5-2021

Improving Logistics Efficiency Through Collaborative Truck Routing

Patrick Dougherty

Improving Logistics Efficiency Through Collaborative Truck Routing

Abstract:

The logistics industry is among the world's largest and most essential. Specifically, trucking is a massive component of the modern logistics system. In 2012, truck transportation carried 68% of all freight tonnage in the U.S. However, trucking currently faces significant problems with efficiency and sustainability. Of all miles driven by trucks yearly in the U.S., 25% are driven with empty loads and 36% are driven with underutilized loads. In addition to this economic inefficiency, the industry faces social and environmental challenges. Driver turnover rates are near 100%, and trucking accounts for a significant level of greenhouse gas emissions. One potential cause of these issues is the segmented nature of the industry which results in transportation service providers routing their shipments independently of each other, therefore increasing the number of miles driven and emissions produced. Collaborative routing, wherein providers work together to share trucks, routes, and resources, is one approach to solving this issue. However, current research has yet to develop a method that is both effective and efficient. This research aims to develop a model for a collaborative routing methodology wherein trucking service providers would share resources to increase load efficiency while decreasing emissions and long-haul drive frequency.

1. Background and Significance

According to the American Trucking Association, $1.4 trillion was spent in the trucking industry in 2016, accounting for 7.5% of the U.S. GDP [1]. Populations around the world rely on trucks to transport goods across vast distances, thus enabling the development of a global economy. The industry also accounts for 7.4 million American jobs, with truck driver as the most common job in 29 states [1,2]. The economic impacts of the industry are significant.

However, as the demand on the logistics industry continues to grow, its core problems become more evident. Trucking struggles significantly with inefficiency in its daily operations. Specifically, the average truck trailer is just 42% full by metric weight [3]. If all trucks were to drive with 100% full loads, a $60 billion savings opportunity could be generated for trucking service providers [3]. The environment also suffers as a result of this inefficiency. In 2011, trucking accounted for 6% of all greenhouse gas emissions in the U.S. [4]. If loads were all packed to full capacity, there could be a decrease of 233 million metric tons of carbon dioxide produced each year [3]. There is a pressing need to discover new ways of reducing the carbon footprint while maintaining efficient operations that move goods at the pace consumers around the world enjoy today, and collaborative truck routing could be one approach of confronting this task.

Currently, providers operate independently from one another without sharing loads, resources, or facilities. The concept of horizontal collaboration in the trucking industry has been in development for more than a decade. The Physical Internet initiative, an ongoing approach to improving logistics sustainability, addresses challenges such as modular shipping containers and digital connectivity between loads [5]. However, large-scale implementation of collaboration has yet to be achieved. Efforts have been made to model collaborative truck routing in the past, however, existing methods lack the required computational efficiency to scale the optimization up to real-world transportation networks. Progress toward such methodologies could serve as additional motivation for logistics providers to consider implementing collaborative elements to their supply chains.

2. Literature Review

The idea of collaboration within the logistics industry has been developing as a research area for some time. The Physical Internet Initiative, originating from Professor Benoit Montreuil in 2006, is an ideological approach to combatting the sustainability challenges involved with the current methods of transportation, handling, storage, realization, supply, and usage of physical objects throughout the world [5]. Montreuil asserts that the logistics system must experience a similar revolution to that of the information and telecommunications community decades ago, when the digital world required fundamental changes [5]. Taking inspiration from the physical transportation and logistics systems, the digital world developed a new framework for itself: the information highway [5]. This approach proved massively successful for the digital community and led to the development of the Digital Internet, which describes the interconnection between networks that allows the transmission of formatted data packets in a standard way. This

revolution clearly had and continues to have incredible impact on the methods by which information is communicated across the globe [5].

Montreuil argues that the physical logistics system should in turn take inspiration from the digital world to address the environmental, economic, and social challenges currently faced by the industry. He named this approach the Physical Internet to provide a vision for how this revolution could occur [5]. Among the main characteristics of this vision are modular shipping containers (termed as π-containers), which would mimic digital packets by being world-standard, smart, green, and varied in modular sizes to maximize the efficiency of container storage and transportation [5]. Another characteristic of the Physical Internet is the Open Global Supply Web, which suggests a shift from private supply networks to an open network that contains product realization centers, distribution centers, warehouses, distributors, and retailers that are open and interconnected [5]. This concept is of particular interest to the following research, as it lays the ideological foundation for how transportation service providers could collaborate to route vehicles for maximum efficiency in terms of cost, environmental impact, and employee satisfaction. The Physical Internet vision has grown dramatically over the last decade, specifically with regards to research involvement, and this research is intended to provide additional justification to this initiative's potential implementation.

There has not been extensive research previously published with regards to the specific elements of collaborative truck routing. One paper written by R. Steven Roesch outlined a problem description based on a traditional pickup and delivery problem that, given a set of transportation requests, a set of delivery resources, and a transportation network consisting of nodes and arcs, determines the routes of transportation requests, the routes of delivery resources, and the assignment of transportation requests to delivery resources [6]. The collaborative element introduced by Roesch is allowing transportation requests to be transferred between delivery resources at pre-specified transshipment nodes [6]. A transfer involves physically moving a transportation request from one delivery resource (i.e. truck) to another in order to obtain better resource utilization and lower overall costs [6]. The main concern with this model, however, is its ability to identify near-optimal solutions within feasible amounts of time [6]. This research, while conceptually interesting, is not very practical for real world-sized problems. However, multiple concepts from this paper, including transshipment nodes, were adopted and/or modified for the purposes of this research. Additionally, research by Matthew Walters presented an optimization model to quantify the benefits of a collaborative supply chain [7]. His results were promising in that the collaborative model consistently outperformed the more traditional, non-collaborative for metrics such as total cost and average truck utilization [7].

3. Modeling

This section will address the design and development of the collaborative truck routing model. In 3.1, an overview of the problem will be provided. In 3.2, the model will be fully formulated and described.

3.1 Problem Description

This implementation of collaborative truck routing is modeled as a logistics network of nodes, trucks, and loads. Given a set of loads, a set of trucks, and fixed routes for each truck through the network of nodes, the model determines the load swapping decisions to minimize the total cost of transportation. Nodes are separated into three categories: origin nodes, demand nodes, and transshipment nodes. Origin nodes are the starting locations of all trucks, with each truck having a unique origin node. Demand nodes are nodes that require a certain quantity of loads to be delivered. Each load within the system is pre-destined for one demand node. Transshipment nodes, which enable the collaborative element of the model, are nodes that trucks can stop at on their way to demand nodes. While at a transshipment node, a truck may swap any number of loads with other trucks that are also present at that node in that time period.

