OPTIMIZATION OF BUS ROUTES FOR KAUNDINYA PUBLIC SCHOOL BASED ON SIMULATION METHOD IN PROMODEL

Rishav Ranjan

Global Automotive & Mfg
University of Michigan
1075 Beal Ave
Ann Arbor, MI 48109

Akash Prakash Chitra Iyer

Department of Industrial & Oper Eng
University of Michigan
1891 IOE Building 1205
Beal Ave, Ann Arbor, MI 48109

Roshan Kumar Dash

Automotive Engineering
University of Michigan
1075 Beal Ave
Ann Arbor, MI 48109

Yatharth Sanjay Kumar Bhatt

Department of Industrial & Oper Eng
University of Michigan
1891 IOE Building 1205
Beal Ave, Ann Arbor, MI 48109

ABSTRACT

The paper uses discrete event simulation along with optimization using python for a school in rural region of India. The objective of this paper is to optimize the routes used by the buses, so that it can cater to all children and minimize the fuel consumed in the process. The data was gathered directly from the school and google map, along with app script was used to create a driving distance matrix. The route was optimized using a greedy algorithm in Python. The current state and the optimized state were modeled in ProModel. Paired T-test was done on the output using Minitab. Cost calculations were done to predict a cost saving.

1. INTRODUCTION

Kaundinya Public School is situated in Jhumri Telaiya, Jharkhand India. It is a small school, which serves the children in the town. The project aims to simulate the transportation system employed by the school to ferry the children. The school currently has 300 children. Some of the children use their own means of transport. 173 are ferried by a collection of buses and vans. Figure 1 shows the routes taken by the buses and vans. The number on the locations refers to the sequential order of the stops. Each different color on the location is served by a different vehicle.

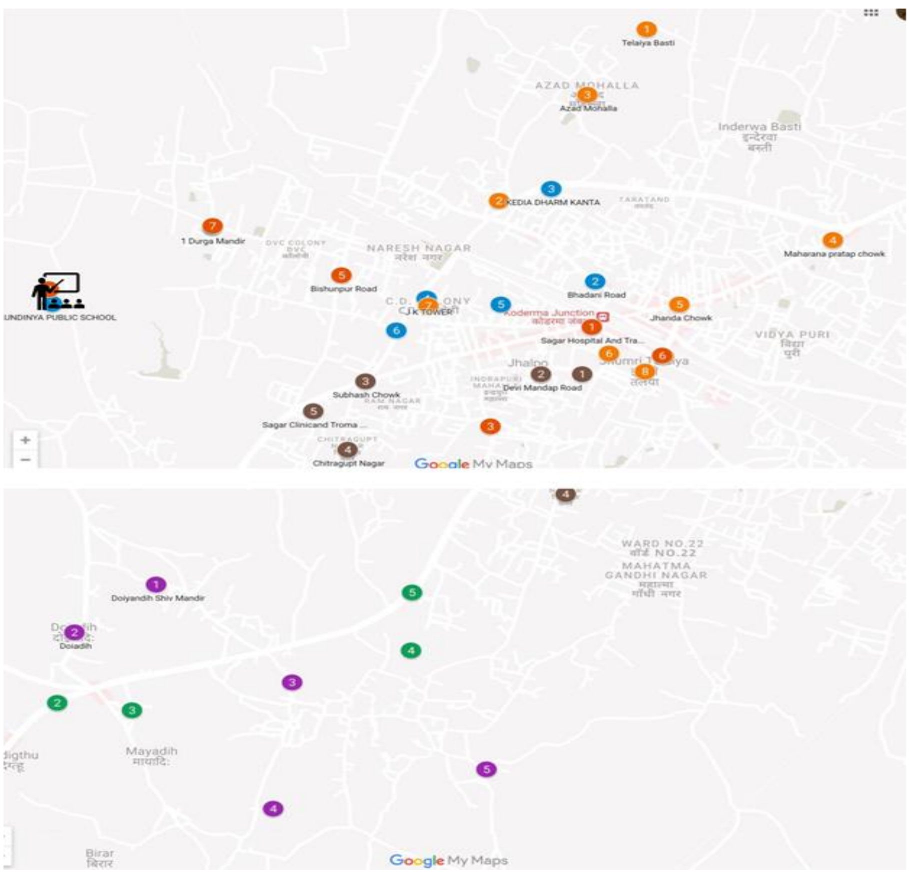

Figure 1 Route of School Buses

1.1 Literature Review

We identified 2 research papers having worked on similar challenges previously. Mazzulla, Gabriella. (2009). An activity-based system of models for student mobility simulation. European Transport Research Review. 1. 163-174. 10.1007/s12544-009-0017-2, this paper provides a system of models for student mobility simulation on a university campus. This paper provides a system of models for student mobility simulation on a university campus. The system of models is sequential, in which each choice of dimension is linked to the previous one and involves the choices made at a lower level, and it is made up of multinomial and hierarchical Logit models. The analyst's aim is to establish the relative effect of each attribute on the overall utility associated with each option by the individuals. In the paper titled "An activity-based system of models for student mobility simulation" by Mazzulla (2009), the author presents a system of models for student mobility simulation on a university campus. The paper aims to provide a tool for transport system planning and adapts an activity-based approach. The system of models is sequential and made up of multinomial and hierarchical Logit models, in which each choice of dimension is linked to the previous one and involves the choices made at a lower level.

1.2 Problem Description and Importance

The transportation of students forms a major part of the costs for the school. The school is looking at optimizing the transportation system so that it can serve the maximum number of children at the lowest possible cost. At present, there are 6 vehicles which are operating on 7 different routes. One of the vans takes two trips. The aim of this project is to analyze the system and help in making the following decisions:

- Optimal routes that should be taken by the buses and vans
- Find out the cost of fuel used by the buses and vans and possible savings after optimization

The model has been created with data obtained from the school. The initial simulations have been verified using the actual data from the school. To answer the first question a model to optimize the routes for each vehicle were developed. The optimized routes were modeled again into the software and fuel usage data was used to determine a possible cost reduction.

1.4 Questions to be solved

Some of the key questions that we want to solve are -

- What are the optimal routes that should be taken by the buses and vans to ferry the required number of children at the lowest possible cost?
- How much fuel is currently being used by the buses and vans, and what potential savings can be achieved through route optimization?
- How can the model be developed to optimize the routes for each bus, taking into consideration factors such as vehicle capacity, time constraints, and student boarding and deboarding processes?
- How can the model be verified and validated using actual data from the school, and what steps can be taken to ensure its accuracy and reliability?
- What are the potential benefits and cost savings that can be achieved through the implementation of the optimized transportation system, and how can these be measured and evaluated?
- How can the findings and recommendations from the simulation project be effectively communicated to school management, stakeholders, and other relevant parties for informed decision-making and implementation?

2. METHODOLOGY

2.1 Data Collection

The data collection process was designed to support the simulation project aimed at optimizing the transportation routes for the Kaundinya school buses. Identifying the best route options for different buses, and potentially reducing travel time, congestion, and gas use are the general goals. The following data has been collected from the school:

- Operation time of the buses
- Distance to each stop
- Time that is taken to reach those stops
- Time that is taken to board and de-board the buses
- Fuel consumption of the buses
- Distribution of the stops on the map
- Number and capacity of buses and vans

The data can be found in the appendix.

