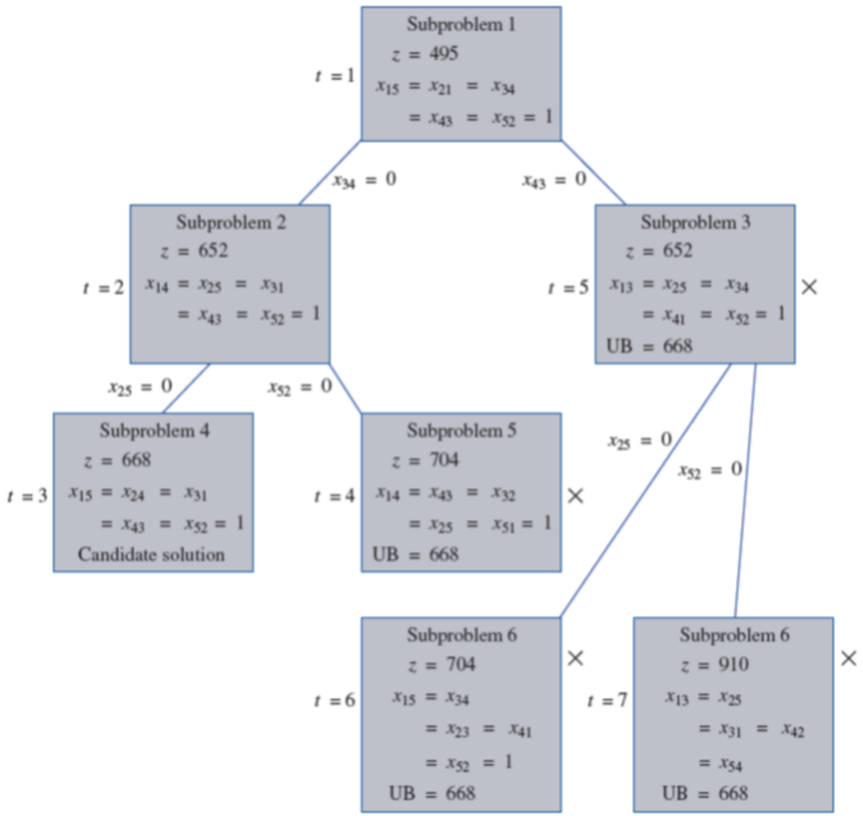
$$u_2 - u_5 + 5x_{25} \leq 4 \text{ and } u_5 - u_3 + 5x_{52} \leq 4$$

Using these,

$$\left.\begin{array}{l} u_2 - u_5 + 5x_{25} \leq 4 \\ u_5 - u_2 + 5x_{52} \leq 4 \end{array}\right\} \Rightarrow 5(x_{25} + x_{52}) \leq 8 \Leftrightarrow x_{25} = 0 \vee x_{52} = 0$$

The Branch-and-Bound Method

TSP



The Branch-and-Bound Method

TSP

	City 1	City 2	City 3	City 4	City 5
City 1	M	132	217	164	58
City 2	132	M	290	201	79
City 3	217	290	M	M	303
City 4	164	201	113	M	196
City 5	58	79	303	196	M

	City 1	City 2	City 3	City 4	City 5
City 1	M	132	217	164	58
City 2	132	M	290	201	M
City 3	217	290	M	M	303
City 4	164	201	113	M	196
City 5	58	79	303	196	M

Cost Matrices for Sub-Problem 2 and 3

The Branch-and-Bound Method

TSP

	City 1	City 2	City 3	City 4	City 5
City 1	M	132	217	164	58
City 2	132	M	290	201	M
City 3	217	290	M	M	303
City 4	164	201	113	M	196
City 5	58	79	303	196	M

	City 1	City 2	City 3	City 4	City 5
City 1	M	132	217	164	58
City 2	132	M	290	201	79
City 3	217	290	M	M	303
City 4	164	201	M	M	196
City 5	58	M	303	196	M

