

Dynamic BEST Optimization

Submitted in partial fulfilment of the requirements
Of the degree of

Bachelor of Engineering

by

Jaineel Nandu (60001150029)
Jyotsna Shenoy (60001150051)

Project Guide:
Prof. Mayur Parulekar



Electronics Engineering
Dwarkadas J. Sanghvi College of Engineering
Mumbai University
2018-2019

Certificate

This is to certify that the project entitled **Dynamic BEST Optimization** is a bonafide work of **Jaineel Nandu (60001150029)** and **Jyotsna Shenoy (60001150051)** submitted to the University of Mumbai in partial fulfilment of the requirement for the award of the degree of “**Bachelor of Engineering**” in Electronics Engineering.

Internal Guide

Internal Examiner

Prof. Mayur Parulekar

Head of Department

External Examiner

Prof. Prasad Joshi

Principal

Dr. Hari Vasudevan

Project Report Approval for B. E.

This project report entitled **Dynamic BEST Optimization** by **Jaineel Nandu (60001150029)** and **Jyotsna Shenoy (60001150051)** is approved for the degree of **Electronics Engineering**.

Examiners

1.-----

2.-----

Date:

Place:

Declaration

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

(Signature)

(Name of student and SAP no.)

Date:

Abstract

In a densely populated city like Mumbai where urban transportation systems are always hanging by a thread, the mass transportation system can only be relied on, if the service is reliable, timely and caters to all class of people. The railways & buses hence form the backbone of the city. The core objective is to identify the current deployment of buses from depots based on routes to cover the entire city, the total carrying capacity of buses, frequency of buses & overlapping routes for buses. This would be used to design a dynamical system model which would be able to estimate the dynamic deployment of buses from depots, identify & estimate passenger traffic in all clusters & create an inter-network for route hopping on buses to reduce passenger waiting time at bus- stops as well as to provide a balanced capacity load sharing on profitable or maximum utilization routes.

The sole objective is to provide the users with a robust & highly inter-connectable, independent & reliable mass transportation system which ensures minimum waiting time for passengers, profitability for the service provider & reduced network congestion in the city which helps us work towards a smart & planned city even with all the current constraints.

Contents			
CHAPTER NO.	SECTION NO.	TOPIC	PAGE NO.
		Certificate	ii
		Project Report Approval for B. E.	iii
		Declaration	iv
		Abstract	v
1.	1.1	Introduction	1-4
2.	2.1	Review of Literature	5-10
3.	3.1	System Model/Architecture	11
	3.2	Network flow optimization architecture	13
	3.3	Genetic Algorithm	16
	3.3	Technology stack used	23
4.	4.1	Proposed System Implementation	27
	4.2	Individual Module Components	29
	4.3	Matrix models of input parameters	29
	4.4	Real time parameters provided by BEST	31
	4.5	Data Analysis, Extraction, Parsing, Processing	34
	4.5	Complete Program Flow	42
5	5.1	Results and Discussions	45
	5.2	Data Extraction and Evaluation	47
	5.2	Time based module evaluation using Genetic algorithm	50
6	6.1	Conclusions and scope	55
	6.2	Conclusion	57
	6.2	Scope	58
7		REFERENCES & BIBLIOGRAPHY	59-62
8		Acknowledgments	63

Chapter 1

Introduction

1.1 Introduction

With an ever-increasing road traffic and with projects like metro rail or sea links taking a more than estimated time it is necessary to remodel the existing infrastructure that we have or redesign our mass transportation systems. We look at the current scenario of BEST such as the frequency of buses, load carrying capacity of buses, frequency and fixed time of departure of buses, peak routes and scheduling algorithms to analyse the interconnectivity and design evolutionary strategies that would dynamically adapt to the stochastically distributed network.

The current system employed by the BEST though it encompasses a complete coverage of the city it has still witnessed a shortfall in passengers over time. The reasons that could be attributed to this follows:

- (a) Erratic timings of buses –Though bus timings from the depots are fixed the passengers at the node bus-stops between source and destination have no estimated time of arrival of the bus. This creates the problem of positive feedback i.e. more passengers take the rickshaw or cabs thus leading to more traffic on the roads and in turn creates more congestion.
- (b) Mismanagement of bus frequency routes due to fixed timing –This problem is created as multiple buses ply on the same route from different depots, now considering that there would be traffic jams and other factors there would be integrated problems that these buses would reach the same nodes at close intervals which would lead to the problem of one full capacity bus while the consecutive one could go empty on the same route and also delayed arrivals. This also leads to mismanagement of passengers and they are left stranded for long time intervals.
- (c) Interconnectivity – Since no interconnectivity or bus hopping options are available due to ticketing restrictions and no connectivity between junctions with no ETA estimation people stay put in buses irrespective if they are caught in jams however if the bus routes have interconnectivity at junctions there could bus hopping if route connectivity and fare chart is displayed.
- (d) Bus stop locations –It is observed that buses that are plying on the same route will have bus stops in close proximity due to this a person who can catch one

of these buses waits on one of these stops but if a bus of different stop comes over he has to run towards the proximity stop, in this case if an indicator indicates the ETA of the bus he can as well wait at that particular stop thus leading to load distribution of passengers among the particular buses that they are looking for.

- (e) Maximum capacity limitation of passengers by applying load distribution – It has been observed that there are peak timings such as the office hours when more buses need to be deployed so that capacity sharing is done but how are these buses to be scheduled such that there is minimum waiting time for passengers is the purpose of our design. This leads us to design the network flow optimization architecture which tackles the dynamic deployment of buses at various junction nodes such that traffic jam routes are bypassed temporarily and passengers ahead of traffic nodes are picked up without latency.
- (f) Bus Location and Bus Allocation from Depot– It is necessary to know the real time position of buses and hence the need to design a signalling system at the depot which would do a real time tracking of the buses and thus provide dynamic allocation of the buses from the different depots along the routes based on number of people waiting at a particular stop.

The possible solution for this is to retrieve the real time data from the BEST, collect information of the distances and node positions of the BEST bus stops and encode it through an algorithm which will provide a possible outcome of all the existing parameters as well as some optimum data which can be presented to the BEST to make the required changes. While the nodes/bus stops have to be kept fixed/static throughout, the BEST can accommodate to remodelling of the existing system. The possible changes that can be proposed are:

1. Addition of buses: Deploying more buses to high frequency nodes and also removal of idle buses in the existing system.
2. Route revamp: Restructuring the routes to deploy buses where there is more demand and directing buses away from lower frequency nodes.
3. Improved scheduling: Deploying more buses at peak hours (current scenario being that BEST buses are being deployed at fixed time sections thereby following a static timetable) taking into consideration a time section parameter at the input.

Chapter 2

Review of Literature

2.1 Review of Literature

The Brihanmumbai Electricity Supply and Transport (BEST) undertaking is the civic transport and electricity provider public body based in Mumbai, Maharashtra, India. It was originally set up in 1873 as a tramway company called “Bombay Tramway Company Limited”. The company set up a captive thermal power station at the Wadi bundar in November 1905 to generate electricity for its trams and positioned it to also supply electricity to the city and re-branded itself to “Bombay Electric Supply and Tramways (BEST) Company. In 1926, BEST became an operator of motor buses. In 1947, the BEST became an undertaking of the Municipal Corporation and rebranded itself to “Bombay Electric Supply and Transport (BEST)”. It now operates as an autonomous body under the Municipal Corporation. BEST operates one of India’s largest fleet of buses. The bus transport service covers the entire city and also extends its operations outside city limits into neighbouring urban areas.

BEST uses CNG and conventional diesel buses. BEST buses have a fleet which comprises of single-decker diesel buses, CNG buses, Midi buses and double decker buses. All are tagged with a route number and its corresponding destination. In 2005, BEST decided to hire buses from private operators instead of procuring new buses to cut costs.

BEST bus routes are spread citywide and to neighbouring cities. BEST operates inter-city services to three areas beyond the municipal limits of Mumbai City; into the limits of the bordering corporations of Navi-Mumbai, Thane and Mira-Bhayandar. BEST supplements suburban rails, which is the mass carrier in the Mumbai regions. It is for this reason that BEST always gives priority for feeder routes over other routes. The majority of BEST buses consist of ordinary routes. Limited bus services which skips minor stops is used on long routes, high-capacity routes, and routes that provides connectivity beyond Mumbai city.

As of 2015, the BEST runs a total of 3,600 buses, ferrying 5 million passengers over 443 routes, and has a workforce strength of 38,000, which includes 22,000 bus drivers and conductors.