2.2 Time Study

The members of the project team worked with the school staff and local drivers to get real-time data. This made the data more accurate and reliable. The drivers were requested to track their travel time from one location to another using a GPS device and report the data back to the school staff. This approach helped minimize the potential for recall bias and increased the granularity of the data. Recall bias, also known as memory bias, refers to the systematic error that occurs when participants in a study inaccurately remember and report past events or experiences. In the context of the data collection process for this project, recall bias could have potentially affected the accuracy of the travel time data if drivers were asked to provide this information based on their memory rather than tracking it in real-time.

There are a total of 44 bus stops, which makes a total of 1892 permutations (44P2) of distances which need to be plotted for the distance matrix. Getting all these distances are essential to formulate a multiple-traveling salesman optimization problem. Google app script was used along with the coordinates of all the bus stops to get the driving distance for all the permutations. The distance matrix is attached in the appendix, along with the code.

In order to account for potential variations in travel times due to factors such as traffic conditions and weather, the data collection process was spread over multiple days and at different times of the day. The team members assumed that the variations in travel times would follow a normal distribution, allowing for the generation of more accurate and robust simulations.

The gathered data was then cleaned, formatted into standardized units, and the used to calculate the following:

Bus	Total Distance (KM)	Total Time (HR)	Speed = Distance/Time (Km/hr)
JH 2288	11.35	0.4993	22.73182455
JH 05CN 8105	24.05	1.0493	22.92004193
JH 0953 Eco Round (1)	34.6	1.2327	28.06846759
JH 0953 Eco Round (2)	20.45	0.8677	23.56805347
BROAV 4446	16.05	0.698	22.99426934
JH 12J 1892	12.4	0.8307	14.92716986

The above data can be visualized using the following graph:

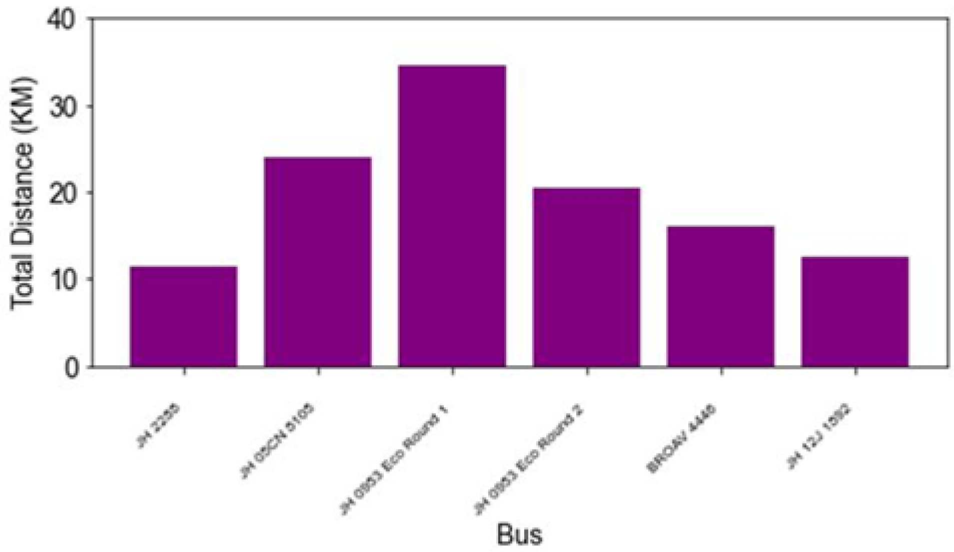

Figure 2 Distance traveled by each bus

The first bar plot represents the total distance (in kilometers) traveled by each vehicle

A subset of the obtained distance matrix can be seen in the figure below:

Bus stop Code	AAA	AM	BD	BHR	BIR	BR	CC
AAA		4.791	6.736	3.416	2.412	4.466	2.97
AM	4.782		16.491	2.538	2.371	1.815	2.092
BD	7.651	9.985		8.59	8.442	9.66	8.144
BHR	3.415	2.546	15.106		1.004	2.221	0.592
BIR	2.412	2.38	14.959	1.005		2.055	0.559
BR	4.466	1.823	16.175	2.222	2.055		1.776
CC	2.969	2.1	14.66	0.592	0.558	1.775	
CH	9.043	14.172	5.029	12.777	12.629	13.847	12.331
CN	3.869	3.001	14.476	1.606	1.458	2.675	1.16
DM	2.074	3.341	7.858	1.966	0.962	3.016	1.52
DM1	2.074	3.341	7.858	1.966	0.962	3.016	1.52
DO	3.78	6.114	4.479	4.72	4.572	5.789	4.274
DR	3.839	2.97	15.114	0.893	1.428	2.063	1.016
DS	3.162	5.496	5.337	4.102	3.954	5.171	3.656

Figure 3 Subset of Distance Matrix

3. MODEL

3.1 Model Description

The models simulate the pick-up and drop off of children. The setting used are as follows:

- Entity:
  There are seven buses, each with a maximum capacity of 30. They all travel on different routes. In order to simplify the model for the purpose of the project, entity is kept as "van_xxxx". For example vav_1892 represents the first bus.

- Locations:
  All the bus stops associated with the buses are the locations. There are a total of 44 bus stops. The return location is the parking lot of the school.

- Resource:
  A resource named "bus" is used solely for the purpose of assigning shifts to each of the buses.

- Process:
  The buses leave the school premises at 7am in the morning. They go to various bus stops and come back to school. Then they leave again at 6pm in the evening, visit the bus stops and come back to the school. There are a total of 44 bus stops. To simplify the models, the time travelled for some of the bus stops are combined into one stop.

- Shift assignment:
  The shift starts at 7 am in the morning and finishes once all the buses are back. Then the shifts start again at 6pm in the evening and finishes when all the buses are back to the school.

- Variable:
  Variables called "fuel_usageX" where x is the bus number is used to calculate the total fuel used by the buses.

Two Promodel models were made. The first model represents the current state of the system. The second model represents the proposed optimized system (the optimization is discussed in the next section). The total fuel usage was calculated for both the models.

3.2 Model Assumption

There are a few assumptions that have been made before creating the model.

- It is assumed that there are no breakdowns in the vehicles, and they are always available when required.
- Drivers are always available to drive vehicles.
- There are no accidents.
- The students arrive at the bus stops on time and if they don't the buses leave as vehicles are not allowed to wait for students. All the vehicles are capable of town speeds and do not have any maximum limit below it.
- There are no extraordinary traffic jams which delay the vehicles.
- The effect of seasons and weather is not being accounted for.
- The students getting in and out of vehicle takes a much smaller time when compared to the time spent in travel, hence it is ignored because it would make the calculation complicated. Each of the 44 stops have different number of students getting in and out. Moreover, it will have no significant impact on the distance travel and the fuel consumption.
- There are no human errors in boarding and deboarding of the children
- The vehicles do not deviate from the designated routes.

3.2 Input Modeling

Although the distance traveled data is used for optimizing the route, the promodel model was made using the time taken between stops. Data for time taken between the bus stops was recorded over a month and the stat fit software was used to fit a curve.

3.3 Model Optimization

Problem Description
The given problem is a classic example of a Vehicle Routing Problem with Capacity Constraints (CVRP). In this problem, a fleet of vehicles is to be routed to pick up children from various bus stops and transport them to a common destination (e.g., school). Each vehicle has a fixed seating capacity, and the objective is to minimize the total distance traveled by all vehicles. The problem can be represented as a graph, where each node corresponds to a bus stop and each edge represents the distance between two nodes.

