Airline Planning Algorithm

Oleguer Canal - Fernando Garcia - Federico Taschin - Catherine Weldon

KTH - Royal Institute of Technology

Introduction

The *Vehicle Routing Problem* is a well-known dilemma. It involves combinatorics, optimization methods and integer programming. The goal is to optimize the route of a vehicle fleet to maximize locations reached while minimizing costs, often represented as time.

In this study, this problem was applied in the context of airline route planning. The goal was to identify the optimal route for a network of airplane fleets so that the airline maximizes its income while providing transportation for passengers.

1 Previous work

The Vehicle Routing Problem is a widely studied topic by computer scientists. Much research has been conducted on this topic as the solution is of high value to many different industries. Despite all of this research, it is still a problem without a scalable solution due to its high complexity in many environments.

Route optimization is of high importance to many companies as it has the has the potential to optimize costs and resources in large scale and complex problems.

2 Complexity

This problem has an intrinsically high complexity, making it difficult to be solved when considered at a large scale.

The complexity of this problem is ***Non-deterministic Polynomial Time Complete*** (*NP-Complete*). These kind of problems have no solution that can easily be computed because as the scale increases, the computational complexity increases at a above polynomial rate. Nevertheless, there exist several mechanisms which allow approximations to its solutions. In this project we take a look at some of them.

Fig. 1: Airline Routing

Problem Breakdown

This variation of the problem has been split into three different sub-problems, which vary in complexity. *Problem 0* is the most *relaxed* variation while *Problem 2* is the most complex one. The purpose of these variations is to assess the feasibility of each implementation.

Problem 0

This environment is defined as: single-agent, deterministic and fully-observable.

- At initial state at time t_0, all passengers and the plane are at a fixed location.
- The objective is to maximize the amount of people moved substracting the flight costs (set to one for every flight).
- The problem concludes when either all passengers arrive at their destination or a deadline is reached.

This problem has been solved by means of **Integer Programming**, ***PDDL*** and **Reinforcement Learning**.

Problem 1

This environment is defined as: cooperative multi-agent, deterministic and fully-observable.

- At initial state at time t_0, all passengers and planes are at a fixed location.
- The objective is to maximize the amount of people moved in a specified amount of time, substracting flight costs (set to be equal to flight time).
- Each plane has different capacity.
- The problem concludes when either all passengers arrive at their destination or a deadline is reached.

This problem has been solved by means of ***PDDL*** and **Reinforcement Learning**.

Problem 2

This environment is defined as: cooperative multi-agent, stochastic, and fully-observable.

- At initial state at time t_0, all passengers and planes are at a fixed location.
- The objective is to maximize the amount of people moved in a specified amount of time, substracting flight costs (set to be equal to flight time).
- Each plane has different capacity, and each person has a probability to miss a flight.
- The problem concludes when either all passengers arrive at their destination or the deadline is reached.

This problem has been solved by means of **Reinforcement Learning**.

GitHub repository of the project: `https://github.com/OleguerCanal/vehicle_routing_problem`

Airline Planning Algorithm

Oleguer Canal - Fernando Garcia - Federico Taschin - Catherine Weldon

KTH - Royal Institute of Technology

Integer Programming

For the *Integer Programming* approach, the problem was solved using *AMPL*, a modeling tool specifically designed for optimization problems. Because the problem's constraints involved multiple "if-statements" and the maximization called for multiple variables, the optimization problem was non-linear.

1 What is *Integer Programming*

Integer programming is a mathematical optimization method in which all variables are restricted to be integers.

2 Simplified Implementation

Parameters:

E_{n} Destination for each passenger n

Variables:

$$x_{\text{ijt}} = \begin{cases} 1 & \text{If flight travels from Origin i to Destination j at point t} \\ 0 & \text{Otherwise} \end{cases}$$

$$m^{n}{}_{\text{ijt}} = \begin{cases} 1 & \text{if passenger n travels from Origin i to Destination j at point t} \\ 0 & \text{Otherwise} \end{cases}$$

Objective $\rightarrow$ MAXIMIZE Profit (people moved - flight cost)

$$\sum_{n=1}^{N} x_{\text{ijt}} m^{n}{}_{\text{ijt}} E^{n}{}_{\text{j}} - \sum_{ijt} x_{\text{ijt}}$$

3 Conclusions

- **Problem 0** was solvable under certain conditions with Integer Programming.
- Default *AMPL* solver, Minos does not guarantee a global optimum for non-linear problems.
- Many of the formulations with stricter time-limits or additional passengers resulted in non-optimal routes.
- *AMPL* computing limits are also a problem, as the solver would return a solution before completing all necessary iterations if the scale was too large.
- **Problem 1** and **Problem 2** were not implemented with Integer Programming methods.

