# PhD Title

Franco Peschiera

## Contents

- Maintenance planning is more important.

## The maintenance planning problem

Three main concepts in all maintenance planning:

- Resources.

*time*
Examples: industrial production, nurse rostering, aircraft.

# The maintenance planning problem

Three main concepts in all maintenance planning:

- Resources.
- Tasks.

*time*
Examples: industrial production, nurse rostering, aircraft.

## The maintenance planning problem

Three main concepts in all maintenance planning:

- Resources.
- Tasks.
- Recovery tasks.

*time*
Examples: industrial production, nurse rostering, aircraft.

# An MFMP problem

- Aircraft.

Objective:

# An MFMP problem

- Aircraft.
- Missions.

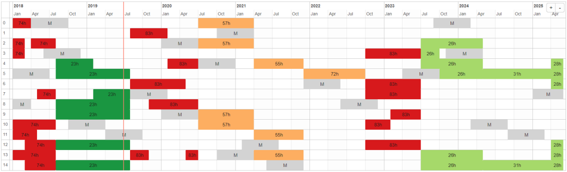
Objective:

# An MFMP problem

- Aircraft.
- Missions.
- Maintenances.

Objective:

## An MFMP solution



## Encoding of an MFMP solution

$$a_{it} = \begin{cases} -1 & \textit{check} \\ 0 & \textit{no assignment} \\ j & \textit{mission } j \end{cases}$$

Table format: a solution $x$ is represented by a matrix $A = \mathbb{Z}^{I \times T}$.
Patterns: $p \in \mathcal{P}$
$p = \{a_{i0}, a_{it}, a_{it+1}, ..., a_{iT}\}$
Pattern format: a solution $x$ is represented by a mapping $f : \mathcal{I} \to \mathcal{P}$.

# Outline

1. Introduction
2. **Complexity and exact methods**
3. Pattern-like modeling and machine learning
4. Graph-based VND
5. Conclusions

## Complexity analysis

safdasdf

## Exact methods

$a_{jit}$ : $=1$ if mission $j \in J$ in period $t \in \mathcal{T}_j$ is realized with aircraf
0 otherwise.

$m_{it}$ : $=1$ if aircraft $i \in I$ starts a check in period $t \in \mathcal{T}$, 0 other

$u_{it}$ : flown time (continuous) by aircraft $i \in I$ during period $t \in$

$$u_{it} \geq \sum_{j \in \mathcal{J}_t \cap \mathcal{O}_i} a_{jti} H_j \qquad t = 1, ..., \mathcal{T}, i \in \mathcal{I} \qquad (1)$$

$$u_{it} \geq U^{min}(1 - \sum_{t' \in \mathcal{T}_t^s} m_{it'}) \qquad t = 1, ..., \mathcal{T}, i \in \mathcal{I} \qquad (2)$$

$$rft_{it} \leq rft_{i(t-1)} - u_{it} + H^M m_{it} \qquad t = 1, ..., \mathcal{T}, i \in \mathcal{I} \qquad (3)$$

$$rft_{it} \in [0, H^M] \qquad t \in \mathcal{T}, i \in \mathcal{I} \qquad (4)$$

## Contents

1. Introduction.

# Exact methods

$a_{jit}$ : =1 if mission $j \in J$ in period $t \in \mathcal{T}_j$ is realized with aircraf
0 otherwise.

$m_{it}$ : =1 if aircraft $i \in I$ starts a check in period $t \in \mathcal{T}$, 0 other

$u_{it}$ : flown time (continuous) by aircraft $i \in I$ during period $t \in$

$$u_{it} \geq \sum_{j \in \mathcal{J}_t \cap \mathcal{O}_i} a_{jti} H_j \qquad t = 1, ..., \mathcal{T}, i \in \mathcal{I} \qquad (1)$$

$$u_{it} \geq U^{min}(1 - \sum_{t' \in \mathcal{T}_t^s} m_{it'}) \qquad t = 1, ..., \mathcal{T}, i \in \mathcal{I} \qquad (2)$$

$$rft_{it} \leq rft_{i(t-1)} - u_{it} + H^M m_{it} \qquad t = 1, ..., \mathcal{T}, i \in \mathcal{I} \qquad (3)$$