A standard supply chain problem contains routing, vehicle procurement, facility location and demand satisfaction decisions. The direction of this research is to make progress towards solving the collaborative version of a standard supply chain problem. Our work studies a particular subproblem of this general supply chain problem with collaboration that assumes that trucks are assigned to fixed routes, with each truck having a predetermined path through the network to a specified demand node. To integrate fixed routes into the model, the location parameter was utilized. Notated as $z_{s,t,v}$ this binary parameter is equal to 1 if truck v is at transshipment node s in time period t and is equal to 0 otherwise. This parameter provides the locations of all trucks in all time periods of the model (excluding the first and last time periods) and therefore define the opportunities for collaborative trailer usage at transshipment nodes. Throughout the remainder of this work, we define collaboration to occur when loads are exchanged between trailers at intermediate transshipment nodes of our network. For simplicity, we refer to this exchange as a swap. For two or more trucks to swap loads, they must be present at the same transshipment node in the same time period. Figure 1 shows a simplified network with two origin nodes, two transshipment nodes, two demand (destination) nodes, and two vehicles, wherein the values of $z_{s,t,v}$ are displayed for each transshipment node.

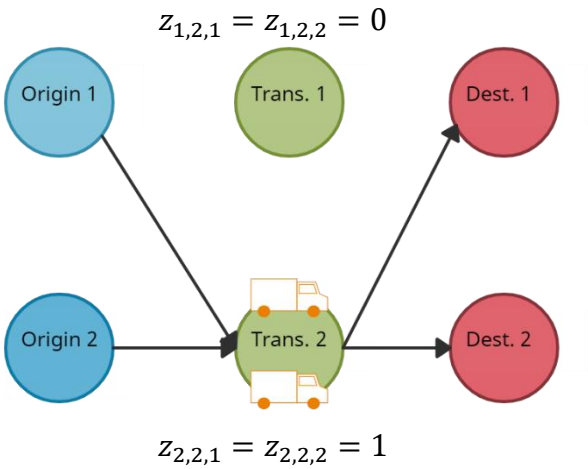

Figure 1: The binary location parameter specifies whether or not a certain truck is at a certain transshipment node in certain time period. When multiple trucks are at a transshipment node at the same time, they may swap loads.

3.2 Model Formulation

Parameters:

D set of demand nodes/types.

S set of transshipment nodes.

T set of time periods.

V set of vehicles.

g per load handling cost.

h per vehicle holding cost.

$p_{v,d}$ quantity of loads of type d on vehicle v at t = 0.

$z_{s,t,v}$ = 1 if vehicle v is at node s in period t, 0 otherwise

u_v capacity of vehicle v $\forall v \in V$.

b_{vd} = 1 if vehicle v is destined for demand node d, 0 otherwise $\forall d \in D$.

q_d quantity demanded by demand node d $\forall d \in D$.

Decision Variables:

x_{dvst} = number of loads of type d gained or lost from vehicle v at transshipment node s in time period t

$y_{vt} = 1$ if vehicle v contains any loads in time period t, 0 otherwise

Objective Function:

Minimize

$$\sum_{s\in S}\sum_{d\in D}\sum_{v\in V}\sum_{t\in T} g\, x^{+}_{dvst}\, z_{s,t,v} + \sum_{v\in V}\sum_{t\in T} h y_{vt} \tag{1}$$

Subject to:

$$\sum_{v\in V} x_{dvst} z_{s,t,v} = 0 \qquad \forall d \in D, \forall s \in S, \forall t \in T \tag{2}$$

$$\sum_{d\in D}(p_{v,d} + \sum_{s\in S}\sum_{1}^{t} x_{dvst} z_{s,t,v}) \le u_v y_{vt} \qquad \forall v \in V, \forall t \in T \tag{3}$$

$$\sum_{v\in V} b_{vd}(p_{v,d} + \sum_{\forall t\in T}\sum_{s\in S}(x_{dvst} z_{s,t,v})) \ge q_d \qquad \forall d \in D \tag{4}$$

$$\sum_{s\in S}\sum_{d\in D}(\, x^{+}_{dvst} z_{s,t,v} + x^{-}_{dvst} z_{s,t,v}) \le 2u_v y_{vt} \quad \forall v \in V, \forall t \in T \tag{5}$$

$$x_{dvst} z_{s,t,v} = x^{+}_{dvst} z_{s,t,v} - x^{-}_{dvst} z_{s,t,v} \qquad \forall v \in V, d \in D, \forall s \in S, \forall t \in T \tag{6}$$

$$x^{+}_{dvst}, x^{-}_{dvst}, y_{vt} \ge 0 \qquad \forall v \in V, d \in D, \forall s \in S, \forall t \in T \tag{7}$$

The objective function (1) minimizes the total transportation cost required to deliver all loads based on the number of load swaps that occur (handling cost) and the number of time periods trucks spend with freight onboard (holding cost). Constraint (2) ensures that the total number of loads within the system does not change. Constraint (3) ensures trucks do not carry more loads than their capacity permits. Constraint (4) ensures that each demand node receives the quantity of loads it demands by the end of the time horizon. Constraint (5) relates the x and y

decision variables to ensure that y represents a binary variable for whether or not a truck has a load on it in a given time period. Constraint (6) separates the net change in loads into loads gained and loads swapped so that load swaps can be properly penalized in the objective function. Constraint (7) ensures non-negativity for the decision variables.

4. Data Generation

To generate a representative supply chain, this research uses pseudo-randomly generated data to test the collaborative truck routing model. 10 model instances were developed using Microsoft Excel. Each instance represents one logistics scenario, with a specified generation of nodes, trucks, loads, demand types, and truck routes. To facilitate the generation of these instances, a script was written in Visual Basic for Applications. The full code is included in the appendix of this document. This script allows a user to input four base parameters through a user form: the number of transshipment nodes, the number of time periods, the number of trucks, and the number of demand types. This user form can be seen in Figure 2. The script then automates the generation of instances in the form of rows of cells within an Excel spreadsheet.

