Project Assignment in 42114 Integer Programming

Jesper Larsen
jesla@dtu.dk

Technical University of Denmark – Fall 2022

Introduction

The project period formally starts on **Friday 30 September** and the report must be handed in at the latest **Friday 4 November at 12:00**. Questions to be answered are clearly marked in "question boxes". Please read the text and the questions carefully before answering them.

Info: Your final report must be uploaded to the appropriate group assignment on DTU Learn of the course. The report is expected to be not more than 5 pages long **excluding** tables, figures and appendices and must be written in Danish or in English. The project may be solved in groups and permissible group sizes are 1, 2 and 3 persons.

1 Envelope production

Your background in Operations Research gives you an interesting task for a producer of envelopes (yes, envelopes still exist). This company makes the finest envelopes imaginable on some exquisite paper to the Royal Danish Court (RDC). The envelopes are square.

A number of documents should be put into the envelopes, exactly one document per envelope. Each document consist of one sheet of paper. The documents are square and side lengths of the documents that are to be put into the envelopes are between $120mm$ and $900mm$. The size of document i is denoted s_i.

To create some kind of order there is not an envelope for each possible size, that would simply be too expensive. Instead the RDC would like to work with S different sizes of envelopes. Each envelope will contain exactly one document, so we can ignore the "height"/"thickness" of the envelope. There are N documents in the sample data you have received.

The question is to decide on the size of the envelopes, so as to minimise the total amount of (surface area) of paper used to produce the envelopes.

1.1 The Assignment formulation

Your first idea is an assignment formulation. For $i = 1, \ldots, N$ and $j = 1, \ldots, S$ define x_{ij} by:

$$x_{ij} = \begin{cases} 1 & \text{if document } i \text{ is assigned to envelope type } j \\ 0 & \text{Otherwise} \end{cases}$$

Furthermore, we also define the variable a_j = area of envelope j for $j = 1, \ldots, S$. This results in the following model:

$$\begin{array}{lllll} \min & \displaystyle\sum_{i=1}^{N}\sum_{j=1}^{S} a_j x_{ij} & & & \\ \text{s.t.} & \displaystyle\sum_{j=1}^{S} x_{ij} = 1 & i = 1, \ldots, N & & \text{(assign)} \\ & x_{ij} = 1 \text{ then } a_j \geq s_i^2 & i = 1, \ldots, N; j = 1, \ldots, S & & \text{(fit)} \\ & x_{ij} \in \{0, 1\} & i = 1, \ldots, N; j = 1, \ldots, S & & \\ & a_j \geq 0 & j = 1, \ldots, S & & \end{array}$$

This mixed integer programming model (MIP) has two challenges: 1) the objective function is not linear and 2) the (fit) constraint is still not written as a constraint.

Question 1

Introduce the variable y_{ij} defined by $y_{ij} = a_j x_{ij}$ and use this to establish a model with a linear objective and linear constraints.

Question 2

Implement the MIP model in Julia. With this assignment also follows the file "envelopeF22.jl". The Julia file contains variable definitions and constraints that does not need to be changed, you need to introduce new variables and "rewritten" version of (fit) and objective function.

Question 3

The file also contains a test problem containing ten items and $S = 3$.

- Solve the problem to optimality as a MIP and solve LP relaxation of the MIP problem.
- Report the optimal values of both problems and the optimal solution of the MIP.
- Will there be more than one optimal solution of the MIP problem?
- Comment on the value of the LP relaxation.

1.2 The partitioning formulation

Another idea for a MIP model for the problem relies on defining the variable z_{ij} by:

$$z_{ij} = \begin{cases} 1 & \text{if documents } i \text{ through } j \text{ are put in the same size envelope (with edge size } s_j) \\ 0 & \text{Otherwise} \end{cases}$$

Here we introduce that the documents are *ordered* from the smallest to the largest. Therefore in this model if $z_{ij} = 1$ then all documents $i, i+1, \ldots, j-1, j$ are placed in the same size envelope – of edge size s_j in order to waste as little paper as possible.
The cost coefficient c_{ij} for z_{ij} can be calculated as $\sum_{i \le k \le j} s_j^2 - s_k^2$.

