Lecture 7

Synthesis, discrete optimisation and mixed-integer programming
Why discrete optimisation

Process synthesis
- System integration (integration of units, flowsheets and networks)
- Knowledge models (i.e. logical constraints, propositional logic)

Discontinuous systems
- Discontinuities (i.e. cost models of economics of scale)
- Procedural models (i.e. if in region A use this model; otherwise use model B)
Modelling with binary variables

Binary variable

\[ y = \begin{cases} 
1 & \text{if } p \text{ True} \\
0 & \text{if } p \text{ False} 
\end{cases} \]

- Integer formulation can resolve discontinuities
- Mixed-integer linear programming
“Either or” constraints

Original problem
If p is True then
\[ f_1(x) \leq 0 \]
else
\[ f_2(x) \leq 0 \]

\[ f_1(x) - M y \leq 0 \]
\[ f_2(x) - (1 - y) M \leq 0 \]

\( M \): large number

\[ \implies \] • Introduce binary variable for statement p
  • If p is True \( \Rightarrow y = 0 \)
  • If p is False \( \Rightarrow y = 1 \)

\[ y = 0 \rightarrow f_1(x) \leq 0 \]
\[ y = 1 \rightarrow f_2(x) \leq 0 \]
Modelling discontinuous functions

\[
\text{Cost} = \begin{cases} 
ax + b & L \leq x \leq U \\
0 & \text{otherwise}
\end{cases}
\]

\[
y = \begin{cases} 
1 & \text{if } x \in [L, U] \\
0 & \text{otherwise}
\end{cases}
\]

\[
x - U \cdot y \leq 0
\]

\[
-x + Ly \leq 0
\]

\[
\text{cost} = ax + b \cdot y
\]

\[
x \leq 0 \implies x = 0 \\
x \geq 0 \implies \text{cost} = 0
\]

\[
\text{cost} = ax + b \\
L \leq x \leq U
\]
Integer programming in nonlinear problems

Introduce $y_i$ binary variables:

$$y_i = \begin{cases} 1 & B_{i-1} \leq x \leq B_i \\ 0 & \text{otherwise} \end{cases}$$

$$\text{cost} = \sum_{i=1}^{n} (a_i y_i + b_i x_i)$$

$$x - B_i y_i \leq 0$$

$$-x + B_{i-1} y_i \leq 0$$

$$\text{cost} = a_k + b_k x$$

$$B_{k-1} \leq x \leq B_k$$

$$\text{cost} = a_j + b_j x$$

$$B_{j-1} \leq x \leq B_j$$
Systems integration

Main questions
– what units to use?
– How to interconnect these units
– sizes and operating conditions

Mathematical programming approach
(1) Generate structure that contains all possible connections: superstructure
(2) Formulate an optimisation problem based upon the superstructure and optimise it.
Superstructure example

\[ F_1 = F_2 = S_1 = S_2 = 0 \]

\[ F_2 = 0, S_2 = S_3 = 0 \]
Systems integration

• Advantages
  – Simultaneously addresses structural and parameter optimisation
  – Considers all interactions among the units
  – Provides systematic framework for the evaluation of the available design scenarios

• Disadvantages
  – Difficult or expensive to solve
  – Optimum relies on the postulated superstructure

• Modelling superstructures
  – Continuous approach
  – Mixed integer approach
Continuous vs Discrete formulation:

An Example
Continuous case

Superstructure \[ \Rightarrow \quad \min f(x) \quad \Rightarrow \quad \text{If } x_i \to 0, \text{ the stream associated with unit } i \text{ is eliminated} \]
\[ h(x) = 0 \quad g(x) \leq 0 \]

Example: Selection of materials

<table>
<thead>
<tr>
<th>Material</th>
<th>Yield</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Material A</td>
<td>80% yield</td>
<td>Cost = 55 F^{0.6}</td>
</tr>
<tr>
<td>Material B</td>
<td>66.7% yield</td>
<td>Cost = 40 F^{0.6}</td>
</tr>
</tbody>
</table>

Objective: Minimise cost

Cost of F: $50/unit

Market demand: 10 units
Superstructure

**Step 1:** Generation of superstructure

\[ x_0 \rightarrow x_1 \rightarrow \text{I} \rightarrow z_1 \rightarrow \text{II} \rightarrow x_2 \rightarrow z_2 \]

\[ z_1, z_2: \text{yields for A and B} \]

