# Description of Paper

## Introduction

This paper aims to propose a MIP formulation for the Pickup-and-Delivery Problem with Transshipment (PDPT). The formulation is shown to have polynomial size. In addition, the author shows alternative formulations that could capture problems with or without time window, heterogenous vehicles and flexible fleet size. Including transshipment in the model has been demonstrated to improve optimum objective and thus shows promising real-world value.

## Motivation and Literature Review

So far, only a few papers have addressed PDPT. In three seminal papers, **Drexl (2012a, 2012b, 2013)** introduced different classes of problems that included different vehicle type such as lorries, tractors, trailer, and semi-trailers. The paper brought forth the difficulty in developing adequate solution and emphasized on the issues in real world problems. Specifically, Milk collection in rural Germany.

A complete MIP model for PDPT is studied in **Cortes Matamala and Contardo (2010)**. This paper uses decision variables as base ad nodes in the network structure. The model contains 23 different sets of constraint. A relatively smaller problem is addressed.

The result presented by **Kerivin, Larcroix, Mahjoub, and Quilliot (2008)** presented an altered PDPT problem where demand may be split from pickup to delivery. More complex problems arise where the paper proposed two MIP formulations. A branch-and-cut algorithm was used to solve the problem. Similarly, **Grunert and Sebastian (2000)** also proposed instances where demand can be split. In this paper, a long-haul transportation problem is proposed based on acyclic network with transshipment options specifically address the postal and package delivery in Germany. A discrete time model where integer variables for the vehicle flows and continuous variables for the request flows is used. Neither implementation of a framework and performance analysis of model was conducted.

A MIP model on supply chain optimization that addressed PDPT in the paper by **Dondo, Mendeza, and Cerda (2009)**. The paper address commodities distributions from productions plants using distribution centers. A MIP model for PDPT with 25 different constraints was proposed by **Takoudjou, Deschamps, and Dupas (2012)**. The problem was generated and solved using heuristics. An altered PDPT, which “shuttle routes” are considered between pickup and delivery, is presented by **Masson, Ropke, Lehuede and Peton (2013)**. Three models were proposed, and branch-and-cut-and-price method is used. An extensive analysis is provided with promising results. In the paper, **Masson Lehuede, and Peton (2013a, 2013b)**, a heuristic and neighborhood searches method is used to solve real-world problems, specifically, aircraft route planning and transportation for the disabled. **Oretal (2000)** explored multiple theoretical results for PDPT model using tabu search algorithm for solving with less and equals to one transshipment per request and two transshipment locations. Both randomly generated and real-world problems are studied.

The concept of transshipment is not new, and it is easy to see that it has great application potential in real-world problems. Although several problems have been studied, this NP hard problem has yet to be easily adaptable for commercial purposes. In this paper, the author address PDPT with or without time window with a MIP approach. Very importantly to note that, the number of constraints and variables in the model are polynomial.

## Modelling Approach

The model is decomposed into two parts. One part specifically deals with the flow of vehicles. The other part, built on the first part, models the transport of requests. Both parts use traditional routing problem formulation.

### Flow of Vehicles

1.

2.

3.

Equation 1-3 are flow conservation constraints for vehicles, , that details the departure of vehicle, the return of vehicle and its routing.

12.

13.

14.

Equation 12-14 are subtour elimination constraints, that uses the pseudo-order variable to form a set of polynomial sized sub-tour elimination constraints.

### Flow of Requests

4.

5.

6.

7.

Equation 4-7 are flow conservation constraints for customer requests . Note that in addition to constraints that similar to vehicle’s, here we have a special transshipment constraint that allows the interchange of orders, as long as the total amount is conserved.

8.

9.

Equation 8 and 9 details the relationship between vehicle and requests. The vehicles need to be on specific path and have sufficient capacity.

15.

16.

17.

18.

Equation 15-18 details the time window constrains. It uses a big-M formulation to detail the time consumed during transportation. Then it forces that the vehicle must arrive before leave, and appropriate bounds on arrival and departure times.

19.

20.

Similarly, for the transshipment to happen, the two vehicles must be present at transshipment node at the same tine.

