

Integer Programming Introduction

Divisibility assumption of LP allows for fractions: Produce 7.8 units of a product, buy 12500.33 liters of oil, hire 12.123 people for full time.

Clearly some activities cannot be done in fractions and must be specified as integers.

A small example:

Consider producing chairs and tables using only 21 m^2 of wood. Each chair (table) requires 6 (7) m^2 of wood. Each chair is sold at \$12 and each table is sold at \$13.

Let C and T denote the number of tables and chairs produced.

Maximize : $12C + 13T$

Subject to

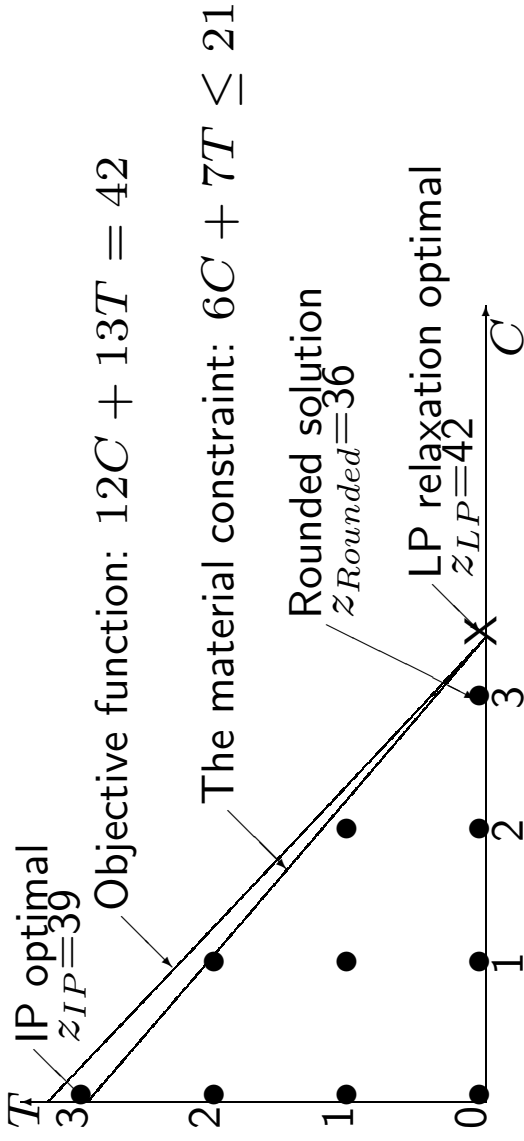
$$6C + 7T \leq 21 \quad (1)$$

$$C, T \geq 0 \quad (2)$$

$$C, T \text{ int} \quad (3)$$

Integer Programming: Rounding

Solve the LP graphically and round the LP solution:



Rounded solutions of LP relaxations are not necessarily optimal.

How about another approach; Evaluate all the integer solutions in the feasible region and pick the best? Practically impossible.

Two approaches are common: *Branch and Bound technique*, and *Cutting planes*.

A Knapsack Problem

Jean Luc (an MBA student) is going to study at most 40 hours/week in the next term and considering to take some of the following courses:

Operations Research	Supply Chain Management	Information Technology	Finance
9	7	5	4
Marketing	Organizational Behaviour	Italian Cinema	Russian
5	3	7	10

Completing each of these courses increases Jean Luc's chances of finding a job. But the contributions of courses towards this:

Operations Research	Supply Chain Management	Information Technology	Finance
0.10	0.14	0.06	0.09
Marketing	Organizational Behaviour	Italian Cinema	Russian
0.08	0.03	0.04	0.05

What courses Jean Luc should take to maximize his chances of finding a job?

A Knapsack Problem

Decision Variables

For each course, one by one, he answers the question whether a course is taken. Then he will know what he is taking.