**Step 2:** Mathematical model

\[
\begin{align*}
\text{min} & \quad 55x_1^{0.6} + 40x_2^{0.6} + 50x_0 \\
\text{Subject to} & \quad x_0 = x_1 + x_2 \\
& \quad z_1 = 0.8x_1 \\
& \quad z_2 = 0.667x_2 \\
& \quad z_1 + z_2 = 10 \\
& \quad x_1, x_2, x_0 \geq 0
\end{align*}
\]

\[ \Rightarrow \text{DOF} = 1 \]
Optimal solution

• Optimum depends on the initial point
Discrete formulation

Cost approximation

\[ \text{Cost} = \text{Fixed cost} + \text{Variable cost} \]

Mathematical formulation

\[ \begin{align*}
\min \ & \ (75y_1 + 14x_1) + (55y_0 + 10x_2) + 50x_0 \\
\text{Subject to} \ & \ x_0 = x_1 + x_2 \\
\ & \ z_1 = 0.8x_1 \\
\ & \ z_2 = 0.667x_2 \\
\ & \ z_1 + z_2 = 10 \\
\ & \ x_0, x_1, x_2 \geq 0
\end{align*} \]
## Optimal solution

<table>
<thead>
<tr>
<th>Options for $y$</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>$y_1$  $y_2$</td>
<td>Infeasible</td>
</tr>
<tr>
<td>0     0</td>
<td>875  →  Optimum (irrespective of initial point)</td>
</tr>
<tr>
<td>1     0</td>
<td>955</td>
</tr>
<tr>
<td>0     1</td>
<td>930</td>
</tr>
<tr>
<td>1     1</td>
<td></td>
</tr>
</tbody>
</table>

- Optimum independent of the initial point
- Exhaustive enumeration require examination of $2^n$ combinations
Modelling logic

Mutually exclusive choices
Select between (only one choice is possible)
- Employer A ($y_A$)
- Employer B ($y_B$)
\[ y_A + y_B = 1 \]

Select between (at least one choice is possible)
- Transportation path A ($y_A$)
- Transportation path B ($y_B$)
\[ y_A + y_B \geq 1 \]

Select between (at most one choice is possible)
- Investment scheme A ($y_A$)
- Investment scheme B ($y_B$)
- Investment scheme C ($y_C$)
\[ y_A + y_B + y_C \leq 1 \]
Contingency decision

(a) B can only happen if A happens

\[ y_B - y_A \leq 0 \]

\[
\begin{align*}
y_A = 0 & \quad \Rightarrow \quad y_B \leq 0 \Rightarrow y_B = 0 \\
y_A = 1 & \quad \Rightarrow \quad y_B \leq 1 \quad \text{or} \quad y_B = 1
\end{align*}
\]

(b) If A does not exist, everything related/around A should be eliminated. Say \( Var_A \geq 0 \)

\[
Var_A - y_A \cdot UB_{Var_A} \leq 0
\]

\[
\begin{align*}
y_A = 0 & \quad \Rightarrow \quad 0 \leq Var_A \leq 0 \\
y_A = 1 & \quad \Rightarrow \quad Var_A \leq UB_{Var_A}
\end{align*}
\]

\[ \Rightarrow Var_A = 0 \]
Mixed-integer linear programming (MILP)

- Need to keep the model as linear

## Example

<table>
<thead>
<tr>
<th>Contingency</th>
<th>Linear formulation</th>
<th>Nonlinear formulation</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a)</td>
<td>$y_B - y_A \leq 0$</td>
<td>$y_B \cdot y_A = 0$</td>
</tr>
<tr>
<td>(b)</td>
<td>$Var_A - y_A \cdot UB \leq 0$</td>
<td>$Var_A \cdot y_A = 0$</td>
</tr>
</tbody>
</table>

Several methods to solve

(Essentially) impossible or difficult to solve
MILP techniques

- “Brute force” approach
- Relaxation method and “Naive’s approach”
- Branch and Bound: Numerous algorithms
- Benders decomposition
- Cutting plane algorithms
Brute force approach

Solve LPs for all 0-1 combinations of the

\[ y_i \quad i = 1, 2, \ldots n \]