## Datasets

The author starts from the well-known problem instances from Li and Lim (2001). The following steps are applied to create the datasets used in this paper.

|  |  |
| --- | --- |
|  | Procedure |
| Step 1 | Pick subsets of the ﬁrst 10 nodes and the ﬁrst 14 nodes |
| Step 2 | Pair pickup-and-delivery requests |
| Step 3 | Randomly generated origin and ﬁnal depots for each vehicle, the number of vehicles equals to the number of requests |
| Step 4 | Associated a cost factor with each vehicle |
| Step 5 | Vary the capacities of the vehicles randomly, at least one vehicle is given enough capacity for carrying the largest transport load request. |
| Step 6 | Adjust the time windows to only 50% of the originals |

# Implementation Details

The following work is completed in this project:

1. A GAMS model of the paper’s formulation of the illustrative example as described in section 3. The model is solved by CPLEX and found correct solution, which we use as reference model.
2. A data-processing python file that handles 1) download the original dataset. 2) As the author describes in section 6, slice part of the dataset. 3) Randomizes vehicle depot location, cost factor, vehicle capacity and etc. 4) Rewrite the dataset to readable format. Since the random data generation process is only vaguely described in the paper, a separate manual detailing data generation process is written as attachment, in hopes that if the paper is to be reproduced again, data can be generated properly.
3. Computational tests consist of large number of different cases to run through. To automate the process, we rewrote our model to python-based modelling package Pyomo.
   1. The reference case is first solved to demonstrate correctness.
   2. Then cases provided by Li and Lim (2001), each with 10 nodes and 4 variants (time window and transshipment), are solved by GUROBI (the solver author uses).
   3. Cases with 14 nodes, as mentioned in the paper, take huge time to solve. Because of that, we only took the first case and solved it to optimality. This case will be used to demonstrate the advantage of adding transshipment capability to the model.
4. Visualization is important to vehicle routing problem, especially for reproducing this paper where optimum result is unknown. Therefore, for the last part, we focused on automated generation of vehicle routing diagrams for inspection. A mini plotting program is written by us, which takes a solved Pyomo model and automatically produces an interactive routing diagram. [Please use this link to view an example plot (14 nodes with time window and transshipment)](https://plot.ly/~yuyuez/1).

# Challenges

## Model Formulation

### Transshipment Flow Conservation Equation

This equation conserves the flow of customer requests () across different vehicles. However, as specifically mentioned by author in section 6, “following which they are solved again with the same dataset but allowing transshipment at all nodes in the network.”, transshipment nodes include pick up and drop off nodes. Therefore, we propose to add and subtract appropriate amount when the conservation is calculated.

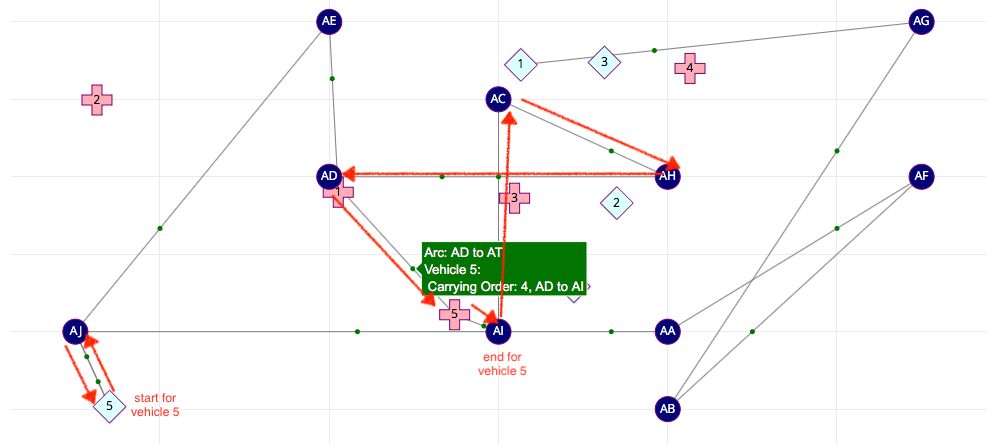
### Normal Node Flow Conservation Equation

Similar to the above formulation, we need to exclude the pickup and delivery node as their flow is obviously not conserved.