For example, if OR is taken, the answer will be a “yes” for OR, otherwise a “no”. “yes” $\equiv 1$ and “no” $\equiv 0$. Also let x_{OR} be the question. Then, $x_{OR} = 1$ implies a “yes” answer for OR.

$$x_j = \begin{cases} 1 & \text{if course } j \text{ is taken} \\ 0 & \text{otherwise} \end{cases}.$$

Constraints

There is only one constraint: 40 hours are available for studies.

$$9x_{OR} + 7x_{SC} + 5x_{IT} + 4x_{Fi} + 5x_{Ma} + 3x_{OB} + 7x_{IC} + 10x_{Ru} \leq 40$$

A Knapsack Problem

Objective Function

Writing contributions of each course and summing:

$$\text{Maximize } 0.10x_{OR} + 0.14x_{SC} + 0.06x_{IT} + 0.09x_{Fi} + 0.08x_{Ma} + 0.03x_{OB} + 0.04x_{IC} + 0.05x_{Ru}$$

IP Formulation

$$\text{Max } 0.10x_{OR} + 0.14x_{SC} + 0.06x_{IT} + 0.09x_{Fi} + 0.08x_{Ma} + 0.03x_{OB} + 0.04x_{IC} + 0.05x_{Ru}$$

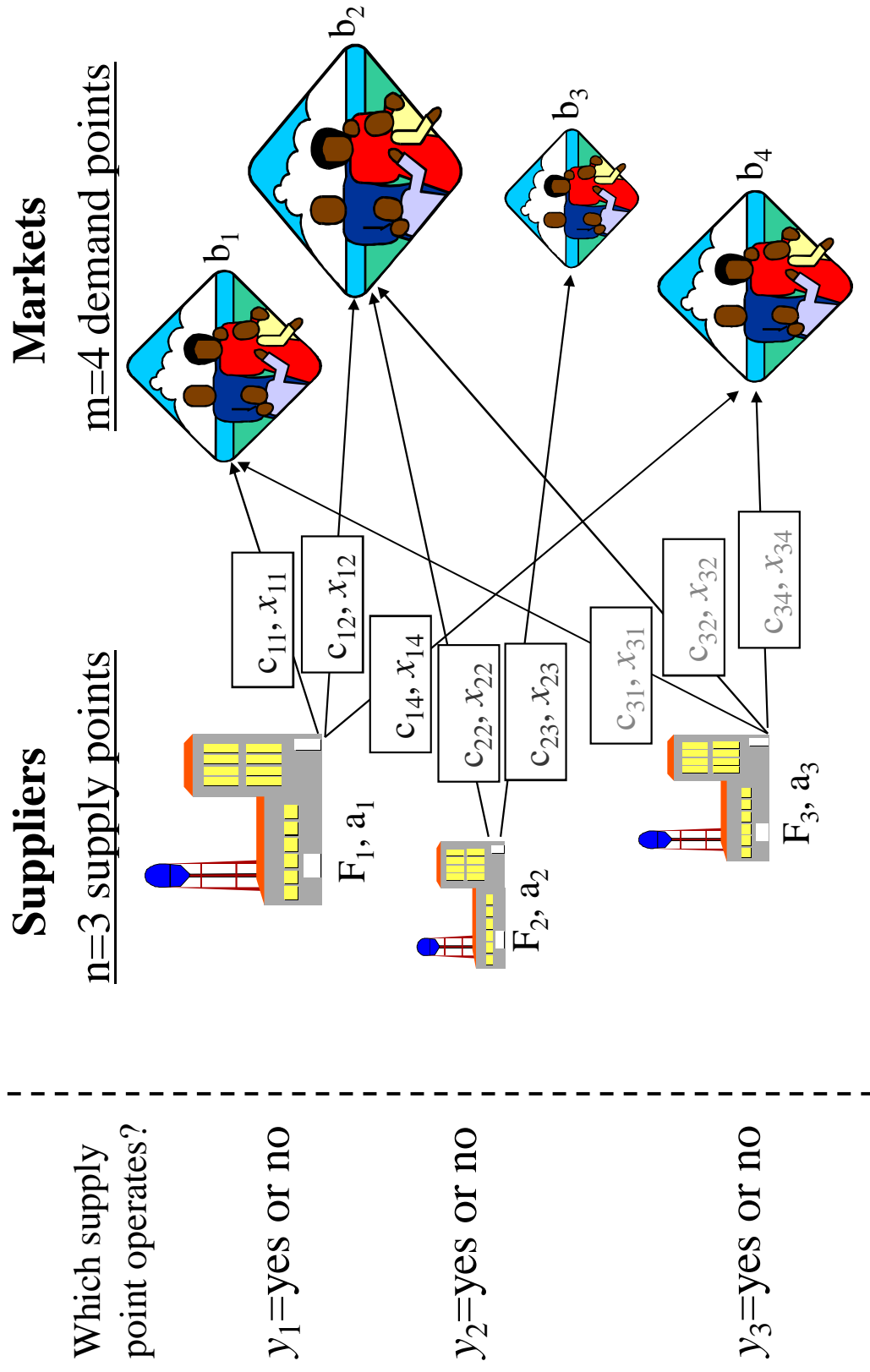
St

$$9x_{OR} + 7x_{SC} + 5x_{IT} + 4x_{Fi} + 5x_{Ma} + 3x_{OB} + 7x_{IC} + 10x_{Ru} \leq 40$$
$$x_j \in \{0, 1\}$$

With solution $x_{OR} = 1$, $x_{SC} = 1$, $x_{IT} = 1$, $x_{Fi} = 1$, $x_{Ma} = 1$ and $x_{IC} = 1$, which courses are taken, any slack time?

Binary Programs? Origins of the knapsack problem.

A Facility Location Problem



A Facility Location Problem

This problem is very similar to the transportation problem. Only difference is that we want to decide on warehouse locations as well as flows. There are m locations where we can open up warehouses. To open up a warehouse we pay a cost of F_i .

Decision Variables

Reminiscent from the transportation problem, flows from i to j :

x_{ij} = Amount of flow from warehouse i to retailer j