Code Overview
The provided code is a Python script that uses the OR-Tools library, a popular optimization library developed by Google, to solve the CVRP. The script is organized as follows:

- Import necessary libraries: numpy, pandas, and OR-Tools related modules.
- Define helper functions print_solution, read_distance_matrix_from_csv, and the main function main.
- Read the distance matrix and bus stop labels from a CSV file using the read_distance_matrix_from_csv function.
- Initialize the routing model and set the number of vehicles, vehicle capacity, and demands at each bus stop.
- Define callback functions for distance and demand computation.
- Add a capacity dimension to the routing model to handle the capacity constraints.
- Set the search parameters and solve the problem using the routing model.
- Print the solution.

Solver and Search Strategy
The script uses the pywrapcp.RoutingModel class from the OR-Tools library as the solver for the CVRP. The search strategy employed in this script is the PATH_CHEAPEST_ARC, which is a greedy strategy that constructs a route by successively adding the arc with the least cost. This strategy is specified using the routing_enums_pb2.FirstSolutionStrategy.PATH_CHEAPEST_ARC enumeration.

Explanation of Important Details

1. The distance matrix and bus stop labels are read from a CSV file using the pandas library. The distance matrix represents the pairwise distances between bus stops, and the labels mapping is a dictionary that maps indices to bus stop names.
2. The pywrapcp.RoutingIndexManager is used to manage indices in the routing model, and the pywrapcp.RoutingModel is initialized with the number of nodes, vehicles, and the depot index (which is set to 0).
3. The vehicle_capacity variable is set to 30, which represents the fixed capacity of each vehicle.
4. The demands variable is a list that contains the number of children to be picked up at each bus stop.
5. The demand_callback and distance_callback functions are defined to compute the demand and distance between nodes, respectively. These functions are registered as callbacks in the routing model using the routing.RegisterTransitCallback and routing.RegisterUnaryTransitCallback methods.

6. The capacity dimension is added to the routing model using the routing.AddDimensionWithVehicleCapacity method. This ensures that the capacity constraints are satisfied during the optimization process.
7. The default routing search parameters are used with the PATH_CHEAPEST_ARC strategy. The problem is solved using the routing.SolveWithParameters method, which returns an assignment object containing the optimal solution.
8. The print_solution function is used to display the solution, including the routes for each vehicle, the total distance traveled, and the number of children picked up.

Finally, this script provides an efficient and easy-to-understand solution to the Vehicle Routing Problem with Capacity Constraints using the OR-Tools library. The script reads the distance matrix and demands from a CSV file, initializes the routing model, and solves the problem using a greedy search strategy. The solution from this model is used as input for the optimized system's ProModel.

Code Output:

Vehicle Route	Optimal Routes	Distance of the route	Children Picked up
0	AAA (0 children) -> MM (2 children) -> GCM (15 children) -> MG (2 children) -> MA (10 children) -> AAA	14 km	29
1	AAA (0 children) -> SA (3 children) -> BD (6 children) -> GH (9 children) -> CH (8 children) -> RT (2 children) -> AAA	22 km	28
2	AAA (0 children) -> JK (3 children) -> BHR (7 children) -> GC (3 children) -> AM (1 children) -> TB (1 children) -> MP (3 children) -> JC (6 children) -> BR (4 children) -> KDM (1 children) -> AAA	11 km	29
3	AAA (0 children) -> NB (1 children) -> SC (4 children) -> GD (8 children) -> CN (4 children) -> ST (3 children) -> GR (3 children) -> PL (2 children) -> GB (3 children) -> AAA	14 km	28
4	AAA (0 children) -> FM (2 children) -> DS (7 children) -> DO (4 children) -> SH (1 children) -> VK (2 children) -> PT (4 children) -> PB (3 children) -> SM (6 children) -> GS (1 children) -> AAA	10 km	30
5	AAA (0 children) -> DM (1 children) -> DM1 (4 children) -> BIR (3 children) -> KD (4 children) -> GG (1 children) -> DR (5 children) -> VP (1 children) -> VKP (1 children) -> CC (5 children) -> AAA	5 km	25
	TOTAL	76 km	169

OUTPUT VISUALIZATION

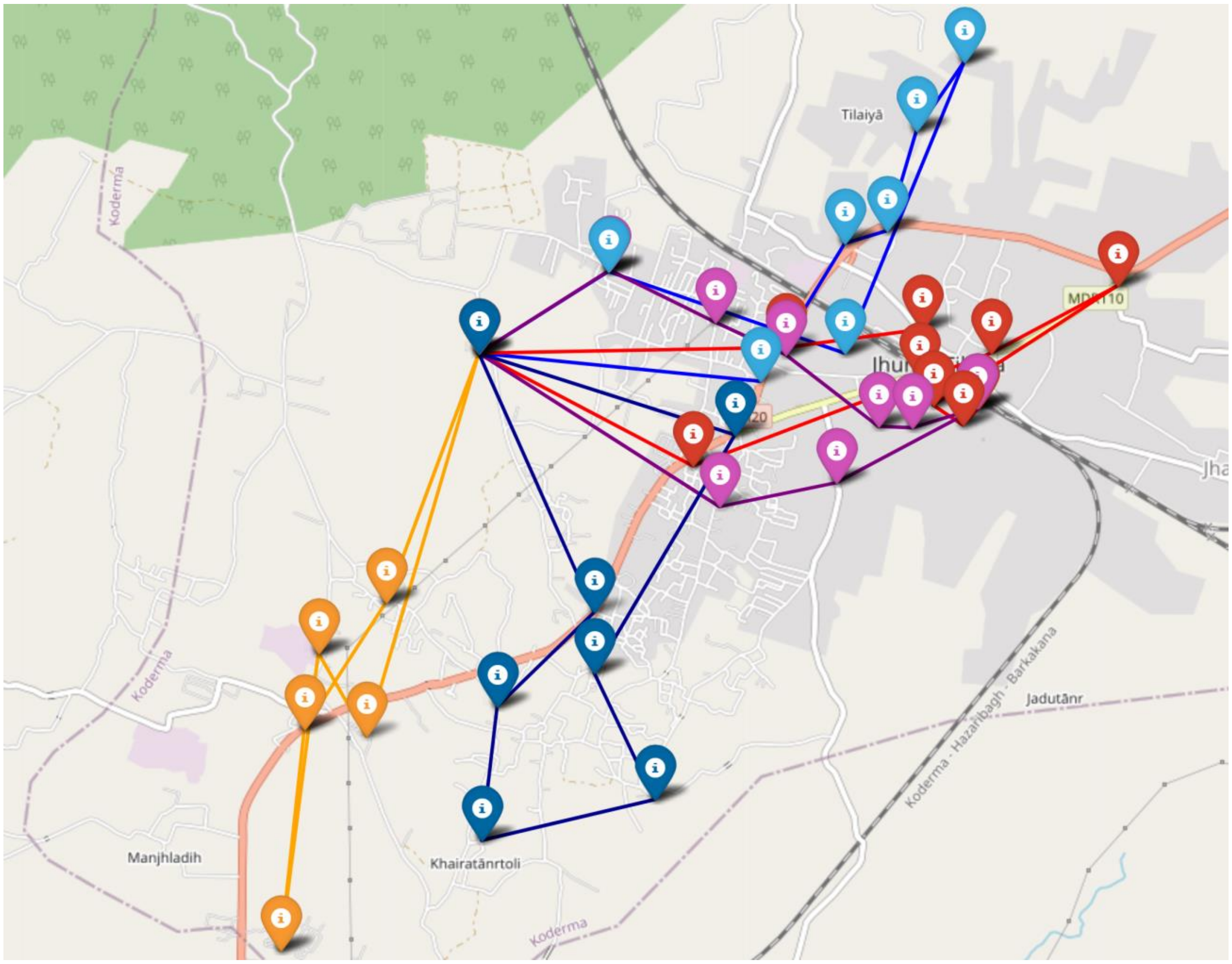

Figure 4: Optimal Route Visualization in Python

Output Explanation

The given output represents the optimal solution for the Vehicle Routing Problem with Capacity Constraints, as computed by the provided script. The output displays the routes for each vehicle, the distance traveled on each route, and the number of children picked up.