### Vehicle Start and End Depot Subtour

Equation 1 and 2, both equations characterize the start and end depot for vehicles. One thing to note here is that only the “vehicle leaving start depot” and “vehicle entering end depot” is constrained. This leads to potential for reverse flow, where vehicle enters the start depot and leaves the end depot, because these nodes are excluded from flow conservation equation 3.

Note that this is not a problem, as any violation will form a subtour and should be caught by sub-tour elimination constraints. However, during tests, we still found sub-tour formation in our solutions.

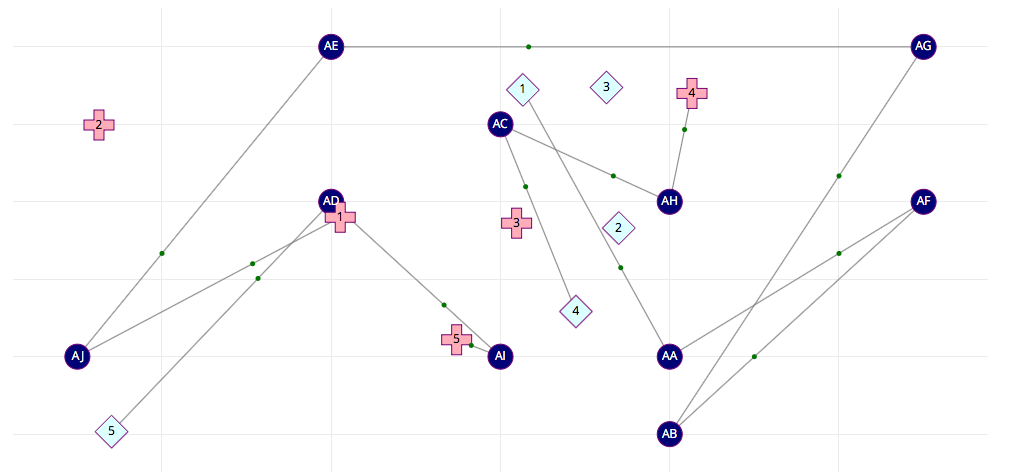


If we look at the subtour elimination constraints proposed by the authors:

It is easy to see that any that leaves start depot or enter end depot is not constrained. Therefore, although their reverse counterparts are registered, the subtour will not be eliminated because equation 14 still holds.

We propose to simply add two constraints to stop vehicle from entering a start depot, as well as leaving an end depot.

The following is the solution after the constraints are added.



In this new solution, we see vehicle 5, as intended, goes to AD and delivers to AI, before goes to the end depot. As such, the original “cheated” delivery from AC to AH is now handled by vehicle number 4.

### Typo in Equation 19

Indices for should be .

### Speed up Computation

In the formulation proposed by the author, every arc is a feasible arc as its distance and cost can be computed. In practice, based on triangle inequality and the fact that cost and time for each arc is proportional to the distance traveled, it is easy to see that there is no benefit for any vehicle to travel to other vehicles’ start and end depot. Therefore, we propose that connections between all start and end depot are not feasible and thus should be removed.

## Data Generation

As discussed previously, a lot of the data for computation is randomized, which has caused some problems. Here is a list of assumptions and adjustments we have applied to the generation process.

|  |  |
| --- | --- |
|  | Additional Assumption / Adjustments |
| Pair Request | In the original problem, pickup and delivery quantity is not paired. We will choose the pickup amount as the request amount |
| Cost Factor |  |
| Vehicle Capacity |  |
| Time Window | Since pickup and delivery are paired arbitrarily, there is no guarantee that the time window is feasible. We calculate the transport time for each order and make sure that there is in between |

## Unresolved Issues

Although the authors gave clear instructions on which parameter to randomly generate, it is still challenging because the value and distribution of the random process is unknown. Note that variables like vehicle capacity, and the pairing process, have huge impact on the final solution. Currently there is no way for us to confirm that our generated data fits closely to the data the authors use, as such there is little point in comparing results.

In addition, although we attempted to guarantee feasibility of the problem by controlling the time window, it is easy to see that, when only one vehicle is able to transport the load for multiple orders, it may not be able to meet the all the time window constraints. Similarly, we have encountered other scenarios where the problem is infeasibility. Unfortunately, it is out of the scope of this project to write a script to robustly generate data that is guaranteed to be feasible.

