# DeepFreight: A Model-free Deep-reinforcement-learning-based Algorithm for Multi-transfer Freight Delivery

Jiayu Chen[1], Abhishek K. Umrawal[1], Tian Lan[2], and Vaneet Aggarwal[1]
[1]Purdue University, [2]The George Washington University

## Introduction & Background

➢ **Motivation:**

- On one hand, the U.S. trucking industry shipped **11.84** billion tons of goods in 2019 and had a market of **791.7** billion dollars.
- On the other hand, transportation is currently the largest source (**29%**) of greenhouse gas emissions in the U.S., and passenger cars and trucks account for more than **80%** of this section.

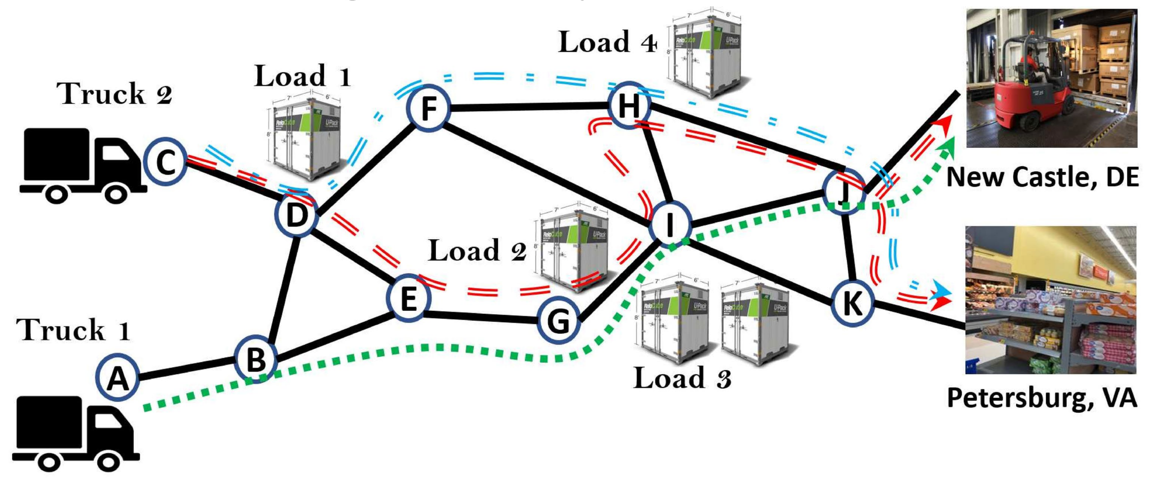
For the sake of better utilization of trucks and less fuel consumption, innovations to enable intelligent and efficient freight transportation are of central importance.

➢ **Related Works:**

- Typically, the freight delivery problem is modeled as a variant of **'Vehicle Routing Problem (VRP)'** (Dantzig and Ramser 1959), and many heuristics and metaheuristics algorithms have been adopted as the solution, which are model-based.
  Con: high time complexity and poor scalability.
- Reinforcement learning aided approaches have been adopted in **passenger ride-sharing systems** (Ghosh, and Aggarwal 2019); further, approaches that allow passengers to go over multiple hops have been proposed (Singh, Alabbasi, and Aggarwal 2019) to greatly increase the ride availability and operation efficiency.

Inspired by the multi-hop ride-sharing system, we propose **multi-transfer freight delivery**, where goods can be dropped off in the middle and carried further by another truck before reaching the goal.

➢ **Multi-transfer Freight Delivery:**




- Load 1 and 4 need to be shipped to Petersburg, VA, while Load 2 and 3 to New Castle, DE; Two trucks located at zone A and zone C, are with capacity R1 = 4 and R2 = 5.
- They can:

a) serve Load 1 and 4 using Truck 2 (i.e. the dashed-dotted blue line) and Load 2 and 3 using Truck 1 (i.e. the dotted green line);
b) have the two trucks pick up their loads first and then meet at zone H to fold the loads into a single truck for the final delivery.