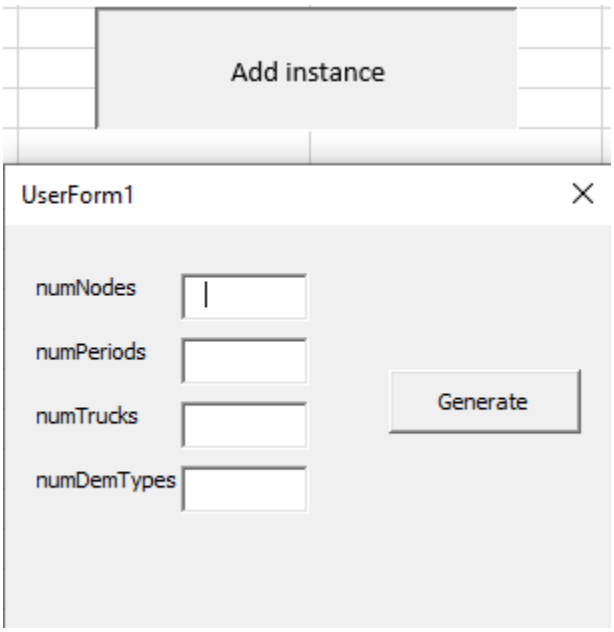

Figure 2: The user form takes in values for the desired number of nodes, number of time periods, number of trucks, and number of demand types for an instance and automates the generation of most other parameters in the instance.

When the user enters values for each field in the form and clicks “Generate,” the code executes. First, the entered values are stored as variables and printed to a worksheet in the first four columns. Next, each truck is assigned a maximum capacity as a random number in the range

from 1 to 5 loads. These capacities are then printed to the worksheet. Next, the initial load allocation headers are printed. Each truck is manually assigned an initial quantity of loads of each demand type such that the truck's total starting cargo is less than or equal to its capacity. The initial load allocations are inserted manually by the user.

Next, the binary location parameters, which essentially act as the routes for all trucks through the network, are generated. First, the header cells for each parameter are printed. The number of location parameters created is dependent on the input values from the user. The calculation is shown below.

$$Number\ of\ location\ parameters = numNodes \times (numPeriods - 2) \times numTrucks$$

The location parameters are three-dimensional arrays with numNodes (the number of transshipment nodes in the network), numPeriods (the number of time periods) and numTrucks (the number of trucks in the network) as the indices, respectively. It is important to note that location parameters are not created for the first and final time periods. This is because in time t = 1, all trucks are at their origin nodes. At the final time period, all trucks have arrived at destination nodes. Since the decisions of the model are the swaps that are made at intermediate transshipment nodes, it is unnecessary to consider these time periods. The script prints headers for each location parameter in the form of [numNodes, numPeriods, numTrucks]; for example, [1][2][3] represents whether truck 3 is at transshipment node 1 in time period 2, with a value of 1 meaning it is present, and a value of 0 meaning it is not.

Binary values are then generated for all location parameters in the instance. To randomly generate routes for each truck, the following procedure is executed: for each combination of numPeriods and numTrucks, a random number is generated between 1 and the total number of transshipment nodes. This number determines which transshipment node that truck will be at in that time period. The script then stores that random number as the numNodes index in a location parameter, followed by the associated numPeriods and numTrucks indices. Next, the script searches through the existing location parameter headers in that instance row to find the cell location which matches each index. When that cell is found, the cell directly underneath is given a value of 1, affirming that the truck is at that transshipment node in that time period. This process is repeated until every combination of time periods and trucks has been assigned to a transshipment node. Finally, the script populates the remaining location parameter cells with 0s to ensure that a truck cannot be located at more than one transshipment node in the same time period. An example of location parameter generation in one instance can be seen in Figure 3.

[1][2][1]	[1][2][2]	[1][2][3]	[1][2][4]	[1][2][5]
1	0	0	0	1

Figure 3: Binary location parameters are generated for each instance. In this case, truck 1 and truck 5 are both present at transshipment node 1 in time period 2 and can swap loads with each other.

5. Experiment

In 5.1, the translation of the model and data files into an AMPL context will be described. In 5.2, the results of the model solutions on multiple instances will be discussed.

5.1: Model Solution

To compute solutions to the collaborative logistics model, the AMPL modeling platform was used. AMPL (A Mathematical Programming Language) is a commercial optimization software that is powerful for large-scale mathematical computing. AMPL integrates a modeling language that allows the description of data, variables, objectives, and constraints. It also includes a command language for browsing models and analyzing results.

First, the model formulation outlined in section 3.2 was translated to a .mod format to enable successful AMPL integration. This involves the description of each set, parameter, and decision variable as well as the objective function and constraints. The .mod file can be seen in the appendix. Next, each of the 10 generated instances described in section 4 were translated into .dat files using the appropriate AMPL syntax. The .dat files for each instance can be seen in the appendix.

To solve the model with each instance, the CPLEX optimization package was utilized. CPLEX enables the solution of integer programs, such as this model which represents a collaborative logistics environment. CPLEX calculates the optimal objective function value for a particular model and data file and returns the associated values of decision variables for each solution. Therefore, it can be used for this model to determine the optimal swapping decisions for each instance.

5.2 Results

Each instance was solved to optimality in AMPL and the results were recorded. For each instance, the results include the optimal objective function value, which represents the total cost of the solution, as well as the decision variables that describe the swapping activity that occurs in the network. Table 1 shows the key parameters for each instance as well as the resulting total cost and number of swaps in the optimal solution. Parameters g and h, which represent the per load swapping cost and per vehicle per time period holding cost respectively, were held constant at values of 1 and 10 across all instances.

Instance	# of Transshipment Nodes	# of Time Periods	# of Trucks	# of Demand Nodes	Total Loads Demanded	Total Cost	Total # of Swaps
1	2	5	3	2	5	92	2
2	4	6	5	4	17	215	15
3	5	6	7	5	21	292	12
4	3	4	4	3	9	88	8
5	4	7	8	4	35	419	19
6	3	5	3	2	9	94	4
7	6	7	10	5	32	517	17
8	5	8	6	3	16	367	7
9	2	7	4	4	8	204	4
10	6	8	7	2	22	430	10

Table 1: This table contains the key parameters and results for each instance tested.

Results varied across all instances as each instance was unique in its parameter values. Generally, it appears that instances that are larger in scope (higher numbers of transshipment nodes, demand nodes, time periods, and trucks) promoted more frequent swapping activity.

Table 2 shows the swapping decision results for instance 5 in which there are 4 transshipment nodes, 7 time periods, 8 trucks, and 4 demand nodes. Interestingly, all swaps occur in either time period 2 or time period 4, and there are no swaps made at transshipment node 1. This demonstrates the optimizable nature of collaborative truck routing whereby logistics planners can determine the optimal network design, including the number and placement of transshipment nodes, to produce the most cost and time efficient routing solutions possible.

