On Demand Public Transport

SUMMARY

Table of Contents

Introduction

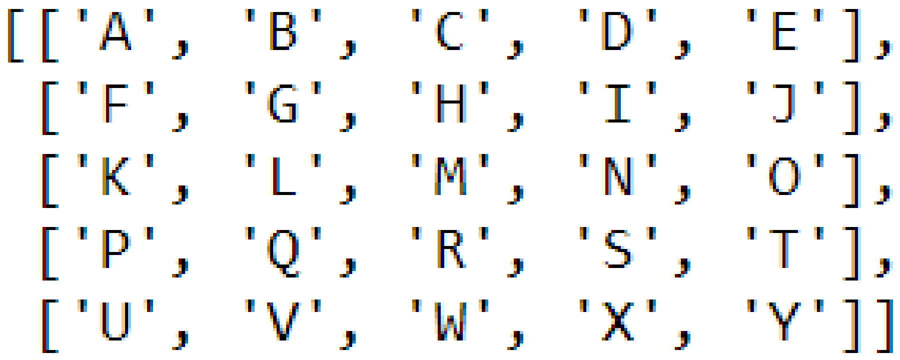
Mobility on demand ridesharing services have transformed transportation by bringing point to point transportation convenience. However, the benefits these services have to offer are limited by their capacities rendering them as taxi equivalents. Here we propose applying mobility on demand with high capacity buses. This concept of on demand public transport through dynamic routing offers several benefits over fixed route public transport such as follows:

1. Reduced waiting and travel time for the customers
2. Accommodate changing customer demand and traffic conditions
3. Reduced traffic congestion as high capacity buses will serve a greater demand

Here in this summary we will look at a dynamic routing simulation and compare its results with that of a taxi like service simulation and will observe how closely do both public transport services match.

Random Initializations

To begin with comparing the two different means of transport we first had to create a simulation of a single city's demand. For that purpose, we created our own city and mapped it onto a grid like structure of 5x5 to make the distance and time calculations simple and the shortest path route easy to visualize. Each block of a city is identified by the letters of the alphabet. The figure is shown below:



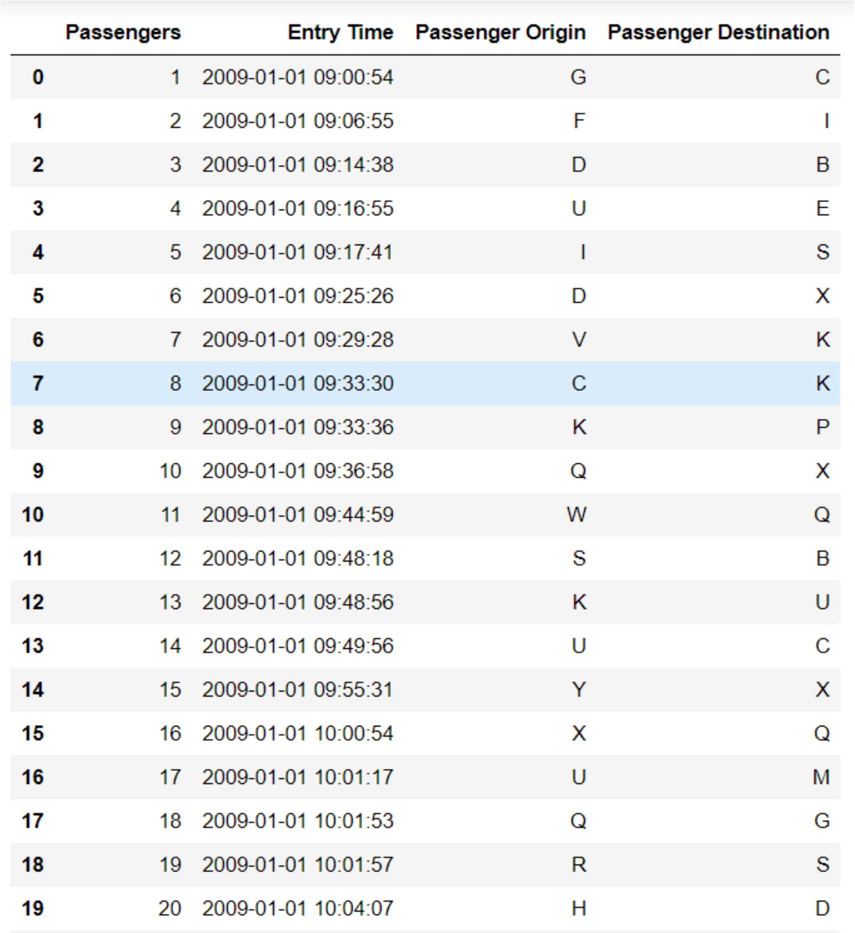
```
[['A', 'B', 'C', 'D', 'E'],
 ['F', 'G', 'H', 'I', 'J'],
 ['K', 'L', 'M', 'N', 'O'],
 ['P', 'Q', 'R', 'S', 'T'],
 ['U', 'V', 'W', 'X', 'Y']]
```