Type of bus services and routes: ^[7]

Ordinary	Ordinary routes are the most common, with buses on these routes stopping at all stops. Buses travelling on these routes are identified by a white route number on a black background.
Limited	Buses on these routes stop only at major stops and skip all the minor stops in between on high volume routes. They used to have a marginally higher fare than ordinary buses and are identified by the route number in red on a white background. In 2008, the fares of Limited and Ordinary buses were made equal. The route number ends with LTD.
Special	These buses travel on select routes covering railway termini and the central business districts. These routes have a fare marginally higher than the 'Limited' routes and are identified by the route number in white on a red background.
Express/Corridor	These buses service long-distance intra-city routes and have fares that are the same as the Special routes, but with a fewer number of stops. They have route numbers indicated in red on a yellow background. These buses usually skip the flyovers.

The routes operated by BEST can be broadly classified in the following categories.

- Feeder Routes: These routes feed the railway stations either from the residential complexes or business districts.
- East-west connectors: These are the routes, which run east/west, where railways have no role to play and connect the western suburb with the eastern suburb.
- Trunk routes: These routes run south-north through the city and are almost parallel to the railways.
- AC express routes: These route runs on western and eastern express highways, to provide faster services to the commuters.
- AC standard routes: These are air-conditioned routes across the city.

The aforementioned routes are further classified into types depending on the number of stops the bus takes and type of bus used.

BEST is composed of a transport department and an electricity department. With respect to survey of its financial years, the transport department has incurred heavy losses while the electricity department has availed satisfactory profits giving the company a particular net loss every year. Newer management techniques, such as retrenching of excess staff and closure of less patronized routes, have reduced losses in recent year.

Though BEST is a government-owned company, it does not receive any financial assistance from the BMC, or the state government. BEST also earns revenues by way of advertisements on its buses and bus-stops. The BEST, being a public utility company, cannot increase electricity and bus fares to maximize its profits. An increase, when affected, is usually carried out to curb losses due to inflation. BMC approval is needed before such an increase goes into effect.

BEST has been a quintessential part of life in Mumbai. Among its future plans is the "digitization project", wherein all underground cables, sub-stations, street lights and bus-stops would be tracked digitally through the geographical information system. It also plans to connect all its electricity meters through a network, so that the readings can be taken remotely, and in real-time, thus obviating the need for monthly manual door-to-door inspection.

Chapter 3

System Model/Architecture

3.1 Network flow optimization architecture

3.1.1 Erratic Bus Timings – Consider the network model as shown in Fig 2.1 which would be considered as a base network to illustrate the problem of network congestion and the solution to erratic bus timings. The roadmap proposed does not involve any change or overhaul in the existing infrastructure however would only suggest the remodelling of the existing distribution and allocation of buses on different routes. Consider A, E, F and D as junction nodes while B and C are depot nodes that is bus allocation can be done from B and C. Based on GPS data we would be able to find the route congestion in the network as well as if the bus if GPS enabled the current locations on the grid can be seen on the signalling system at the depot.

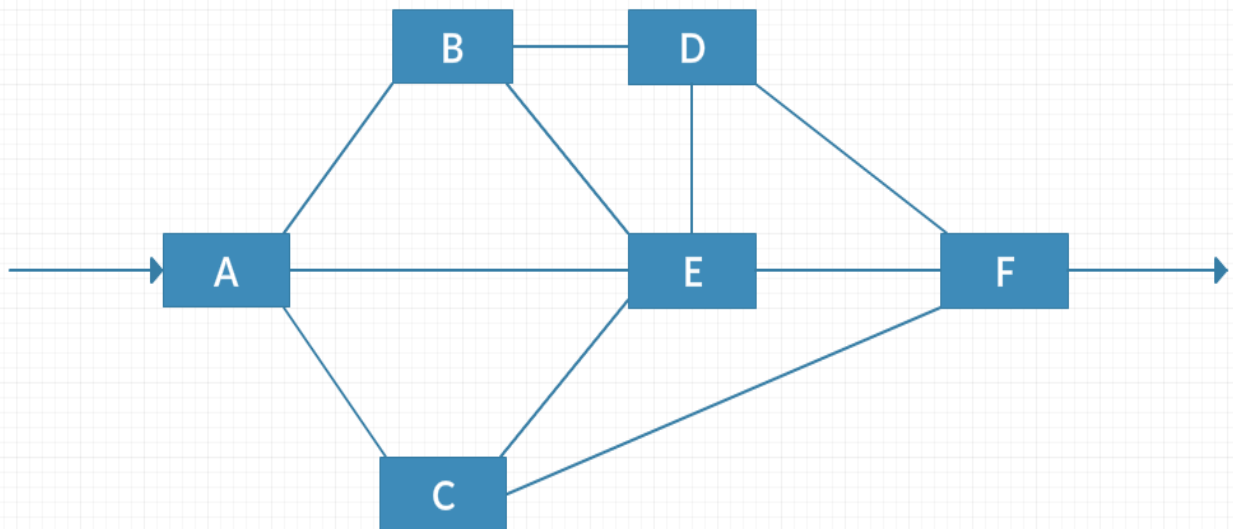


Fig. 3.1: Network Model for Congestion Control

Now let us consider a scenario where the segment A to E has become congested due to traffic and there are passengers stranded between points A to E to F. Bus no A1 is now stuck between junction A and E however there is information on the GPS platform that the network after E is completely decongested.

Under the system of static/allocated fixed time departure of the buses the buses entering from A to E would continually face this network congestion problem but if dynamic allocation is enabled then depot nodes B and C would be authorized to deploy a bus from either B to E or from C to E immediately rather than at the fixed time. This has effectively led to bypassing of the congested route and

picking up of passengers after point E thereby reducing the latency of the system and increasing the efficiency of the system. The dynamic bus allocation would also depend on constraints such as maximum number of trip times a driver is supposed to do or what is the current status of the frequency of that bus which means the time of arrival of the driver after he has completed his prior trip. This ensures that no bus is currently adding to the traffic from A to E and at the same time ensuring that passengers after the affected spot are not delayed. The passengers in the affected segment will be carried by the buses which are currently on that route itself in the traffic jam.

3.1.2 Mismanagement of bus frequency and routes due to fixed timing – It has been observed that due to fixed timings from depots there is rampant mismanagement of buses. Many buses of the same number are often deployed from the depots thus leading to unnecessary and avoidable losses for the BEST. This is very frequently. This not only exhausts the number of times the bus can frequented but because of that after these buses are deployed the next buses did not arrive for the next half hour. It was also observed that buses from different depot scheduled were in such a way that they were reaching the junction nodes and intermittent nodes either at the same time or in close interval. This leads us to design a system to synchronize scheduled timings between the depot nodes so that arrival times are evenly spaced considering traffic or other constraints.

Thus, the need to sensitize the driver as well as synchronizing of bus timings is highly necessary if the loss-making unit is to change into profitability. These buses if used effectively can actually reduce the waiting time of the passengers. It is often observed that in the time slots of morning 6-7.30 am when the traffic was least still multiple buses on the same route are sent by the depot which lead us to see that almost all these buses were completely empty, instead if this were to be dynamically synchronized such that there would be evenly spaced buses this would lead to win-win situation for the BEST as well as for the passengers. This would also lead to more utilization rate of the buses during peak timings.

3.1.3 Interconnectivity – This has been a major problem associated with our urban transport system that though node junction design for interconnectivity has been completely absent. This means if we consider the expanse of the western line in Mumbai say from Churchgate to Virar then we can divide the city in to

bus will have a greater number of trips but of a shorter distance. It also makes sure that as the distance covered is less so forward propagation of network congestion across the length of the city is reduced. Now at the junction nodes there will be fixed departures for arterial nodes based on connecting buses and dynamic departures from junction nodes based on traffic conditions. These considering now that the distance travelled by junction to arterial nodes is less the frequency of buses would be increased based on profitability, peak times and channel utilization.

Our main aim of the system model is to use a search algorithm which will define an objective function to maximize some useful parameters like passenger demand, route efficiency and optimum combination of buses and minimize cost incurred to BEST. For now, we intend to look into a solution that needs no/less infrastructural changes in the system. We propose to look into Genetic algorithm which is touted to be used in transportation problems to optimize the system. Genetic algorithm, which is normally in the biotechnology domain, can have factors correlating with certain parameters mentioned here. Here is a detailed theoretical insight of the algorithm:

3.2 Genetic Algorithm

Genetic algorithms are search algorithms based on the mechanics of natural selection and natural genetics. They combine survival of the fittest among string structures with a structured yet randomized information exchange to form a search algorithm with some innovative flair of human search. In every generation, a new set of artificial creatures (strings) is created using bits and pieces of the fittest of the old; an occasional new part is tried for good measure. While randomized, genetic algorithms are no simple random walk. They efficiently exploit historical information to speculate on new search points with expected improved performance.