PDDL

This is a planning problem, therefore it can easily be expressed with PDDL and solved with a PDDL solver.

1 What is *PDDL*

Programming Domain Definition Language (*PDDL*) is a standardized planning language for AI systems.

2 Simplified Implementation

Passengers are grouped by final destination to reduce branching factor. A plane can either board all or none of the passengers in a group. The used actions are the following:

- **Action BOARD**(plane, group, city):
  - **Preconditions**: The group and plane must be in the same city. Deadline must not be reached.
  - **Effects:** Group is boarded onto the plane and is no longer in the city. Onboard passengers' counter and plane stopwatch are set accordingly.
- **Action FLY**(plane, from, to):
  - **Preconditions**: Plane did not reach deadline and is in city `from`.
  - **Effects**: Move the plane in city `to`, increase plane stopwatch by the time between `from` and `to`.
- **Action UNBOARD**(plane, group, city):
  - **Preconditions**: Group is in plane and plane is in city.
  - **Effects**: Move group to the city, set empty seats in the plane, set group stopwatch accordingly.

3 Conclusions

- **Problem 0** was solvable with either equal or more optimal results than the other approaches for each problem variation. However, the number of evaluated states quickly explodes as the problem size increases, resulting in very long computation times and/or out of memory errors.
- **Problem 1** was solvable for small problem instances. The PDDL solver can handle a great number of passengers when grouping is possible, but fails in all other cases.
- A great limitation was given by the scarcity of properly working open source PDDL solvers. Although many solvers (Metric-FF, FastDownward, ENHSP) were tried, only one (ENHSP) was able to handle the problem, and not all of its search/heuristic options returned a solution.

Reinforcement Learning

An *Active Reinforcement Learning* solution is presented in order to model stochasticity, added constraints, and solve problems within a larger state space and bigger branching factor. In particular, the algorithm used is *Q-Learning* with *epsilon-greedy* exploration.

1 What is *Reinforcement Learning*

Reinforcement Learning is a machine learning area concerned with how software agents should take actions in a given environment such that some notion of a cumulative reward is maximized.

2 Simplified Implementation

- **State**:
  - Cities considered and distances between them
  - List of people with current and desired cities
  - List of landed planes and their locations
  - List of ongoing flights
- **Action**:
  - List of all possible combinations of flights given current state
- **Reward**:
  - Moving people to their final location
  - Moving planes towards cities with more people
  - Minimizing total time
  - Minimizing airtime

Note: Stochasticity is added in the form of probability of people missing a flight once the action has been taken

3 Conclusions

- **Branching factor:** Better manages larger state spaces compared to other methods but it is still a problem.
- **Meta-parameters:** Highly susceptible to: episode number, learning rate, discount factor, epsilon, reward weights ...
- **Quality Table Limitations:** Simple state-action table can be improved with a deep network instead.
- **Algorithm generalization:** Current defined state space could easily be tested with different algorithms such as simulated annealing[?] or genetics algorithm [?].

Airline Planning Algorithm

Oleguer Canal - Fernando Garcia - Federico Taschin - Catherine Weldon

KTH - Royal Institute of Technology

Problem 0: IP vs PDDL vs RL

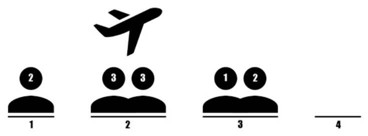

Fig. 1: P0-1

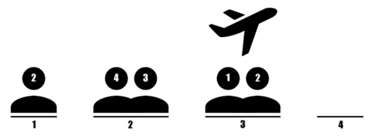

Fig. 2: P0-2

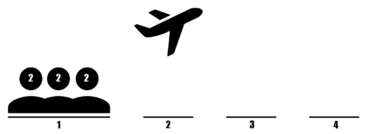

Fig. 3: P0-3

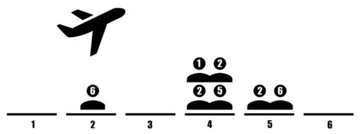

Fig. 4: P0-4

Tab. 1: Score (people dest. - flight cost)

Score	IP	PDDL	RI
P0-1	1	**2**	**2**
P0-2	0	**1**	**1**
P0-3	**1**	**1**	**1**
P0-4	N/A	**1**	**1**

Tab. 2: Iterations

Iterations	IP	PDDL	RI
P0-1	1794	11647	**6**
P0-2	45	46349	**8**
P0-3	1422	544	**16**
P0-4	Constrain Limit	11872177	**37**