Here we can see that block A is directly connected to block B, G and K, likewise block H is connected to block B, C, D, G, I, L, M, N. Here the shortest path from block F to block to W can be more than 1 because of the structure. For e.g. it can be

- F -> L -> R -> W or
- F -> K -> Q -> W or
- F -> L -> Q -> W

The reason for this is because the vertical, horizontal and diagonal distances to the adjacent block is always the same for e.g. the distance between A to B, G, and F are all the same.

After creating a city, there had to be a demand for going from one place to the other. For this purpose, we randomly initialized a demand for a single day between 9:00 AM to 9:30 PM and made sure that no passenger's origin is his destination. A snippet of the table of the travel demand is shown below:



	Passengers	Entry Time	Passenger Origin	Passenger Destination
0	1	2009-01-01 09:00:54	G	C
1	2	2009-01-01 09:06:55	F	I
2	3	2009-01-01 09:14:38	D	B
3	4	2009-01-01 09:16:55	U	E
4	5	2009-01-01 09:17:41	I	S
5	6	2009-01-01 09:25:26	D	X
6	7	2009-01-01 09:29:28	V	K
7	8	2009-01-01 09:33:30	C	K
8	9	2009-01-01 09:33:36	K	P
9	10	2009-01-01 09:36:58	Q	X
10	11	2009-01-01 09:44:59	W	Q
11	12	2009-01-01 09:48:18	S	B
12	13	2009-01-01 09:48:56	K	U
13	14	2009-01-01 09:49:56	U	C
14	15	2009-01-01 09:55:31	Y	X
15	16	2009-01-01 10:00:54	X	Q
16	17	2009-01-01 10:01:17	U	M
17	18	2009-01-01 10:01:53	Q	G
18	19	2009-01-01 10:01:57	R	S
19	20	2009-01-01 10:04:07	H	D

In the time frame of 12 and a half hours, we created a demand for 200 people whose entry time into the system can be seen above in the table. The time that the passenger entered into the system was randomly allotted to all of them.

After that all that was left was to assign each bus to a starting location which was also done randomly but making sure at the start no two buses were from the same block.

Supply Parameters

The supply parameters refer to the different decision variables which we decided to use for our simulation. These parameters can be simply altered to more realistic settings to get a more accurate result of the simulations.

Number of Passengers	200
Fleet Size	10
Bus Capacity	10
Waiting Time Threshold	12 minutes
Travel Time Threshold	24 minutes
Uniform Speed of Bus	50 Km/hr.

Given that the uniform speed of the bus is 50 Km/hr. That means the time between two adjacent blocks is 6 minutes. This will later help in our calculations and observing the simulation results.

Later in the Dynamic Bus Router Simulation section we will discuss the entire process of the simulation and how a bus is selected for a particular bus but here I would first like to clear what these thresholds are in detail.