Route for vehicle 0:

The vehicle starts at AAA (the school) and picks up children from MM, GCM, MG, and MA bus stops before returning to AAA. The total distance traveled on this route is 14 km, and 29 children are picked up.

Route for vehicle 1:

The vehicle begins at AAA and picks up children from SA, BD, GH, CH, and RT bus stops, then returns to AAA. The route covers a distance of 22 km, and 28 children are picked up.

Route for vehicle 2:

Starting at AAA, this vehicle visits JK, BHR, GC, AM, TB, MP, JC, BR, and KDM bus stops before returning to AAA. The total distance for this route is 11 km, and 29 children are picked up.

Route for vehicle 3:

This vehicle starts at AAA and picks up children from NB, SC, GD, CN, ST, GR, PL, and GB bus stops, then returns to AAA. The distance traveled on this route is 14 km, and 28 children are picked up.

Route for vehicle 4:
The vehicle begins its route at AAA and visits FM, DS, DO, SH, VK, PT, PB, SM, and GS bus stops before returning to AAA. The total distance of this route is 10 km, and 30 children are picked up.

Route for vehicle 5:
Starting at AAA, the vehicle picks up children from DM, DM1, BIR, KD, GG, DR, VP, VKP, and CC bus stops, then returns to AAA. The distance covered on this route is 5 km, and 25 children are picked up.

The output also prints a summary of the total distance traveled by all vehicles, which is 76 km, and the total number of children picked up, which is 169. This optimal solution ensures that all children are picked up from their designated bus stops while minimizing the total distance traveled by the vehicles, adhering to their capacity constraints.

3. VALIDATION AND VERIFICATION

3.1 Model Verifications

Two team members, who were not involved in creating the model were used for verification. They went through all locations, entities etc. and verified the data. Then they went over the processing and confirmed the code. Moreover, additional simulations were performed by changing various data used in the model to ensure that all the data is being used by the model.

3.2 Model Validation

The model is an accurate representation of the school and all the stops. Google map was used to plot the stops. The screen shot of the map was scaled in promodel. To validate the model, the timings of the current state of the model were matched with the actual data obtained from the school. The co-driver in school buses were asked to record the arrival times for each of the stops and for all the school buses. Further, the total fuel used by each of the buses in a week was recorded. The data which was obtained was very close to the mean value of fuel usage obtained by the model.
The optimized model uses one less school bus. Since, changing the bus timings and routes is a long process for the school. It was not possible to permanently change the routes within the time allotted for the project. As a result, in order to verify the result, a dry run of the buses was made. In this, the buses were made to take the optimized routes. In the end, the total distance traveled by the buses was calculated and validated with the model. The result was within 2 standard deviations of the model, hence validating the result.

3.3 Data Validation

The distance traveled between each of the stops in the current system was directly taken from the school. The odometer of the buses was used to measure the distance between the stops. However, in order to optimize the route, we need a distance matrix which has distance between all the stops. A 44 x 44 distance matrix will lead to 1936 combinations. Getting so many distances manually was difficult for the purpose of the project. Using coordinates to get distance would be in accurate, since it would not account for the actual driving distance. To solve this issue, a google app script was used. The coordinates of the bus stop was fed into the app script and driving distance using google maps was obtained. 50 of the distances given by the google maps was validated by physically traveling between those locations.

4. RESULTS

Prior to running the simulation, we established a "fuel usage" variable for each bus. For example, "fuel usage1" indicates the fuel consumption of van-1 during one trip. By computing the average of each fuel usage variable, we can obtain the "Average fuel consumption" of the corresponding van for a single trip.

Following the simulation, which compared Baseline Vs Optimized for a year, we collected data for each fuel usage variable as presented below. Our simulation model aims to minimize fuel consumption, and we employed this methodology to quantify the "optimized route" model.

Variable Summary						
Name	Total Changes	Average Time Per Change (Min)	Minimum Value	Maximum Value	Current Value	Average Value
fuel usage1	458.00	1,146.02	0.00	2.57	1.67	1.80
fuel usage2	458.00	1,146.01	0.00	2.31	1.16	1.50
fuel usage3	457.00	1,147.11	0.00	4.20	2.90	3.12
fuel usage4	457.00	1,147.15	0.00	5.32	3.60	3.57
fuel usage5	457.00	1,147.13	0.00	3.73	3.26	2.46
fuel usage6	457.00	1,148.50	0.00	3.21	1.90	2.08
fuel usage7	457.00	1,148.66	0.00	6.46	4.31	4.77

Figure 5: Baseline Fuel Usage

Variable Summary						
Name	Total Changes	Average Time Per Change (Min)	Minimum Value	Maximum Value	Current Value	Average Value
fuel usage1	396.00	1,323.77	0.00	2.69	2.18	2.16
fuel usage2	396.00	1,323.82	0.00	3.76	3.03	3.22
fuel usage3	396.00	1,325.50	0.00	3.33	3.04	2.79
fuel usage4	396.00	1,325.51	0.00	3.35	2.55	2.77
fuel usage5	396.00	1,325.48	0.00	2.67	2.01	2.08
fuel usage6	395.00	1,327.07	0.00	2.11	1.39	1.50

Figure 6: Optimized Fuel Usage

5. OUTPUT ANALYSIS

The model was replicated 10 and the value of total fuel consumption record.

Replication	Total Fuel Consumption	
	Current Route	Optimized Route
1	14.52	19.3
2	14.49	19.3
3	14.46	19.32
4	14.53	19.37
5	14.55	19.38
6	14.54	19.31
7	14.5	19.34
8	14.53	19.35
9	14.53	19.33
10	14.49	19.34

Paired T-test was conducted on the result to establish statistical significance.

Descriptive Statistics

Sample	N	Mean	StDev	SE Mean
Current Route	10	14.5140	0.0280	0.0088
Optimized Route	10	19.3340	0.0276	0.0087

Test

Null hypothesis H_0: μ_difference = 0
Alternative hypothesis H_1: μ_difference ≠ 0

T-Value	P-Value
-511.24	0.000

A P-value of 0.00 rejects the null hypothesis and establishes a statistically significant difference in the mean of fuel consumption.