Genetic algorithms have been developed and have been found fruitful to 1) Abstract and rigorously explain the adaptive process of natural systems. 2) Design artificial systems software that retains the important mechanisms of natural systems. The central theme of research of genetic algorithms has been robustness, the balance between efficiency and efficacy necessary for survival in many different environments. The implications of robustness for artificial systems are manifold. If artificial systems can be made more robust, costly redesigns can be

reduced or eliminated. If higher levels of adaptations can be achieved, existing systems can perform their functions longer and better.

Therefore, where robustness performance is desired, nature does it better, the features for self-repair, self-guidance and reproduction are the rules in biological systems; the secrets of adaptations and survival are best learned from careful study of biological examples. Genetic algorithms are theoretically and empirically proven to provide robust search in complex spaces and thus establishes validity in function optimization and control applications. Having been established as a valid approval to problems requiring efficient and effective search, genetic algorithms have widespread applications in business, scientific and engineering circles. These algorithms are computationally simple, yet powerful in their search for improvement. Furthermore, they are not fundamentally limited by restrictive assumptions about the search space. In order for genetic algorithms to surpass their more traditional cousins (other conventional search algorithms, in quest for robustness.

Genetic algorithms must differ in some very fundamental ways. Genetic algorithms are different from more normal optimization and search procedures in four ways:

1. Genetic Algorithms work with a coding of the parameter set, not the parameter themselves.
2. Genetic algorithms search from a population of points, not just a single point.
3. Genetic algorithms use payoff (objective function) information, not derivatives or other auxiliary knowledge.
4. Genetic algorithms use probabilistic transition rules, not deterministic rules.

Genetic algorithms require the natural parameter set of optimization problem to be coded as a finite length string over some finite alphabet. As an example, suppose we wish to optimize a function $f(x)$ on an integer interval. With genetic algorithms, the first step of the optimization process is to code the parameter x as a finite length string. Genetic algorithms exploit coding similarities in a very general way; as a result, they are largely unconstrained by the limitations of other methods. (Continuity, derivative existence, unimodality and so on)

In many optimization methods, we move from a single point in the decision space to the next using some transition rule to determine the next point. These point-to-point methods are dangerous because it is a perfect prescription for locating false peaks in multimodal (many peaked) search spaces. By contrast, genetics algorithms work from a rich database of points simultaneously (a population of strings), climbing many peaks in parallel; thus, the probability of finding a false peak is reduced over methods that go point-to-point.

Genetic algorithm thus starts with a population of strings and thereafter generates successive population of strings as against other techniques which might start off with a set of strings, apply some transition rules, and generate a new trial of strings. Assume a problem statement using genetic algorithm, successive populations can be generated simultaneously from a single set of population. By working from a population of well-adapted diversity instead of a single point, this characteristic of handling data parallel contributed to genetic algorithm robustness. Many search techniques require much auxiliary information in order to work properly. For example, gradient techniques need derivatives (calculated analytically or numerically) in order to be able to climb the current peak, and other local search procedures like greedy techniques of combinatorial optimization requires access to most if not all tabular parameters. By contrast, genetic algorithms have no need for all this auxiliary information: they are blind! To perform an effective search for better and better structures, they only require payoff values (objective function values) associated with individual strings. This characteristic makes genetic algorithm a more canonical method than many search schemes. After all, every search problem has a metric relevance to the search.

Unlike many methods, genetic algorithms use probabilistic transition rules to guide their search. Genetic algorithms use random choice as a tool to guide a search towards regions of the search space with likely improvement.

Taken together: These 4 differences- direct use of coding, search from a population, blindness to auxiliary information, and randomized operators; contribute to genetic algorithm's robustness and resulting advantage of other more commonly used techniques.

The mechanics of a simple genetic algorithm involve nothing more complex than copying strings and swapping partial strings. The simplicity of operation and power of effect are two main attractions of genetic algorithm. The procedure of Genetic Algorithms starts with randomly producing a set of chromosomes of a

population, and these set of genomes enter into what is called a Fitness or Objective Function.

3.2.1 Fitness Function (also known as the **Evaluation or Objective Function**) evaluates how close a given solution is to the optimum solution of the desired problem. It determines how fit a solution is. In genetic algorithms, each solution is generally represented as a string of binary numbers, known as a **chromosome**. We have to test these solutions and come up with the best set of solutions to solve a given problem. Each solution indicates how close it came to meeting the overall specification of the desired solution. This score is generated by applying the **fitness function** to the test, or results obtained from the tested solution.

The algorithm is composed of 3 operators:

1. Reproduction
2. Crossover
3. Mutation

3.2.2 Reproduction

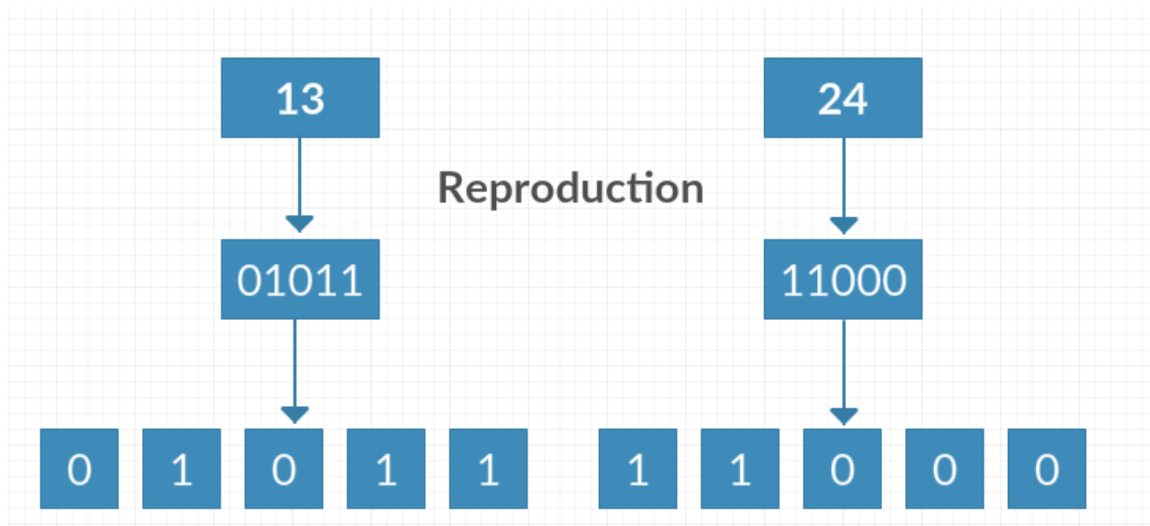


Fig. 3.3 Example of reproduction

Reproduction is a process in which individual strings are copied according to their objective function values. Intuitively, we can think of the function as a measure of profit, utility and goodness that we want to maximize. Copying strings according to their fitness values means that the strings with the higher values have a higher probability of contributing one or more offspring in the next generation.

3.2.3 Crossover

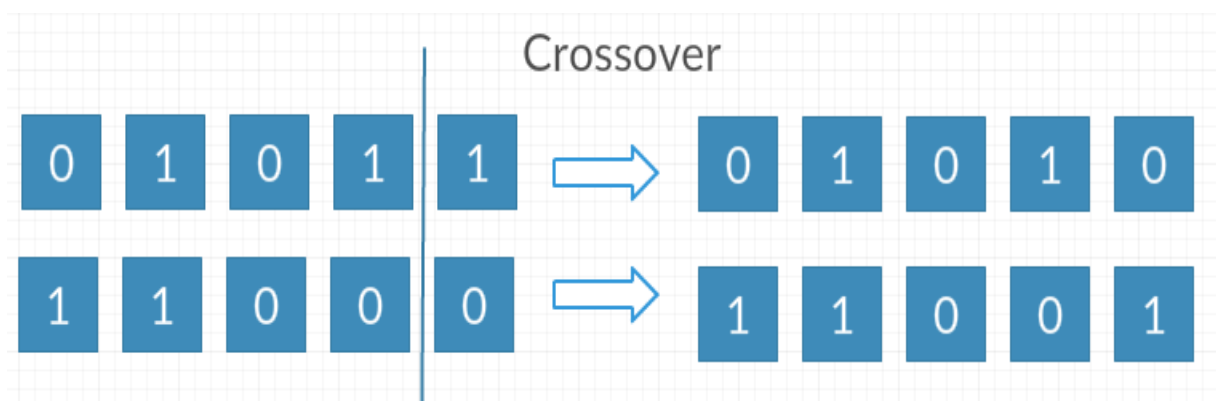


Fig. 3.4 Example of crossover

After reproduction, simple crossover may proceed in two steps: Members of the newly reproduced strings in the mating pool are mated at random. Second, each

pair of strings undergoes crossover as follows: an integer position 'k' along the string length on (1, l-1). Two new strings are created by swapping all characters between positions k+1 and l inclusively. Two new strings are created by swapping all characters between positions k+1 and l inclusively. The resulting crossover yields two new strings where strings are a part of a new generation.