I. **Travel Time Threshold**

These threshold times can be basically understood as promise times. When a bus picks you up, you are given an expected alight time of when you will reach you desired destination. So, imagine when person A enters the system and the bus 4 is selected for him, he is given an initial expected alight time based on the route that bus 4 is taking for all the passengers on the bus. Now the bus 4 promises that person A will reach his destination before his initial expected alight time plus the travel time threshold. So that means if a second person B is now entering the system (He/She is not selected by Bus 4 yet), bus 4 will add his origin and destination in its optimized route and see whether the dynamic bus route change causes any of the already existing bus 4 passenger's expected alight time to be more than his promised time.

Expected Alight Time after Adding new passenger <= Expected Initial Alight Time + Travel Time Threshold

If this constraint is not being met, then Bus 4 doesn't select that passenger as it will heavily affect his current passengers expected time. This calculation is made by all buses and if for all the buses this constraint isn't met the passenger B is rejected.

II. **Waiting Time Threshold**

Like the Travel Time Threshold, the Waiting Time Threshold also works like a promise. For a e.g. a bus is selected to pick up a passenger, if the bus isn't currently where the passenger is then the passenger will have to wait for the bus to reach that block. So now let's imagine that passenger A is to be picked up by Bus 2 and on its way to picking up that passenger another passenger B enters the system (He/She is not selected by Bus 2 yet). Now passenger A already has an initial expected waiting time for when Bus 2 will pick passenger A up. Now Bus 2 will check if it picks up passenger B through dynamic route change will it affect the pickup time of passenger A. If it doesn't cause the expected pick up time of passenger A to be more than his/her expected initial time plus the waiting time threshold then Bus 2 will pickup passenger B otherwise it will reject him.

Expected Waiting Time after Adding new passenger <= Expected Initial Waiting Time + Waiting Time Threshold

This concept will further be cleared in the Dynamic Bus Routing simulator section.

Shortest Path Algorithm

For the shortest path algorithm, we use the Floyd-Warshall Algorithm to see the shortest distance between two specific blocks and also allows us to construct the shortest path between both of those blocks. This algorithm is used to find the shortest path from one place to another however, when implementing this with the multiple passengers arriving at multiple different origins who have to get off at multiple different destinations. The problem with putting all the origins and destinations together and solving this through the Travel Salesman Problem is that we need to keep a check that for all passengers we pick up we first have to reach at their origin before we go to their destination.

So, for this case we use the Nearest Neighbor approach. In this approach we add all the origins of the passengers that are yet to come on board and the destinations of all those passengers who are already on the bus. We consider these locations as places the bus has to stop at to either pick or drop off passengers, lets call these places checkpoints. Now the algorithm starts from where it currently is and sees which checkpoint is the closest to it and sets that checkpoint to the next place on its route. It then looks through the list of checkpoints and sees which is closest to the checkpoint it just set. It keeps doing this till all the places have been selected and we now have a shortest path that the bus can take. One thing to keep in mind is that when we pick up a passenger from a checkpoint, we add that passengers destination to the list of checkpoints to choose from as now that passenger is on board and we would like his destination to be in the shortest path route.

Dynamic Bus Routing Simulation

In this section we will go over the entire simulation process that we have designed in different steps.

1. A demand is realized (A passenger enters the system with an entry time stamp)
2. All the buses (Empty or with existing passengers) calculate their cost function without the addition of the new passenger.
   Cost Function Before:
   - SUM Of All Passengers [Excluding New Passenger] (Alight Time - Entry Time)
3. Remember this Expected Alight Time is calculated based on the most optimized route we could find through the Nearest Neighbor Algorithm.
4. Then for all the Buses we add the Origin and Destination of the new passenger and from our Nearest Neighbor Algorithm to find the most optimized route for all the existing passengers in the bus and the new passenger.
5. Based on this optimized route we calculate the Expected Alight Time for all the passengers including the new passenger based on our optimized route, this is done for all buses in our fleet.
6. Then we use the same cost function but this time we include the new passenger
   Cost Function After:
   - SUM Of All Passengers [Including New Passenger] (Alight Time - Entry Time)
7. After we have both Cost Function values, we check the difference between the latter Cost Function and the former Cost Function for all of the buses.