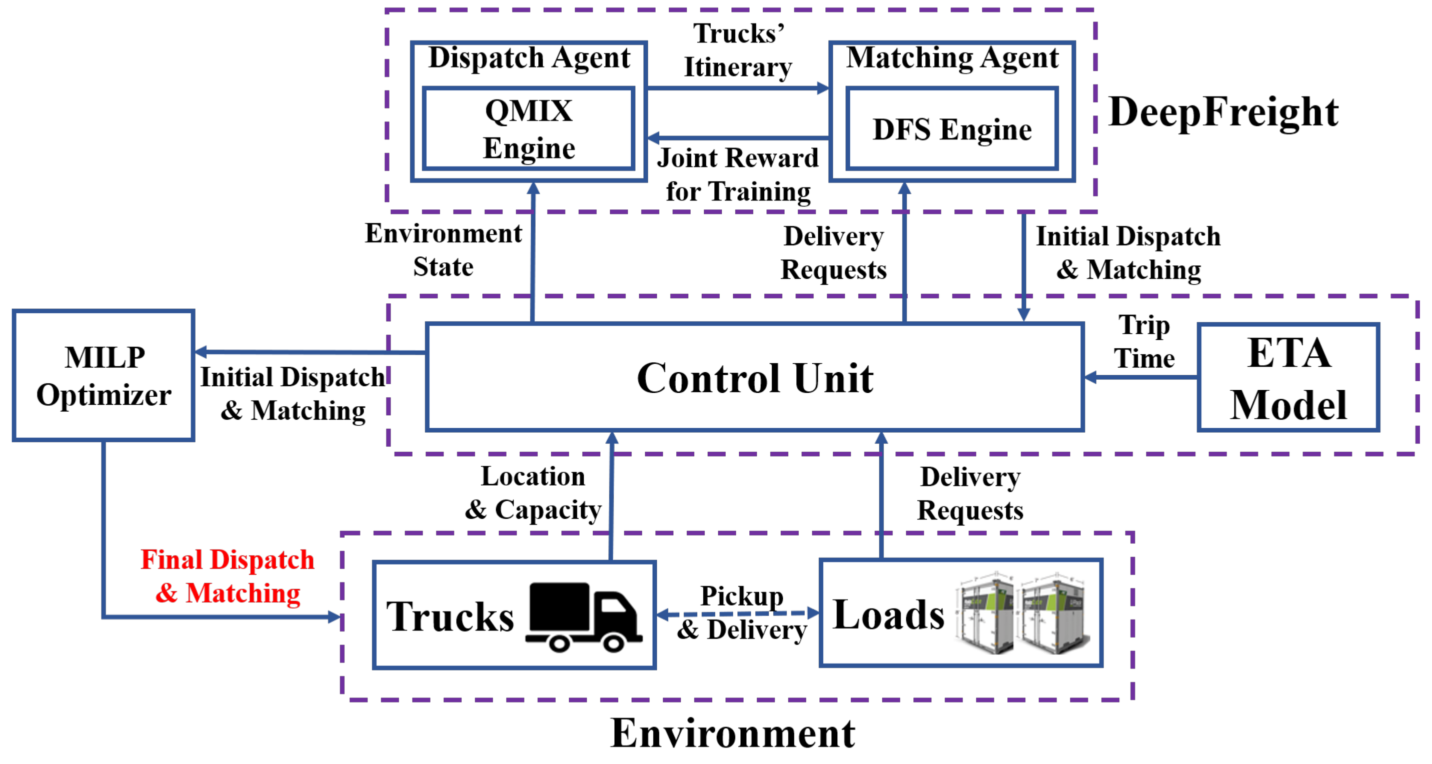
Apparently, multi-transfer delivery provides additional elasticity for the truck scheduling problem and improves the fleet usage.

## Algorithm

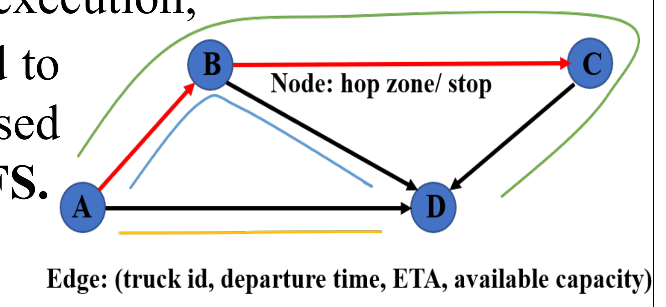
➢ **Freight Delivery Problem:**

- At the beginning of each day, we can get a list of delivery requests, including the source, destination and size of each package;
- The target is to complete these requests using a fixed number of trucks within a time limit which can be one or two days, aiming at maximizing the number of served requests and minimizing the total driving time of the fleet;
- To fulfill this target, we need to schedule the itinerary of the fleet and the assignment of the packages to the trucks;
- Note that a package can be assigned to more than one truck along the route from its origin to its destination (multi-transfer).