3.2.4 Mutation

Mutation is a genetic operator used to maintain genetic diversity from one generation of a population of genetic algorithm chromosomes to the next. Mutation alters one or more gene values in a chromosome from its initial value. In mutation, the solution may change entirely according to user-definable mutation probability. This probability should be set low. If it is set to high, the search will turn into a primitive random search.

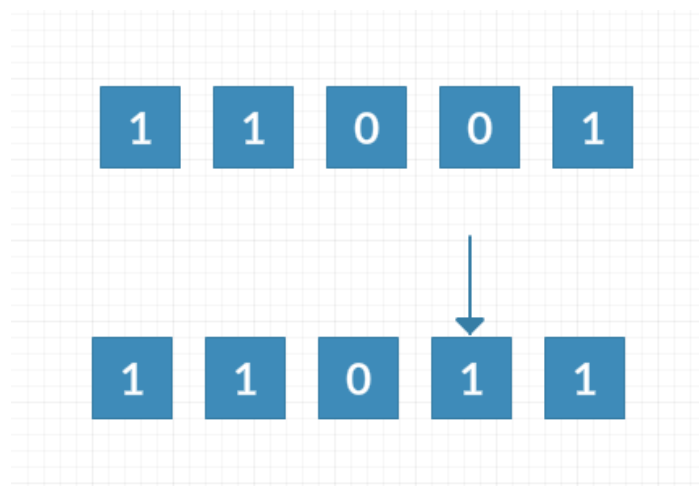


Fig 3.5 Example of Bit string mutation

Bit string mutation: The mutation of bit strings ensues through bit flips at random positions, the probability of mutation of a bit being inverse of the length of the binary vector. (For example: a 5-bit vector will have a mutation probability of (1/5) or 20%).

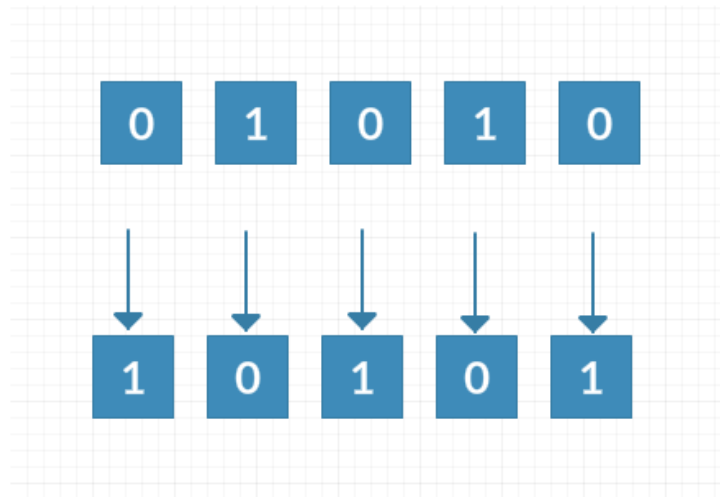


Fig 3.6 Example of Flip bit mutation

Flip bit mutation: This mutation operator takes the chosen genome and inverts all the bits (i.e. if the genome bit is 1, it is changes to 0 and vice versa)

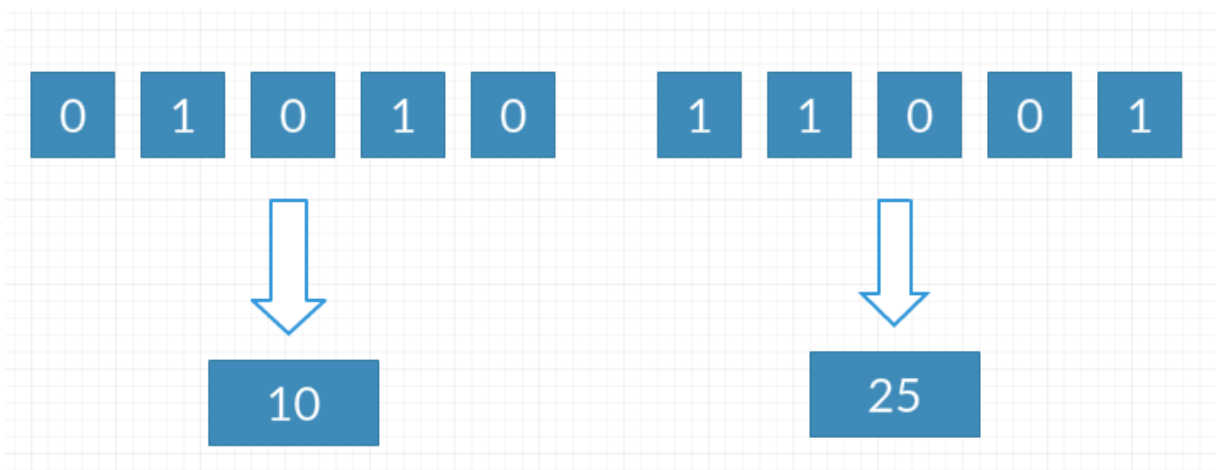


Fig. 3.7 Retention

3.3 Technology stack used:

3.3.1 PhpMyAdmin: It is an open-source software tool written in PHP (Hypertext Pre-processor) and JavaScript, intended to handle the administration of MySQL databases over the web through a graphic user interface. PhpMyAdmin supports a wide range of operations on MySQL and MariaDB.

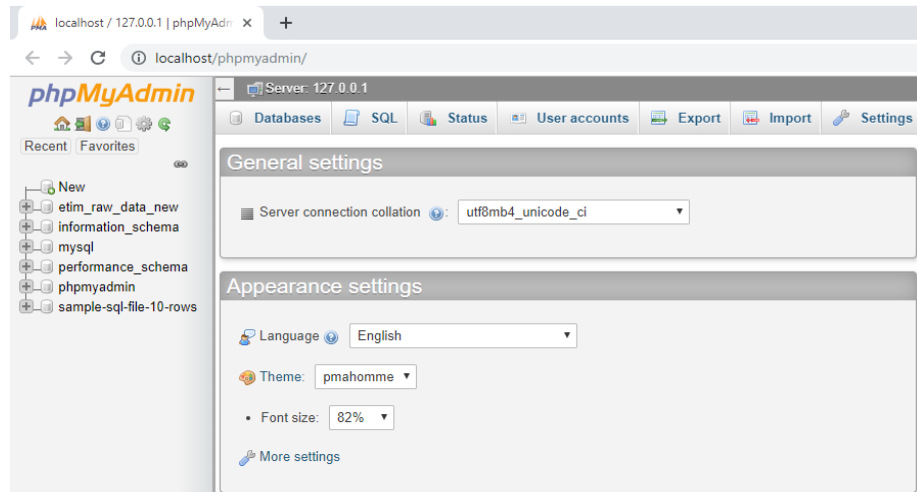


Fig 3.8 PhpMyAdmin Homepage

It supports frequently used MySQL operations like:

- Browse, drop databases, tables, views, fields and indexes.
- Create, copy, drop, rename and alter databases, tables, fields and indexes.
- Execute, edit and bookmark any SQL-statement, even batch queries
- Manage MySQL user accounts and privileges.

Other major features include:

- Importing data from CSV and SQL
- Exporting data from various formats: CSV, XLSX, Open Document Spreadsheets, Word, etc.
- Administering multiple servers
- Intuitive web interfacing with selected tools.

The databases on PhpMyAdmin can be accessed on a localhost server (localhost/phpMyAdmin or 127.0.0.1/xampp/) via the basic configuration of XAMPP, which is a free and open-source cross-platform web server solution stack developed by Apache Friends mainly consisting of Apache HTTP server

and the MariaDB database. XAMPP stands for x-os (x-os means it can be used for any operating system), apache, MySQL, php, Perl.