8. Then from all the Cost Differences of all the buses we check which Bus has the minimum Cost Difference and that Bus is the candidate bus to be selected for the new passenger.
9. However, here is where the Travel Time and Waiting Time Threshold comes into play. We know that every passenger when enter the system and is selected by a Bus is given an initial Expected Alight Time and initial Expected Board Time and based on the supply parameters set for Waiting and Travel Time Threshold they are also given a Promise Board Time and Promise Alight Time as shown in the equations below.
   Promise Alight Time *= Expected Initial Alight Time + Travel Time Threshold*
   Promise Board Time *= Expected Initial Board Time + Waiting Time Threshold*

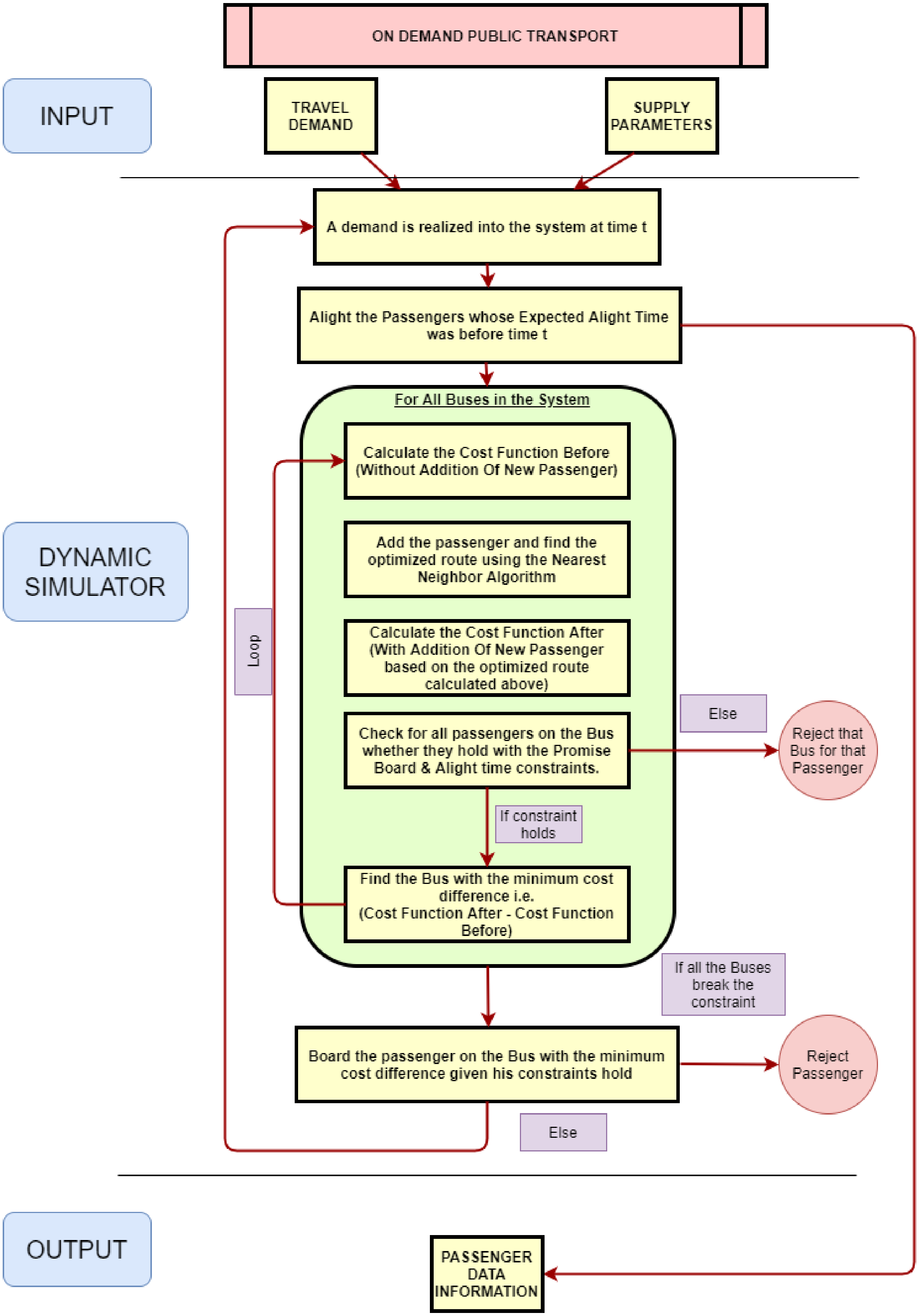
10. And we know that no passenger's Expected Alight Time should exceed his/her Promise Alight Time nor should his/her Expected Board Time surpass his Promise Board Time.
11. And we know that Expected Alight and Board Times may change due to dynamic bus routing, where a new passenger could be selected by a bus and then the route is changed to a more optimized route based on all the passengers start and stop destinations
12. So, for each Bus we check the passenger's Expected Alight Times and see that no existing passenger's Expected Alight Time becomes more than his Promise Alight Time due to dynamic route change.
13. This is also checked for the Expected Board Time with respect to Promise Board Time. And if any single existing passenger's Expected Alight Time and Expected Board Time exceed their Promise Alight Time or Promise Board Time, then that new passenger is rejected from that Bus. And that passenger is then considered for the next bus with the minimum Cost Difference
14. All Buses are checked that both these constraints are holding, otherwise if the constraints don't hold for all of the Buses the passenger is rejected from all buses.

