Lecture 7: Using MILP to solve decision problems

- Decision problems
- Absolute values
- Transformation of a logical “or” into a logical “and”
- A formal definition of MILP
- Example: holding patterns
- Posing the number of holding patterns as a MILP
- A real example from Air Traffic Control

Graphical interpretation of decision problems

Some variables are on a grid, some are not:

Discrete order of arrival: deciding the order

First kind of decision variables: order of arrival

Now, every time you see an absolute value

Absolute value: not linear, not affine \( \rightarrow \) difficult

Absolute value can be expressed as a logical disjunction
(this is just a fancy way to say « or »).

Another way to express this:

\[ |t_1 - t_2| \geq \Delta \]

« and » is easy, « or » is difficult

Reminder: you have already used « and » many times

Minimize:

\[ c(x_1, x_2) = a_1x_1 + a_2x_2 \]

Subject to:

\[ a_1x_1 \leq c_{\text{max}} \]
\[ a_2x_2 \geq a_{\text{min}} \]
\[ a_1x_1 + a_2x_2 \geq a_{\text{min}} \]
\[ a_2x_2 \geq 2a_1x_1 \]
\[ a_2x_2 \leq 2a_1x_1 \]

All these are logical “and”

Transformation of an « or » into an « and »

Let us pick a very large number \( M \)
Let us pick a decision variable \( d \in \{0, 1\} \)

The two following statements are equivalent:

or

\[ \begin{cases} t_1 - t_2 \geq \Delta \\ t_2 - t_1 \geq \Delta \end{cases} \]

and

\[ \begin{cases} t_1 - t_2 \geq \Delta - Md \\ t_1 - t_2 \leq -\Delta + M(1 - d) \end{cases} \]
Two cases to investigate:

Case 1: $d=0$

$t_1 - t_2 \geq \Delta - Md$ becomes $t_1 - t_2 \geq \Delta$

Therefore, the second condition can be discarded.

Case 2: $d=1$

$t_1 - t_2 \leq -\Delta + M(1 - d)$ becomes $t_1 - t_2 \leq -\Delta + M$

i.e. $t_1 - t_2 \leq -\Delta + 1000000$

This is always true (just take $M$ large enough). Therefore, the second condition can be discarded.

Case 1: $d=0$

$t_1 - t_2 \geq \Delta - Md$ becomes $t_1 - t_2 \geq \Delta$

$t_1 - t_2 \leq -\Delta + M(1 - d)$ becomes $t_1 - t_2 \leq -\Delta + M$

i.e. $t_1 - t_2 \leq -\Delta + 1000000$

This is always true (just take $M$ large enough). Therefore, the second condition can be discarded.
Logical explanation

Two cases to investigate:

Case 1: \( d = 0 \)
\[
 t_1 - t_2 \geq \Delta - Md \quad \text{becomes} \quad t_1 - t_2 \geq \Delta
\]

Case 2: \( d = 1 \)
\[
 t_1 - t_2 \leq -\Delta + M(1-d) \quad \text{becomes} \quad t_1 - t_2 \leq -\Delta
\]
\[
 t_1 - t_2 \geq \Delta - Md \quad \text{becomes} \quad t_1 - t_2 \geq \Delta - M
\]
\[
 \text{i.e.} \quad t_1 - t_2 \geq \Delta - 1000000
\]

This is always true (just take \( M \) large enough). Therefore, the second condition can be discarded.

Summary

Two cases to investigate:

Case 1: \( d = 0 \)
\[
 t_1 - t_2 \geq \Delta - Md \quad \text{becomes} \quad t_1 - t_2 \geq \Delta
\]

Case 2: \( d = 1 \)
\[
 t_1 - t_2 \leq -\Delta + M(1-d) \quad \text{becomes} \quad t_1 - t_2 \leq -\Delta
\]

Depending on the value of \( d \):

Case 1: \( d = 0 \) \( t_1 - t_2 \geq \Delta \)

Case 2: \( d = 1 \) \( t_2 - t_1 \geq \Delta \)

In other words \( t_1 - t_2 \geq \Delta \quad \text{or} \quad t_2 - t_1 \geq \Delta \)

Transformation of an “or” into an “and”

Let us pick a very large number \( M \)
Let us pick a decision variable \( d \in \{0, 1\} \)

The two following statements are equivalent:

\[
\begin{align*}
\text{and} & \quad \begin{cases}
 t_1 - t_2 \geq \Delta - Md \\
 t_1 - t_2 \leq -\Delta + M(1-d)
\end{cases} \\
\text{or} & \quad \begin{cases}
 t_1 - t_2 \geq \Delta \\
 t_2 - t_1 \geq \Delta
\end{cases}
\end{align*}
\]

Why is it useful?