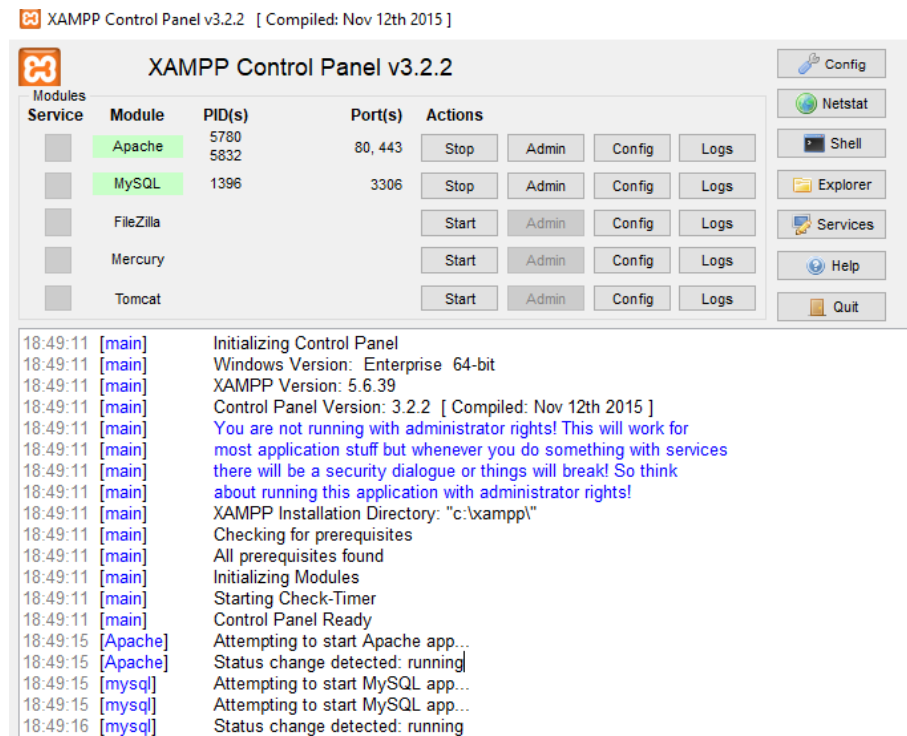


Fig 3.9 XAMPP Control Panel

XAMPP's designers intended it for use only as a development tool, to allow website designers and programmers to test their work without any access to the internet. It provides support for creating and manipulating databases in MariaDB and SQLite among others. It also has the ability to serve web pages on the World Wide Web. Apart from PhpMyAdmin, XAMPP comes with a number of other modules like WordPress, OpenSSL, MediaWiki and Joomla.

XAMPP offers MySQL (Database server) and Apache (Webserver) in a single setup and can be managed using XAMPP starter. Hit the start button under actions for MySQL and Apache and clicking on MySQL Admin will directly redirect the user to the localhost or 127.0.0.1, alternately it can also be typed manually on the browser's address bar.

For an IP address/Host Name at 127.0.0.1, MySQL server is connected to Port 3306 and Apache Webserver is connected to Port 80,443. Another feature of PhpMyAdmin is that it can create user accounts and grant privileges. It can be

used to store user names, passwords, email addresses and privileges. By default, the username is set as “root” and password is “”.

3.3.2 Python: Python is an open-source, integrated, high-level, general-purpose programming language, created by Guido van Rossum and first released in 1991. Python has a design philosophy that emphasizes code readability and provides constructs that enable clear programming on both small and large scales and it has a syntax that allows programmers to express concepts in fewer lines of code than might be used in languages such as C++ or Java.

Python features a dynamic type system and automatic memory management and supports multiple programming paradigms, including object-oriented, imperative, functional and procedural.



Fig 3.10 Python 3.7

Python has a large standard library, commonly cited as one of Python's greatest strengths, providing tools suited to many tasks. This is deliberate and has been described as a "batteries included" Python philosophy. Modules for creating graphical user interfaces, connecting to relational databases, generating pseudorandom numbers, arithmetic with arbitrary precision decimals, manipulating regular expressions, and doing unit testing are also included.

Rather than having all of its functionality built into its core, Python was designed to be highly extensible. This compact modularity has made it particularly popular as a means of adding programmable interfaces to existing applications.

In 2000, Python 2.0 was released, followed by upgradation to Python 3.0 in 2008. Python 2.7's end-of-life date is set to 2020 out of a concern that large amount of code cannot be forward-ported to Python 3. The latest version currently is Python 3.7.

3.3.2.1 Libraries used within Python:

NumPy: NumPy (Numerical Python computing) is a general-purpose array-processing package. At the core of its package, is the ndarray object. It has sophisticated functions useful for linear algebra, fourier transform and random number capabilities. It is also an efficient multi-dimensional container of generic data. It can seamlessly and speedily integrate with a wide variety of databases.

Matplotlib: Matplotlib (Mathematical plotting library) is a visualisation library for plotting and creating 2D graphs and plots from python scripts. Its numerical mathematics extension is NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter and wxPython.

Pandas: Written in Python, Cython and C; Pandas is a software library for data manipulation and analysis. It offers data structures and operations for manipulation numerical tables and time series.

PyMySQL: PyMySQL (Python+MySQL) is a pure- Python MySQL client library which works with interfacing Python to MySQL or MariaDB. Its main features are the cursor and connect which help to interact with the databases.

OpenPyxl: OpenPyxl (Python+Excel) allows Python programs to read, write and modify Excel spreadsheets using the workbook function.

Datetime: In Python, data, time and datetime classes provide a number of functions to deal with dates, times and time intervals. Date and datetime are an object in Python, so during manipulation objects are manipulated, not string or timestamps.

Chapter 4

Proposed System Implementation

Since we are working to transform an existing static system and make it dynamic, we need to feed in real time data for input parameters as well. Hence, we plan to build a time-based module over a set mathematical models which will contain information of the distance between two nodes, nodal bus stops and the passenger demands. These inputs will be in form of matrices and this data will be encoded in the algorithm, starting from an initial population or sample set after which it goes through several steps of generations until a most suitable generation is produced in form of the required outputs. We desire to present this outputted information to BEST, expected to improve the current system.

4.1 Individual Module Components and Assumptions:

4.1.1 No. of time slots/time size: Since this is a model based on time consideration, it is imperative to have time slots or a time size with subsequently a number of prime input parameters to have a strong dependency on it.

4.1.2 Bus Running Cost per minute, Ticket Price, Fixed Cost for B.E.S.T: These input parameters are to be either declared explicitly within the environment or are kept in generalized format to be inputted by the user. The units for the following are: Rs/minute, Rs/person and Rs/bus respectively.

4.2 Matrix models of input parameters:

4.2.1 Route matrix

Table no. 4.1 Route matrix

Route Matrix	Nodes	D	D		
		N1	N2	N3	N4
<i>Routes</i>					
R1		3	1	2	0
R2		3	1	0	2
R3		1	3	2	0
R4		1	3	0	2

This is a defined route matrix of the order (4*4) having information of a combination of buses being deployed on a specific set of nodes. The order of how a bus is deployed on the route across several set of nodes is determined by the

non-zero values specified within the matrix. For example, Route “R1” has a bus deployed across Node “N2”→Node “N3”→Node “N1” in particular order. Hence the values ‘1’, ‘2’ and ‘3’ are assigned respectively to those nodes corresponding to Route R1. If the bus on the same route is not deployed across a particular node, that corresponding value within the corresponding unit cell is assigned as null value or zero. For example, the bus on Route “R1” is not deployed on Node “N4”, hence ‘0’ is assigned. The Routes “R1-R4” will be given a combination which will signify whether a bus is deployed across the route or not. “R1-R4”→[1 0 1 0] means that a bus is deployed only from R1 and R3 and no bus is deployed from R2 and R4.

4.2.2 Demand matrix

Table no. 4.2 Demand matrix

Demand for T1-T2		Source				Total Demand in the entire network between this time slot
	Node	N1	N2	N3	N4	
Destination	N1	0	1	4	3	23
	N2	2	0	1	1	
	N3	4	3	0	0	
	N4	2	2	0	0	

This is a defined demand matrix of the order (4*4) which has information of the demand from one node to the other, both axes are defined by nodal points. The column nodes are source points and the row nodes are the destination points. By default, the diagonal elements are null values since source and destination cannot have a common node. The rest of the nodes will carry finite whole number values. For example, there are 4 passengers getting up at source Node 3 and getting down at the destination Node 1. This single matrix is defined for one range of time slots. The total demand in the entire network for the given time slot is also stored, here 23. The number of demand matrices in the system is decided by the number of time slots that are available or defined by the user.