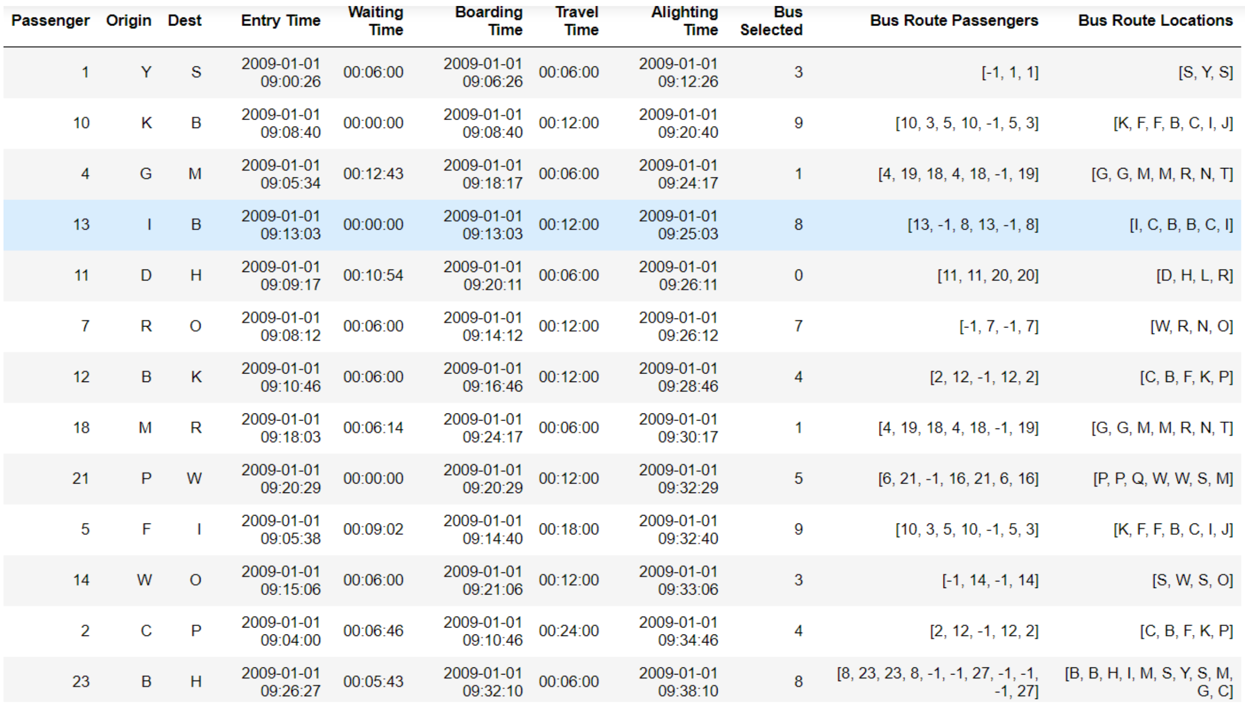
Input Dynamic Bus Routing Simulator

Travel Demand	**Supply Parameters**
1. Time of Entry into the System	1. Fleet Size
2. Origin	2. Number Of Passengers
3. Destination	3. Bus Capacity
	4. Waiting Time Threshold
	5. Travel Time Threshold
	6. Uniform Speed of Bus

Output Dynamic Bus Routing Simulator

Passenger Output Data
1. Waiting Time
2. Boarding Time on the Bus
3. Travel Time
4. Alight Time from the Bus
5. Bus Selected
6. Current Bus Route
7. Current Passengers on Bus
8. Average Bus Capacity

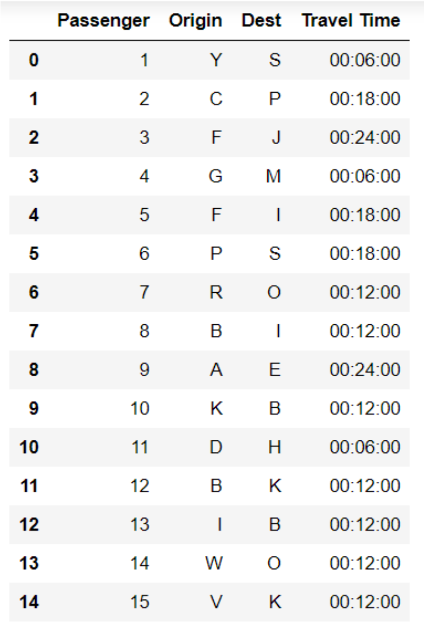

ON DEMAND PUBLIC TRANSPORT
INPUT
TRAVEL DEMAND
SUPPLY PARAMETERS
A demand is realized into the system at time t
Alight the Passengers whose Expected Alight Time was before time t
DYNAMIC SIMULATOR
For All Buses in the System
Calculate the Cost Function Before (Without Addition Of New Passenger)
Add the passenger and find the optimized route using the Nearest Neighbor Algorithm
Calculate the Cost Function After (With Addition Of New Passenger based on the optimized route calculated above)
Check for all passengers on the Bus whether they hold with the Promise Board & Alight time constraints.
Loop
Else
Reject that Bus for that Passenger
If constraint holds
Find the Bus with the minimum cost difference i.e. (Cost Function After - Cost Function Before)
If all the Buses break the constraint
Board the passenger on the Bus with the minimum cost difference given his constraints hold
Reject Passenger
Else
OUTPUT
PASSENGER DATA INFORMATION



