A Reinforcement Learning Approach for Initialization of Column Generation with Application to Aircraft Recovery Problem

Jingxi Lu
Laboratory of Smart City, Transportation, and Logistics
Shenzhen Research Institute of Big Data
Shenzhen, China
Department of Computer Science
Beijing Normal University-Hong Kong Baptist University United International College
Zhuhai, China
1624539843@qq.com

Xiongwen Qian*
Laboratory of Smart City, Transportation, and Logistics
Shenzhen Research Institute of Big Data
Shenzhen, China
qianxiongwen@sribd.cn

***Abstract*—Column Generation is a crucial technique for addressing large-scale combinatorial problems, particularly in logistics and transportation scenarios like the aircraft recovery problem. Yet, the initialization of columns within this framework remains an unexplored topic that significantly affects the efficiency of the solving process. In this study, we propose a novel reinforcement learning approach to design initial columns for the aircraft recovery problem. This approach is conceptualized as a decision-making process led by an agent, leveraging a Graph Attention Network combined with Proximal Policy Optimization to identify routes that minimize reduced costs within the aircraft's flight connection network. Preliminary computational results validate the effectiveness of the RL approach, demonstrating its capacity to enhance the overall speed of the column generation process. Notably, the trained policy exhibits robust generalization across different networks, rapidly producing high-quality initial columns to reduce runtime without additional training.**

***Keywords*—reinforcement learning, column generation, column initialization, aircraft recovery problem**

I. Introduction

Column Generation is a widely utilized method for solving large scale combinatorial problems with application in logistics [1-3] and transportation [4-6]. For example, column generation based heuristic has been devised to solve the aircraft recovery problem (ARP) and achieved high-quality results [5]. However, how to initialize the columns of the column generation framework when solving the aircraft recovery problem remains an unexplored topic. The initialization method has a substantial impact on the efficiency of the entire solving process. Fundamentally, the initial columns represent preliminary estimations of the optimal ones. In the most extreme scenario, where these initial estimations are entirely accurate, the best solutions are achieved right away.

For the aircraft recovery problem, due to its significance to airline operations, many scholars have conducted research. Teodorović and Guberinić [7] were pioneers in examining this issue, proposing a network flow-based heuristic method that uses branch-and-bound to sequentially plan flight strings for each aircraft. Argüello et al. [8] developed a greedy, random adaptive search algorithm employing a neighborhood search strategy. Cao and Kanafani [9] established a quadratic 0-1 programming model for the recovery of multiple fleets and introduced an approximate linear programming method for solving it. Rosenberger et al. [10] created a set-partitioning-based mathematical model to reschedule flight strings for each aircraft, where only pre-specified aircraft were allowed to change flights, generating new flight strings. Eggenberg et al. [11] used a recovery network with specific constraints to solve the problem of abnormal flight recovery, where time is discretized into individual time windows that require manual adjustment. Liang et al. [5] considered the problem of abnormal flight recovery under airport capacity constraints and maintenance flexibility conditions. Recently, Wen et al. [12] proposed an innovative method for rerouting aircraft to meet maintenance demands that emerge during the operational phase, utilizing a column generation technique. Li et al. [13] address the issue of disrupted flight recovery by incorporating two practical elements: firstly, the introduction of an alternative recovery strategy that includes altering flight durations, and secondly, the integration of aircraft assignment limitations. Zang et al. [14] enhance the selection process for recovery options by considering the variable characteristics of airport capacity changes across different locations and times, alongside the dynamic aspects of resolving flight delays.

From the analysis of the above literature, it is evident that significant progress has been made in solving the flight recovery problem. However, the focus has mainly been on classic integer programming and heuristic algorithms, with less consideration given to using artificial intelligence algorithms to accelerate the optimization process.

In this work, we propose a reinforcement learning (RL) approach to help design the initial columns (routes) in the aircraft recovery problem. The routing process is modelled as a decision-making process by an agent. Graph Attention Network (GAT) and Proximal Policy Optimization (PPO) are integrated to identify routes with negative reduced cost in the flight connection network of the aircraft recovery problem. At each decision-making step, the agent observes the information from the entire flight connection network and chooses one node to add to the currently extending route. The information includes the departure time, arrival time, and delay, etc. of all flights under consideration. The agent repeatedly picks one node (flight) in the network until a route is formed; then the route's return is calculated. The agent seeks to maximize the route's return so as to minimize the reduced cost associated with the generated route. Preliminary computational results confirm that the reinforcement learning approach finds good quality routes which help to reduce the total runtime of the

979-8-3315-0863-0/24/$31.00 ©2024 IEEE
DOI 10.1109/LESCDT63857.2024.00029

entire column generation process. What's more, the policy obtained by training on one flight connection network is able to generalize to other new flight connection networks, which means high quality initial columns of the new instances of the aircraft recovery problem can be found quickly without training.