Problem 1: PDDL vs RL

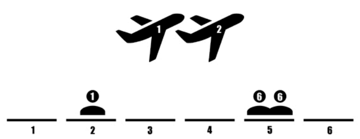

Fig. 5: P1-1

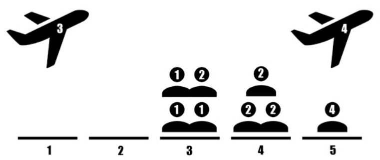

Fig. 6: P1-2

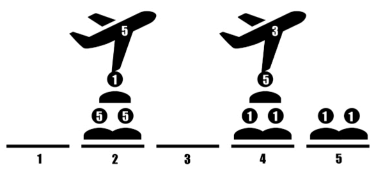

Fig. 7: P1-3

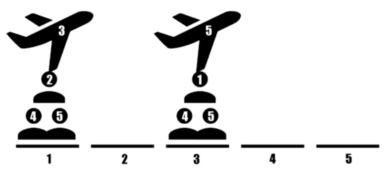

Fig. 8: P1-4

Tab. 3: Score (people dest. - flight cost)

Score	PDDL	RL
P1-1	-1	**0**
P1-2	**3** (with grouping)	2
P1-3	**5** (with grouping)	3
P1-4	OutOfMemory	**0**

Tab. 4: Iterations

Score	PDDL	RL
P1-1	16816	**5087**
P1-2	118215	**3997**
P1-3	2189489	**9185**
P1-4	OutOfMemory	**7555**

Note: This experiment has also been done employing distance between cities.

Problem 2: RL

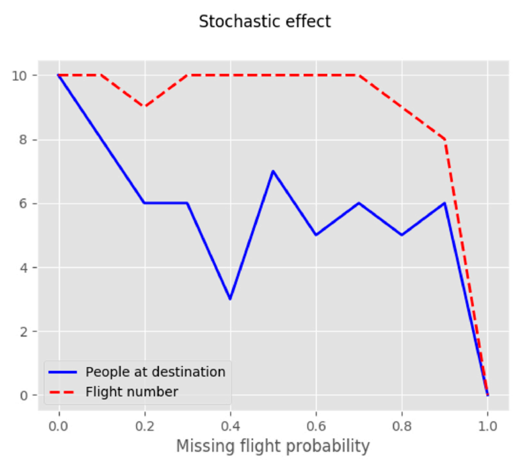

Fig. 9: P1-1

We added a probability of a person missing a flight. Results show:

- With low missing probabilities, the RL agent is able to perform at the same level.
- With middle range probabilities, the agent continues doing the same number of flights, though bringing less passengers.
- With high probabilities, the agent realizes its better not to move any plane.

AIRLINE PLANNING ALGORITHM

Oleguer Canal - Fernando Garcia - Federico Taschin - Catherine Weldon

KTH - Royal Institute of Technology

Future Work

1 Solution Improvements

The formulation of this routing problem has a very high branching factor and complexity level which makes it difficult to apply to real-world scenarios.

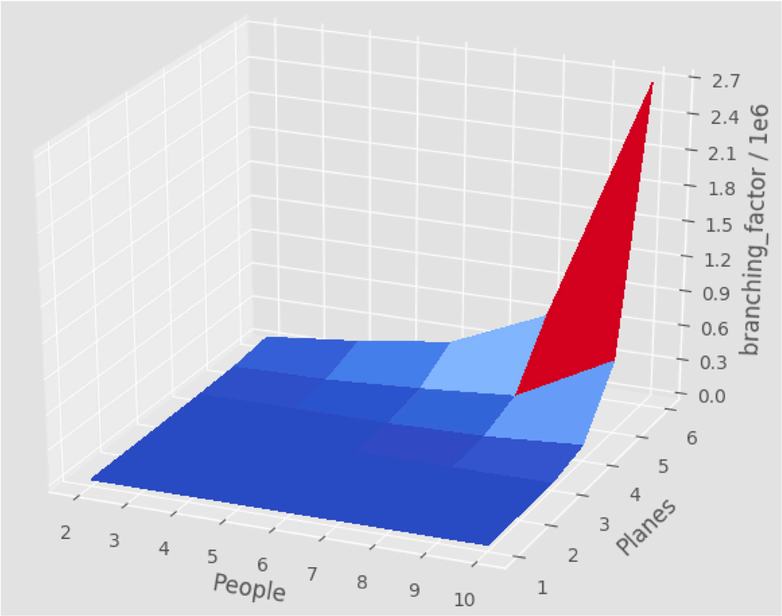

Fig. 1: **Problem 2** Branching Factor as a function of Passengers and Planes

Scalability in the Vehicle routing problem, which is NP-hard, is currently an unsolved problem. To manage this branching factor and allow for increased scalability, the next steps would be to implement the following:

- Different heuristic algorithms such as the Clarke-Wright Algorithm [12] which is commonly implemented in the VRP to find quicker feasible solutions
- Deep Q Learning with Reinforcement Learning [11] to optimize the learning time

Both of these solutions would limit the branching factor and allow for solving larger problems.