# Results

## Illustrative Example

The base model structure is implemented in both GAMS and Pyomo and solved to optimality. The result has been checked to be the same as the result listed in the paper. (Obj: 15)

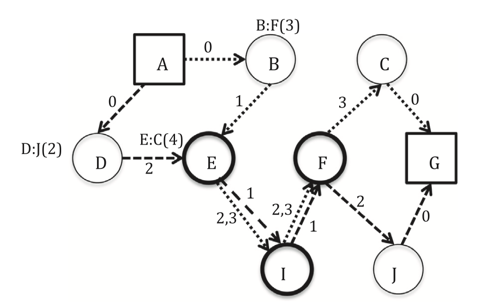
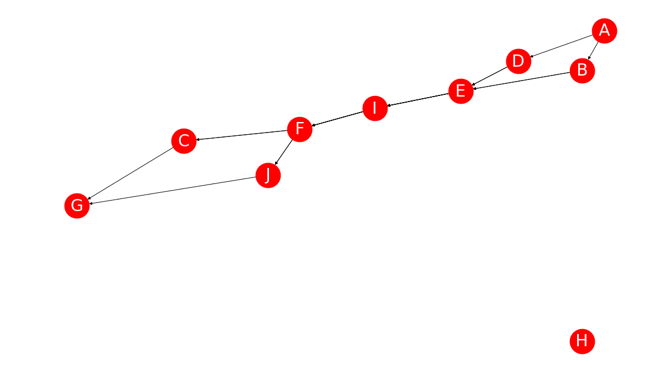


Figure 1 Left: Pyomo, Right: Paper, Our GAMS and Pyomo model produces the same result as paper.

## 10-Node Example

The paper solved 4 variants of the model for each case: 1) no time window no transshipment. 2) no time window yes transshipment. 3) force time window no transshipment. 4) force time window yes transshipment. We solved the above cases with our independently generated data, using GUROBI with a time limit of 1000 second. The results are listed below:

Table 1 NO Time Window

|  | NO Transshipment | | | YES Transshipment | | |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Case | Obj | CPU time | Gap | Obj | CPU time | Gap | Opt Gap |
| lc101.dat | 524.65 | 12.9 | 0 | 524.65 | 11.7 | 0 | 0.00% |
| lc102.dat | 942.43 | 553.1 | 0 | 942.43 | 276.9 | 0 | 0.00% |
| lc103.dat | 688.02 | 1000 | 0.09 | 671.24 | 340.1 | 0 | 2.50% |
| lc104.dat | 633.64 | 7.5 | 0 | 618.77 | 5.1 | 0 | 2.40% |
| lc106.dat | 525.6 | 31.9 | 0 | 525.6 | 12.1 | 0 | 0.00% |
| lc107.dat | 1064.7 | 1000 | 0.07 | 1064.7 | 1000 | 0.11 | 0.00% |
| lc108.dat | 860.91 | 1000 | 0.03 | 860.91 | 1000 | 0.07 | 0.00% |
| lc109.dat | 730.57 | 84.7 | 0 | 730.57 | 101.9 | 0 | 0.00% |
| lc203.dat | 3508.13 | 268.8 | 0 | 3508.13 | 167.2 | 0 | 0.00% |
| lc204.dat | 3796.0 | 1000 | 0.02 | 3796.0 | 1000 | 0.06 | 0.00% |

Table 2 YES Time Window

|  | NO Transshipment | | | YES Transshipment | | |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Case | Obj | CPU time | Gap | Obj | CPU time | Gap | Opt Gap |
| lc101.dat | 820.56 | 21.4 | 0 | 814.39 | 47.3 | 0 | 0.76% |
| lc102.dat | 958 | 1000 | 0.01 | 958 | 451.2 | 0 | 0.00% |
| lc103.dat | 671.24 | 1000 | 0.03 | 671.24 | 266.3 | 0 | 0.00% |
| lc104.dat | 648.87 | 10.9 | 0 | 618.77 | 6.0 | 0 | 4.86% |
| lc106.dat | 999.07 | 766.5 | 0 | 886.2 | 216.6 | 0 | 12.74% |
| lc107.dat | 1141.24 | 134.8 | 0 | 1138.28 | 79.2 | 0 | 0.26% |
| lc108.dat | 1041.83 | 140.4 | 0 | 1017.58 | 191.0 | 0 | 2.38% |
| lc109.dat | 795.13 | 207.5 | 0 | 795.13 | 189.1 | 0 | 0.00% |
| lc203.dat | 3548.0 | 1000 | 0.03 | 3548.0 | 572.8 | 0 | 0.00% |
| lc204.dat | 3796.0 | 1000 | 0.06 | 3769.22 | 1000 | 0.07 | 0.71% |