The rest of the paper is organized as follows. In Section II, we review the column generation heuristic to solve the aircraft recovery problem. In Section III, we present the reinforcement learning based column initialization approach. Computational results are demonstrated in Section IV. Section V concludes the paper.

II. Column Generation Heuristics for ARP

The aircraft recovery problem involves rescheduling flights and reassigning aircraft in real-time to minimize recovery costs for airlines following disruptions. In the previous study [5], a column generation heuristic is proposed to solve the problem. The heuristic is structured around a master problem that selects routes for aircraft and subproblems that generate these routes. In the master problem, routes are selected for the aircraft to operate. The optimal dual values of these flights are then used as inputs to the subproblems, facilitating the creation of new routes. If any generated routes exhibit negative reduced costs, they are incorporated back into the master problem. This iterative process continues until no more routes with negative reduced costs can be identified, indicating that the solution to the current linear programming model is optimal. Finally, to derive an integer solution, the master problem is resolved using integer programming.

The master problem is formulated as a linear relaxation of a set partitioning problem. For a detailed formulation of the master problem, interested readers are referred to the work [5]. On the other hand, the subproblem is based on the flight connection network. Fig. 1 presents an example of this network. In the network, individual flights represent nodes, and their connections are depicted as arcs. Two nodes have a direct connection when the airport where one flight terminates coincides with the airport where the subsequent flight originates. Flights departing earlier are directed towards those departing later. To facilitate the process of determining a route for the aircraft in the subproblem, dummy source and sink nodes are integrated into the network for the start available airports and end available airports. The challenge within the subproblem is to discover the shortest path from the source node to the sink node in the connection network. The goal is to minimize the reduced cost, which corresponds to the length of the shortest path found.

The travelling cost of one arc is

$$\overline{\beta}_{c,f} = \beta_{c,f}^{swap} + \beta_{c,f}^{delay} - \pi_f \qquad (1)$$

where $\beta_{c,f}^{swap}$ is the cost of changing a flight f's planned aircraft to aircraft c, $\beta_{c,f}^{delay}$ is the cost of flight delay, and π_f is the dual variable for flight f of the covering constraint in the master problem. Observe that in the subproblem, the travelling cost of one arc is not fully given, because $\beta_{c,f}^{delay}$ is also a decision variable, whereas $\beta_{c,f}^{swap}$ and π_f are given as parameters. In the column initialization phase, we adopt a reinforcement learning approach to find the shortest path on the flight connection network, and later transfer the trained policy to other problem instances.

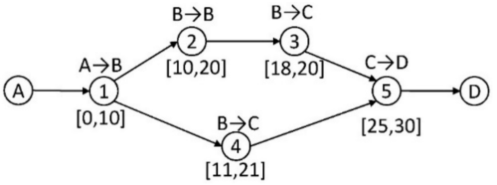

Fig. 1. Illustration of a connection network (Numbers in brackets represent a flight's scheduled departure and arrival times, converted to integers) [5]

III. Column Initialization via Reinforcement Learning

A. Reinforcement Learning Framework

The reinforcement learning methodology is structured as a Markov Decision Process (MDP), which serves as a robust mathematical framework designed for making sequential decisions. At every decision point, the agent obtains the current states from the environment and decides on actions accordingly. Specifically, within the context of column initialization, the agent selects which network node to visit, this decision being guided by the detailed information pertaining to the various nodes of the network.

The state of nodes is defined as follows. The node information and topology are included in the primitive embedding for nodes. The primitive node embedding for i-th node in N is a 9-dimensional vector n_i with each element as: 1) the ID of the node, 2) whether the node is a maintenance node, 3) the scheduled departure time of the flight, 4) the scheduled arrival time of the flight, 5) the delay of the flight, 6) whether the departure airport of the flight is the start airport of the assigned aircraft, 7) whether the arrival airport of the flight is the end airport of the assigned aircraft, 8) if the node has been picked, 9) whether the flight i used to belong to aircraft ar_k. The state of the node is defined as the following vector: $s_i = (n_i,\ m_i,\ dt_i, at_i, d_i,\ wd_i,\ wa_i, p_i,\ b_i)$.