Cost Matrices for Sub-Problem 4 and 5

The Branch-and-Bound Method

TSP

We can define the following IP for the TSP:

$$\min z = \sum_{i} \sum_{j} c_{ij} x_{ij}$$

$$\sum_{i=1}^{N} x_{ij} = 1, \quad j = 1,2, \dots, N$$

$$\sum_{j=1}^{N} x_{ij} = 1, \quad i = 1,2, \dots, N$$

$$u_i - u_j + N x_{ij} \leq N - 1, \quad \forall i, j \quad i \neq j$$

$$x_{ij} \in \mathbb{Z}, \quad \forall i, j$$

$$u_j \geq 0, \quad \forall j$$

The Branch-and-Bound Method

2 Heuristics for the TSP

- The Nearest-Neighbor (NN)
- The Cheapest-Instertion (CI)

The Branch-and-Bound Method

TSP-The NN

	City 1	City 2	City 3	City 4	City 5
City 1	0	132	217	164	58
City 2		0	290	201	79
City 3			0	113	303
City 4				0	196
City 5					0

The Branch-and-Bound Method

TSP-The NN

- Begin at city 1.
- Go to city 5 which is the nearest city to city 1.
  - 1-5
- Go to city 2 which is the nearest to city 5, the most recently visited.
  - 1-5-2
- Go to city 4 which is the nearest to city 2, the most recently visited.
  - 1-5-2-4

The Branch-and-Bound Method

- Go to city 3 which is the nearest to city 4, the most recently visited.
  - 1-5-2-4-3
- Go to city 1 which is the nearest to city 3, the most recently visited.
  - 1-5-2-4-3-1

- A tour is completed. Stop.

The Branch-and-Bound Method

Arc Replaced	Arcs Added to Sub-Tour	Added Cost
(1,5)	(1,2)-(2,5)	$c_{12} + c_{25} - c_{15} = 153$
(1,5)	(1,3)-(3,5)	$c_{13} + c_{35} - c_{15} = 462$
(1,5)	(1,4)-(4,5)	$c_{14} + c_{45} - c_{15} = 302$
(5,1)	(5,2)-(2,1)	$c_{52} + c_{21} - c_{51} = 153$
(5,1)	(5,3)-(3,1)	$c_{53} + c_{31} - c_{51} = 462$
(5,1)	(5,4)-(4,1)	$c_{54} + c_{41} - c_{51} = 302$

Sub-Tour: 1-2-5-1

The Branch-and-Bound Method

Arc Replaced	Arcs Added to Sub-Tour	Added Cost
(1,2)	(1,3)-(3,2)	$c_{13} + c_{32} - c_{12} = 375$
(1,2)	(1,4)-(4,2)	$c_{14} + c_{42} - c_{12} = 233$
(2,5)	(2,3)-(3,5)	$c_{23} + c_{35} - c_{25} = 514$
(2,5)	(2,4)-(4,5)	$c_{24} + c_{45} - c_{25} = 318$
(5,1)	(5,3)-(3,1)	$c_{53} + c_{31} - c_{51} = 462$
(5,1)	(5,4)-(4,1)	$c_{54} + c_{41} - c_{51} = 302$

Sub-Tour: 1-4-2-5-1

The Branch-and-Bound Method

Arc Replaced	Arcs Added to Sub-Tour	Added Cost
(1,4)	(1,3)-(3,4)	$c_{13} + c_{34} - c_{14} = 166$
(4,2)	(4,3)-(3,2)	$c_{43} + c_{32} - c_{42} = 202$
(2,5)	(2,3)-(3,5)	$c_{23} + c_{35} - c_{25} = 514$
(5,1)	(5,3)-(3,1)	$c_{53} + c_{31} - c_{51} = 462$