Demand Type	Truck	Transshipment Node	Time Period	Net Swaps
1	1	4	4	-3
1	8	4	4	3
2	1	3	2	-1
2	3	3	4	-1
2	5	2	4	-4
2	6	2	4	4
2	6	3	2	1
2	7	3	4	1
3	1	3	2	1
3	1	4	4	3
3	2	3	4	-1
3	3	3	4	1
3	6	3	2	-1
3	8	4	4	-3
4	2	3	4	1
4	5	2	4	4
4	6	2	4	-4
4	7	3	4	-1

Table 2: This table contains the swapping decision results (values of x_{dvst}) for all swaps in the solution of instance 5.

Another interesting finding was the discovery that, as the total number of loads demanded in the instance increased, the number of swaps in the solution increased linearly. To test this, instance 5 was expanded into 4 versions, with the first version having its initial loads and quantity demanded subtracted by 5 (from the original instance) for each demand type. Each successive version added 1 demand of each type, such that 4 total demand was added each time. With each new version, the number of swaps and total cost increased by 4, just as the total demand did. The resulting number of swaps and total costs for each version are shown in Table 3.

Version of Instance 5	Total Loads Demanded	Total Number of Swaps	Solution Cost
1	15	3	403
2	19	7	407
3	23	11	411
4	27	15	415

Table 3: This table shows the linear increase of number of swaps and solution cost with an increase in total loads demanded in the instance.

Conclusions

This paper modeled a subproblem of the overall collaborative supply chain problem. The results demonstrate that truck routing that includes the ability for trucks (potentially from different logistics providers) to stop at pre-determined transshipment facilities along their paths to swap loads can take advantage of this collaboration to produce optimal solutions. This research focused specifically on fairly small instances to demonstrate the feasibility of the collaborative routing methodology. The model can be utilized in a decomposition procedure to solve larger problems of similar design. It also provides a testing platform for gathering insights on when swapping is advantageous within a logistics network. The computational results of this research show that this type of subproblem can be approximately solved using a linear program without significant sacrifice in solution quality, which suggests that the subproblem can be successfully solved quickly in future applications. This research also included the creation of a test instance generator, which can be used in the future to support other research efforts within collaborative logistics. The generator can also be used to quickly generate results for sensitivity analysis on the current model.

There are multiple next steps to advance this subproblem to continue the development of collaborative logistics research. In this paper, the instances used for testing were relatively small compared to the size of real-world logistics networks. Datasets that are closer in scale to real-world sizes can be acquired and tested with this model to determine the computational limits in solving the integer program using a black box optimization solver. Further research could also expand upon the concept of a neighborhood-search based heuristic that defines neighborhoods as a result of possible swaps at a transshipment node. Finally, there could be an exploration of the NP-hardness of this subproblem to determine whether a polynomial time algorithm may be possible.

References

[1] "Reports, Trends & Statistics." *American Trucking Associations*.

[2] Bui, Quoctrung. "Map: The Most Common* Job In Every State." *NPR*, NPR Planet Money, 5 Feb. 2015.

[3] William Ferrell, Kimberly Ellis, Phil Kaminsky & Chase Rainwater (2019) Horizontal collaboration: opportunities for improved logistics planning, International Journal of Production Research, DOI: 10.1080/00207543.2019.1651457

[4] Environmental Protection Agency, 16 July 2019. Fast Facts on Transportation Greenhouse Gas Emissions, www.epa.gov/greenvehicles/fast-facts-transportation-greenhouse-gas-emissions.

[5] Montreuil, B., Meller, R.D., & Ballot, E. (2010). Towards a Physical Internet: the Impact on Logistics Facilities and Material Handling Systems Design and Innovation.

[6] Roesch, R. Steven. (2016). Transportation Service Provider Collaboration Problem: Potential Benefits and Solution Approaches. Virginia Tech.

[7] Walters, Matthew (2021). Quantifying the Benefits of a Collaborative Supply Chain Network using a Discrete-Time Vehicle Routing Model. University of Arkansas.

Appendix

a. *VBA Script for Instance Generation:*

```
Private Sub UserForm_Initialize()
Dim TcNodes As Integer
Dim TcPeriods As Integer
Dim TcTrucks As Integer
Dim TcDem As Integer

End Sub

Private Sub GenerateButton_Click()

Worksheets("Sheet1").Activate
Dim emptyRow As Long
emptyRow = WorksheetFunction.CountA(Range("A:A")) + 1


Dim truckCap() As Variant
ReDim truckCap(TcTrucks)
Dim i As Integer
Dim j As Integer
Dim firstLocCell As Integer


Cells(emptyRow, 1).Value = "numNodes"
Cells(emptyRow, 2).Value = "numPeriods"
Cells(emptyRow, 3).Value = "numTrucks"
Cells(emptyRow, 4).Value = "numDemTypes"
Cells(emptyRow + 1, 1).Value = TcNodes
Cells(emptyRow + 1, 2).Value = TcPeriods
Cells(emptyRow + 1, 3).Value = TcTrucks
Cells(emptyRow + 1, 4).Value = TcDem


For i = 1 To TcTrucks
   Cells(emptyRow, 4 + i).Value = "Truck" & i & " Cap"
   truckCap(i) = Int((5 - 1 + 1) * Rnd + 1)
```

```
   Cells(emptyRow + 1, 4 + i).Value = truckCap(i)
Next i

Dim emptyCol As Long
emptyCol = Cells(emptyRow, Columns.Count).End(xlToLeft).Column + 1

For i = 1 To TcTrucks
   For j = 1 To TcDem
      Cells(emptyRow, emptyCol).Value = "Truck " & i & " Initial D" & j & " loads"
      emptyCol = emptyCol + 1
   Next j
Next i

emptyCol = Cells(emptyRow, Columns.Count).End(xlToLeft).Column + 1
firstLocCell = emptyCol

For i = 1 To TcNodes
  For j = 2 To TcPeriods - 1
      For k = 1 To TcTrucks
        Cells(emptyRow, emptyCol).Value = "[" & i & "]" & "[" & j & "]" & "[" & k & "]"
        emptyCol = emptyCol + 1
      Next k
   Next j
Next i

Dim randNode As Integer
Dim numLocsPerNode As Integer
   numLocsPerNode = TcTrucks * (TcPeriods - 2)
Dim selectedNode As String
Dim nodeCol As Integer

For j = 2 To TcPeriods - 1
   For k = 1 To TcTrucks
      randNode = Int((TcNodes - 1 + 1) * Rnd + 1)
      selectedNode = "[" & randNode & "]" & "[" & j & "]" & "[" & k & "]"
      nodeCol = Rows(emptyRow).Find(What:=selectedNode, LookIn:=xlValues,
LookAt:=xlWhole, SearchOrder:=xlByColumns, SearchDirection:=xlNext,
MatchCase:=False).Column
      Cells(emptyRow + 1, nodeCol).Value = 1
   Next k
```

```
Next j

emptyCol = Cells(emptyRow, Columns.Count).End(xlToLeft).Column
Dim rng As Range

For Each rng In Range(Cells(emptyRow + 1, firstLocCell), Cells(emptyRow + 1,
emptyCol))
   If IsEmpty(rng) Then
      rng.Value = 0
   End If
Next


End Sub
```