This is basically a set partitioning problem for all possible numbers in between 1 and N and we have to select exactly one interval for each document.

$$
\begin{aligned}
\min \quad & \sum_{i=1}^{N} \sum_{j=i}^{N} c_{ij} z_{ij} \\
\text{s.t.} \quad & \sum_{i=1}^{k} \sum_{j=k}^{N} z_{ij} = p_k \quad k = 1, \ldots, N && \text{(partitioning)} \\
& \sum_{i=1}^{N} \sum_{j=i}^{N} z_{ij} = L && \text{(total)} \\
& z_{ij} \in \{0, 1\} \qquad i = 1, \ldots, N; j = i, \ldots, N
\end{aligned}
$$

We need a variable z_{ij} for $j \ge i$. This can be calculated as $i + (i-1) + (i-2) + \ldots + 1$.

Question 4

In the presented model there are a number of parameters that needs to be assigned a value. What value should L have? And what about the p_k parameters?

1.3 Lagrangian relaxation

You consider to build a Lagrangian heuristic based on the partitioning formulation. Therefore you need to describe the Lagrangian relaxation of the problem. You decide to make the Lagrangian relaxation that keeps the constraint (total) as a constraint, but move the remaining constraints into the objective function. Denote the Lagrangian multiplier for constraint k as u_k.

Question 5

Present the Lagrangian relaxation for the partitioning formulation by relaxing the (partitioning) constraints. For fixed u how would you solve the Lagrangian relaxation? *tip: the summations become quite complicated, so in order to get a better understanding it can be beneficial to set N to a low number like 3 or 4 and then write up the relaxation in detail and from this generalize.*

Question 6

Given an optimal solution to the Lagrangian relaxation as defined above, define a Lagrangian heuristic that generates an feasible solution for the original problem. Your Lagrangian heuristic cannot rely on the values of the Lagrangian multipliers.

Question 7

Given the following Lagrangian relaxation solution $z_{25} = z_{46} = z_{8,10} = 1$ and all other variables zero. Use your Lagrangian heuristic and present the feasible solution it generates.

2 Production planning

A company is producing a variety of products at its production plant. In total, n different products can be produced. Producing a product is denoted a *job*. A job consists of a number of *operations*. An operation o_{ik} is determined by the job number i and the sequence number k. We assume that each job contain the same number of operations. This number we denote m, so job i contains operations $o_{i1}, o_{i2}, \ldots, o_{im}$ in that order.

Each operation o_{ik} has an associated processing time p_{ik} and on what machine the operation will take place d_{ik}. Let M be the set of machines, then we have $d_{ik} \in M$.

The total number of operations in the problem is $n \times m$. Let O be the set of all operations. Each machine has to process exactly n operations, one from each job, but it cannot process more than one at a time. As soon as a machine start processing an operation it has to finish it before a new operation can be processed.

The objective is to determine the start time for each operation on the specific machine corresponding to the operation such that the finishing time for the last operation is as early as possible (this is also known as the *makespan*).

2.1 Modelling the problem

For each operations o_{ik} we define a variable t_{ik}, which denotes the starting time for the operation. In addition, we also need a variable t that will measure the finish time. The production planning model now becomes:

$$
\begin{array}{lll}
min & t & \\
\text{s.t.} & t_{im} + p_{im} \leq t & i = 1, \ldots, n \\
 & t_{ik} + p_{ik} \leq t_{i(k+1)} & i = 1, \ldots, n; k = 1, \ldots, m-1 \\
 & (t_{ik} + p_{ik} \leq t_{jl}) \text{ or } (t_{jl} + p_{jl} \leq t_{ik}) & i, j, k, l \text{ with } d_{ik} = d_{jl} \\
 & t_{ij} \geq 0 & i = 1, \ldots, n; j = 1, \ldots, m \\
 & t \geq 0 &
\end{array}
$$