$$rft_{it} \in [0, H^M] \qquad t \in \mathcal{T}, i \in \mathcal{I} \qquad (4)$$

# Contents

1. Introduction.
2. Complexity analysis and exact methods.

## Exact methods

$a_{jit}$ : =1 if mission $j \in J$ in period $t \in \mathcal{T}_j$ is realized with aircraf
0 otherwise.
$m_{it}$ : =1 if aircraft $i \in I$ starts a check in period $t \in \mathcal{T}$, 0 other
$u_{it}$ : flown time (continuous) by aircraft $i \in I$ during period $t \in$

$$u_{it} \geq \sum_{j \in \mathcal{J}_t \cap \mathcal{O}_i} a_{jti} H_j \qquad t = 1, ..., \mathcal{T}, i \in \mathcal{I} \qquad (1)$$

$$u_{it} \geq U^{min}(1 - \sum_{t' \in \mathcal{T}_t^s} m_{it'}) \qquad t = 1, ..., \mathcal{T}, i \in \mathcal{I} \qquad (2)$$

$$rft_{it} \leq rft_{i(t-1)} - u_{it} + H^M m_{it} \qquad t = 1, ..., \mathcal{T}, i \in \mathcal{I} \qquad (3)$$

$$rft_{it} \in [0, H^M] \qquad t \in \mathcal{T}, i \in \mathcal{I} \qquad (4)$$

## Contents

# Exact methods

| | |
|---|---|
| $a_{jit}$ | : =1 if mission $j \in J$ in period $t \in \mathcal{T}_j$ is realized with aircraf<br>0 otherwise. |
| $m_{it}$ | : =1 if aircraft $i \in I$ starts a check in period $t \in \mathcal{T}$, 0 otherw |
| $u_{it}$ | : flown time (continuous) by aircraft $i \in I$ during period $t \in$ |

$$u_{it} \geq \sum_{j \in \mathcal{J}_t \cap \mathcal{O}_i} a_{jti} H_j \qquad t = 1, ..., \mathcal{T}, i \in \mathcal{I} \qquad (1)$$

$$u_{it} \geq U^{min}(1 - \sum_{t' \in \mathcal{T}_t^s} m_{it'}) \qquad t = 1, ..., \mathcal{T}, i \in \mathcal{I} \qquad (2)$$

$$rft_{it} \leq rft_{i(t-1)} - u_{it} + H^M m_{it} \qquad t = 1, ..., \mathcal{T}, i \in \mathcal{I} \qquad (3)$$

$$rft_{it} \in [0, H^M] \qquad t \in \mathcal{T}, i \in \mathcal{I} \qquad (4)$$

# Contents

1. Introduction.
2. Complexity analysis and exact methods.

# Outline

1. Introduction
2. Complexity and exact methods
3. **Pattern-like modeling and machine learning**
4. Graph-based VND
5. Conclusions

## New formulation

$a_{ijtt'}$ : =1 if aircraft $i$ starts an assignment to mission $j$ at the b of period $t$ and finishes at the end of period $t'$, zero otherwi

$m_{ip}$ : =1 if aircraft $i$ uses check pattern $p$, zero otherwise. each p has a single feasible combination of check starts for an aircraf the whole planning (usually only 1-2 checks per aircraft).

$$\sum_{(j,t,t')\in\mathcal{JTT}_{ic}} a_{ijtt'} H'_{jtt'} + U'_{tc} \leq H^M + M(1 - m_{ip})$$

$$i \in \mathcal{I}, p \in \mathcal{P}, c \in \mathcal{C}_p \tag{5}$$

(6)

# Formulation

$$\text{Max} \sum_{i \in \mathcal{I}, p \in \mathcal{P}} m_{ip} \times W_p \tag{7}$$

$$\sum_{i \in \mathcal{I}, p \in \mathcal{P}_t} m_{ip} \leq C^{max} \qquad t \in \mathcal{T} \tag{8}$$

$$\sum_{i \in \mathcal{I}_j, (t_1, t_2) \in \mathcal{T}_{jt}} a_{ijt_1t_2} \geq R_j \qquad j \in \mathcal{J}, t \in \mathcal{T}\mathcal{J}_j \tag{9}$$

$$\sum_{p \in \mathcal{P}_t} m_{ip} + \sum_{j \in \mathcal{J}_t \cap \mathcal{J}_i} \sum_{(t_1, t_2) \in \mathcal{T}_{jt}} a_{ijt_1t_2} \leq 1 \qquad t \in \mathcal{T}, i \in \mathcal{I} \tag{10}$$

$$\sum_{(j,t,t') \in \mathcal{J}\mathcal{T}\mathcal{T}_{ic}} a_{ijtt'} H'_{jtt'} + U'_{tc} \leq H^M + M(1 - m_{ip}) \qquad i \in \mathcal{I}, p \in \mathcal{P}, c \in \mathcal{C}_p$$

## New vs old formulation

1. It uses 3 times the number of constraints and 3 times the number of variables.

## New vs old formulation

1. It uses 3 times the number of constraints and 3 times the number of variables.

- variables: 11000 => 28000.