b. Collaborative Truck Routing Model .mod file:

```
#Parameters and Sets
set D;
set S;
set T;
set V;
param g;
param h;
param p{v in V, d in D};
param z{s in S, t in T, v in V};
param u{v in V};
param q{d in D};
param b{v in V, d in D};

#Decision Variables
var x{d in D, v in V, s in S, t in T} integer;
var x_plus{d in D, v in V, s in S, t in T} integer >= 0;
var x_minus{d in D, v in V, s in S, t in T} integer >= 0;
var y{v in V, t in T} binary integer >= 0;

#Model
minimize Cost: sum{s in S} sum{d in D} sum{v in V} sum{t in T} g * x_plus[d,v,s,t] *
z[s,t,v] + sum{v in V} sum{t in T} h * y[v,t];
```

```
subject to loads{d in D, s in S, t in T}: sum{v in V} x[d,v,s,t] * z[s,t,v] = 0;
subject to capacity{v in V, t in T}: sum{d in D} (p[v,d] + sum{s in S} sum{1..t}
x[d,v,s,t] * z[s,t,v]) <= u[v] * y[v,t];
subject to demand{d in D}: sum{v in V} (b[v,d] * (p[v,d] + (sum{t in T} sum{s in S} (
x[d,v,s,t] * z[s,t,v])))) >= q[d];
subject to netswaps{v in V, d in D, s in S, t in T}: x[d,v,s,t] * z[s,t,v] = x_plus[d,v,s,t] *
z[s,t,v] - x_minus[d,v,s,t] * z[s,t,v];
subject to fixedCost{v in V, t in T}: sum{s in S} sum{d in D} (x_plus[d,v,s,t] * z[s,t,v] +
x_minus[d,v,s,t] * z[s,t,v]) <=  2*u[v] * y[v,t];
```

c. *Instance 1 .dat file*

```
set D := 1 2;
set S := 1 2;
set T := 2 3 4;
set V := 1 2 3;
param g := 1;
param h := 10;

param p:
        1       2:=
1       1       1
2       1       1
3       1       0;

param b:
        1       2 :=
1       1       0
2       0       1
3       1       0;

param z :=
        [1, *, *]       2 1 1   2 2 1   2 3 1   3 1 0   3 2 0   3 3 0   4 1 0   4 2 0   4 3 0
        [2, *, *]       2 1 0   2 2 0   2 3 0   3 1 1   3 2 1   3 3 1   4 1 1   4 2 1   4 3 1;

param u :=
1       2
2       2
3       3;
```

```
param q :=
1       3
2       2;
```

d. Instance 2 .dat file

```
set D := 1 2 3 4;
set S := 1 2 3 4;
set T := 2 3 4 5;
set V := 1 2 3 4 5;
param g := 1;
param h := 10;

param b:
        1       2       3       4:=
1       0       0       1       0
2       0       1       0       0
3       0       0       0       1
4       1       0       0       0
5       0       1       0       0;


param p:
        1       2       3       4:=
1       1       1       0       1
2       1       1       1       0
3       1       1       1       0
4       0       0       2       2
5       0       1       2       1;

param z :=
        [1, *, *]       2 1 1   2 2 0   2 3 0   2 4 0   2 5 1   3 1 0   3 2 0   3 3 0   3 4 1
        3 5 0   4 1 1   4 2 0   4 3 0   4 4 1   4 5 1   5 1 0   5 2 0   5 3 0   5 4 0   5 5 0
        [2, *, *]       2 1 0   2 2 1   2 3 0   2 4 0   2 5 0   3 1 0   3 2 1   3 3 0   3 4 0
        3 5 0   4 1 0   4 2 0   4 3 0   4 4 0   4 5 0   5 1 1   5 2 1   5 3 0   5 4 0   5 5 1
        [3, *, *]       2 1 0   2 2 0   2 3 1   2 4 0   2 5 1   3 1 1   3 2 0   3 3 1   3 4 0
        3 5 0   4 1 0   4 2 1   4 3 1   4 4 0   4 5 0   5 1 0   5 2 0   5 3 0   5 4 1   5 5 0
        [4, *, *]       2 1 0   2 2 1   2 3 0   2 4 1   2 5 0   3 1 0   3 2 0   3 3 0   3 4 0
        3 5 1   4 1 0   4 2 0   4 3 0   4 4 0   4 5 0   5 1 0   5 2 0   5 3 1   5 4 0   5 5 0;
```

param u :=
1 3
2 3
3 3
4 4
5 4;

param q :=
1 3
2 4
3 6
4 4;

e. Instance 3 .dat file

set D := 1 2 3 4 5;
set S := 1 2 3 4 5;
set T := 2 3 4 5;
set V := 1 2 3 4 5 6 7;
param g := 1;
param h := 10;
param b:

	1	2	3	4	5 :=
1	1	0	0	0	0
2	0	0	0	1	0
3	0	1	0	0	0
4	0	0	0	0	1
5	0	0	1	0	0
6	0	0	0	1	0
7	0	0	0	0	1;

param p:

	1	2	3	4	5:=
1	2	0	1	0	1
2	2	1	0	2	1
3	0	1	0	0	0
4	0	0	0	1	1
5	1	1	1	0	0
6	1	1	0	1	0

```
7       0       0       0       1       1;

param z :=
        [1, *, *]       2 1 0   2 2 0   2 3 1   2 4 1   2 5 0   2 6 0   2 7 0   3 1 0   3 2 0
        3 3 0   3 4 0   3 5 0   3 6 1   3 7 1   4 1 0   4 2 0   4 3 0   4 4 0   4 5 1   4 6 1
        4 7 0   5 1 0   5 2 0   5 3 0   5 4 0   5 5 0   5 6 0   5 7 0
        [2, *, *]       2 1 1   2 2 0   2 3 0   2 4 0   2 5 1   2 6 1   2 7 0   3 1 0   3 2 1
        3 3 0   3 4 0   3 5 0   3 6 0   3 7 0   4 1 1   4 2 0   4 3 0   4 4 0   4 5 0   4 6 0
        4 7 0   5 1 0   5 2 0   5 3 0   5 4 0   5 5 0   5 6 0   5 7 0
        [3, *, *]       2 1 0   2 2 1   2 3 0   2 4 0   2 5 0   2 6 0   2 7 0   3 1 0   3 2 0
        3 3 0   3 4 1   3 5 1   3 6 0   3 7 0   4 1 0   4 2 1   4 3 0   4 4 1   4 5 0   4 6 0
        4 7 1   5 1 0   5 2 0   5 3 0   5 4 0   5 5 0   5 6 0   5 7 0
        [4, *, *]       2 1 0   2 2 0   2 3 0   2 4 0   2 5 0   2 6 0   2 7 1   3 1 1   3 2 0
        3 3 1   3 4 0   3 5 0   3 6 0   3 7 0   4 1 0   4 2 0   4 3 1   4 4 0   4 5 0   4 6 0
        4 7 0   5 1 0   5 2 0   5 3 0   5 4 0   5 5 0   5 6 0   5 7 0
        [5, *, *]       2 1 0   2 2 0   2 3 0   2 4 0   2 5 0   2 6 0   2 7 0   3 1 0   3 2 0
        3 3 0   3 4 0   3 5 0   3 6 0   3 7 0   4 1 0   4 2 0   4 3 0   4 4 0   4 5 0   4 6 0
        4 7 0   5 1 1   5 2 1   5 3 1   5 4 1   5 5 1   5 6 1   5 7 1;

param u :=
1       5
2       6
3       1
4       2
5       3
6       4
7       2;

param q :=
1       6
2       4
3       2
4       5
5       4;
```

f. Instance 4 .dat file

```
set D := 1 2 3;
set S := 1 2 3;
set T := 2 3;
```

```
set V := 1 2 3 4;
param g := 1;
param h := 10;
param b:
        1       2       3 :=
1       1       0       0
2       0       1       0
3       0       0       1
4       0       0       1;

param p:
        1       2       3 :=
1       1       0       0
2       1       1       0
3       1       1       1
4       0       1       2;

param z :=
        [1, *, *]       2 1 0   2 2 1   2 3 1   2 4 0   3 1 0   3 2 0   3 3 0   3 4 0
        [2, *, *]       2 1 0   2 2 0   2 3 0   2 4 1   3 1 0   3 2 1   3 3 0   3 4 1
        [3, *, *]       2 1 1   2 2 0   2 3 0   2 4 0   3 1 1   3 2 0   3 3 1   3 4 0;


param u :=
1       2
2       2
3       3
4       3;

param q :=
1       3
2       3
3       3;
```

g. Instance 5 .dat file

```
set D := 1 2 3 4;
set S := 1 2 3 4;
set T := 2 3 4 5 6;
set V := 1 2 3 4 5 6 7 8;
```

```
param g := 1;
param h := 10;
param b:
        1       2       3       4 :=
1       0       0       1       0
2       0       0       0       1
3       0       0       1       0
4       1       0       0       0
5       0       0       0       1
6       0       1       0       0
7       0       1       0       0
8       1       0       0       0;


param p:
        1       2       3       4:=
1       1       1       1       0
2       0       1       1       1
3       0       0       3       0
4       2       1       0       1
5       1       1       0       2
6       1       2       1       1
7       0       3       1       2
8       2       2       2       1;

param z :=
        [1, *, *]       2 1 0   2 2 1   2 3 1   2 4 0   2 5 0   2 6 0   2 7 0   2 8 0   3 1 0
        3 2 0   3 3 1   3 4 0   3 5 0   3 6 1   3 7 0   3 8 1   4 1 0   4 2 0   4 3 0   4 4 0
        4 5 0   4 6 0   4 7 0   4 8 0   5 1 1   5 2 0   5 3 0   5 4 0   5 5 0   5 6 0   5 7 1
        5 8 0   6 1 0   6 2 0   6 3 1   6 4 0   6 5 0   6 6 0   6 7 0   6 8 0
        [2, *, *]       2 1 0   2 2 0   2 3 0   2 4 1   2 5 1   2 6 0   2 7 0   2 8 0   3 1 0
        3 2 0   3 3 0   3 4 1   3 5 1   3 6 0   3 7 0   3 8 0   4 1 0   4 2 0   4 3 0   4 4 0
        4 5 1   4 6 1   4 7 0   4 8 0   5 1 0   5 2 1   5 3 0   5 4 0   5 5 0   5 6 0   5 7 0
        5 8 1   6 1 0   6 2 1   6 3 0   6 4 0   6 5 0   6 6 1   6 7 0   6 8 0
        [3, *, *]       2 1 1   2 2 0   2 3 0   2 4 0   2 5 0   2 6 1   2 7 1   2 8 0   3 1 0
        3 2 0   3 3 0   3 4 0   3 5 0   3 6 0   3 7 1   3 8 0   4 1 0   4 2 1   4 3 1   4 4 1
        4 5 0   4 6 0   4 7 1   4 8 0   5 1 0   5 2 0   5 3 0   5 4 0   5 5 1   5 6 0   5 7 0
        5 8 0   6 1 1   6 2 0   6 3 0   6 4 1   6 5 1   6 6 0   6 7 0   6 8 0
        [4, *, *]       2 1 0   2 2 0   2 3 0   2 4 0   2 5 0   2 6 0   2 7 0   2 8 1   3 1 1
        3 2 1   3 3 0   3 4 0   3 5 0   3 6 0   3 7 0   3 8 0   4 1 1   4 2 0   4 3 0   4 4 0
```

```
4 5 0  4 6 0  4 7 0  4 8 1  5 1 0  5 2 0  5 3 1  5 4 1  5 5 0  5 6 1  5 7 0
5 8 0  6 1 0  6 2 0  6 3 0  6 4 0  6 5 0  6 6 0  6 7 1  6 8 1;

param u :=
1       3
2       3
3       3
4       4
5       4
6       5
7       6
8       7;

param q :=
1       7
2       11
3       9
4       8;
```

h. Instance 6 .dat file

```
set D := 1 2;
set S := 1 2 3;
set T := 2 3 4;
set V := 1 2 3;
param g := 1;
param h := 10;
param b:
        1       2 :=
1       1       0
2       0       1
3       1       0;

param p:
        1       2 :=
1       2       1
2       1       2
3       2       1;

param z :=
```