For the information of the assigned aircraft, we created a dummy node as the final endpoint. The dummy node contains the current aircraft's start and end times, the rest of the variables are filled with -1. For the aircraft ar_k the dummy node is defined as the following vector: $s_{dummy} = (-1, -1,$ $sar_c\ , ear_c, -1, -1, -1,\ -1, -1)$. When the dummy node is selected, the search for the initial column (route) is terminated. The elements of the state are normalized.

In this paper, the action set, A, is defined as all nodes on the flight connection network. The action selected at each decision making step t for aircraft ar_k, is at $a_t \in$ A, which represents an aircraft select a node to join its route. The action space is defined as $A = \{1, 2, \ldots, |N| + 1\}$.

The reward function defines the agent's gain of each decision step and indicates whether its behavior is good or bad in a particular state within the environment. To minimize the path cost (reduced cost), the reward is set to the negative value of the arc travelling cost $\overline{\beta}_{c,f}$, which is calculated in Equation 1. When the dummy node is selected, the reward is set to twice the path cost to encourage the agent to choose a longer path with a positive sum of rewards.

B. Proposed Model

1) Encoder: Graph Attention Network (GAT) [15] is a powerful graph neural network architecture, which uses attention mechanism to propagate node information effectively. The primitive embeddings for nodes introduced in previous section are firstly extended by a full connection layer.

$$\tilde{n}_i = W_n * n_i \tag{2}$$

The inputs of the GAT are the embedding $\tilde{n}_i$, and the output is an updated node embedding $n_{\{\mathrm{GAT,i}\}}$. The attention weight vector for pair i, j is calculated as:

$$w_{\{i,j\}} = LeakyReLU(W_L * \tilde{n}_i) \tag{3}$$

$$\widetilde{w}_{\{i,j\}} = \frac{\exp{(w_{\{i,j\}})}}{\sum_j \exp{(w_{\{i,j\}})}} \tag{4}$$

$$n_{\{GAT,i\}} = \tilde{n}_i + \textstyle\sum_j \widetilde{w}_{\{i,j\}} \otimes \tilde{n}_j \tag{5}$$

where ⊗ represents element-wise multiplication between vectors, and $W_n, W_{\{i,j\}}, W_L$ are weight matrices to be learned. The output of GAT provides a more comprehensive representation of graph topology. This paper employs multiple layers of GAT to ensure that each node has a wider acceptance domain, thereby increasing the possibility of exchanging information with remote nodes. The output from the last layer provides the final version of the node embed. Each node embedding contains its own information as well as information from related nodes.

2) Decoder: The decoder operates using an attention mechanism, producing the pointer vector u_i. This vector is subsequently transmitted to a softmax layer, which constructs a probability distribution over the subsequent candidate node. To avoid selecting nodes that should not be chosen, an action mask is necessary. We set the action mask of the nodes with excessively long delays to 1, ensuring they will not be selected. Similar to pointer networks, the attention mechanism and pointer vector u_i are defined as:

$$u_i = \begin{cases} v^{\mathrm{T}} \cdot \tanh{(W_r r_i + W_q q)} & \text{if action mask } i \ = \ 0, \\ -\infty & \text{otherwise} \end{cases} \tag{6}$$

where W_r and W_q are trainable matrices, q is a query vector from the hidden variable of the Gated Recurrent Unit (GRU) [16], and r_i is a reference vector containing the information of the context of all nodes. The distribution policy over all candidate node is given by:

$$\pi_\theta(\mathbf{a}_i|\mathbf{s}_i) = \mathbf{p}_i = \mathrm{softmax}(\mathbf{u}_i) \tag{7}$$

We predict the next visited node by sampling from the policy $\pi_\theta(\mathbf{a}_i|\mathbf{s}_i)$.

3) Training process: We utilize the actor-critic framework of Reinforcement Learning to train the parameters of both the encoder and decoder. Let the parameter set for the encoder and decoder be denoted as θ. In our approach, the actor and critic networks share the same architecture, with the exception that the critic's decoder includes two additional feed-forward layers. The first of these layers is a dense layer with ReLU activation, followed by a linear layer. The parameter set for the value network is denoted as ϕ. The training process of the actor-critic cycle is outlined as follows.