Sub-Tour: 1-3-4-2-5-1

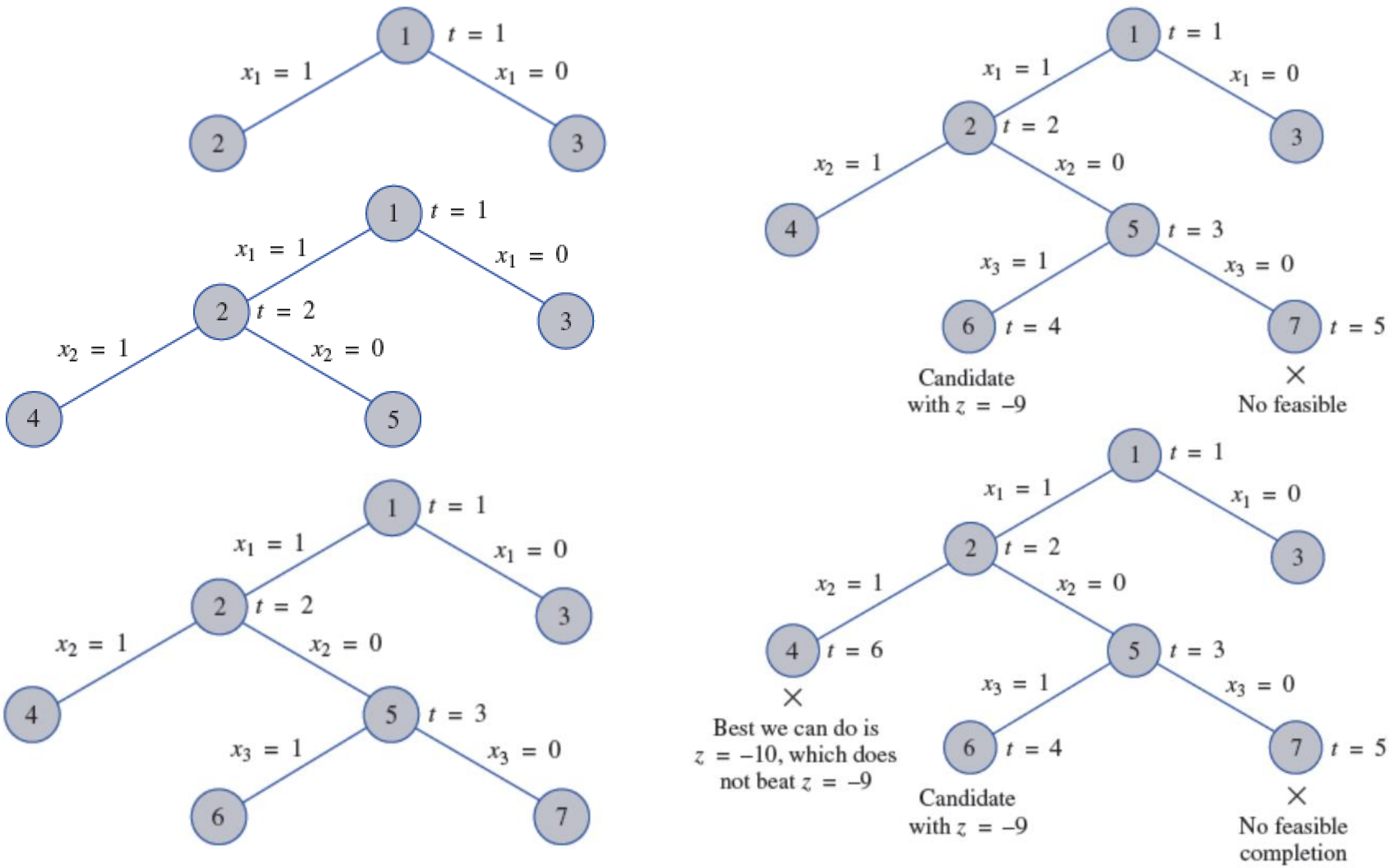
Implicit Enumeration

Example:

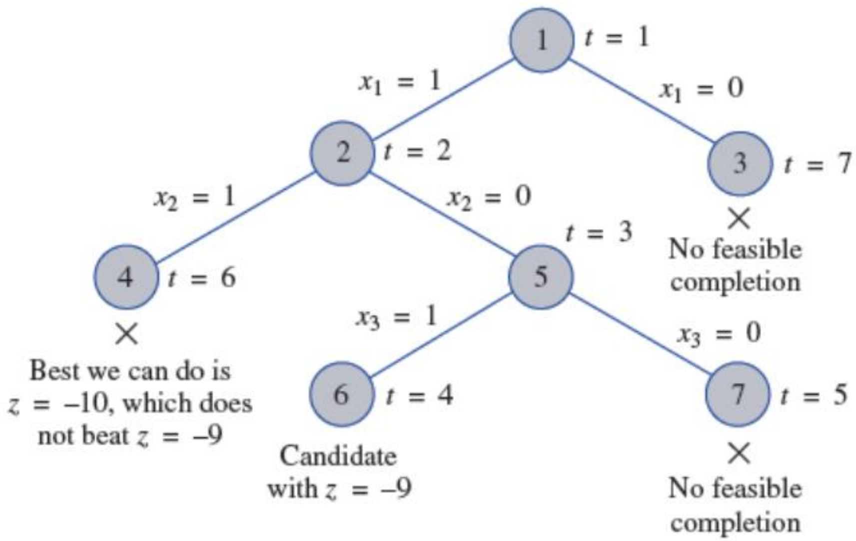
$$\max z = -7x_1 - 3x_2 - 2x_3 - x_4 - 2x_5$$

$$\begin{array}{rcrcrcrcrcr}
-4x_1 & - & 2x_2 & + & x_3 & - & 2x_4 & - & x_5 & \leq & -3 \\
-4x_1 & - & 2x_1 & - & 4x_3 & + & x_4 & + & 2x_5 & \leq & -7 \\
x_1 & , & x_2 & , & x_3 & , & x_4 & , & x_5 & \in & \{0,1\}
\end{array}$$

Implicit Enumeration



Implicit Enumeration



NODE 1:

At the beginning all variables are free.

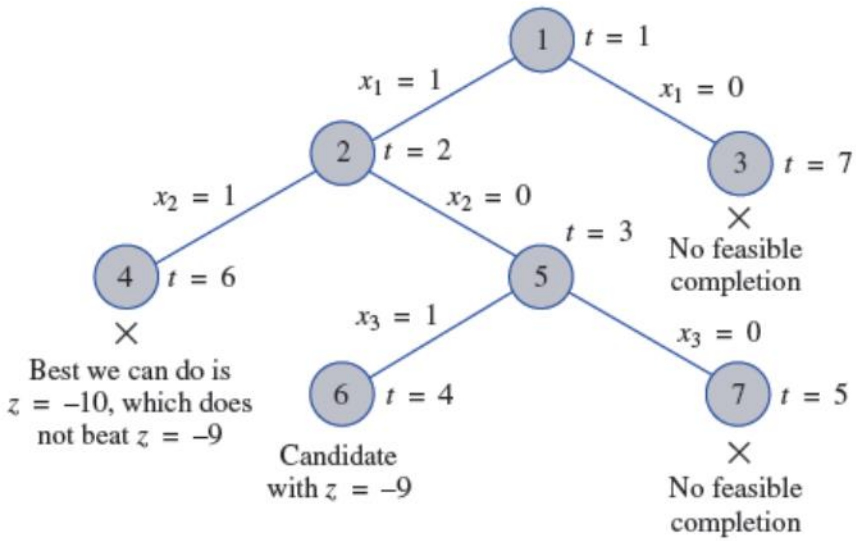
Is best completion feasible?

NO!

Is there a feasible completion?

YES!

Implicit Enumeration



NODE 2:

x_1: fixed, others free

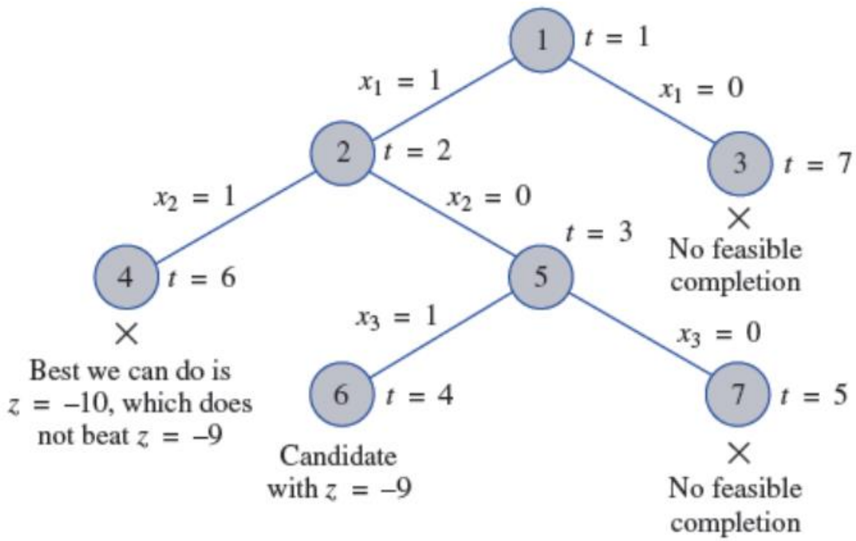
Is best completion feasible?

NO!

Is there a feasible completion?

YES!

Implicit Enumeration



NODE 5:

x_1, x_2: fixed, others free

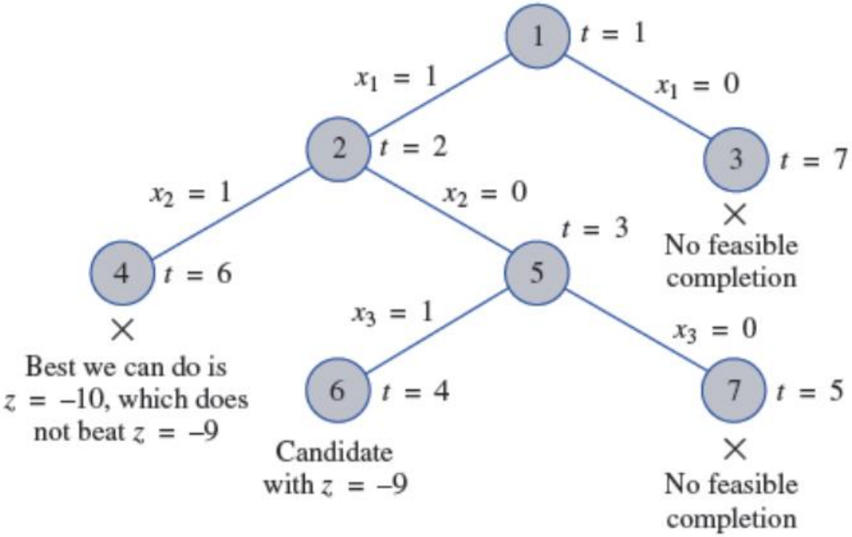
Is best completion feasible?

NO!

Is there a feasible completion?

YES!

Implicit Enumeration



NODE 6:

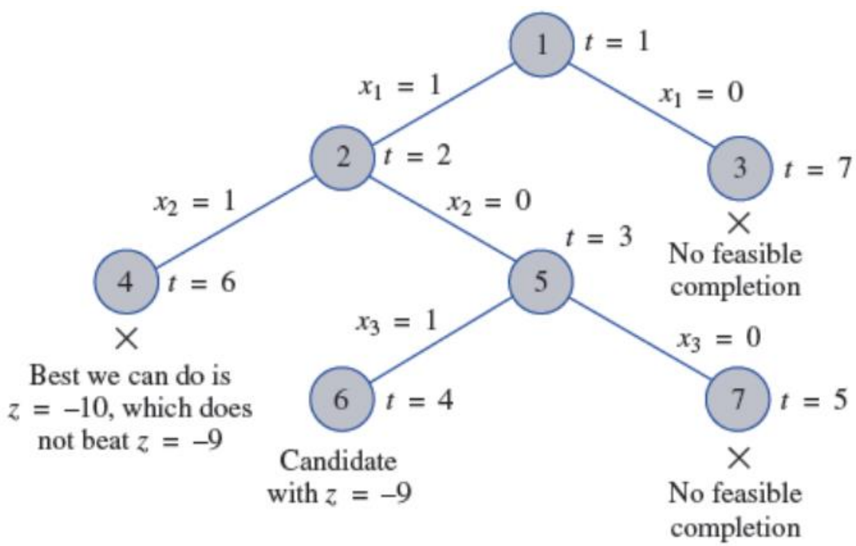
x_1, x_2, x_3: fixed, others free

Is best completion feasible?