➢ **Algorithm Framework:**




- The proposed algorithm, DeepFreight, decomposes the problem into two components: truck-dispatch and package-matching;
- The dispatch policy determines a sequence of dispatch decisions for each truck, which is trained through a centralized training with decentralized execution algorithm **QMIX** (Rashid et al. 2018) to consider the cooperation among the trucks when training and make the multi-agent dispatch scalable for execution;
- The matching policy is then executed to assign the requests to the trucks based on the route of each truck through **DFS.**

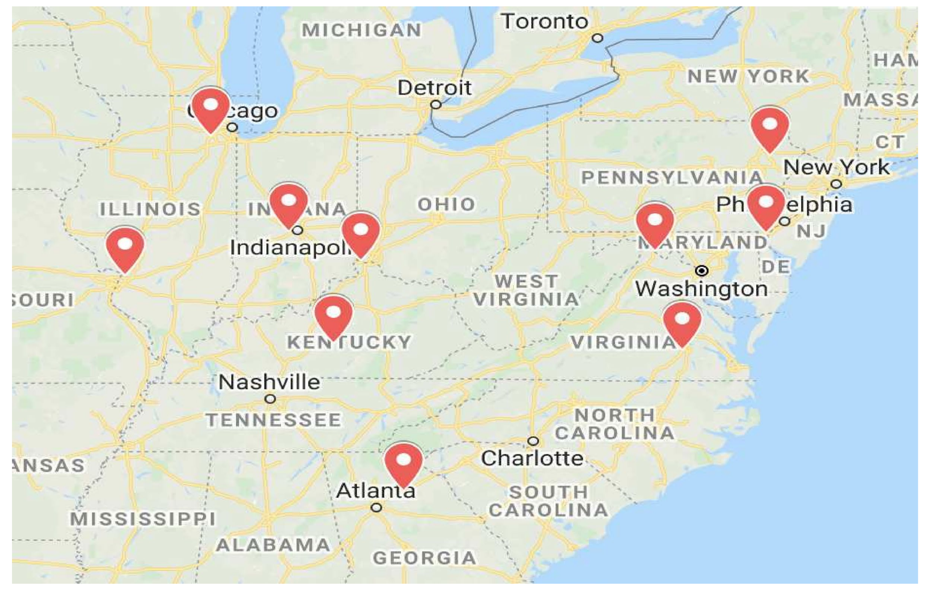



- Further, a **MILP** solver is adopted to optimize the initial dispatch & matching decisions:

a) If a truck does not transport enough goods per unit time, its route is deleted and the requests it fulfills are marked unassigned;
b) Remaining routing problem is solved exactly with a **MILP** solver.