To calculate the advantage by Monte Carlo method:

$$\delta_{\mathrm{MC}}^{(t)} \leftarrow \textstyle\sum_{k=0}^{T-t-1} \gamma^k r_{t+k} - \hat{v}(s_t, \phi) \tag{8}$$

To train the critic network:

$$\phi \leftarrow \phi + \alpha_\phi \delta_{\mathrm{MC}}^{(t)} \nabla_\phi \hat{v}(s_t, \phi) \tag{9}$$

To train the actor via clipped surrogate objective Proximal Policy Optimization method:

$$\begin{aligned} & L^{CLIP}(\theta) \\ & = \hat{\mathbb{E}}_t[min(r_t(\theta)\delta_{\mathrm{MC}}^{(t)}, \mathrm{clip}(r_t(\theta), 1 \quad (10) \\ & - \epsilon, 1 + \epsilon)\delta_{\mathrm{MC}}^{(t)})] \end{aligned}$$

Let α_ϕ denote the learning rate for the critic. The term $r_t(\theta)$ is the ratio of the new policy to the old policy, with ϵ set to 0.2.

IV. Computational Result

To evaluate the effectiveness of the reinforcement learning based column initialization approach, computational experiments were carried out in this work. The test scenarios and corresponding data are derived from the previous study [5]. The experiments utilized a desktop computer featuring an NVIDIA GeForce RTX 3090 GPU and an Intel Core i5-13400 CPU, running Ubuntu 22.04 LTS. All programs were coded in Python 3.

We employed the Proximal Policy Optimization framework within reinforcement learning, where the policy network was equipped with Graph Attention Networks as encoders and GRU-based pointer networks as decoders to generate initial columns. The data of Scenario 1 are used to train the policy network within the reinforcement learning framework. Later, the obtained policy network is used to infer the action (which node to choose) in the other scenarios (and Scenario 1). Fig. 2 and Table I illustrates the results of the tests of all scenarios. We compared the RL initialization approach with the originally feasible route initialization approach which was adopted in the work [5].

TABLE I. Computational Result of Column Initialization using RL and Originally Feasible Routes

Scenario	1	2	3	4
Number of Aircraft	12	44	83	16
Number of Nodes	95	611	443	83
Number of Connections	1024	49227	13128	466
Total Runtime by Original Initialization (s)	15.56	158319.81	7831.53	2.74
Total Runtime by RL Initialization (s)	11.79	45582.32	3118.46	2.54
Reduction	24.23%	71.21%	60.18%	7.3%

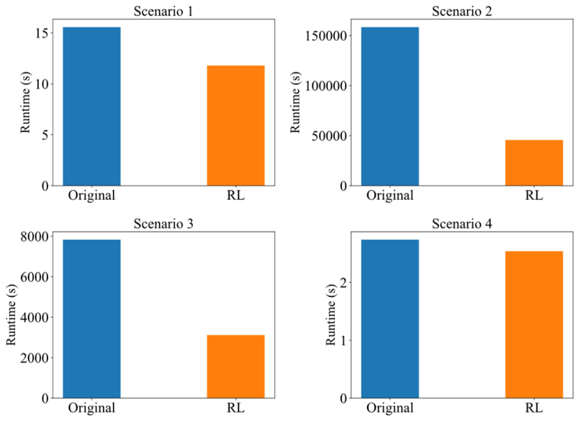

Fig. 2. Total runtime comparison: original vs RL initialization

Our study demonstrates significant performance improvements in column generation for the flight recovery problem by employing reinforcement learning-generated initial columns compared to using originally feasible routes. Across various scenarios, such as Scenario 1 involving 12 aircraft and 95 nodes where RL initialization reduced column generation time from 15.56 seconds to 11.79 seconds, and scenarios with 44 aircraft and 611 nodes where it decreased from 158319.8 seconds to 45582.3 seconds, RL consistently accelerated the solution process. This enhancement is attributed to the integration of Proximal Policy Optimization with Graph Attention Networks and GRU networks, enabling our RL model to effectively capture complex node relationships and routing dynamics. The model trained on data from Scenario 1 displayed performance across all experiments, showcasing its generalizability and stability in diverse flight recovery scenarios. Our findings underscore the broad applicability and efficacy of RL in column initialization tasks, validating its superiority over traditional methods across varying problem scales and complexities.