From the optimized fuel usage, we get total of (2.16 + 3.22 + 2.79 + 2.77 + 2.08 + 1.50) = 14.52 Liters.
Other than the baseline which used total of (1.80 + 1.50 + 3.12 + 3.57 + 2.46 + 2.08 + 4.77) = 19.3 Liters

Fuel saved per trip = Baseline – Optimized
= 19.3 – 14.52
=4.78 Liter
Fuel saved per day = 4.78 * 2 = 9.56 Liter
Fuel saved in 1year = 9.56 * 300 = 2868 Liters
Cost of Petrol = $1.22 / Liter
Total cost saved = Cost of 1 van + Cost of petrol saved in 1 year
= $20,000 + $ (1.22*2868)
= $20,000 + $ 3497

=$23,497

By adopting the optimized model, we can potentially save around $3500 per year on gas expenses.

5. CONCLUSION

In this paper, we have attempted to optimize the route for school buses. We have created two models and compared their result. The optimization led to cost savings for the school. However, the optimization model is non-convex, and it might be giving the local optimum. The team was limited by the computing power, and it is possible that a better combination exists. A model can be created which accounts for changes in number of children in the school. Despite the limitations of optimization of non-convex problems, the project will be able to save a significant amount of money for the school, while satisfying all other constraints.

APPENDIX

Python Code

```
import numpy as np
import pandas as pd
```

```
from ortools.constraint_solver import routing_enums_pb2
from ortools.constraint_solver import pywrapcp

def print_solution(manager, routing, assignment, demands, labels_mapping):
    total_distance = 0
    total_children_picked = 0
    for vehicle_id in range(manager.GetNumberOfVehicles()):
        index = routing.Start(vehicle_id)
        plan_output = 'Route for vehicle {}:\n'.format(vehicle_id)
        route_distance = 0
        route_children = 0
        while not routing.IsEnd(index):
            children = demands[manager.IndexToNode(index)]
            plan_output += ' {} ({} children) ->'.format(labels_mapping[manager.IndexToNode(index)],
children)
            route_children += children
            previous_index = index
            index = assignment.Value(routing.NextVar(index))
            route_distance += routing.GetArcCostForVehicle(previous_index, index, vehicle_id)
        plan_output += ' {}\n'.format(labels_mapping[manager.IndexToNode(index)])
        plan_output += 'Distance of the route: {} km\n'.format(route_distance)
        plan_output += 'Children picked up: {}\n'.format(route_children)
        print(plan_output)
        total_distance += route_distance
        total_children_picked += route_children
    print('Total Distance of all routes: {} km'.format(total_distance))
    print('Total children picked up: {}'.format(total_children_picked))

def read_distance_matrix_from_csv(file_path):
    df = pd.read_csv(file_path, index_col=0)
    labels_mapping = {idx: label for idx, label in enumerate(df.index)}
    return df.values.astype(int), labels_mapping

def main():
    # Read distance matrix and labels mapping from the CSV file
    distance_matrix, labels_mapping =
read_distance_matrix_from_csv(r"C:\Users\akash\Downloads\Distance_AAA.csv")

    manager = pywrapcp.RoutingIndexManager(len(distance_matrix), 6, 0)
    routing = pywrapcp.RoutingModel(manager)

    # Fixed capacity for each vehicle
    vehicle_capacity = 30

    # Replace random demand with the exact number of kids to be picked at the bus stops
```

```
    demands = [0, 1, 6, 7, 3, 4, 5, 8, 4, 1, 4, 4, 5, 7, 2, 3, 3, 15, 8, 1, 9, 3, 1, 6, 3, 4, 1, 10, 2, 2, 3, 1, 3, 2, 4, 2,
3, 4, 1, 6, 3, 1, 2, 1, 1, 1]

    def demand_callback(from_index):
        from_node = manager.IndexToNode(from_index)
        return demands[from_node]

    def distance_callback(from_index, to_index):
        from_node = manager.IndexToNode(from_index)
        to_node = manager.IndexToNode(to_index)
        return distance_matrix[from_node][to_node]

    transit_callback_index = routing.RegisterTransitCallback(distance_callback)
    routing.SetArcCostEvaluatorOfAllVehicles(transit_callback_index)

    demand_callback_index = routing.RegisterUnaryTransitCallback(demand_callback)
    routing.AddDimensionWithVehicleCapacity(
        demand_callback_index,
        0,  # null capacity slack
        [vehicle_capacity] * 6,  # capacities of all vehicles
        True,  # start cumul to zero
        'Capacity'
    )

    search_parameters = pywrapcp.DefaultRoutingSearchParameters()
    search_parameters.first_solution_strategy = (
        routing_enums_pb2.FirstSolutionStrategy.PATH_CHEAPEST_ARC)

    assignment = routing.SolveWithParameters(search_parameters)
    if assignment:
        print_solution(manager, routing, assignment, demands, labels_mapping)

if __name__ == '__main__':
    main()
```