[1, *, *]	2 1 0	2 2 0	2 3 0	3 1 0	3 2 0	3 3 1	4 1 0	4 2 0	4 3 0
[2, *, *]	2 1 0	2 2 0	2 3 1	3 1 0	3 2 0	3 3 0	4 1 1	4 2 1	4 3 0
[3, *, *]	2 1 1	2 2 1	2 3 0	3 1 1	3 2 1	3 3 0	4 1 0	4 2 0	4 3 1;

param u :=
1 4
2 3
3 5;

param q :=
1 5
2 4;

i. Instance 7 .dat file

set D := 1 2 3 4 5;
set S := 1 2 3 4 5 6;
set T := 2 3 4 5 6;
set V := 1 2 3 4 5 6 7 8 9 10;
param g := 1;
param h := 10;
param b:

	1	2	3	4	5:=
1	1	0	0	0	0
2	0	0	0	1	0
3	0	0	0	0	1
4	0	1	0	0	0
5	0	0	1	0	0
6	0	0	0	0	1
7	0	1	0	0	0
8	1	0	0	0	0
9	0	0	1	0	0
10	0	0	1	0	0;

param p:

	1	2	3	4	5:=
1	2	2	0	0	0

```
2       0       0       1       1       1
3       1       0       0       0       2
4       0       1       0       1       0
5       0       1       1       0       0
6       1       0       1       1       1
7       0       1       0       0       0
8       2       0       0       1       1
9       1       1       2       1       0
10      1       1       2       0       0;


param z :=
        [1, *, *]       2 1 1   2 2 0   2 3 0   2 4 0   2 5 0   2 6 0   2 7 0   2 8 1   2 9 0
        2 10 0  3 1 0   3 2 0   3 3 1   3 4 0   3 5 0   3 6 0   3 7 0   3 8 0   3 9 0   3 10 0
        4 1 0   4 2 0   4 3 0   4 4 0   4 5 0   4 6 0   4 7 0   4 8 0   4 9 0   4 10 0  5 1 1
        5 2 0   5 3 0   5 4 1   5 5 0   5 6 1   5 7 1   5 8 0   5 9 0   5 10 1  6 1 0   6 2 0
        6 3 0   6 4 0   6 5 0   6 6 0   6 7 0   6 8 1   6 9 1   6 10 0
        [2, *, *]       2 1 0   2 2 0   2 3 0   2 4 0   2 5 0   2 6 0   2 7 0   2 8 0   2 9 0
        2 10 0  3 1 0   3 2 0   3 3 0   3 4 0   3 5 0   3 6 1   3 7 0   3 8 0   3 9 1   3 10 1
        4 1 0   4 2 0   4 3 0   4 4 0   4 5 0   4 6 1   4 7 0   4 8 0   4 9 1   4 10 0  5 1 0
        5 2 0   5 3 0   5 4 0   5 5 0   5 6 0   5 7 0   5 8 0   5 9 1   5 10 0  6 1 1   6 2 0
        6 3 1   6 4 0   6 5 0   6 6 0   6 7 1   6 8 0   6 9 0   6 10 0
        [3, *, *]       2 1 0   2 2 1   2 3 0   2 4 0   2 5 1   2 6 0   2 7 0   2 8 0   2 9 0
        2 10 1  3 1 0   3 2 0   3 3 0   3 4 0   3 5 1   3 6 0   3 7 0   3 8 0   3 9 0   3 10 0
        4 1 0   4 2 0   4 3 0   4 4 0   4 5 0   4 6 0   4 7 0   4 8 0   4 9 0   4 10 0  5 1 0
        5 2 0   5 3 0   5 4 0   5 5 0   5 6 0   5 7 0   5 8 0   5 9 0   5 10 0  6 1 0   6 2 1
        6 3 0   6 4 0   6 5 0   6 6 1   6 7 0   6 8 0   6 9 0   6 10 0
        [4, *, *]       2 1 0   2 2 0   2 3 0   2 4 0   2 5 0   2 6 0   2 7 0   2 8 0   2 9 0
        2 10 0  3 1 1   3 2 0   3 3 0   3 4 1   3 5 0   3 6 0   3 7 1   3 8 1   3 9 0   3 10 0
        4 1 0   4 2 0   4 3 1   4 4 0   4 5 0   4 6 0   4 7 0   4 8 0   4 9 0   4 10 1  5 1 0
        5 2 0   5 3 0   5 4 0   5 5 1   5 6 0   5 7 0   5 8 0   5 9 0   5 10 0  6 1 0   6 2 0
        6 3 0   6 4 0   6 5 0   6 6 0   6 7 0   6 8 0   6 9 0   6 10 1
        [5, *, *]       2 1 0   2 2 0   2 3 0   2 4 1   2 5 0   2 6 0   2 7 0   2 8 0   2 9 0
        2 10 0  3 1 0   3 2 1   3 3 0   3 4 0   3 5 0   3 6 0   3 7 0   3 8 0   3 9 0   3 10 0
        4 1 1   4 2 1   4 3 0   4 4 0   4 5 0   4 6 0   4 7 1   4 8 0   4 9 0   4 10 0  5 1 0
        5 2 0   5 3 1   5 4 0   5 5 0   5 6 0   5 7 0   5 8 1   5 9 0   5 10 0  6 1 0   6 2 0
        6 3 0   6 4 0   6 5 0   6 6 0   6 7 0   6 8 0   6 9 0   6 10 0
        [6, *, *]       2 1 0   2 2 0   2 3 1   2 4 0   2 5 0   2 6 1   2 7 1   2 8 0   2 9 1
        2 10 0  3 1 0   3 2 0   3 3 0   3 4 0   3 5 0   3 6 0   3 7 0   3 8 0   3 9 0   3 10 0
        4 1 0   4 2 0   4 3 0   4 4 1   4 5 1   4 6 0   4 7 0   4 8 1   4 9 0   4 10 0  5 1 0
```