YES!

Hence, we have a candidate!

Implicit Enumeration



NODE 7:

x_1, x_2, x_3: fixed, others free

Is best completion feasible?

NO!

Is there a feasible completion?

NO!

Hence, we fathom the node!

The Cutting Plane Algorithm

Like the branch-and-bound approach, the cutting plane algorithm can be used to solve IPs. The summary of the algorithm is as follows:

The Cutting Plane Algorithm

Step 1) Find the optimal tableau for the LPR. If all variables in the optimal solution assume integer values, then we have found an optimal solution to the IP. Otherwise, proceed to Step (2).

Step 2) Pick a constraint in the LPR optimal tableau whose right-hand side has the fractional part closest to 1/2. This constraint will be used to generate a cut.

The Cutting Plane Algorithm

Step 2-a) For the constraint identified in Step (2), write its right-hand side and each variables's coefficient in the form $[x] + f$ where $0 \leq f < 1$.

Step 2-b) Rewrite the constraint used to generate the cut as

all terms with integer coefficients = all terms with fractional coefficients

The cut is then

all terms with fractional coefficients ≤ 0

The Cutting Plane Algorithm

Step 3) Use the dual simplex to find the optimal solution to the LP relaxation, with the cut as an additional constraint. If all variables assume integer values in the optimal solution, we have found an optimal solution to the IP. Otherwise, pick the constraint with the most fractional right-hand side and use it to generate another cut, which is added to the tableau. We continue this process until we obtain a solution in which all variables are integers. This will be an optimal solution to the IP.

The Cutting Plane Algorithm

Example: Consider the below model and optimal solution for the LPR:

$$\max z = 8x_1 + 5x_2$$

$$\begin{array}{rcrcr} x_1 & + & x_2 & \leq & 6 \\ 9x_1 & + & 5x_2 & \leq & 45 \\ x_1 & , & x_2 & \geq & 0 \\ x_1 & , & x_2 & \in & \mathbb{Z} \end{array}$$

	z	x_1	x_2	s_1	s_2	RHS
z	1	0	0	1.25	0.75	41.25
x_2	0	0	1	2.25	– 0.25	2.25
x_1	0	1	0	– 1.25	0.25	3.75

The Cutting Plane Algorithm

	z	x_1	x_2	s_1	s_2	RHS
z	1	0	0	1.25	0.75	41.25
x_2	0	0	1	2.25	– 0.25	2.25
x_1	0	1	0	– 1.25	0.25	3.75

To apply the cutting plane method, we begin by choosing any constraint in the LPR's optimal tableau in which a basic variable is fractional. We arbitrarily choose the second constraint, which is

$$x_1 - 1.25s_1 + 0.25s_2 = 3.75$$

The Cutting Plane Algorithm

Now write each variable's coefficient and the constraint's right hand side in the form $[x] + f$ where $0 \leq f < 1$ as

$$x_1 - 2s_1 + 0.75s_1 + 0s_2 + 0.25s_2 = 3 + 0.75$$

Putting all terms with integer coefficients on the left side and all terms with fractional coefficients on the right side yields

$$x_1 - 2s_1 + 0s_2 - 3 = 0.75 - 0.75s_1 - 0.25s_2$$

Now, we add the following constraint to the LPR which is called a cut:

$$0.75 - 0.75s_1 - 0.25s_2 \leq 0$$

The Cutting Plane Algorithm

We can show that

- Any feasible point for the IP will satisfy the cut.
- The current optimal solution to the LP relaxation will not satisfy the cut.

The Cutting Plane Algorithm

Thus, a cut “cuts off” the current optimal solution to the LP relaxation, but not any feasible solutions to the IP. When the cut to the LP relaxation is added, we hope we will obtain a solution where all variables are integer-valued. If so, we have found the optimal solution to the original IP. If our new optimal solution (to the LPR plus the cut) has some fractional-valued variables, then we generate another cut and continue the process. Gomory (1958) showed that this process will yield an optimal solution to the IP after a finite number of cuts.