Passenger	Origin	Dest	Entry Time	Waiting Time	Boarding Time	Travel Time	Alighting Time	Bus Selected	Bus Route Passengers	Bus Route Locations
1	Y	S	2009-01-01 09:00:26	00:06:00	2009-01-01 09:06:26	00:06:00	2009-01-01 09:12:26	3	[-1, 1, 1]	[S, Y, S]
10	K	B	2009-01-01 09:08:40	00:00:00	2009-01-01 09:08:40	00:12:00	2009-01-01 09:20:40	9	[10, 3, 5, 10, -1, 5, 3]	[K, F, F, B, C, I, J]
4	G	M	2009-01-01 09:05:34	00:12:43	2009-01-01 09:18:17	00:06:00	2009-01-01 09:24:17	1	[4, 19, 18, 4, 18, -1, 19]	[G, G, M, M, R, N, T]
13	I	B	2009-01-01 09:13:03	00:00:00	2009-01-01 09:13:03	00:12:00	2009-01-01 09:25:03	8	[13, -1, 8, 13, -1, 8]	[I, C, B, B, C, I]
11	D	H	2009-01-01 09:09:17	00:10:54	2009-01-01 09:20:11	00:06:00	2009-01-01 09:26:11	0	[11, 11, 20, 20]	[D, H, L, R]
7	R	O	2009-01-01 09:08:12	00:06:00	2009-01-01 09:14:12	00:12:00	2009-01-01 09:26:12	7	[-1, 7, -1, 7]	[W, R, N, O]
12	B	K	2009-01-01 09:10:46	00:06:00	2009-01-01 09:16:46	00:12:00	2009-01-01 09:28:46	4	[2, 12, -1, 12, 2]	[C, B, F, K, P]
18	M	R	2009-01-01 09:18:03	00:06:14	2009-01-01 09:24:17	00:06:00	2009-01-01 09:30:17	1	[4, 19, 18, 4, 18, -1, 19]	[G, G, M, M, R, N, T]
21	P	W	2009-01-01 09:20:29	00:00:00	2009-01-01 09:20:29	00:12:00	2009-01-01 09:32:29	5	[6, 21, -1, 16, 21, 6, 16]	[P, P, Q, W, W, S, M]
5	F	I	2009-01-01 09:05:38	00:09:02	2009-01-01 09:14:40	00:18:00	2009-01-01 09:32:40	9	[10, 3, 5, 10, -1, 5, 3]	[K, F, F, B, C, I, J]
14	W	O	2009-01-01 09:15:06	00:06:00	2009-01-01 09:21:06	00:12:00	2009-01-01 09:33:06	3	[-1, 14, -1, 14]	[S, W, S, O]
2	C	P	2009-01-01 09:04:00	00:06:46	2009-01-01 09:10:46	00:24:00	2009-01-01 09:34:46	4	[2, 12, -1, 12, 2]	[C, B, F, K, P]
23	B	H	2009-01-01 09:26:27	00:05:43	2009-01-01 09:32:10	00:06:00	2009-01-01 09:38:10	8	[8, 23, 23, 8, -1, -1, 27, -1, -1, -1, 27]	[B, B, H, I, M, S, Y, S, M, G, C]

Comparison to Taxi Like Services

Then for the same Travel Demand data we implement a simulation where the passenger goes directly from his origin to his destination as if he was the only passenger in the taxi. We are doing this so we can compare our dynamic bus routing results with that of a taxi like service. We assume here that the taxi travels at the same speed as the bus and that the taxi chooses the shortest route. Then we can compare how close is our travel time from the best travel time solution.



	Passenger	Origin	Dest	Travel Time
0	1	Y	S	00:06:00
1	2	C	P	00:18:00
2	3	F	J	00:24:00
3	4	G	M	00:06:00
4	5	F	I	00:18:00
5	6	P	S	00:18:00
6	7	R	O	00:12:00
7	8	B	I	00:12:00
8	9	A	E	00:24:00
9	10	K	B	00:12:00
10	11	D	H	00:06:00
11	12	B	K	00:12:00
12	13	I	B	00:12:00
13	14	W	O	00:12:00
14	15	V	K	00:12:00

Results

	Dynamic Bus Routing	Taxi Like Service
Average Travel Time	**20 Minutes**	**14 Minutes**
Minimum Travel Time	**6 Minutes**	**6 Minutes**
Maximum Travel Time	**1 Hour 6 Minutes**	**24 Minutes**
Average Waiting Time	**21 Minutes**	**-**