## 14-node Example

14-nodes case is much more computational expensive and even in the paper, the authors reported solution time of over 10,000 sec. Therefore, we didn’t test our model on all of the cases. However, we did pick out the first case and solved the model.

|  | NO Transshipment | | | YES Transshipment | | |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Case | Obj | CPU time | Gap | Obj | CPU time | Gap | Opt Gap |
| lc101.dat | 3293.96 | 119.4 | 0 | 3221.9 | 2000 | 0.16% | 2.2% |

As can be seen, transshipment offers bigger benefit at larger cases. If you look at the appendix, where we plotted both results side by side, it can be easily seen that transshipment allows the selection of shorted path, as well as cheaper vehicle.

# Comparison and Discussion

Note that because of the random data generation process, we cannot directly compare optimum value and CPU time with the paper. Nevertheless, judging by the results we listed above, we are confident that we have reproduced the trends the authors want to show us.

First, transshipment offers similar percentage increases as the paper. In the paper, the benefit of having transshipment ranges from 0.13% to 6.88%, where our own implementation produces a benefit of 0.26% to 12.74%. Note that in both results about 50% of cases report no improvement after transshipment is applied.

The paper argues that this behavior is the result of using Euclidean distance in the objective, which in essence, means that there is more incentive for a direct delivery for each request (shortest distance). Therefore, every time transshipment happens at a cost. We agree with this analysis, and we add that this behavior is also made obvious by the way the author specifies one vehicle for each order, which makes the geotechnical location of the vehicle’s start and depot matters way more than it should (If one vehicle serves only one order). We postulate that if the number of vehicles is far less than the number of orders, which is a real-world scenario, then the effect of transshipment will be much more obvious.

In our computational experiments, it is clear that cases with time window constraints report more improvement than cases without. This behavior is reasonable as cases with time window constraints are less optimized and can benefit more from relaxing to transshipment.