## New vs old formulation

1. It uses 3 times the number of constraints and 3 times the number of variables.

- variables: 11000 => 28000.
- constraints: 13000 => 48000.

## New vs old formulation

1. It uses 3 times the number of constraints and 3 times the number of variables.

- variables: 11000 => 28000.
- constraints: 13000 => 48000.

1. It is still better. Better lineal relaxation, better performance.

## New vs old formulation

1. It uses 3 times the number of constraints and 3 times the number of variables.

- variables: 11000 => 28000.
- constraints: 13000 => 48000.

1. It is still better. Better lineal relaxation, better performance.
2. Can we reduce intelligently the number of variables?

## New vs old formulation

1. It uses 3 times the number of constraints and 3 times the number of variables.

- variables: 11000 => 28000.
- constraints: 13000 => 48000.

1. It is still better. Better lineal relaxation, better performance.
2. Can we reduce intelligently the number of variables?

- we can.

## Predicting pseudo-cuts

Let an optimization problem with input data $x$ and solution space $y \in \mathcal{Y}(x)$. We want to find the optimal solution given a cost function $C(x, y)$:

$$y^*(x) :\equiv arg \min_{y \in \mathcal{Y}(x)} C(x, y)$$

For any solution $y$, we can calculate $N$ several features $g_n(y) \; \forall n \in \{1, .., N\}$.

$$\hat{g}_n(x) \approx g_n(y^*) \; \forall n \in \{1, .., N\}$$

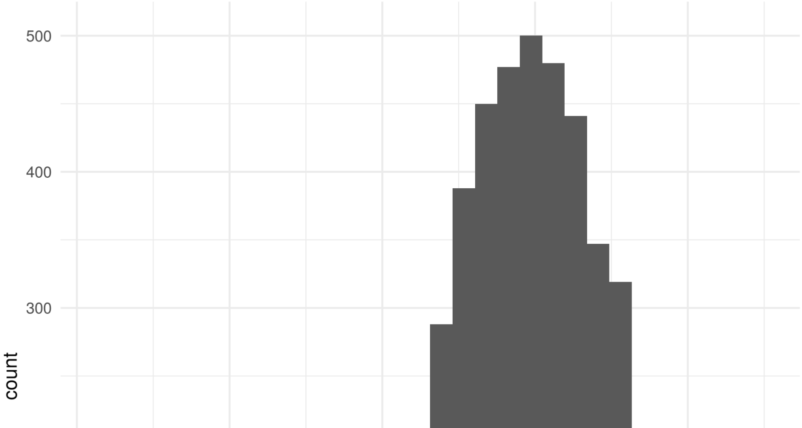
We then use that information to reduce the solution space $\mathcal{Y}'(x) \subset \mathcal{Y}(x)$ and solve it:

$$\hat{y}^*(x) = arg \min_{y \in \mathcal{Y}'(x)} C(x, y)$$

The trick is doing so without losing optimality:

## Distance between maintenances

.pull-left[ * The distance between maintenance has a maximum of $E^M$ periods. * Depending on the instance, the optimal distance can be shorter. * This distance conditions the total number of patterns to create.]
.pull-right[



## Forecasting + Optimization

**We want to**:

1. Train a statistical model to predict the mean distance between maintenances for any given instance.

–

**Benefits**:

–

**The better we're able to predict the optimal distance between maintenances for the whole fleet, the less optimality**

## Forecasting + Optimization

**We want to**:

1. Train a statistical model to predict the mean distance between maintenances for any given instance.
2. Use this information to limit all possible combinations of patterns to generate.

–

**Benefits**:

–

**The better we're able to predict the optimal distance between maintenances for the whole fleet, the less optimality**

## Forecasting + Optimization

**We want to**:

1. Train a statistical model to predict the mean distance between maintenances for any given instance.
2. Use this information to limit all possible combinations of patterns to generate.

–

**Benefits**:

1. **Performance**: a smaller model is easier to solve.

–

**The better we're able to predict the optimal distance between maintenances for the whole fleet, the less optimality**

## Forecasting + Optimization

**We want to**:

1. Train a statistical model to predict the mean distance between maintenances for any given instance.
2. Use this information to limit all possible combinations of patterns to generate.