In addition to flows, answer the question whether warehouse i is opened up. If we answer is “yes” to this question, warehouse i is opened.

$$y_i = \left\{ \begin{array}{ll} 1 & \text{if warehouse } i \text{ opened up} \\ 0 & \text{otherwise} \end{array} \right\}.$$

A Facility Location Problem

Constraints

Demand Constraints: $\sum_{i=1}^m x_{ij} \geq b_j \quad j = 1..n$

Supply Constraints: If warehouse i is not opened up, we can not send any units out of it:

$$y_i = 0 \implies \sum_{j=1}^n x_{ij} = 0$$

On the other hand, if warehouse i is opened, we can send its supply a_i to retailers:

$$y_i = 1 \implies \sum_{j=1}^n x_{ij} \leq a_i$$

When warehouse i is open its supply is a_i , otherwise it is 0. In general, its $y_i a_i$:

$$\sum_{j=1}^n x_{ij} \leq y_i a_i \quad i = 1..m .$$

A Facility Location Problem

Objective Function

The cost of sending materials from warehouses to retailers:

$$\sum_{i=1}^j \sum_{j=1}^n c_{ij}x_{ij}$$

We also pay F_i by opening warehouse i , and 0 otherwise. In either case, we pay $F_i y_i$.
The total cost of warehouses is:

$$\sum_{i=1}^m F_i y_i$$

Summing up the flow and warehouse costs, we drive the total cost to be minimized:

$$\text{Minimize } \sum_{i=1}^j \sum_{j=1}^n c_{ij}x_{ij} + \sum_{i=1}^m F_i y_i$$

A Facility Location Problem

IP Formulation

$$\text{Minimize } \sum_{i=1}^j \sum_{j=1}^n c_{ij} x_{ij} + \sum_{i=1}^m F_i y_i$$

Subject to

$$\sum_{i=1}^m x_{ij} \geq b_j \quad j = 1..n$$

$$\sum_{j=1}^n x_{ij} \leq y_i a_i \quad i = 1..m$$

$$x_{ij} \geq 0 \quad i = 1..m, j = 1..n$$

$$y_i \in \{0, 1\} \quad i = 1..m$$

Mixed Integer Linear Program (MILP)?

