



Modelling with Integer Variables

Jesper Larsen

jesla@man.dtu.dk

Department of Management Engineering
Technical University of Denmark



What is an integer program?

Let us start with a linear program:

$$\begin{array}{ll} \max & cx \\ \text{s.t.} & Ax \leq b \\ & x \geq 0 \end{array}$$

where A is a m by n matrix, c is a vector of size n , b a vector of size m and x is a vector of (decision) variables.



Which is equivalent to writing:

$$\begin{array}{llll} \max & c_1x_1 & +c_2x_2 & \dots +c_nx_n \\ \text{s.t.} & a_{11}x_1 & +a_{12}x_2 & \dots +a_{1n}x_n \leq b_1 \\ & a_{21}x_1 & +a_{22}x_2 & \dots +a_{2n}x_n \leq b_2 \\ & \vdots & \vdots & \vdots \\ & a_{m1}x_1 & +a_{m2}x_2 & \dots +a_{mn}x_n \leq b_m \\ & x_1 \geq 0, & x_2 \geq 0, & \dots x_n \geq 0 \end{array}$$



Mixed Integer Program

Now if *some* but not all variables are integer, we have a **(linear) Mixed Integer Program (MIP)**:

$$\begin{array}{ll} \max & cx + hy \\ \text{s.t.} & Ax + Gy \leq b \\ & x \geq 0, \quad y \geq 0 \text{ and integer} \end{array}$$

where A is a m by n matrix, G is a m by p matrix, c is a vector of size n , h is a vector of size p , b is a vector of size m and x is a vector of (decision) variables, and finally y is a vector of **integer** (decision) variables.



Integer Program (IP)

If *all* variables are integer, we have an **Integer (Linear) Program**:

$$\begin{array}{ll} \max & cx \\ \text{s.t.} & Ax \leq b \\ & x \geq 0 \text{ and integer} \end{array}$$



Binary Integer Program (BIP)

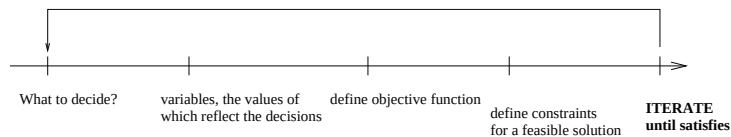
And if all variables are not only integer but restricted to the values 0 or 1, we have a **Binary Integer Program**:

$$\begin{array}{ll} \max & cx \\ \text{s.t.} & Ax \leq b \\ & x \in \{0, 1\}^n \end{array}$$



Different Models:

- How does one model?



Modelling is an **art**.



What is a “good” model?

- It is easy to understand the model,
- it is easy to detect errors in the model, and
- it is easy to compute the optimal solution.



Example 1: The 0-1-Knapsack Problem

We are going hiking. The knapsack has a capacity of b . Since not all items we would like to bring along fits in the knapsack we assign all items a weight w_i and a “profit” c_i .

- Now the goal is to choose the items that we want to bring along so that the capacity of the knapsack is not exceeded and the “profit” is maximized.



Example 2: The (linear) Assignment Problem

There are n people available to carry out n jobs. Each person is assign to carry out exactly one job, and each job need exactly one person assigned. There is an estimated cost c_{ij} if person i is assigned to job j .

- The goal is to the assignment with the minimum cost.



Innovative use of variables in IP

- Big- M notation
- Either-or constraints
- Functions with N possible values
- The Fixed Charge Problem



Example 3: The Uncapacitated Lot Sizing Problem

We need to schedule production over n time periods for a single product. We need to full fill a demand of d_t in time period t . The basic model can has additional data:

- f_t is the fixed cost of producing in period t
- p_t is unit production cost in period t
- h_t is unit storage cost in period t
- The goal is now to schedule our production as cheaply as possible.



k out of N constraints must hold (I)

$$\left. \begin{array}{l} f_1(x_1, x_2, \dots, x_n) \leq d_1 \\ f_2(x_1, x_2, \dots, x_n) \leq d_2 \\ \vdots \\ f_N(x_1, x_2, \dots, x_n) \leq d_N \end{array} \right\} k \text{ must hold}$$

This can be mastered by the following change



k out of N constraints must hold (II)

$$\begin{array}{ll} f_1(x_1, x_2, \dots, x_n) & \leq d_1 + My_1 \\ f_2(x_1, x_2, \dots, x_n) & \leq d_2 + My_2 \\ \vdots & \vdots \\ f_N(x_1, x_2, \dots, x_n) & \leq d_N + My_N \\ \sum_{i=1}^N y_i & = N - k \\ y_i & \in \{0, 1\} \end{array}$$



Either-or constraints (I)

Consider a case where one of two constraints must hold:

$$\begin{array}{ll} \text{Either} & 3x_1 + 2x_2 \leq 18 \\ \text{or} & x_1 + 4x_2 \leq 16 \end{array}$$

We need to reformulate into a mathematical model where all constraints specified must hold.



Either-or constraints (II)

Requirements can be rewritten as:

$$\begin{array}{ll} \text{Either} & 3x_1 + 2x_2 \leq 18 \\ & x_1 + 4x_2 \leq 16 + M \\ \text{or} & 3x_1 + 2x_2 \leq 18 + M \\ & x_1 + 4x_2 \leq 16 \end{array}$$

where M is a very very large positive number.



Either-or constraints (III)

This is equivalent to

$$\begin{aligned} 3x_1 + 2x_2 &\leq 18 + My \\ x_1 + 4x_2 &\leq 16 + M(1 - y) \end{aligned}$$

where y is a binary auxiliary variable. By using one binary variable for each constraint this idea can be generalised to more constraints.



Functions with N possible solutions

We have

$$f(x_1, x_2, \dots, x_n) = d_1 \text{ or } d_2 \text{ or } \dots \text{ or } d_N$$

$f()$ could be either a variable or a constraint.
Equivalent IP is

$$\begin{aligned} f(x_1, x_2, \dots, x_n) &= \sum_{i=1}^N d_i y_i \\ \sum_{i=1}^N y_i &= 1 \end{aligned}$$



The Travelling Salesman



- Tour of Sweden have 24978 nodes.
- TSP record: 528,280,881 nodes.
- Real-life applications of TSP are VLSI design and DNA sequencing.
- For more info see www.tsp.gatech.edu