Now you can pose the problem of earliest arrival time of the last aircraft with decision enabled for order of arrival: you can deal with continuous and discrete variables.

\[
\begin{align*}
\text{min:} & \quad dt_1 + (1-d)t_2 \\
\text{Subject to:} & \quad t_1 - t_2 \geq \Delta - Md \\
& \quad t_1 - t_2 \leq -\Delta + M(1-d) \\
& \quad t_1 \leq b_1 \\
& \quad t_1 \geq a_1 \\
& \quad t_2 \leq b_2 \\
& \quad t_2 \geq a_2
\end{align*}
\]
**Why is it useful?**

Now you can pose the problem of earliest arrival time of the last aircraft with decision enabled for order of arrival: you can deal with continuous and discrete variables.

\[
\begin{align*}
\text{min:} & \quad dt_1 + (1-d)t_2 \\
\text{Subject to:} & \quad t_1 - t_2 \geq \Delta - Md \\
& \quad t_1 - t_2 \leq -\Delta + M(1-d) \\
& \quad t_1 \leq b_1 \\
& \quad t_1 \geq a_1 \\
& \quad t_2 \leq b_2 \\
& \quad t_2 \leq a_2
\end{align*}
\]

A formal definition of a MILP

A Mixed Integer Linear Program is a Linear Program in which some of the variables are continuous, and some are integer.

\[
\begin{align*}
\text{min:} & \quad dt_1 + (1-d)t_2 \\
\text{Subject to:} & \quad t_1 - t_2 \geq \Delta - Md \\
& \quad t_1 - t_2 \leq -\Delta + M(1-d) \\
& \quad t_1 \leq b_1 \\
& \quad t_1 \geq a_1 \\
& \quad t_2 \leq b_2 \\
& \quad t_2 \leq a_2
\end{align*}
\]

**Holding patterns: how many should an aircraft fly?**

A holding pattern delays an aircraft by a fixed amount of time, usually T=3 min.

**Question for ATC:**

how many holding patterns should one aircraft do before it is allowed to land?

CTAS tracks courtesy of NASA Ames

**A holding pattern is a shift by T**

![Diagram of an aircraft area with feasible arrival times indicating aircraft 1, aircraft 2, aircraft 3, and aircraft 4 with a holding pattern indicated as a shift by T between feasible arrival times a1 and b1 for aircraft 1, and a2 and b2 for aircraft 2, with no intersection for aircraft 3 and aircraft 4.](image-url)
A holding pattern is a shift by T

This is the set in which we want to schedule aircraft. We seek one arrival time for each aircraft in each of the colored sets.

A holding pattern can be expressed as a MILP

The number of holding patterns is a decision variable (the decision is actually made by the human Air Traffic Controller).
A holding pattern can be expressed as a MILP

The number of holding patterns is a decision variable (the decision is actually made by the human Air Traffic Controller).

\[
\begin{align*}
t_2 & \geq a_2 + 3T \\
t_2 & \leq b_2 + 3T
\end{align*}
\]

Actually, the human air traffic controller has the possibility to schedule aircraft 2 anywhere in the fourth time interval (i.e. with three holding patterns).

\[
t_2 \in [a_2 + 3T, b_2 + 3T]
\]

This is can be expressed in terms of two linear constraints involving integer and continuous variables

\[
\begin{align*}
t_2 & \geq a_2 + nT \\
t_2 & \leq b_2 + nT
\end{align*}
\]

for any admissible interval for aircraft 2:

An real example from transportation (air traffic control)

Problem: separating aircraft by \( \Delta = 3 \text{ min} \): how to schedule the aircraft so the last aircraft comes as early as possible.
Problem: separating aircraft by $\Delta = 3$ min: how to schedule the aircraft so the last aircraft comes as early as possible.

This is a Mixed Integer Linear Program (MILP):
- Some variables are integer (the order of arrival): 3, 1, 2, 4…
- Some variables are continuous (the times of arrival)
- The problem can be posed as a linear program involving both integer and continuous variables.