Player Selection at Dallas Cowboys

Dallas Cowboys wants to sign 3 new players. There are five players under consideration. If player j demands the salary c_j , make up an IP to decide on which players should be signed to minimize the signing budget considering

1. Player 1, 2 and 4 can play the quarter back position and Cowboys want to bring in at least one new player for this position.
2. Each player has a strength measured in s_j and Cowboys want to add at least S units of strength to the team by hiring some of these 5 players.
3. Player j is likely to have a_j assaults per season and Cowboys imposes a quota of A assaults per season for these players.

Player Selection at Dallas Cowboys

Decision Variables

$$x_j = \begin{cases} 1 & \text{if player } j \text{ is signed} \\ 0 & \text{otherwise} \end{cases}.$$

Constraints

At least one quarterback: $x_1 + x_2 + x_4 \geq 1$

Minimum strength: $\sum_{j=1}^5 s_j x_j \geq S$

Assaults quota: $\sum_{j=1}^5 a_j x_j \leq A$

Objective Function

$$\text{Minimize } \sum_{j=1}^5 c_j x_j$$

Player Selection at Dallas Cowboys: If-then constraints

- If player 1 and 2 are bodies and will come to Dallas only together, add a constraint to guarantee that either they are signed together or they are not signed at all.

$$(x_1 = 1 \implies x_2 = 1) \text{ and } (x_1 = 0 \implies x_2 = 0).$$

Thus add : $x_1 - x_2 = 0$

- If player 1 hates player 3 and will not come if player 3 comes, add an appropriate constraint that does not allow signing player 1 and 3 together.

$$(x_3 = 1 \implies x_1 = 0) \text{ and } (x_1 = 1 \implies x_3 = 0).$$

Thus add: $x_1 + x_3 \leq 1$

- If player 1 is signed and player 3 is not, player 4 is not going to sign. Add an appropriate constraint that does not allow signing player 4 when 1 is signed but 3 is not.

$$(x_1 = 1, x_3 = 0 \implies x_4 = 0)$$

Thus add: $x_4 \leq 1 - x_1 + x_3$.

Player Selection at Dallas Cowboys: If-then constraints

- If player 1 is signed and player 3 is not, player 4 and 5 are not going to sign. Add appropriate constraints that do not allow signing player 4 or 5 when 1 is signed but 3 is not.

$$(x_1 = 1, x_3 = 0 \implies x_4 = 0) \text{ and } (x_1 = 1, x_3 = 0 \implies x_5 = 0)$$

Thus add: $x_4 \leq 1 - x_1 + x_3$ and $x_5 \leq 1 - x_1 + x_3$

- If player 1 is signed and player 3 is not, one of player 4 and 5 must be signed. Add appropriate constraints that signs player 4 or 5 when 1 is signed but 3 is not.

$$((x_1 = 1, x_3 = 0) \implies (x_4 = 1 \text{ or } x_5 = 1))$$

Thus add: $x_4 + x_5 \geq x_1 - x_3$

Integer programming is a very versatile formulation tool, almost to the level that you can formulate daily speech!