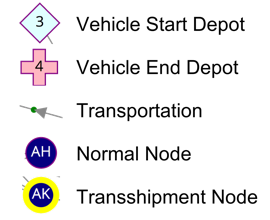
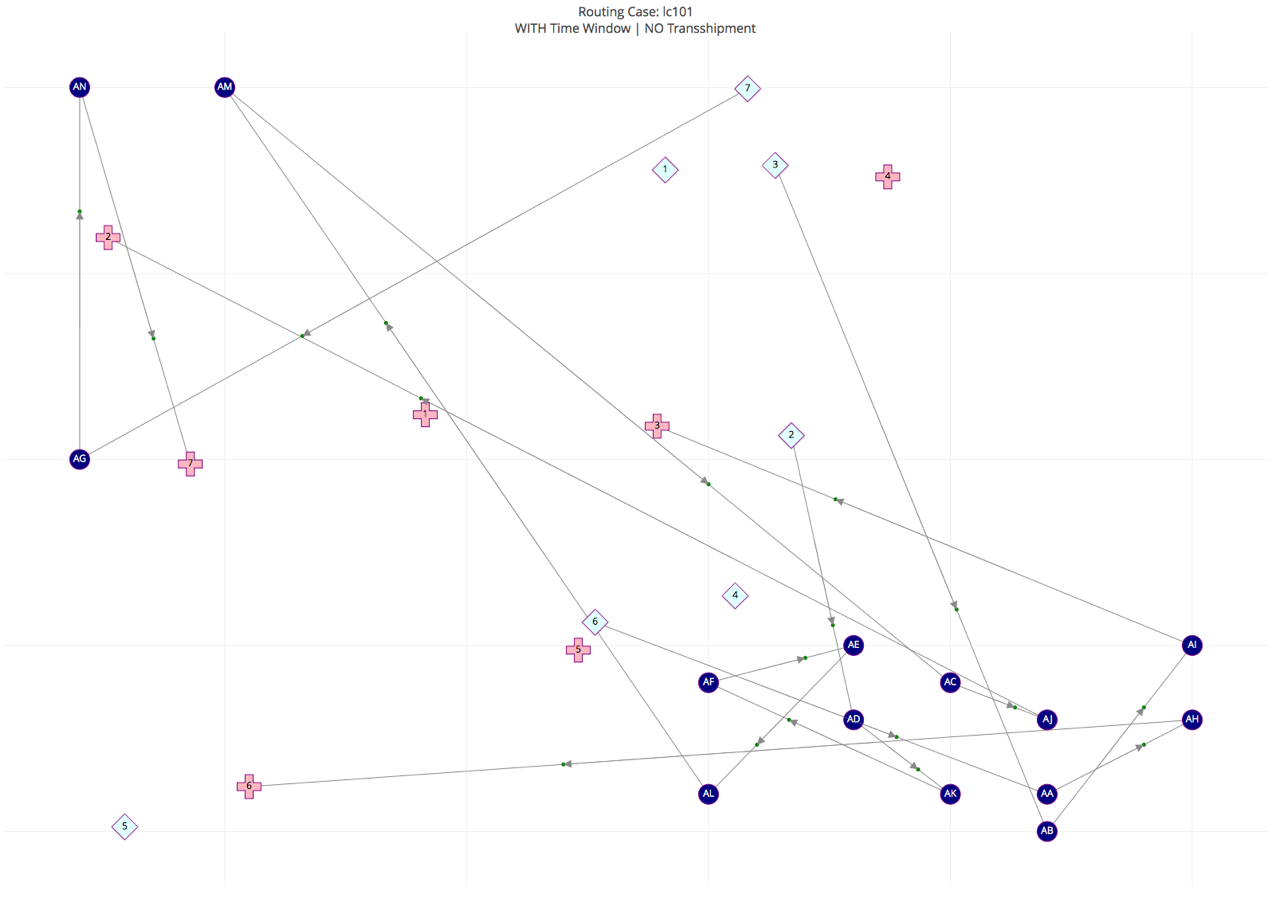
Another interesting find that is different from the paper is that our model appears to solve faster with transshipment allowed. Obviously, a lot of factors could influence this behavior and we suspect that it has a lot to do with our vehicle capacity and time windows.

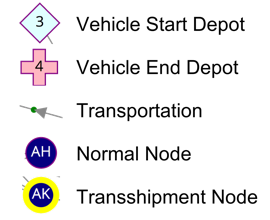
Overall, the results of the paper are reproduced in this project. The randomized data actually produces very similar improvements when comparing transshipment model against non-transshipment models. The fact that two independently generated data produces similar results strengthens the paper’s credibility.