DISTANCE MATRIX

GS	JC	JK	KD	KDM	MA	MG	MM	MP	NB	PB	PL	PT	RT	SA	SC	SH	SM	ST	TB	VK	VKP	VP
3.833	4.713	2.953	3.562	4.209	6.746	4.784	4.66	6.104	2.985	4.454	9.054	4.375	4.71	4.947	3.491	3.953	3.996	3.719	6.012	4.357	4.36	4.204
2.955	2.317	1.829	2.502	1.557	5.867	3.906	3.781	2.698	2.106	3.196	8.176	3.497	8.54	8.777	2.505	3.075	3.118	2.841	2.605	2.671	2.671	3.326
8.607	9.487	8.147	7.593	9.403	11.062	9.1	8.976	11.299	7.891	9.228	13.37	9.149	8.244	1.426	7.522	8.727	8.77	8.035	11.206	9.131	9.134	8.978
1.007	1.517	0.708	1.117	1.964	4.482	2.52	2.396	2.395	0.721	1.258	6.791	1.337	7.173	7.41	1.12	0.757	0.958	1.456	3.767	1.161	1.164	1.008
1.422	2.302	0.542	0.936	1.798	4.335	2.373	2.249	3.693	0.574	2.043	6.643	1.964	6.17	6.407	0.973	1.542	1.585	1.308	3.601	1.946	1.949	1.793
2.639	0.501	1.513	2.186	0.835	5.551	3.59	3.465	1.392	1.79	1.38	7.86	1.664	8.224	8.461	2.189	1.262	1.768	2.525	3.044	0.855	0.855	1.214
1.009	1.889	0.262	0.671	1.518	4.036	2.074	1.95	3.414	0.275	1.63	6.345	1.551	6.727	6.964	0.674	1.129	1.172	1.01	3.321	1.533	1.536	1.38
12.794	13.674	12.334	11.78	13.59	15.249	13.287	13.163	15.486	12.078	13.415	17.557	13.336	10.399	5.062	11.709	12.914	12.957	12.222	15.393	13.318	13.321	13.165
1.623	2.503	1.163	0.609	2.418	3.852	1.89	1.766	4.314	0.907	2.244	6.16	2.165	7.627	7.864	0.538	1.743	1.786	0.825	4.221	2.147	2.15	1.994
2.383	3.263	1.503	1.823	2.759	5.296	3.334	3.21	4.654	1.469	3.004	7.604	2.925	5.832	6.069	1.934	2.503	2.546	2.269	4.562	2.907	2.91	2.754
2.383	3.263	1.503	1.823	2.759	5.296	3.334	3.21	4.654	1.469	3.004	7.604	2.925	5.832	6.069	1.934	2.503	2.546	2.269	4.562	2.907	2.91	2.754
4.736	5.616	4.276	3.722	5.532	7.191	5.229	5.105	7.428	4.021	5.357	9.499	5.278	5.746	4.815	3.651	4.856	4.899	4.164	7.335	5.26	5.263	5.107
0.646	1.561	1.132	1.199	2.388	4.49	2.528	2.404	2.439	1.497	0.832	6.798	0.829	7.597	7.834	1.076	0.801	0.449	1.463	4.191	1.205	1.208	0.694
4.118	4.998	3.658	3.104	4.914	6.573	4.611	4.487	6.81	3.402	4.739	8.881	4.66	5.588	3.846	3.033	4.238	4.281	3.546	6.717	4.642	4.645	4.489
5.612	6.492	5.152	4.598	6.408	8.067	6.105	5.981	8.304	4.896	6.233	10.375	6.154	4.094	2.351	4.527	5.732	5.775	5.04	8.211	6.136	6.139	5.983
2.897	4.469	3.129	2.575	4.384	1.876	2	1.746	6.28	2.873	4.21	4.184	4.131	9.593	12.835	2.504	3.709	3.752	2.791	6.187	4.113	4.116	3.96
2.28	1.355	1.154	1.827	0.854	5.192	3.231	3.106	2.778	1.431	2.234	7.501	2.822	7.865	8.102	1.83	2.4	2.443	2.166	2.685	1.709	1.709	2.068
2.883	3.763	2.423	1.869	3.678	2.076	0.48	0.176	5.574	2.167	3.504	4.384	3.425	8.887	13.036	1.798	3.003	3.046	2.085	5.481	3.407	3.41	3.254
1.108	2.927	1.587	1.033	2.842	3.443	1.481	1.356	4.738	1.331	2.668	5.751	2.393	8.051	8.288	0.962	2.167	2.014	1.249	4.645	2.571	2.574	2.418
0.293	2.057	1.457	0.903	2.712	4.194	2.232	2.108	2.935	1.201	1.459	6.502	1.456	7.921	8.158	0.78	1.297	1.077	1.167	4.516	1.701	1.704	1.321
11.237	12.117	10.777	10.223	12.033	13.692	11.73	11.606	13.929	10.521	11.858	16	11.779	8.842	3.505	10.152	11.357	11.4	10.665	13.836	11.761	11.764	11.608
11.205	12.085	10.744	10.191	12	0.543	11.698	11.574	13.896	10.489	12.706	2.851	11.747	13.884	11.502	10.12	11.325	11.368	10.633	13.803	11.729	11.732	11.576
	1.886	1.142	1.208	2.398	4.499	2.537	2.413	2.764	1.506	1.287	6.807	1.284	7.607	7.844	1.086	1.126	0.905	1.472	4.201	1.53	1.533	1.149
1.886		2.016	2.088	1.338	5.379	3.417	3.293	0.89	2.386	0.878	7.687	1.162	8.487	8.724	1.965	0.76	1.266	2.352	3.547	0.353	0.353	0.712
1.126	2.006		0.674	1.256	4.038	2.077	1.952	3.152	0.277	1.747	6.347	1.668	6.711	6.948	0.677	1.246	1.289	1.012	3.059	1.65	1.653	1.497
1.208	2.088	0.674		1.929	3.485	1.523	1.399	3.825	0.418	1.829	5.793	1.75	7.185	7.422	0.123	1.328	1.371	0.458	3.732	1.732	1.735	1.579
2.111	1.473	0.985	1.659		5.024	3.062	2.938	3.018	1.263	2.732	7.332	2.653	7.696	7.933	1.662	2.231	2.274	1.997	2.925	1.827	1.827	2.482
10.663	11.543	10.202	9.649	11.458		11.156	11.032	13.354	9.947	12.163	2.308	11.205	13.342	10.96	9.578	10.783	10.826	10.091	13.261	11.187	11.19	11.034
3.005	3.885	2.545	1.991	3.8	1.962		0.298	5.696	2.289	3.626	4.27	3.547	15.304	12.922	1.92	3.125	3.168	2.207	5.603	3.529	3.532	3.376
2.707	3.587	2.247	1.693	3.502	2.266	0.304		5.398	1.991	3.328	4.574	3.249	8.711	13.226	1.622	2.827	2.87	1.909	5.305	3.231	3.234	3.078
2.764	0.89	2.718	3.392	1.733	6.757	4.795	4.671		2.996	1.756	9.065	2.04	9.429	9.666	3.395	1.638	2.144	3.73	4.413	1.231	1.231	1.59
1.139	2.019	0.277	0.418	1.533	3.783	1.821	1.697	3.429		1.76	6.091	1.681	6.742	6.979	0.421	1.259	1.302	0.757	3.336	1.663	1.666	1.51
1.287	0.977	1.763	1.829	2.566	5.12	3.158	3.034	1.855	2.127		7.428	0.278	8.228	8.465	1.707	0.501	0.382	2.093	4.524	0.621	0.624	0.32
7.249	11.583	10.791	10.238	12.047	3.735	11.745	11.621	12.461	10.536	10.6		10.878	13.931	11.549	10.167	11.107	10.982	10.68	13.85	11.227	11.23	10.92
1.284	1.261	1.683	1.75	2.85	5.041	3.079	2.955	2.139	2.048	0.278	7.349		8.148	8.385	1.627	0.785	0.379	2.014	4.742	0.905	0.908	0.497
9.706	10.586	6.711	8.692	7.967	10.504	8.542	8.418	9.862	8.99	10.327	12.812	10.248		6.303	8.621	9.826	9.869	7.477	9.77	10.23	10.233	10.077
8.944	9.824	8.484	7.93	9.74	11.399	9.437	9.313	11.636	8.228	9.565	13.707	9.486	6.303		7.859	9.064	9.107	8.372	11.543	9.468	9.471	9.315
1.086	1.965	0.677	0.123	1.932	3.414	1.452	1.328	3.828	0.421	1.706	5.722	1.627	7.141	7.378		1.205	1.248	0.387	3.736	1.609	1.612	1.456
1.126	0.76	1.261	1.328	2.349	4.619	2.657	2.533	1.638	1.626	0.501	6.927	0.785	7.726	7.963	1.205		0.973	1.592	4.32	0.404	0.407	0.251
0.904	1.365	1.303	1.37	2.559	4.661	2.699	2.575	2.243	1.668	0.382	6.969	0.379	7.768	8.005	1.247	0.972		1.634	4.362	1.009	1.012	0.244
2.569	3.449	2.109	1.555	3.364	3.129	1.167	1.043	5.26	1.853	3.19	5.437	3.111	8.573	14.088	1.484	2.689	2.732		5.167	3.093	3.096	2.94
5.814	3.625	4.688	5.362	3.703	8.727	6.765	6.641	2.736	4.966	4.491	11.035	4.775	11.399	11.636	5.365	4.373	5.977	5.7		3.966	3.966	4.325
1.529	0.353	1.665	1.732	1.942	5.023	3.061	2.937	1.231	2.03	0.522	7.331	0.806	8.13	8.367	1.609	0.404	0.91	1.996	3.9		0	0.355
1.533	0.353	1.669	1.735	1.942	5.026	3.064	2.94	1.231	2.033	0.526	7.334	0.81	8.134	8.371	1.613	0.408	0.914	1.999	3.9	0		0.359
1.397	0.965	1.532	1.599	2.554	4.89	2.928	2.804	1.843	1.897	0.706	7.198	0.468	7.997	8.234	1.476	0.271	0.243	1.863	4.591	0.609	0.612	

