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

- J. Clausen *et al.* “Disruption management in the airline industry — concepts, models and methods”, 2009
- R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2017
- V. Mnih *et al.*, “Playing Atari with deep reinforcement learning”, 2013

Work done here

Machine learning technique to discover interesting swap combinations

Outline

Outline

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"

Outline

Principle

Reinforcement learning

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

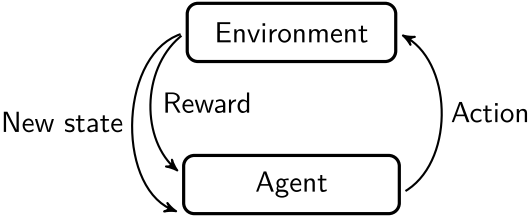

Figure: Reinforcement learning principle

Theoretical basis

State $s \in \mathcal{S}$, action $a \in \mathcal{A}$.

Maximised value

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

Bellman equation

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

- Dynamic programming
- Monte Carlo simulations

Q learning algorithm

procedure Q-LEARNING(Q)
  $s \leftarrow$ initial state
  while episode not finished **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')
    $s \leftarrow s'$
  end while
end procedure

Lookup table implementation

$$Q = \begin{bmatrix} Q(s_0, a_0) & Q(s_0, a_1) & \cdots & \\ Q(s_1, a_0) & Q(s_1, a_1) & \cdots & \\ \vdots & \vdots & & \\ & & Q(s, a) & \\ & & & \end{bmatrix}$$

Update formula

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

$$Q(s, a) \leftarrow Q(s, a) + \alpha \left(r + \max_{a'} Q(s', a') - Q(s, a) \right) \tag{3}$$

Choosing an action

Bandit methods

Maximise reward, minimise regret

Upper confidence bound

$$\underbrace{Q_t(s, a)}_{\text{exploitation}} +$$

Choosing an action

Bandit methods

Maximise reward, minimise regret

Upper confidence bound

$$\underbrace{Q_t(s, a)}_{\text{exploitation}} + c \underbrace{\sqrt{\frac{\ln t}{N_t(s, a)}}}_{\text{exploration}} \tag{4}$$

Final algorithm

procedure $\textsc{Q-learning}(Q, c, \alpha, \mathcal{A})$

  $s \leftarrow$ initial state

  while episode not finished **do**

    $a \leftarrow \textsc{ChooseAction}(\mathcal{A}, c)$

    $(r, s') \leftarrow \textsc{SimulationStep}(s, a)$

    $Q(s, a) \leftarrow Q(s, a) + \alpha \left[r_t + \max_{a' \in \mathcal{A}} Q(s', a') - Q(s, a) \right]$

    $s \leftarrow s'$

  end while

end procedure

Implementing the training

Hyperparameters

- Exploitation exploration trade off
- Initial Q value

Learning rate

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

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

Chaining training sessions

$$Q^1 \xrightarrow{\text{training}} Q^2 \xrightarrow{\text{training}} \cdots \xrightarrow{\text{training}} Q^* \qquad (7)$$

State space

Observation

Partial information of the environment, $\mathcal{O}$ the set of observations,

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

Choice of ϕ

- Carries enough information
- But not too specific
- Time independent

Outline

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

Learning process

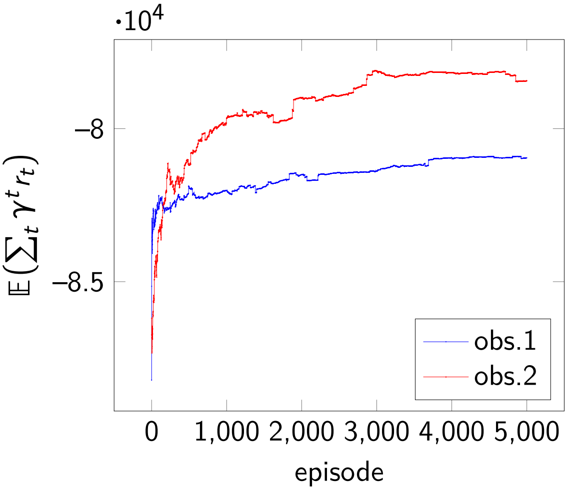

Figure: Average maximum Q values over 5000 episodes.

Comparing with idle behaviour

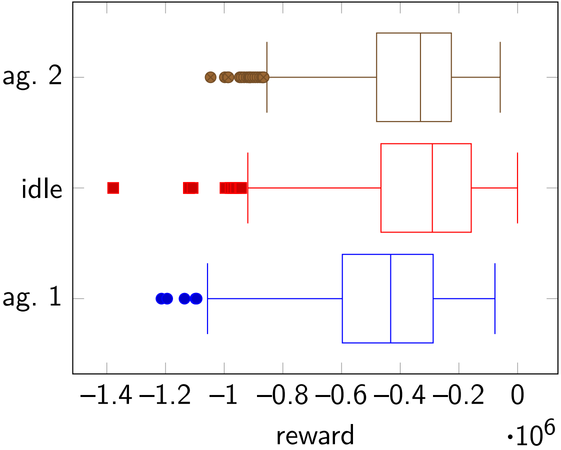

Figure: Comparing the idle behaviour with the agent.

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