—

**Benefits**:

1. **Performance**: a smaller model is easier to solve.
2. **User feedback**: direct feedback about the solution without needing to solve any model.

—

**The better we're able to predict the optimal distance between maintenances for the whole fleet, the less optimality**

# Forecasting + Optimization

**We want to**:

1. Train a statistical model to predict the mean distance between maintenances for any given instance.
2. Use this information to limit all possible combinations of patterns to generate.

–

**Benefits**:

1. **Performance**: a smaller model is easier to solve.
2. **User feedback**: direct feedback about the solution without needing to solve any model.
3. **More stable solutions**: Every aircraft flies an amount that is closest to the mean of the fleet.

–

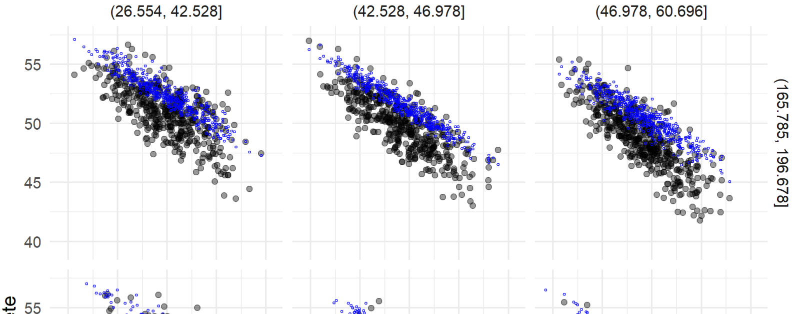
**The better we're able to predict the optimal distance between maintenances for the whole fleet, the less optimality**

# Prediction model

.pull-left[ * **Technique**: *Quantile regressions* to estimate upper and lower bounds. * **Training**: 5000 small instances. * **Input features**: * mean flight demand per period, * total remaining flight hours at start (init), * variance of flight demand, * demand of special missions, * number of period where flight demand is cut in two. * **Output features**: mean distance between maintenances.]

–

.pull-right[



# Experiments

- Number of instances: medium (1000), large (1000) and very large (1000).

–

Largest instances have 60 aircraft, 90 periods, ˜30 missions (4 active missions at any given time).

$$m_{ip} = 0 \qquad p_{t'} - p_t < \hat{\mu}^{lb}_{t'-t} - tol \qquad (13)$$

## Experiments

- Number of instances: medium (1000), large (1000) and very large (1000).
- Time limit at 3600 seconds.

Largest instances have 60 aircraft, 90 periods, ˜30 missions (4 active missions at any given time).

$$m_{ip} = 0 \qquad p_{t'} - p_t < \hat{\mu}^{lb}_{t'-t} - tol \qquad (13)$$

## Experiments

- Number of instances: medium (1000), large (1000) and very large (1000).
- Time limit at 3600 seconds.
- We seeded instance generation for better comparison.

–

Largest instances have 60 aircraft, 90 periods, ~30 missions (4 active missions at any given time).

$$m_{ip} = 0 \qquad p_{t'} - p_t < \hat{\mu}^{lb}_{t'-t} - tol \qquad (13)$$

## Experiments

- Number of instances: medium (1000), large (1000) and very large (1000).
- Time limit at 3600 seconds.
- We seeded instance generation for better comparison.
- CPLEX running 1 thread.

–

Largest instances have 60 aircraft, 90 periods, ˜30 missions (4 active missions at any given time).

$$m_{ip} = 0 \qquad p_{t'} - p_t < \hat{\mu}^{lb}_{t'-t} - tol \qquad (13)$$

## Experiments

- Number of instances: medium (1000), large (1000) and very large (1000).
- Time limit at 3600 seconds.
- We seeded instance generation for better comparison.
- CPLEX running 1 thread.

–

Largest instances have 60 aircraft, 90 periods, ˜30 missions (4 active missions at any given time).

1. Create forecasting model based in 5000 small instances.

$$m_{ip} = 0 \qquad p_{t'} - p_t < \hat{\mu}^{lb}_{t'-t} - tol \qquad (13)$$

## Experiments

- Number of instances: medium (1000), large (1000) and very large (1000).
- Time limit at 3600 seconds.
- We seeded instance generation for better comparison.
- CPLEX running 1 thread.

–

Largest instances have 60 aircraft, 90 periods, ˜30 missions (4 active missions at any given time).