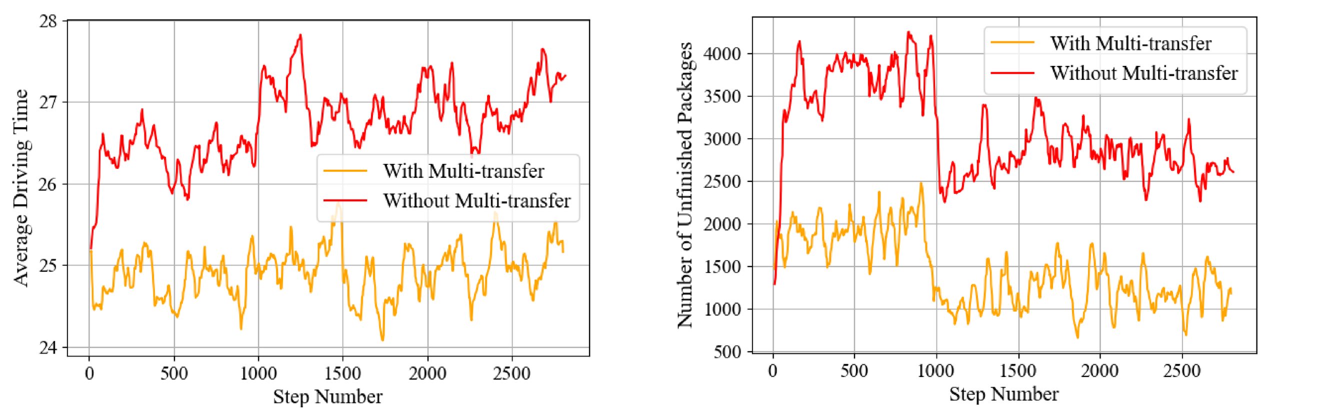
## Experiments

➢ **Simulation Setup:**



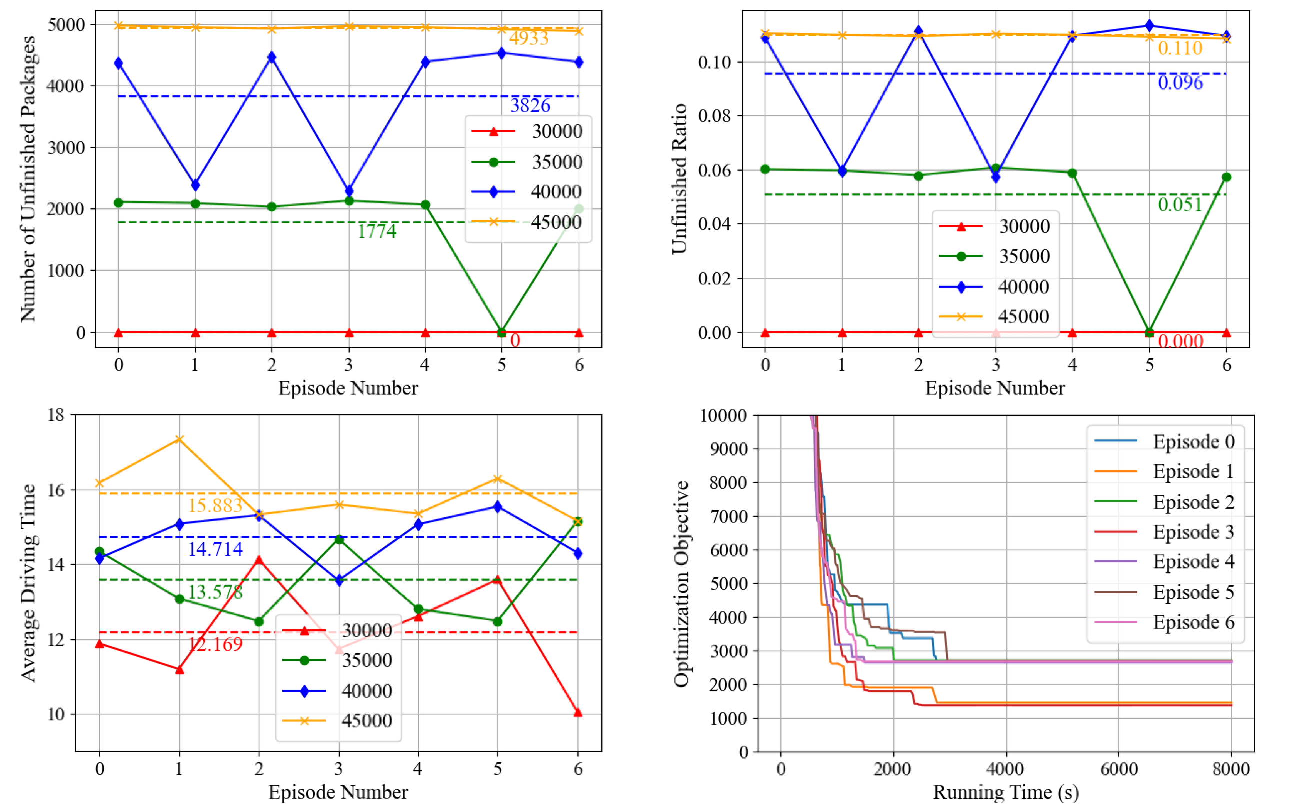

- Ten Amazon distribution centers in the eastern U.S. are chosen as the origins and destinations of delivery requests;
- Twenty trucks are required to complete 40000 randomly generated requests within 2 days.