V. Conclusion

In this study, we propose an innovative reinforcement learning method for column initialization during column generation, especially for aircraft recovery problems. By constructing the column initialization problem as a Markov decision process and incorporating Graph Attention Network and Proximal Policy Optimization algorithms, effectively identify initial columns that minimize reduced costs in the flight connection network. Our computational experiment results show that this reinforcement learning-based initialization method has significant advantages in accelerating the column generation process compared with using the original route as the initial column. The trained RL strategy demonstrates robust generalization across different network scenarios and can quickly produce high-quality initial columns without additional training. This study provides a new way for reinforcement learning to improve the efficiency of large-scale combinatorial problem-solving, and also opens up a new vision for future research in the field of logistics and transportation optimization. In future, we anticipate further refinement of this RL-based approach to accelerate the process of solving the subproblems along with the whole column generation process. Additionally, exploring the adaptability of this method to other combinatorial optimization problems, such as vehicle routing and crew scheduling—previously addressed using the column generation algorithm—could expand its applicability. By continuously enhancing the learning algorithms and broadening the range of problems they can address, this study establishes a foundation for enhancing the influence of operations research and related fields.

Acknowledgment

This research was supported in part by the National Natural Science Foundation of China (NSFC) under Grant 72101162, and the Internal Project Fund from Shenzhen Research Institute of Big Data under Grant J00220230007 and Grant P00120230005.

References

[1] Y. Tao, E. P. Chew, L. H. Lee, and Y. Shi, "A column generation approach for the route planning problem in fourth party logistics," *Journal of the Operational Research Society,* vol. 68, pp. 165-181, 2017.

[2] J. Volland, A. Fügener, and J. O. Brunner, "A column generation approach for the integrated shift and task scheduling problem of logistics assistants in hospitals," *European Journal of Operational Research,* vol. 260, pp. 316-334, 2017.

[3] G. Kozanidis, "Column generation for optimal shipment delivery in a logistic distribution network," *Sustainable Logistics and Transportation: Optimization Models and Algorithms,* pp. 87-112, 2017.

[4] G. Brønmo, B. Nygreen, and J. Lysgaard, "Column generation approaches to ship scheduling with flexible cargo sizes," *European Journal of Operational Research,* vol. 200, pp. 139-150, 2010.

[5] Z. Liang, F. Xiao, X. Qian, L. Zhou, X. Jin, X. Lu*, et al.*, "A column generation-based heuristic for aircraft recovery problem with airport capacity constraints and maintenance flexibility," *Transportation Research Part B: Methodological,* vol. 113, pp. 70-90, 2018/07/01/ 2018.

[6] J. S. Neufeld, M. Scheffler, F. Tamke, K. Hoffmann, and U. Buscher, "An efficient column generation approach for practical railway crew scheduling with attendance rates," *European Journal of Operational Research,* vol. 293, pp. 1113-1130, 2021.

[7] D. Teodorović and S. Guberinić, "Optimal dispatching strategy on an airline network after a schedule perturbation," *European Journal of Operational Research,* vol. 15, pp. 178-182, 1984/02/01 1984.

[8] M. F. Argüello, J. F. Bard, and G. Yu, "A Grasp for Aircraft Routing in Response to Groundings and Delays," *Journal of Combinatorial Optimization,* vol. 1, pp. 211-228, 1997.

[9] J. M. Cao and A. Kanafani, "Real-time decision support for integration of airline flight cancellations and delays part I: mathematical formulation," *Transportation Planning and Technology,* vol. 20, pp. 183-199, 1997/03/01 1997.

[10] J. M. Rosenberger, E. L. Johnson, and G. L. Nemhauser, "Rerouting Aircraft for Airline Recovery," *Transportation Science,* vol. 37, pp. 408-421, 2003.

[11] N. Eggenberg, M. Salani, and M. Bierlaire, "Constraint-specific recovery network for solving airline recovery problems," *Computers & Operations Research,* vol. 37, pp. 1014-1026, 6// 2010.

[12] X. Wen, X. Sun, H.-L. Ma, and Y. Sun, "A column generation approach for operational flight scheduling and aircraft maintenance routing," *Journal of Air Transport Management,* vol. 105, p. 102270, 2022.

[13] J. Li, K. Li, Q. Tian, and P. R. Kumar, "An improved column generation algorithm for the disrupted flight recovery problem with discrete flight duration control and aircraft assignment constraints," *Computers & Industrial Engineering,* vol. 174, p. 108772, 2022.

[14] H. Zang, J. Zhu, Q. Zhu, and Q. Gao, "A proactive aircraft recovery approach based on airport spatiotemporal network supply and demand coordination," *Computers & Operations Research,* vol. 165, p. 106599, 2024.

[15] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," presented at the The Sixth International Conference on Learning Representations (ICLR), Vancouver 2018.

[16] K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk*, et al.*, "Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation," presented at the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, 2014.