Bus stop Code	AAA	AM	BD	BHR	BIR	BR	CC	CH	CN	DM	DM1	DO	DR	DS	FM	GB	GC	GCM	GD	GG	GH	GR	GS
AAA		4.791	6.736	3.416	2.412	4.466	2.97	9.043	3.908	2.074	2.074	4.39	3.824	4.233	2.738	5.444	4.107	4.906	4.353	4.271	7.486	6.555	3.833
AM	4.782		16.491	2.538	2.371	1.815	2.092	12.873	2.97	3.332	3.332	12.669	2.946	13.621	6.568	4.566	1.063	4.028	3.415	3.285	11.316	5.677	2.955
BD	7.651	9.985		8.59	8.442	9.66	8.144	4.185	7.939	8.924	8.924	4.479	8.598	5.698	4.292	9.76	9.301	9.222	8.384	8.302	2.628	10.871	8.607
BHR	3.415	2.546	15.106		1.004	2.221	0.592	11.506	1.585	1.965	1.965	11.284	0.786	12.236	5.201	3.181	1.862	2.642	2.03	1.178	9.949	4.292	1.007
BIR	2.412	2.38	14.959	1.005		2.055	0.559	10.503	1.438	0.962	0.962	11.137	1.413	12.089	4.198	3.033	1.696	2.495	1.883	1.593	8.946	4.144	1.422
BR	4.466	1.823	16.175	2.222	2.055		1.776	12.557	2.654	3.016	3.016	12.353	2.063	13.305	6.252	4.25	0.853	3.712	3.099	2.969	11	5.361	2.639
CC	2.969	2.1	14.66	0.592	0.558	1.775		11.06	1.139	1.519	1.519	10.838	1	11.79	4.755	2.735	1.416	2.196	1.584	1.18	9.503	3.846	1.009
CH	9.043	14.172	5.029	12.777	12.629	13.847	12.331		12.126	13.111	13.111	8.666	12.785	9.885	6.447	13.947	13.488	13.409	12.571	12.489	1.557	15.058	12.794
CN	3.869	3.001	14.476	1.606	1.458	2.675	1.16	18.663	0	2.419	2.419	10.654	1.614	11.606	5.655	2.55	2.316	2.012	0.494	1.318	17.106	3.661	1.623
DM	2.074	3.341	7.858	1.966	0.962	3.016	1.52	10.165	2.399		0	5.512	2.374	5.355	3.86	3.994	2.657	3.456	2.844	2.554	8.608	5.105	2.383
DM1	2.074	3.341	7.858	1.966	0.962	3.016	1.52	10.165	2.399	0		5.512	2.374	5.355	3.86	3.994	2.657	3.456	2.844	2.554	8.608	5.105	2.383
DO	3.78	6.114	4.479	4.72	4.572	5.789	4.274	8.666	4.068	5.053	5.053		4.727	0.858	1.652	5.889	5.43	5.351	4.513	4.431	7.109	7	4.736
DR	3.839	2.97	15.114	0.893	1.428	2.063	1.016	19.301	1.593	2.389	2.389	11.292	0	12.244	5.625	3.188	2.286	2.65	1.755	0.818	17.744	4.299	0.646
DS	3.162	5.496	5.337	4.102	3.954	5.171	3.656	7.942	3.45	4.435	4.435	0.858	4.109		1.494	5.271	4.812	4.733	3.895	3.813	6.385	6.382	4.118
FM	2.738	6.99	4.14	5.596	5.448	6.665	5.15	6.447	4.944	3.86	3.86	1.652	5.603	1.494		6.765	6.306	6.227	5.389	5.307	4.89	7.876	5.612
GB	5.835	4.967	12.499	3.572	3.424	4.641	3.126	16.686	2.036	4.385	4.385	8.677	3.58	9.629	11.123		4.282	1.922	2.051	3.284	15.129	1.685	2.897
GC	4.107	1.464	15.816	1.863	1.696	0.853	1.417	12.198	2.295	2.657	2.657	11.994	2.271	12.946	5.893	3.891		3.353	2.74	2.61	10.641	5.002	2.28
GCM	5.129	4.261	12.7	2.866	2.718	3.935	2.42	16.887	1.33	3.679	3.679	8.878	2.874	9.83	11.324	0.774	3.576		1.344	2.578	15.33	1.885	2.883
GD	4.293	3.425	14.066	2.03	1.882	3.099	1.584	18.253	0.494	2.843	2.843	10.244	1.755	11.196	6.079	2.141	2.74	1.603		1.401	16.696	3.252	1.108
GG	4.163	3.295	14.818	1.178	1.752	2.969	1.197	19.005	1.297	2.713	2.713	10.996	0.818	11.948	5.949	2.892	2.61	2.354	1.401		17.448	4.003	0.293
GH	7.486	12.615	3.472	11.22	11.072	12.29	10.774	1.557	10.569	11.554	11.554	7.109	11.228	8.328	4.89	12.39	11.931	11.852	11.014	10.932		13.501	11.237
GR	10.248	12.582	11.166	11.188	11.04	12.257	10.742	15.353	10.537	11.521	11.521	7.344	11.196	8.296	9.79	3.012	11.898	11.82	10.982	10.9	13.796		11.205
GS	3.849	2.98	15.123	1.007	1.438	2.655	1.026	19.31	1.602	2.399	2.399	11.301	0.646	12.253	5.635	3.197	2.296	2.659	1.108	0.293	17.753	4.308	
JC	4.729	2.326	16.003	1.517	2.318	0.502	1.906	20.19	2.482	3.279	3.279	12.181	1.561	13.133	6.515	4.077	1.356	3.539	2.927	2.057	18.633	5.188	1.886
JK	2.953	1.838	14.662	0.709	0.542	1.513	0.263	11.044	1.142	1.503	1.503	10.84	1.117	11.792	4.739	2.737	1.154	2.199	1.587	1.457	9.487	3.848	1.126
KD	3.427	2.512	14.109	1.117	0.969	2.186	0.671	18.296	0.588	1.823	1.823	10.287	1.199	11.239	5.213	2.183	1.827	1.645	1.033	0.903	16.739	3.294	1.208
KDM	3.938	1.287	15.648	1.694	1.527	0.971	1.248	12.029	2.127	2.488	2.488	11.826	2.102	12.778	5.724	3.722	0.612	3.184	2.572	2.442	10.472	4.833	2.111
MA	9.706	12.04	10.624	10.646	10.498	11.715	10.2	14.811	9.995	10.979	10.979	6.802	10.654	7.754	9.248	5.368	11.356	11.278	10.44	10.358	13.254	5.096	10.663
MG	5.251	4.383	12.586	2.988	2.84	4.057	2.542	16.773	1.452	3.801	3.801	8.764	2.996	9.716	11.21	0.66	3.698	0.122	1.466	2.7	15.216	1.771	3.005
MM	4.953	4.085	12.89	2.69	2.542	3.759	2.244	17.077	1.154	3.503	3.503	9.068	2.698	10.02	11.514	0.964	3.4	0.176	1.168	2.402	15.52	2.075	2.707
MP	5.671	3.019	17.38	3.427	3.26	1.392	2.981	13.762	3.86	4.221	4.221	13.558	2.439	14.51	7.457	5.455	2.344	4.917	4.305	2.935	12.205	6.566	2.764
NB	2.984	2.115	14.407	0.721	0.573	1.79	0.275	11.075	0.886	1.469	1.469	10.585	1.13	11.537	4.77	2.482	1.431	1.943	1.331	1.201	9.518	3.593	1.139
PB	4.47	3.303	15.744	1.258	2.059	1.479	1.647	19.931	2.223	3.02	3.02	11.922	0.832	12.874	6.256	3.818	2.333	3.28	2.668	1.459	18.374	4.929	1.287
PL	10.295	12.629	11.213	11.235	11.087	12.304	10.789	15.4	10.584	11.568	11.568	7.391	11.243	8.343	9.837	3.805	11.945	11.867	11.029	10.947	13.843	3.533	7.249
PT	4.39	3.521	15.665	1.542	1.979	1.763	1.567	19.852	2.144	2.94	2.94	11.843	0.829	12.795	6.176	3.739	2.617	3.201	2.393	1.456	18.295	4.85	1.284
RT	4.71	8.549	8.092	7.174	6.17	8.224	6.728	10.399	9.038	5.832	5.832	5.746	9.697	5.588	4.094	9.202	7.865	8.664	9.483	9.401	8.842	10.313	9.706
SA	4.947	10.322	1.427	8.927	8.779	9.997	8.481	5.062	8.276	9.261	9.261	4.817	8.935	3.846	2.351	10.097	9.638	9.559	8.721	8.639	3.505	11.208	8.944
SC	3.383	2.515	14.038	1.052	0.972	2.189	0.674	18.225	0.517	1.933	1.933	10.216	1.076	11.168	5.169	2.112	1.83	1.574	0.962	0.78	16.668	3.223	1.086
SH	3.968	3.099	15.243	0.757	1.557	1.262	1.145	19.43	1.722	2.518	2.518	11.421	0.801	12.373	5.754	3.317	2.116	2.779	2.167	1.297	17.873	4.428	1.126
SM	4.01	3.141	15.285	1.064	1.599	1.867	1.187	19.472	1.764	2.56	2.56	11.463	0.449	12.415	5.796	3.359	2.457	2.821	2.013	1.076	17.915	4.47	0.904
ST	4.815	3.947	13.752	2.552	2.404	3.621	2.106	17.939	1.016	3.365	3.365	9.93	2.56	10.882	6.601	1.827	3.262	1.289	1.11	2.264	16.382	2.938	2.569
TB	7.641	4.989	19.351	5.397	5.23	4.432	4.951	15.732	5.83	6.191	6.191	15.529	5.805	16.481	9.427	7.425	4.314	6.887	6.275	6.145	14.175	8.536	5.814
VK	4.372	2.679	15.647	1.16	1.961	0.855	1.549	19.834	2.126	2.922	2.922	11.825	1.205	12.777	6.158	3.721	1.709	3.183	2.571	1.7	18.277	4.832	1.529
VKP	4.376	2.679	15.65	1.164	1.965	0.855	1.553	19.837	2.129	2.926	2.926	11.828	1.209	12.78	6.162	3.724	1.709	3.186	2.574	1.704	18.28	4.835	1.533
VP	4.239	3.37	15.514	1.028	1.828	1.467	1.416	19.701	1.993	2.789	2.789	11.692	1.072	12.644	6.025	3.588	2.321	3.05	2.438	1.568	18.144	4.699	1.397