4.2.3 Time Travel Matrix

Table no. 4.3 Time Travel matrix

Bus Travel Time for T1-T2		Source			
	Node	N1	N2	N3	N4
Destination	N1	0	2	3	5
	N2	4	0	6	1
	N3	2	3	0	0
	N4	1	2	0	0

This is a defined time travel matrix of the order (4*4) which has information of the time taken in minutes by the bus to traverse from one node to the other, both axes are defined by nodal points. The column nodes are source points and the row nodes are the destination points. By default, the diagonal elements are null values since source and destination cannot have a common node. The rest of the nodes will carry finite whole number values. For example, the time taken by bus to go from source Node 2 to destination Node 4 is 2 minutes. This single matrix is defined for one range of time slots. The number of time travel matrices in the system is decided by the number of time slots that are available or defined by the user.

4.3 Real Time parameters provided by BEST

The following concerns the BEST database given to us. The database is originally available in .sql extension with a size of 15.1 GB with about 5 crore rows and 35 columns in one single table. It consists of all the data marked for the entries for the month of July 2017.

Table no. 4.4 Column matrix of the real time BEST data

Sr. No.	Column	Sr. No.	Column	Sr. No.	Column
1	depot_cd	13	ticketCode	25	toStageNo
2	dutyDate	14	RUPEE_DENO	26	hexwaybillNo
3	waybillDate	15	FULL_TKT	27	tripNo
4	waybillCollectionDate	16	bottkt	28	tripDirection
5	ticketId	17	machineNo	29	docketNo
6	FULL_AMT	18	ticketDate	30	concessionId
7	FULL_PAX	19	ticketTime	31	serialNo
8	routeNo	20	fromStopcode	32	ep_flag
9	empCode	21	frmStopName	33	sent_flag
10	TICKET_BUS_NO	22	toStopname	34	insertedDate
11	TTL_TKT_ISSUE	23	toStopCode	35	modifiedDate
12	etimTicketCodeId	24	fromStageNo		

These are 36 parameters are registered in the BEST Database when a ticket is purchased. The ones in bold are the ones which will be prime and known with respect to our analysis, the remaining are unique codes which are generated for identification purpose.

4.3.1 depot_cd: This parameter may have a number of depots in region-wise combinations. There are a total of 27 distinct depot_cd values labelled between 01 to 38.

4.3.2 waybillDate: Available full date and time format (for eg: 01-07-2017 04:10:21), this variable aligns exactly with the values of **dutyDate** and **waybillCollectionDate** and **ticketDate**; the latter parameters are only in the full date format.

4.3.3 ticketId: Carries a unique 9-digit id (165180870, 165180871, 165180874..) which is created whenever a ticket is printed for a trip out of the machine.

4.3.4 FULL_AMT: The total amount paid for a ticket for a full trip. This amount is computed on account of an individual passenger or a consolidated set of passengers going together for the journey.

4.3.5 FULL_PAX: The total number of passengers noted for a single dispatched ticket. The minimum number in this column set is 1.

4.3.6 routeNo: For the total number of routes available and covered by BEST in the city of Mumbai is assigned a specific identifier or a unique number to pinpoint a particular route. The routeNo is in form of 4-digit number (eg. 0440, 0021, 1030..)

4.3.7 empCode: Represents a unique 6-digit code assigned to each employee of BEST. (eg. 196589, 107715, 107694..)

4.3.8 TICKET_BUS_NO: Equivalent to a unique vehicle number which is a 4-digit code. (eg. 2343, 2336, 2431)

4.3.9 TTL_TKT_ISSUE: The total number of passengers on a single ticket. The minimum number in this column set is 1.

4.3.10 RUPEE_DENO: Represents a rupee denomination that is received from the passenger(s). Where: **RUPEE_DENO * FULL_PAX= FULL_AMT.**

4.3.11 FULL_TKT: This column contains either 'F' which signifies a full ticket or a 'H' for a half ticket against the single ticket.

4.3.12 machineNo: Contains an 8-digit alphanumeric code (eg: CBD10171, CBD10035) which is assigned to each machine from which tickets are printed.

4.3.13 ticketTime: Contains the timestamp which gives the time at which the ticket was stamped.

4.3.14 fromStopcode, toStopCode: Variable 1 to 3-digit codes globally located which corresponds to **frmStopName** and **toStopname** respectively. (eg: 734, 710, 714)

4.3.15 frmStopName, toStopname: These columns lists carries the source and destination locations respectively. (eg: COLABA DEPOT, LOWER PAREL STATION)

4.3.16 fromStageNo, toStageNo: Variable 1 to 3-digit codes locally assigned which corresponds to **frmStopName** and **toStopname** respectively. (eg: 001,002,013)

4.3.17 hexwaybillNo: A 12-digit code (eg. CBD065BEE7D1, CBD065BEE7D9) that has a relation to the 8-digit **machineNo**.

4.3.18 tripNo: The number of trips covered by the **TICKET_BUS_NO**. The minimum number is 1.

4.3.19 tripDirection: This parameter flags either a 'U' or 'D' which represents 'Up' or 'Down' which a universal reference. As observed, a ticket from R.C. Church to Bandra Reclamation is flagged as 'U' meaning the route is directed from south to north and a ticket from Thakre Chowk Dadar to Colaba depot is flagged as 'D' aka 'Down'.

Other minor parameters: (These parameters are either not yet completely decoded by us or are not holding a relative significance in our analysis)

4.3.20 etimTicketCodeId, ticketCode: The former one is composed of integer numbers at minimum value 1, the latter is associated to it in a way. (for all **etimTicketCodeId**=1, **ticketCode**=00; For other values of **etimTicketCodeId** (greater than 1), the **ticketCode** has an alpha variable code assigned to it.

4.3.21 docketNo, concessionId, serialNo, ep_flag: The docketNo has some unit cells empty or null sets, the which all the latter 3 parameters show a 00 value. To those docket values which hold 14-digit alphanumeric code (eg. 14021500000781) either or all of the 3 columns show a corresponding non-zero value displayed against that ticket.

4.3.22 insertedDate, modifiedDate: These column matrices have a full date and a timestamp displayed.

4.4 Data Analysis, Extraction, Parsing, Processing

4.4.1 Data Importing and Extraction

To upload/import the data (given 15.1 GB BEST Data):

Step 1: Hit start on the action button of the modules Apache webserver and MySQL server on the XAMPP control panel.

Step 2: Click on the MySQL server Admin which takes you to the PhpMyAdmin homepage.

Step 3: Before uploading any file into the server, an empty database needs to be created onto which the file can be imported later. To create a new database table, click on the Databases tab on the homepage or the ‘New’ tab on the extreme left of the page. Alternately, you can manually type http://localhost/phpmyadmin/server_databases.php on the web browser. This will take you to the list of databases that exist in default state or have been created in the past.

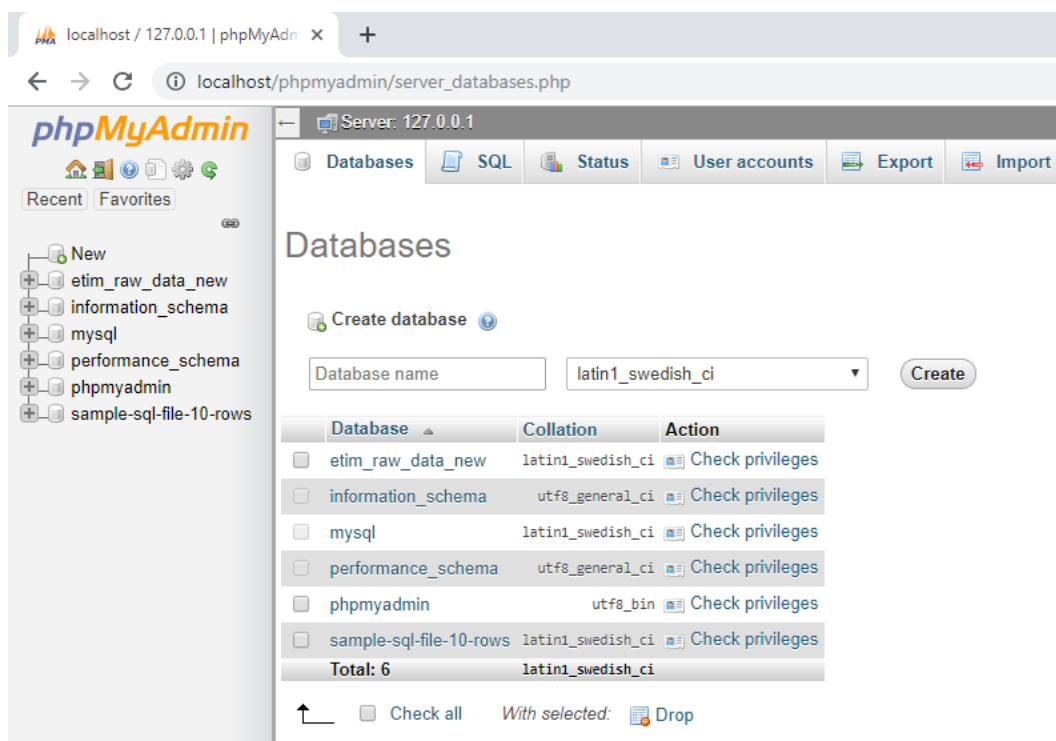


Fig 4.2 Creating a database table on PhpMyAdmin

Step 4: Enter the name of the database and click on Create. (To delete an existing database, select the corresponding checkbox and click on Drop.)