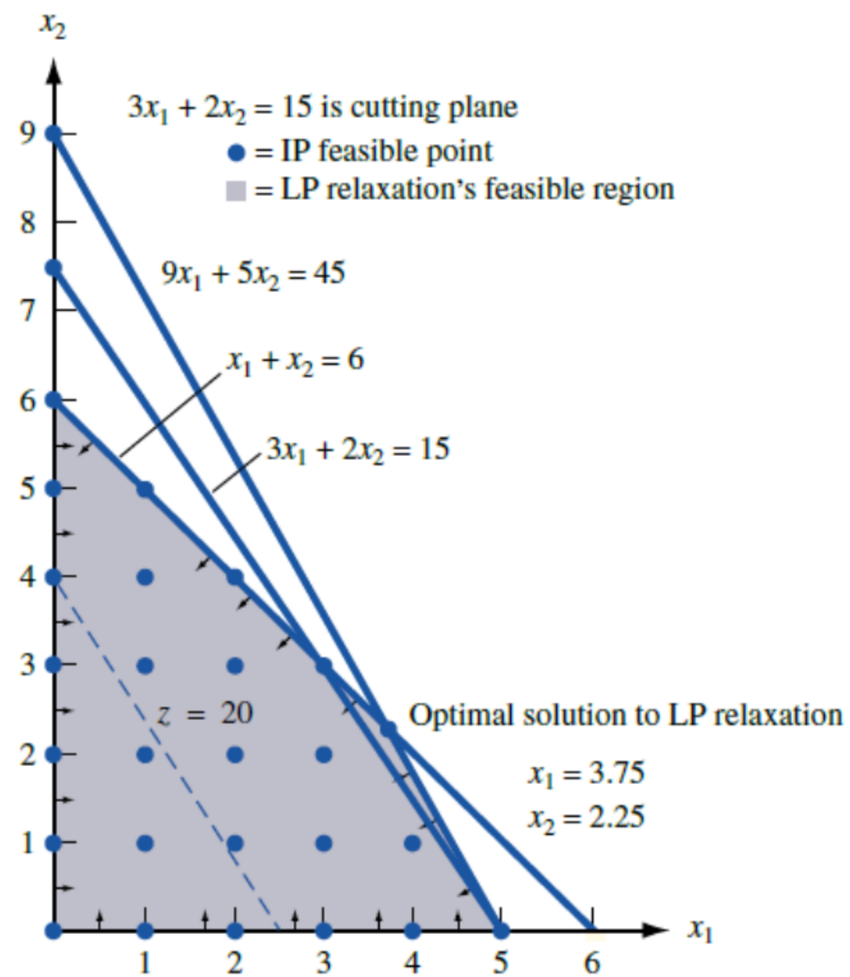
The Cutting Plane Algorithm

After adding the cut, we proceed by the dual simplex as follows:

	z	x_1	x_2	s_1	s_2	s_3	RHS
z	1	0	0	1.25	0.75	0.75	41.25
x_2	0	0	1	2.25	– 0.25	– 0.25	2.25
x_1	0	1	0	– 1.25	0.25	0.25	3.75
s_3	0	0	0	– 0.75	– 0.25	1.00	– 0.75

	z	x_1	x_2	s_1	s_2	s_3	RHS
z	1	0	0	0	0.33	1.67	40
x_2	0	0	1	0	– 1.00	3.00	0
x_1	0	1	0	0	0.67	– 1.67	5
s_1	0	0	0	1	0.33	– 1.33	1

The Cutting Plane Algorithm



The Branch-and-Cut Approach

The Branch-and-Cut Approach

- Problem Pre-Processing
- Cutting Plane Generation
- Branch-and-Bound

The Branch-and-Cut Approach

Problem Pre-Processing

- Fixing Variables
- Eliminating Redundant Constraints
- Tightening Constraints

Thanks… ☺