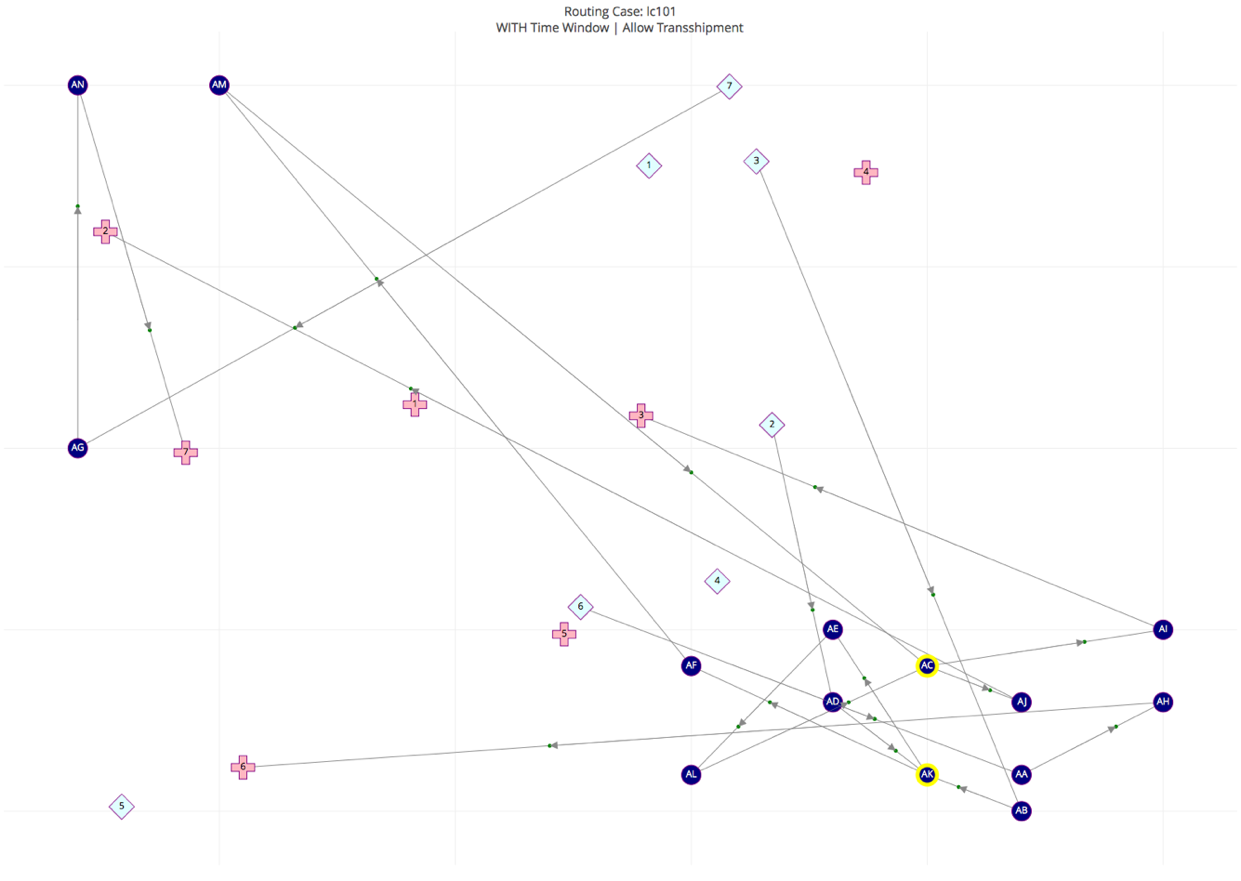
To further demonstrate the benefit of introducing transshipment, two approaches could be adopted in further experiment:

1. Increase the ratio of number of requests to the number of vehicles.
2. Manually generate the transshipment nodes in a more accessible location.
3. Change the travelling cost from point to point Euclidean distance to a tree-graph like structure, where transshipment nodes acts as branching points and therefore vehicles will have to pass those nodes. As a result, no penalty (detour) cost will incur and transshipment benefits will be more obvious.

Appendix – 14-node Comparison

[](https://plot.ly/~yuyuez/3)



[](https://plot.ly/~yuyuez/1)

References

A. Rais, F. Alvelos, M.S. Carvalho, New mixed integer-programming model for the pickup-and-delivery problem with transshipment, European Journal of Operational Research, Volume 235, Issue 3, 2014, Pages 530-539,

Li, H., & Lim, A. (2001). A metaheuristic for the pickup and delivery problem with time windows. In 13th IEEE international conference on tools with artiﬁcial intelligence (pp. 333–340). Los Alamitos, CA: IEEE Computer Society.

Drexl, M. (2012a). Synchronization in vehicle routing – A survey of VRPs with multiple synchronization constraints. Transportation Science, 297–316.

Cortés, C., Matamala, M., & Contardo, C. (2010). The pickup and delivery problem with transfers: Formulation and a branch-and-cut solution method. European Journal of Operational Research, 200, 711–724.

Dondo, R., Méndeza, C., & Cerdá, J. (2009). The supply-chain pick-up and delivery problem with transshipments. In J. Jez´owski & J. Thullie (Eds.), 19th European symposium on computer aided process engineering ESCAPE19 (pp. 1009–1014). Elsevier B.V./Ltd..

Takoudjou, T. R., Deschamps, J. C., & Dupas, R. (2012). A MIP formulation for the pickup and delivery problem with time window and transhipment. In Proceedings of the 14th IFAC symposium on information control problems in manufacturing (Vol. 14-1, pp. 333–338).