Step 5: Before importing the data, hover over the checkbox corresponding to the database name given in the extreme left tab and keep it selected before performing the next step.

Step 6: To import the file onto the empty database table, head to the import tab of PhpMyAdmin or http://localhost/phpmyadmin/server_import.php on the web browser.

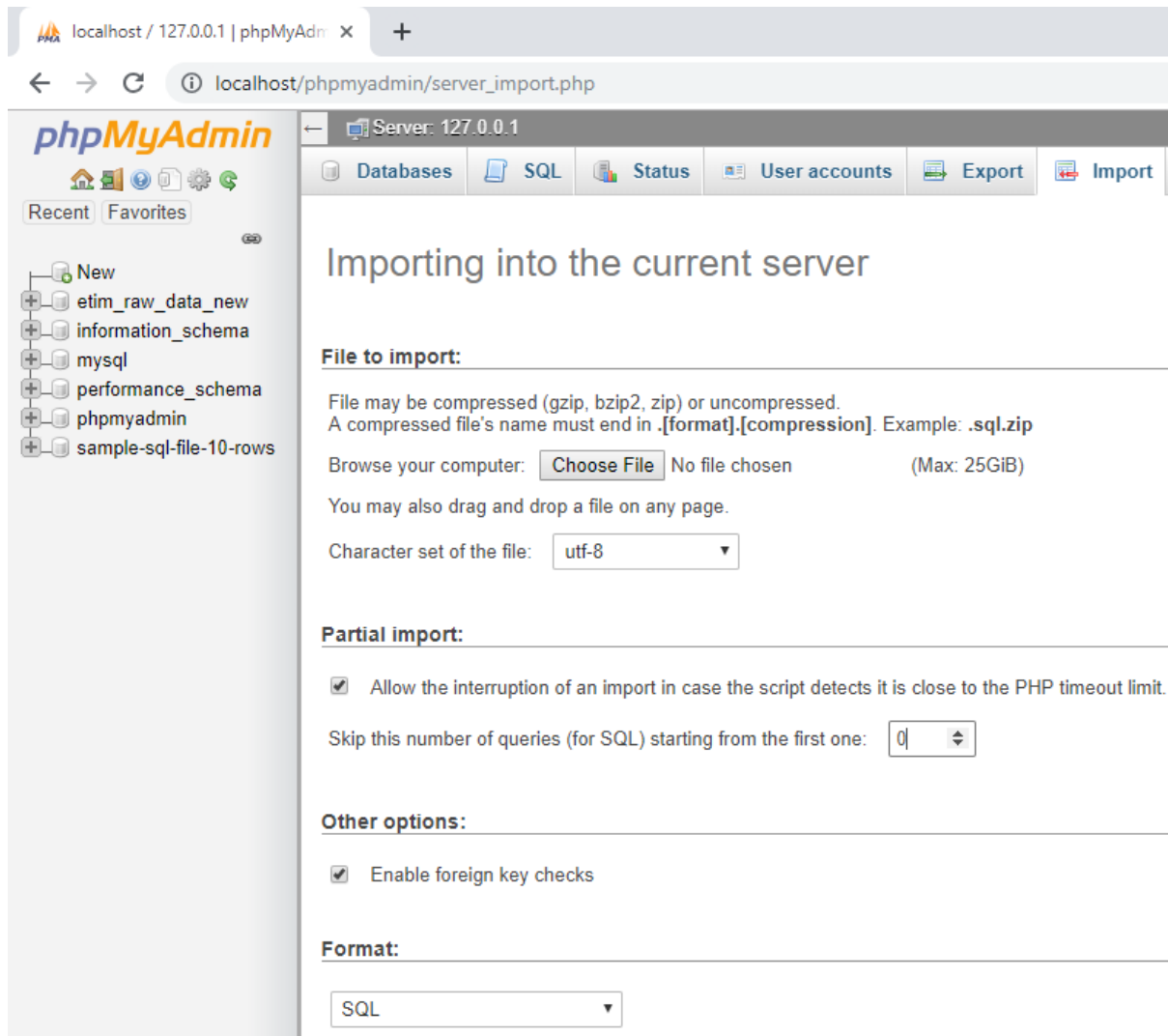


Fig 4.3 Importing a database file on PhpMyAdmin

Step 7: Click on ‘Choose File’ widget under **File to import:** and choose a file from your computer that you wish to upload. Under **Format:** select the option in the dropdown which matches with the file extension to be uploaded.

This import procedure may work only for files in sizes of KB or MB’s

For uploading larger files (the given file is 15 GB) there is an alternate procedure to handle Big Data problems.

In case there is an error observed while importing data, the data can also be uploaded via Command Prompt.

To upload data via Command Prompt(cmd) for Windows OS, perform all the above steps in the previous method mentioned from **Step 1** to **Step 5** as given.

Step 6: Head over to the Command Prompt and modify the path directed to the MySQL bin application of the XAMPP folder in the computer system.

For eg: the default path will be specified as **C:\Users\Jyotsna>** on the cmd.

Type in **cd C:\xampp\mysql\bin** on the same line, it looks like this:

C:\Users\Jyotsna>cd C:\xampp\mysql\bin

C:\xampp\mysql\bin>

Step 7: After the path on the cmd is modified, a unique sql format expression needs to be entered along the line as given below:

The general format to upload a file on MySQL server is *mysql -u "username" -p "databasename" <file_path+file_name.sql*

For eg: for a given file to be uploaded named as **etim_raw_data_new_02_09_17.sql** present in the system under file path **C:\Users\Jyotsna\Desktop\DynamicBEST** database table created which is named as **etim_raw_data_new** and default settings of username given as **"root"**, the format of the entire line looks like this:

C:\xampp\mysql\bin>mysql -u "root" -p etim_raw_data_new <"C:\Users\Jyotsna\Desktop\DynamicBEST\etim_raw_data_new_02_09_17.sql"

Step 8: Hit enter after typing this line format, in case of no errors or missteps the following line appears just after

Enter password:

Under default settings, password is "" (i.e. nothing) in that case simply hit enter, if the database is password protected, enter the password and hit enter. In case of no errors or missteps, you will observe a blinking cursor on the next line which indicates database is currently uploading successfully. After the full database file is uploaded, the cursor stops blinking.

During the process of data importing here, you can observe the status of the data upload on the main page clicking on the created database table. (The file uploaded here is in the database name: **etim_raw_data_new**)