GOOGLE APP SCRIPT CODE

```
function getSeconds(fromAdd, toAdd) {
  var directions= Maps.newDirectionFinder()
```

```
  .setOrigin(fromAdd)
  .setDestination(toAdd)
  .getDirections();
  //DEBUG.PRINT
  //Logger.log(directions.routes[0].legs[0].duration.value);
  //return value
  return directions.routes[0].legs[0].duration.value;
}

function getKm(fromAdd, toAdd) {
  var directions= Maps.newDirectionFinder()
  .setOrigin(fromAdd)
  .setDestination(toAdd)
  .getDirections();
  //DEBUG.PRINT
  //Logger.log(directions.routes[0].legs[0].duration.value);
  //return value
  return directions.routes[0].legs[0].distance.value/1000;
}
```

CHILDREN DATA

Coordinates	Stop Name	Code	Children
24.4477778, 85.!	Azad Mohalla	AM	6
24.4296857, 85.4	Bhondo	BD	4
24.4354273, 85.!	Bhadodih Road	BHR	3
24.4371314, 85.!	Bishunpur Road	BIR	6
24.4367836, 85.!	Bhadani Road	BR	7
24.4353409, 85.!	C.D. Colony	CC	8
24.4623656, 85.4	Chorahi	CH	1
24.4268854, 85.!	Chitragupt Nagar	CN	4
24.4400253, 85.!	Durga Mandir	DM	7
24.4399722, 85.!	1 Durga Mandir	DM1	2
24.4187505, 85.4	Doiadih	DO	4
24.4313311, 85.!	Devi Mandap Road	DR	4
24.4215703, 85.4	Doiyandih Shiv Mand	DS	5
24.4329743, 85.4	Flour Mill	FM	2
24.4176915, 85.!	Gumo (Barawadih)	GB	1
24.4415408, 85.!	Geeta Clinic	GC	15
24.420691, 85.5(	Gumo choudhary mo	GCM	3
24.4237872, 85.!	Gumo Damovir	GD	1
24.4274768, 85.!	Gas Godow Road	GG	3
24.447315, 85.4!	Ghutitanr	GH	8
24.4141875, 85.4	Gumo (Rajput Muhal	GR	9
24.4282408, 85.!	Gandhi School Road	GS	3
24.4354263, 85.!	Jhanda Chowk	JC	3
24.4357862, 85.!	J K TOWER	JK	6
24.431459, 85.5:	Koriyadih	KD	10
24.4422127, 85.!	KEDIA DHARM KANT.	KDM	1
24.4363439, 85.!	Kaundinya Public sch	KPS	4
24.41459, 85.49(	Mahto Ahra	MA	2
24.4210716, 85.!	Main Road Gomoh	MG	3
24.422188, 85.5:	Muslim Muhalla Gun	MM	1
24.4392053, 85.!	Maharana pratap cho	MP	2
24.4339052, 85.!	Nawada Basti	NB	3
24.4314764, 85.!	Phutani chowk, bish	PB	4
24.4023383, 85.4	Police Line	PL	2
24.4293897, 85.!	Pani Tanki Road	PT	2
24.4640625, 85.4	Rohaniya Tand	RT	3
24.4299936, 85.4	Sardarudih	SA	6
24.4309125, 85.!	Subhash Chowk	SC	1
24.4341165, 85.!	Sagar Hospital And T	SH	1
24.4313125, 85.!	Shitla Mata Mandir	SM	3
24.4291533, 85.!	Sagar Clinicand Trom	ST	4
24.451643, 85.5:	Telaiya Basti	TB	2
24.4324062, 85.!	Veer kumar singh ch	VK	1
24.4323781, 85.!	Veer Kuwar Singh Ch	VKP	2
24.4325387, 85.!	Vaishali Press Lane	VP	1