Airline disruption management with aircraft swapping and reinforcement learning

G. Hondet, L. Delgado, G. Gurtner

École nationale de l’aviation civile

December 5, 2018
Introduction

- Lower costs due to airline disruptions
- Usually, Disruption solution man made by rule of thumb
- Aircraft or flight swapping
- Reinforcement learning
Current work

• V. Mnih et al., “Playing Atari with deep reinforcement learning”, 2013

Work done here
Machine learning technique to discover interesting swap combinations
Disruption management with reinforcement learning

Hondet, Delgado, Gurtner

Introduction

Simulator

Specification
Mechanisms
Cost & calibration

Q learning as solver

Principle and method
Q learning algorithm and implementation
Practical training with the simulator

Experiments and results

Experimental setup
Results
Disruption management with reinforcement learning

Hondet, Delgado, Gurtner

Introduction

Simulator Specification
Mechanisms
Cost & calibration

Q learning as solver
Principle and method
Q learning algorithm and implementation
Practical training with the simulator

Experiments and results
Experimental setup
Results
Specification of simulator

Purpose

• Evaluate the delay on a fleet, on a day of operation
• estimate generated costs
• perform actions on the fleet

Does

• model reactionary delay
• include other delays as probability distributions
• simulate aircraft swapping and its consequences

Does not

• model crew management, nor passengers flow
• manage stand-by aircraft,
• modify or cancel legs
Mechanisms

Timestep
\( \forall i \in [1, m], t_i \) is the time of the \( i^{th} \) landing of the day

\((t_1, t_2, \ldots, t_m)\)

Actions
Allow to alter the simulation,

“swap with aircraft \( a \)”

Cost
Immediate cost of a swap

“swapping with \( a \) costs \( c \)”
Cost & calibration

Cost of what
Delay at departure of the flight after swap

Characteristics
- non linear
- increasing derivative
  \[ c(d_1 + d_2) > c(d_1) + c(d_2) \]
- depends on the aircraft type

Calibration
Calibrated against Eurocontrol “Coda Digest 2017”
Disruption management with reinforcement learning
Hondet, Delgado, Gurtner

Introduction
Simulator
Q learning as solver
Principle and method
Q learning
Practical training
Experiments and results
Conclusion

Outline

1. Simulator
   Specification
   Mechanisms
   Cost & calibration

2. Q learning as solver
   Principle and method
   Q learning algorithm and implementation
   Practical training with the simulator

3. Experiments and results
   Experimental setup
   Results
Reinforcement learning

- Interaction between an agent and its environment
- Find a policy $\pi$: state $\rightarrow$ action

Figure: Reinforcement learning principle
Theoretical basis

State $s \in S$, action $a \in A$.

Maximised value

$$E\left(\sum_{t=0}^{T_f} r_t\right) \leftrightarrow Q(s, a) \quad (1)$$

Bellman equation

$$Q^*(s, a) = r(s, a) + \sum_{s' \in S} p(s'|s, a) \max_{a'} Q^*(s', a') \quad (2)$$

- Dynamic programming
- Monte Carlo simulations
Q learning algorithm

\textbf{procedure} \textsc{Q-learning}(Q)\textbf{end procedure}

\begin{verbatim}
  $s \leftarrow$ initial state
  \textbf{while} episode not finished \textbf{do}
    $a \leftarrow$ choose an action from a set
    play $a$, observe reward $r$ and new state $s'$
    $Q \leftarrow$ update $Q$ with ($s, a, r, s'$)
    $s \leftarrow s'$
  \textbf{end while}
\end{verbatim}
Lookup table implementation

\[ Q(s_0, a_0) \quad Q(s_0, a_1) \quad \cdots \]
\[ Q(s_1, a_0) \quad Q(s_1, a_1) \quad \cdots \]
\[ \vdots \quad \vdots \]
\[ Q(s, a) \]

Update formula

State \( s \), action \( a \), reward \( r \) and next state \( s' \).

\[ Q(s, a) \leftarrow Q(s, a) + \alpha \left( r + \max_{a'} Q(s', a') - Q(s, a) \right) \quad (3) \]
Choosing an action

Bandit methods
Maximise reward, minimise regret

Upper confidence bound

\[ Q_t(s, a) + \sqrt{\frac{C}{tN_t(s, a)}} \]

exploitation
Choosing an action

Bandit methods
Maximise reward, minimise regret

Upper confidence bound

\[ Q_t(s,a) + c \sqrt{\frac{\ln t}{N_t(s,a)}} \]

(4)
Final algorithm

procedure Q-LEARNING\( (Q, c, \alpha, A) \)
\[
\begin{align*}
    s & \leftarrow \text{initial state} \\
    \textbf{while} \text{ episode not finished } \textbf{do} \\
    \quad a & \leftarrow \text{CHOOSEACTION}(A, c) \\
    \quad (r, s') & \leftarrow \text{SIMULATIONSTEP}(s, a) \\
    \quad Q(s, a) & \leftarrow Q(s, a) + \alpha \left[ r_t + \max_{a' \in A} Q(s', a') - Q(s, a) \right] \\
    \quad s & \leftarrow s' \\
    \textbf{end while} \\
\end{align*}
\]
end procedure
Implementing the training

Hyperparameters

- Exploitation exploration trade off
- Initial $Q$ value

Learning rate

$$\sum_{n \geq 0} \alpha_n = \infty; \quad \sum_{n \geq 0} \alpha_n^2 \in \mathbb{R} \quad (5)$$

$$\alpha_n = \frac{1}{N_t(s, a)} \quad (6)$$

Chaining training sessions

$$Q^1 \xrightarrow{\text{training}} Q^2 \xrightarrow{\text{training}} \ldots \xrightarrow{\text{training}} Q^* \quad (7)$$
Observation
Partial information of the environment, $\mathcal{O}$ the set of observations,

$$(S, A) \xrightarrow{\phi} (\mathcal{O}, A) \xrightarrow{Q} \mathbb{R}$$

Choice of $\phi$

- Carries enough information
- But not too specific
- Time independent
Disruption management with reinforcement learning

Hondet, Delgado, Gurtner

Outline

1 Simulator
   Specification
   Mechanisms
   Cost & calibration

2 Q learning as solver
   Principle and method
   Q learning algorithm and implementation
   Practical training with the simulator

3 Experiments and results
   Experimental setup
   Results
Experimental setup

- Schedule: Vueling, October 12, 2014
- 6 aircraft, 14 stations, 35 flights

**Observation**
Two different observations tested.

**Disruption**
Artificial delay added.

**Hyperparameters**

\[(p_d, c, q_i) = (0.06, 10, -90000)\]
## Output format

### Spreadsheet like parquet files

**Columns**

- delays
  - atfm delay
  - departure delay
  - miscellaneous delays
  - reactionary delay
  - artificial delay added
  - taxi time
- action and reward
  - action number
  - swap or not
  - cost
  - cumulative reward
- simulation information
  - departure destination
  - departure origin
  - leg duration
  - departure sobt
  - tail number
  - tail number of swapped aircraft
  - time in the simulation
- Q learning data
  - state-action couple visit count
  - Q value
Disruption management with reinforcement learning

Hondet, Delgado, Gurtner

Introduction
Simulator
Q learning as solver
Experiments and results
Experimental setup
Results
Conclusion

Learning process

Figure: Average maximum Q values over 5000 episodes.
Comparing with idle behaviour

Figure: Comparing the idle behaviour with the agent.
Conclusion

Results
Cost reduced in some conditions, not reliable enough. Potential lines of research.

Perspectives

- refine observations
- more sophisticated reinforcement learning techniques
- develop further the simulator