The constraint "$(t_{ik} + p_{ik} \leq t_{jl})$ or $(t_{jl} + p_{jl} \leq t_{ik})$" is called a *disjunctive constraint*. It identifies the order of the operations o_{ik} and o_{jl}; either o_{ik} is processed before o_{jl} or vice versa. If o_{ik} is processed before o_{jl} the starting time t_{jl} of o_{jl} must be larger than or equal to the starting time of o_{ik} (that is t_{ik}) plus the processing time of operation o_{ik}v(which is p_{ik}) and vice versa for the other combination.

Question 8

Rephrase the disjunctive constraints to ordinary conjunctive constraints by introducing binary variables y_{ikjl}. The variable is defined for each pair of operations (o_{ik}, o_{jl}) where $d_{ik} = d_{jl}$ and takes the value 1 if and only if operations o_{ik} preceeds operation o_{jl}.

2.2 Graph representation and longest path

In order to understand the structure of the problem better we define a small test problem. It consists of four jobs and three machines. The operations of the problem are shown in Table 1.

	Operation 1		Operation 2		Operation 3	
	Machine	Duration	Machine	Duration	Machine	Duration
Job1	1	3	2	7	3	9
Job2	2	7	1	10	3	2
Job3	3	3	2	4	1	4
Job4	1	5	3	5	2	6

Table 1: Sequence and processing time for four jobs to be processed on three machines.

One way of visualising the problem is to build a graph, in which each operation corresponds to a vertex. Additionally, the graph has two special vertices, s and t. If operation $o_{i(k+1)}$ is the immediate successor of operation o_{ik}, a directed edge from o_{ik} to $o_{i(k+1)}$ is added in the graph. In addition, there is a directed edge from s to the first operation in each job, and from the last operation in each job to t. Undirected edges are used to model that a set of operations belongs to a specific machine. Two operations o_{ik} and o_{jl} are connected with an undirected edge if and only if they are to be processed on the same machine. These edges are called disjunctive edges. We then have a clique for each machine. A clique is a subgraph with n vertices, in which all pairs of vertices are connected by an undirected edge. To describe the sequence, in which operations are processed on a particular machine, all undirected edges in the clique must be given a direction (corresponding to a precedence). A feasible solution satisfies that the resulting graph is acyclic, and the solution value, the makespan, corresponds to the longest path in the graph starting in s and ending in t. The length of the path is the sum of the processing times of the operations corresponding to the vertices of the path.

Question 9

Visualise the test problem.

2.3 Longest path – dynamic programming

Given a directed acyclic graph the graph represents a feasible solution for the problem. We can find the makespan of a feasible solution by using dynamic programming.

Question 10

Present a dynamic programming method for solving the longest path problem. Give the recursion for the dynamic programming that will compute the length of the longest path.

We specify the order of the jobs as a test of our method:

- Machine1: $o_{11}, o_{41}, o_{22}, o_{33}$
- Machine2: $o_{21}, o_{12}, o_{32}, o_{54}$
- Machine3: $o_{31}, o_{42}, o_{23}, o_{13}$

Question 11

Assign orientations to the edges in the cliques to produce an acyclic graph. Show the graph. Use the dynamic programming method for finding the longest path from s to t to find the makespan. Report the length of the longest path for each of the nodes in the graph as part of determining the makespan, that is, report the intermediate calculations of your dynamic programming method when finding the longest path from s to t.

2.4 Finding a good feasible solution

In Branch-and-Bound algorithms a good initial solution is of great value.

Question 12

Formulate a heuristic to produce a good feasible solution for the production planning problem, and apply the heuristic to the test problem.