2 Additional Considerations

To transform this problem to real world data, there are additional constraints that could be explored. These variations include the following:

- Adding costs, such as fuel, manteinance, crew, taxes
- Adding stochasticity for delays or equipment malfunctions
- Modeling competing airline routes to understand how the increase in supply would impact ticket prices
- Taking into account the optimal scheduling of flight crews
- Considering how seasonality impacts ticket prices

This additional data would increase the effectiveness of the plan, transforming it to a real-world solution.

References

[1] Sohaib Afifi, Duc-Cuong Dang, and Aziz Moukrim. "A Simulated Annealing Algorithm for the Vehicle Routing Problem with Time Windows and Synchronization Constraints". In: *7th International Conference, Learning and Intelligent Optimization (LION 7)*. Vol. 7997. Catania, Italy, Jan. 2013, pp. 259–265. DOI: 10.1007/978-3-642-44973-4_27. URL: https://hal.archives-ouvertes.fr/hal-00916972.

[2] J. Bailey. *How Do Airlines Actually Plan Routes?* 2019. URL: https://simpleflying.com/how-do-airlines-plan-routes/.

[3] Marshall Fisher. *Handbooks in Operations Research and Management Science*. 1995. Chap. Vehicle routing.

[4] *Flight-UI: A Flight Application in Flutter*. Available at https://github.com/ajay1706/Flight-UI?files=1 (Accessed: 2019-10-03).

[5] Anders Forsgren and Margaret H. Wright. "An elementary proof of linear programming optimality conditions without using Farkas' lemma". In: 2014.

[6] Maria Fox and Derek Long. "PDDL2.1: An Extension to PDDL for Expressing Temporal Planning Domains". In: *CoRR* abs/1106.4561 (2011). arXiv: 1106.4561. URL: http://arxiv.org/abs/1106.4561.

[7] *HomePage - Fast Downward Homepage*. Available at http://www.fast-downward.org/ (Accessed: 2019-10-03).

[8] M Kefi and K Ghedira. "A multi-agent model for the Vehicle Routing Problem with Time Windows". In: *WIT Transactions on The Built Environment* 75 (2004).

[9] Gilbert Laporte. "The vehicle routing problem: An overview of exact and approximate algorithms". In: *European Journal of Operational Research* (1992).

[10] Abdul Kadar Muhammad Masum et al. "Solving the Vehicle Routing Problem using Genetic Algorithm". In: *International Journal of Advanced Computer Science and Applications* 2.7 (2011). DOI: 10.14569/IJACSA.2011.020719. URL: http://dx.doi.org/10.14569/IJACSA.2011.020719.

[11] MohammadReza Nazari et al. "Deep Reinforcement Learning for Solving the Vehicle Routing Problem". In: *CoRR* abs/1802.04240 (2018). arXiv: 1802.04240. URL: http://arxiv.org/abs/1802.04240.

[12] Petrică Claudiu Pop et al. "Heuristic Algorithms for Solving the Generalized Vehicle Routing Problem". In: *International Journal of Computers Communications Control* 6.1 (2011), pp. 158–165. ISSN: 1841-9844. DOI: 10.15837/ijccc.2011.1.2210. URL: http://univagora.ro/jour/index.php/ijccc/article/view/2210.

[13] Stuart J Russell and Peter Norvig. *Artificial intelligence: a modern approach*. Malaysia; Pearson Education Limited, 2016.

[14] *Skyscanner Flight Search API: How To Use the API | RapidAPI*. Available at https://rapidapi.com/skyscanner/api/skyscanner-flight-search/details (Accessed: 2019-10-03).

[15] Paolo Toth and Daniele Vigo. *The vehicle routing problem*. SIAM, 2002.

[16] Antonio A Trani. *Applications of Linear Programming Associate Professor of Civil Engineering*. Available at http://www.fast-downward.org/ (Accessed: 2019-10-03).

[17] Sedighe Zibaei, Ashkan Hafezalkotob, and Seyed Sajad Ghashami. "Cooperative vehicle routing problem: an opportunity for cost saving". In: *Journal of Industrial Engineering International* 12.3 (2016), pp. 271–286.