5 2 1 5 3 0 5 4 0 5 5 0 5 6 0 5 7 0 5 8 0 5 9 0 5 10 0 6 1 0 6 2 0
6 3 0 6 4 1 6 5 1 6 6 0 6 7 0 6 8 0 6 9 0 6 10 0;

param u :=
1 4
2 3
3 3
4 2
5 2
6 4
7 1
8 4
9 5
10 4;

param q :=
1 8
2 7
3 7
4 5
5 5;

j. Instance 8 .dat file

set D := 1 2 3;
set S := 1 2 3 4 5;
set T := 2 3 4 5 6 7;
set V := 1 2 3 4 5 6;
param g := 1;
param h := 10;
param b:
	1	2	3 :=
1	0	0	1
2	0	1	0
3	0	0	1
4	1	0	0
5	0	1	0
6	1	0	0;

```
param p:
        1       2       3 :=
1       0       1       2
2       1       1       0
3       1       0       2
4       1       1       0
5       1       2       1
6       1       0       1;

param z :=
        [1, *, *]       2 1 1   2 2 0   2 3 1   2 4 0   2 5 0   2 6 0   3 1 0   3 2 0   3 3 0
        3 4 0   3 5 0   3 6 0   4 1 0   4 2 1   4 3 0   4 4 0   4 5 0   4 6 0   5 1 1   5 2 0
        5 3 0   5 4 0   5 5 0   5 6 0   6 1 0   6 2 0   6 3 0   6 4 0   6 5 0   6 6 1   7 1 0
        7 2 0   7 3 1   7 4 0   7 5 0   7 6 0
        [2, *, *]       2 1 0   2 2 0   2 3 0   2 4 0   2 5 0   2 6 1   3 1 0   3 2 1   3 3 1
        3 4 0   3 5 0   3 6 0   4 1 0   4 2 0   4 3 0   4 4 0   4 5 0   4 6 0   5 1 0   5 2 0
        5 3 0   5 4 0   5 5 0   5 6 0   6 1 0   6 2 0   6 3 1   6 4 0   6 5 1   6 6 0   7 1 1
        7 2 0   7 3 0   7 4 1   7 5 1   7 6 0
        [3, *, *]       2 1 0   2 2 1   2 3 0   2 4 1   2 5 0   2 6 0   3 1 0   3 2 0   3 3 0
        3 4 0   3 5 0   3 6 0   4 1 1   4 2 0   4 3 1   4 4 0   4 5 0   4 6 0   5 1 0   5 2 1
        5 3 0   5 4 1   5 5 0   5 6 1   6 1 1   6 2 1   6 3 0   6 4 1   6 5 0   6 6 0   7 1 0
        7 2 0   7 3 0   7 4 0   7 5 0   7 6 1
        [4, *, *]       2 1 0   2 2 0   2 3 0   2 4 0   2 5 0   2 6 0   3 1 1   3 2 0   3 3 0
        3 4 0   3 5 1   3 6 1   4 1 0   4 2 0   4 3 0   4 4 1   4 5 0   4 6 0   5 1 0   5 2 0
        5 3 0   5 4 0   5 5 1   5 6 0   6 1 0   6 2 0   6 3 0   6 4 0   6 5 0   6 6 0   7 1 0
        7 2 0   7 3 0   7 4 0   7 5 0   7 6 0
        [5, *, *]       2 1 0   2 2 0   2 3 0   2 4 0   2 5 1   2 6 0   3 1 0   3 2 0   3 3 0
        3 4 1   3 5 0   3 6 0   4 1 0   4 2 0   4 3 0   4 4 0   4 5 1   4 6 1   5 1 0   5 2 0
        5 3 1   5 4 0   5 5 0   5 6 0   6 1 0   6 2 0   6 3 0   6 4 0   6 5 0   6 6 0   7 1 0
        7 2 1   7 3 0   7 4 0   7 5 0   7 6 0;

param u :=
1       3
2       3
3       4
4       2
5       4
6       2;

param q :=
```

1 5
2 5
3 6;

k. Instance 9 .dat file

```
set D := 1 2 3 4;
set S := 1 2;
set T := 2 3 4 5 6;
set V := 1 2 3 4;
param g := 1;
param h := 10;

param b:
        1       2       3       4 :=
1       0       0       0       1
2       0       0       1       0
3       1       0       0       0
4       0       1       0       0;


param p:
        1       2       3       4:=
1       0       0       0       1
2       1       1       1       0
3       1       0       0       0
4       1       1       0       1;

param z :=
        [1, *, *]       2 1 0   2 2 1   2 3 0   2 4 0   3 1 0   3 2 1   3 3 1   3 4 0   4 1 1
        4 2 0   4 3 0   4 4 0   5 1 1   5 2 1   5 3 1   5 4 1   6 1 1   6 2 1   6 3 1   6 4 0
        [2, *, *]       2 1 1   2 2 0   2 3 1   2 4 1   3 1 1   3 2 0   3 3 0   3 4 1   4 1 0
        4 2 1   4 3 1   4 4 1   5 1 0   5 2 0   5 3 0   5 4 0   6 1 0   6 2 0   6 3 0   6 4 1;

param u :=
1       1
2       3
3       2
4       4;
```

```
param q :=
1       3
2       2
3       1
4       2;
```

l. Instance 10 .dat file

```
set D := 1 2;
set S := 1 2 3 4 5 6;
set T := 2 3 4 5 6 7;
set V := 1 2 3 4 5 6 7;
param g := 1;
param h := 10;
param b:
        1       2:=
1       1       0
2       0       1
3       0       1
4       1       0
5       1       0
6       0       1
7       1       0;
```

```
param p:
        1       2 :=
1       2       2
2       1       2
3       2       3
4       0       1
5       1       1
6       2       2
7       2       1;
```

```
param z :=
        [1, *, *]       2 1 0   2 2 1   2 3 0   2 4 0   2 5 0   2 6 1   2 7 0   3 1 0   3 2 0
        3 3 0   3 4 0   3 5 0   3 6 1   3 7 0   4 1 0   4 2 0   4 3 1   4 4 0   4 5 0   4 6 0
```

```
        470  510  520  530  541  551  560  570  610  620  630
        640  650  661  670  710  720  730  740  750  760  770
        [2, *, *]    210  220  230  240  250  260  270  310  320
        330  341  351  360  370  411  420  430  440  451  460
        470  511  520  531  540  550  561  570  611  621  630
        641  650  660  670  711  720  730  740  750  761  770
        [3, *, *]    211  220  230  240  250  260  270  311  321
        331  340  350  360  370  410  420  430  441  450  460
        470  510  520  530  540  550  560  570  610  620  631
        640  651  660  671  710  720  730  740  750  760  770
        [4, *, *]    210  220  230  240  251  260  270  310  320
        330  340  350  360  371  410  420  430  440  450  460
        470  510  520  530  540  550  560  570  610  620  630
        640  650  660  670  710  720  731  740  750  760  771
        [5, *, *]    210  220  231  240  250  260  271  310  320
        330  340  350  360  370  410  420  430  440  450  460
        471  510  521  530  540  550  560  571  610  620  630
        640  650  660  670  710  720  730  741  750  760  770
        [6, *, *]    210  220  230  241  250  260  270  310  320
        330  340  350  360  370  410  421  430  440  450  461
        470  510  520  530  540  550  560  570  610  620  630
        640  650  660  670  710  721  730  740  751  760  770;

param u :=
1       4
2       3
3       5
4       1
5       2
6       4
7       3;

param q :=
1       10
2       12;
```