Then select the minimum (or maximum) solution

\[ N_{LP} = 2^n \]

So check

<table>
<thead>
<tr>
<th># variables</th>
<th># LPs</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>32</td>
</tr>
<tr>
<td>100</td>
<td>( \sim 10^{30} )</td>
</tr>
</tbody>
</table>
Relax integer variables as continuous, so

\[ y \in \{0,1\} \implies 0 \leq y \leq 1 \]

Solve the resulting LP problem (whose optimal \textit{solution will not yield} necessarily \textit{integer solutions}). However, there exist some examples for which an integer solution is \textit{always} possible (assignment problem).
Naive’s approach

Basic idea: Generalise the concept and round a non-integer solution to the nearest integer.

\[ y_i^* = 0.8 \rightarrow y_i^* = 1.0 \quad y_i^* = 0.1 \rightarrow y_i^* = 0 \]

Example

\[
\begin{align*}
\min z &= -12y_1 - y_2 \\
y_1 + y_2 &\leq 1 \\
1.2y_1 + 0.5y_2 &\leq 1 \\
y_1, y_2 &\in \{0,1\}
\end{align*}
\]

Optimal solution: \((y_1^*, y_2^*) = (0,1)\)

Relaxed solution

\[
\begin{align*}
y_1^* &= 0.715 \rightarrow 1.0 \\
y_2^* &= 0.285 \rightarrow 0.0
\end{align*}
\]

\((y_1^*, y_2^*) = (1,0)\) is infeasible!
Branch and Bound
Solution search

Basic idea: Avoid exhaustive enumeration of all \{0,1\} combination by exploiting properties.

- Let \( z^* \) be the optimal solution of the binary tree. Then \( z^* \) will be one from \( z_8^*, \ z_9^*, \ldots \ w_{15}^* \).

- At every node \( i \), one can solve a relaxed LP \( z_i^* \) by setting integer values as continuous. If \( i \) is parent to \( j \) then
  \[
  z_i^* \leq z_j^* \text{ (assuming minimisation)}
  \]

So
- If \( z_i^* \) is infeasible \( \Rightarrow \) every thing under \( i \) is infeasible
- Consider that I solve relaxed LPs \( 1, 3, 6, 12 \) and \( 2 \) and that
  \[
  z_1^* \leq z_3^* \leq z_6^* \leq z_{12}^* \\
  z_1^* \leq z_2^*
  \]

Then
\[
  z_1^* \leq z^* \leq z_{12}^*
\]

Say now that \( z_{12}^* \leq z_2^* \)
Pruning the tree

Redundant part of the tree
Remarks

• Properties are never enough to determine moves. Heuristics are typically employed to complete the algorithm.

Usual questions

Which node to expand next?

Depth first: expand the most recently created

Breadth first: expand the node with the lower bound

Which variables should be set to integers?

Pick up the non-integer that is closest to 0.5

Pick up the first variable in the list
Intermediate solutions

• Generating and keeping multiple solutions
• Filtering solution pools (control properties of solutions)
• Use of incumbent filters
• Filling and updating solution pools is a dynamic field
  • E.g. use of big-data technology for acceleration
• Established solvers
  • CPLEX (IBM ILOG)
  • ZOOM
Benders decomposition

General formulation

\[
\begin{align*}
\min & \quad f(x, y) = c^T \cdot y + F(x) \\
x, y & \quad h(x, y) = A \cdot y + H(x) = 0 \\
& \quad g(x, y) = \beta y + G(x) \leq 0 \\
& \quad x \in IR, y \in \{0, 1\}
\end{align*}
\]

Basic idea

“Decompose” the problem into

- a **primal** problem that yields an upper bound (ONLY CONTINUOUS VARIABLES)
- a **master** problem that yields a lower bound (ONLY INTEGER VARIABLES)
- iterate (improving lower and upper bounds) until convergence.
Solution approach

Primal problem

Upper bound

Typically fixes the integer variables

Master problem

Lower bound

Makes use of marginal values of primal problem

Terminate

MI(N)LP formulation
DICOPT++

- Discrete and Continuous OPTimiser (Viswanathan and Grossman, EDRC/CMU)
- Algorithms have provisions to handle non-convexities, but do not necessarily obtain global optima
- Can handle only binary variables
- Usually for small number of integer variables (<50) and “reasonably” smooth formulations