➢ **DeepFreight vs. DeepFreight without Multi-transfer:**



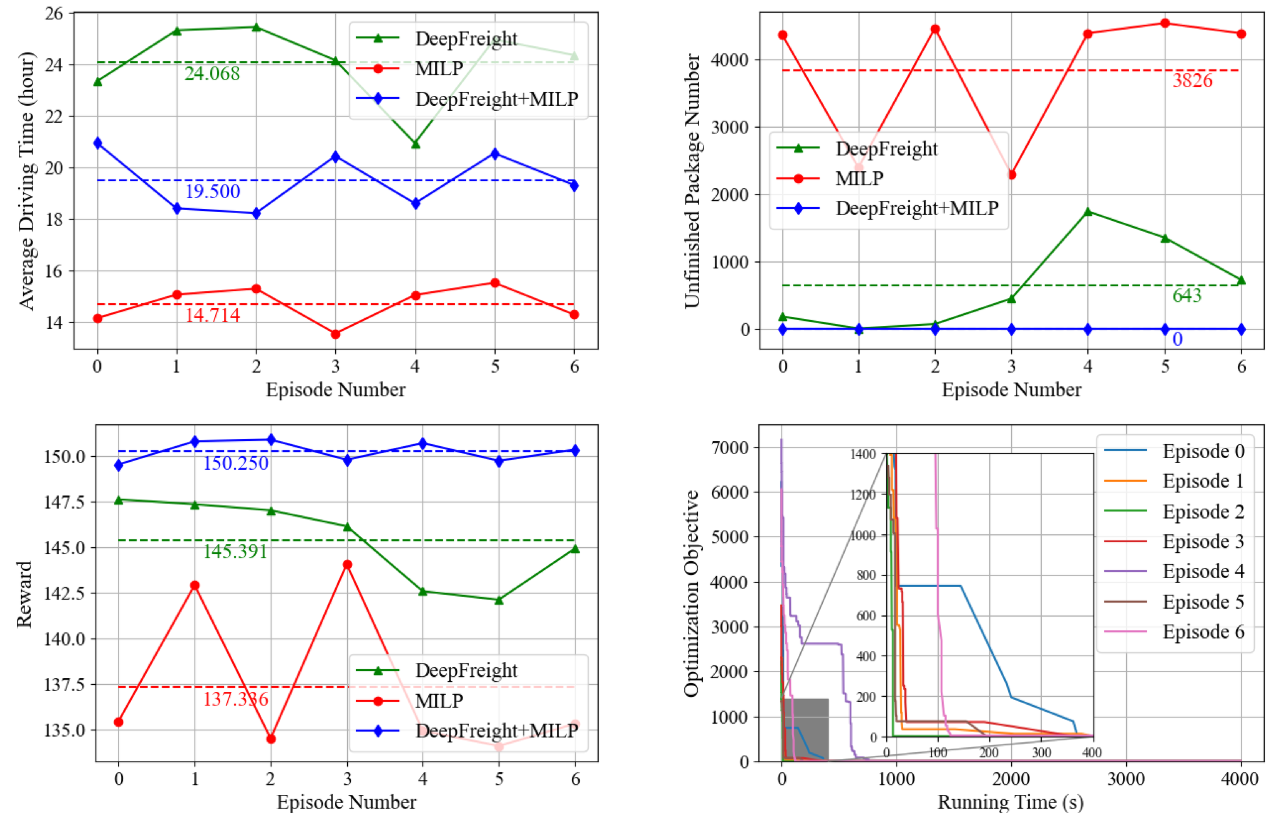

- With the flexibility of multi-transfer, the average driving time is shorter (~2 hours (7.5%) reduction) and the number of unfinished packages is lower (~1500 packages (60%) reduction), which means better utilization of the fleet is realized.

➢ **DeepFreight vs. MILP:**




## Experiments

- With growing number of packages, not only the driving time and the number of unfinished packages increase, the ratio of unfinished packages (normalized by the total package number) also grows, demonstrating the poor scalability of **MILP;**
- The optimization objective converges within 3000s (device: Intel i7-10850H) and then the performance doesn't increase any more, which means the MILP solver cannot find a solution for this task even if given more time.




- As compared to **MILP**, **DeepFreight** is transferable among different problem scenarios and has lower time complexity, but the average driving time and number of unfinished packages need to be further reduced, motivating our design of **DeepFreight+MILP.**

➢ **DeepFreight vs. DeepFreight+MILP:**

- **DeepFreight+MILP** has a further reduction in the average driving time as compared with **DeepFreight**;
- With comparable driving time as **MILP**, **DeepFreight+MILP** realizes a 100% delivery success;
- The convergence curve shows that we can get the routing results for the rest packages within 10 minutes, since most of the packages have been served by the dispatch decisions made by **DeepFreight**.

## Conclusion

The proposed algorithm framework, **DeepFreight+MILP**, performs the best among these algorithms, because it not only has better scalability and lower time complexity but also can ensure a 100% delivery success with fairly low fuel consumption.