Sudoku Problem

Sudoku table of 9 subtables, each with 3 rows and 3 columns

1,1;1	1,2;1	1,3;1	1,1;2	1,2;2	1,3;2	1,1;3	1,2;3	1,3;3
2,1;1	2,2;1	2,3;1	2,1;2	2,2;2	2,3;2	2,1;3	2,2;3	2,3;3
3,1;1	3,2;1	3,3;1	3,1;2	3,2;2	3,3;2	3,1;3	3,2;3	3,3;3
1,1;4	1,2;4	1,3;4	1,1;5	1,2;5	1,3;5	1,1;6	1,2;6	1,3;6
2,1;4	2,2;4	2,3;4	2,1;5	2,2;5	2,3;5	2,1;6	2,2;6	2,3;6
3,1;4	3,2;4	3,3;4	3,1;5	3,2;5	3,3;5	3,1;6	3,2;6	3,3;6
1,1;7	1,2;7	1,3;7	1,1;8	1,2;8	1,3;8	1,1;9	1,2;9	1,3;9
2,1;7	2,2;7	2,3;7	2,1;8	2,2;8	2,3;8	2,1;9	2,2;9	2,3;9
3,1;7	3,2;7	3,3;7	3,1;8	3,2;8	3,3;8	3,1;9	3,2;9	3,3;9

Subtable k

1,1; k	1,2; k	1,3; k
2,1; k	2,2; k	2,3; k
3,1; k	3,2; k	3,3; k

Within subtable k ,
 i, j, k is the entry
in the i th row and
 j th column

Each cell must be filled with a number from 1 to 9. Each number can appear exactly once in each row of the sudoku table, once in each column of the sudoku table and once in each subtable.

Sudoku: Decision Variables+Constraints

Decision Variables

Let $y_{i,j;k}^m$ be 1 when the cell $(i, j; k)$ contains number m ; 0 otherwise.

Constraints

Suppose that some numbers already appear in some cells, list them:

- 3 appears in $(1, 2; 1)$, so $y_{1,2;1}^3 = 1$; 7 appears in $(2, 1; 1)$, so $y_{2,1;1}^7 = 1$.
- . . .
- $[2 @ (2, 1; 9)]$, so $y_{2,1;9}^2 = 1$; $[5 @ (3, 2; 9)]$, so $y_{3,2;9}^5 = 1$.

List \mathcal{L} whose members are $[m @ (i, j; k)]$: Number m already placed in cell $(i, j; k)$.

$$y_{i,j;k}^m = 1 \text{ for each given number } m \text{ in position } (i, j; k).$$

By using the list, we write these constraints with short-hand notation

$$y_{i,j;k}^m = 1 \text{ for } [m @ (i, j; k)] \in \mathcal{L}.$$

Sudoku: Constraints

Each number appears exactly once in each row

$$\sum_{k=1}^3 \sum_{j=1}^3 y_{i,j;k}^m = 1; \quad \sum_{k=4}^6 \sum_{j=1}^3 y_{i,j;k}^m = 1; \quad \sum_{k=7}^9 \sum_{j=1}^3 y_{i,j;k}^m = 1 \quad \text{for } i = 1, 2, 3, m = 1, \dots, 9.$$

Each number appears exactly once in each column

$$\sum_{k \in \{1,4,7\}} \sum_{i=1}^3 y_{i,j;k}^m = 1; \quad \sum_{k \in \{2,5,8\}} \sum_{i=1}^3 y_{i,j;k}^m = 1; \quad \sum_{k \in \{3,6,9\}} \sum_{i=1}^3 y_{i,j;k}^m = 1 \quad \text{for } j = 1, 2, 3,$$

$$m = 1, \dots, 9.$$

Each subtable has exactly one number m

$$\sum_{i=1}^3 \sum_{j=1}^3 y_{i,j;k}^m = 1 \quad \text{for } k, m = 1, \dots, 9.$$

Sudoku: Constraints+Objective

Each cell has a number

$$\sum_{m=1}^9 y_{i,j;k}^m = 1 \quad \text{for } k = 1, \dots, 9, \quad i, j = 1, 2, 3.$$

Objective Function

0*anything goes such as $0y_{1,1;1}^1$.