1. Create forecasting model based in 5000 small instances.
2. Use forecasting model to predict bounds on distance between maintenances: $\hat{\mu}^{lb}_{t'-t}$, $\hat{\mu}^{ub}_{t'-t}$.

$$m_{ip} = 0 \qquad p_{t'} - p_t < \hat{\mu}^{lb}_{t'-t} - tol \qquad (13)$$

## Experiments

- Number of instances: medium (1000), large (1000) and very large (1000).
- Time limit at 3600 seconds.
- We seeded instance generation for better comparison.
- CPLEX running 1 thread.

–

Largest instances have 60 aircraft, 90 periods, ~30 missions (4 active missions at any given time).

1. Create forecasting model based in 5000 small instances.
2. Use forecasting model to predict bounds on distance between maintenances: $\hat{\mu}^{lb}_{t'-t}$, $\hat{\mu}^{ub}_{t'-t}$.
3. Implement the pseudo-cut:

$$m_{ip} = 0 \qquad\qquad p_{t'} - p_t < \hat{\mu}^{lb}_{t'-t} - tol \tag{13}$$

## Experiments

- Number of instances: medium (1000), large (1000) and very large (1000).
- Time limit at 3600 seconds.
- We seeded instance generation for better comparison.
- CPLEX running 1 thread.

–

Largest instances have 60 aircraft, 90 periods, ~30 missions (4 active missions at any given time).

1. Create forecasting model based in 5000 small instances.
2. Use forecasting model to predict bounds on distance between maintenances: $\hat{\mu}^{lb}_{t'-t}$, $\hat{\mu}^{ub}_{t'-t}$.
3. Implement the pseudo-cut:

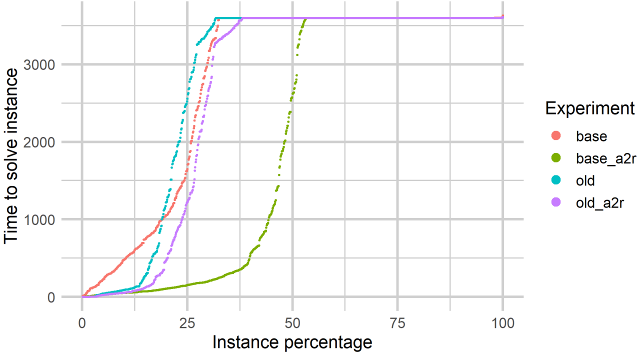
$$m_{ip} = 0 \qquad p_{t'} - p_t < \hat{\mu}^{lb}_{t'-t} - tol \qquad (13)$$

# How good is it (performance)

Faster solutions, more solutions.

–

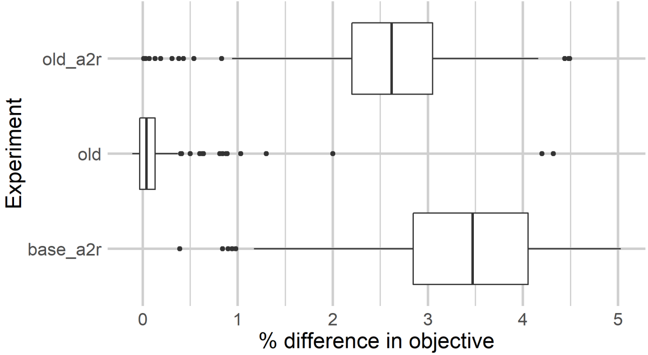
.center[




]

# How good is it (optimality)

For instances were an optimal solution was found (optimum degradation): * 95% of instances had less than 4% gap with real optimal.
.center[




]

## Further steps

- **Better predictions** with better features, or predicting several characteristics of optimal solutions.

# Further steps

- **Better predictions** with better features, or predicting several characteristics of optimal solutions.
- **Predict a distribution** and sample patterns from the distribution instead of predicting patterns.

# Further steps

- **Better predictions** with better features, or predicting several characteristics of optimal solutions.
- **Predict a distribution** and sample patterns from the distribution instead of predicting patterns.
- **Warm-start Column Generation** with a selected subset of potentially good patterns.

# Further steps

- **Better predictions** with better features, or predicting several characteristics of optimal solutions.
- **Predict a distribution** and sample patterns from the distribution instead of predicting patterns.
- **Warm-start Column Generation** with a selected subset of potentially good patterns.
- **Automatize prediction** so it can be easily integrated in other problems.

# Outline

# Outline

NULL