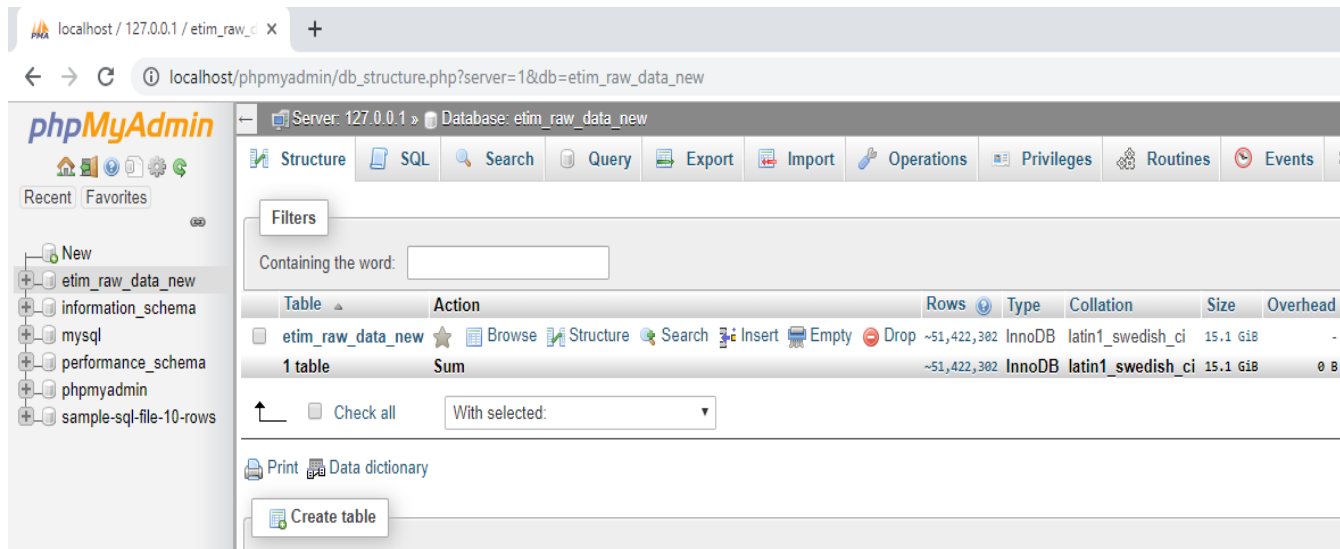


Fig 4.4 Imported Database statistics on PhpMyAdmin

In the course of data import, as the cursor is blinking; the numbers under rows and the size tab keeps getting updated as you refresh the page with time. After the full file upload completion as seen above, the number of rows uploaded from the file were 51,422,302 and data size 15.1 GB.

4.4.2 Parsing Data:

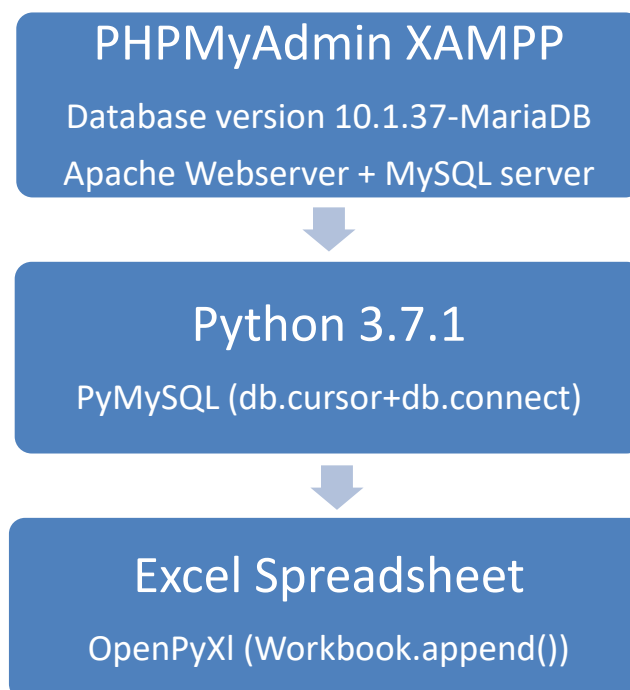


Fig 4.5 Program Flow of Parsing Data

To export data from the MySQL database into more editable formats like Excel spreadsheets, Word or text files; PhpMyAdmin has a feature to export certain number of rows from the data to a selected format.

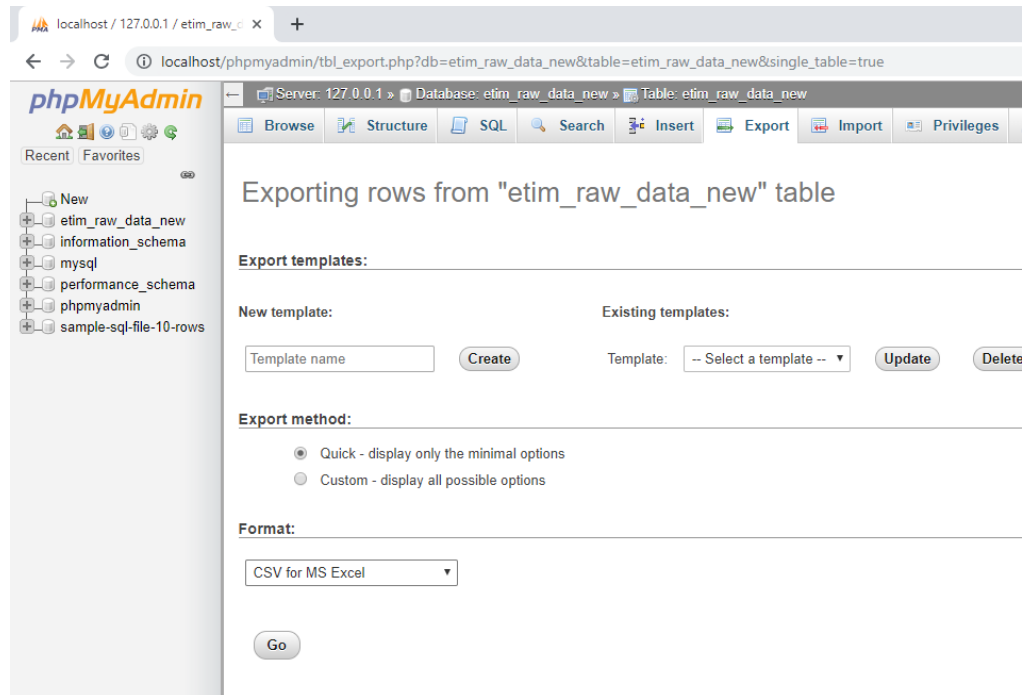


Fig 4.6 Exporting data via PhpMyAdmin

The export option may not have many features to analyse data and may take a lot of time for outputting the data in the files which is not feasible enough in Big data problems. A better and faster alternative is to make it interact with a Python interface and use Python library characteristics to extract any kind and any amount of data in general format files:

To create a MySQL+Python+Spreadsheet interface mainly 2 Python libraries are required: PyMySQL and OpenPyxl

Step 1: To open a database connection, declare a variable which will connect PyMySQL to a list of database arguments.

```
db=pymysql.connect(host="localhost",user="root",passwd="",db="etim_r  
aw_data_new")
```

Step 2: Prepare a cursor object to execute a SQL query and point to set of data
cursor=db.cursor()

Step 3: Fetch rows from the data
data=cursor.fetchall()

Step 4: To push the required set of data into excel (CSV or .xlsx) use Workbook function from OpenPyxl and save the file in a desired path in raw form using appropriate file extension.

```
book=Workbook()
```

```
book.save(r"C:\Users\Jyotsna\Desktop\excel_sheet_name.xlsx")
```

The purpose of parsing data is to analyse only a required set of data into more readable formats. Extracting only a required set of data can be controlled via the SQL query.

4.4.3 Processing Data:

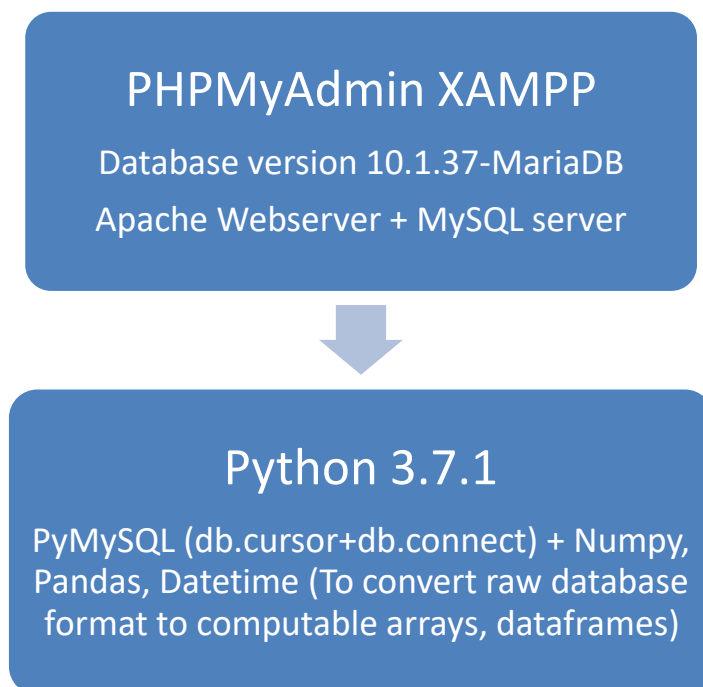


Fig 4.7 Program Flow of Processing Data

In the previous segment, a specific set of data is taken from the entire database and pulled into spreadsheets for data analysis only. For an application of evaluating data, the Workbook OpenPyXL function is to be eliminated and raw form of data is to be converted to computable data. This procedure is constructed and modified in such a way that it can take in data from MySQL database and evaluate the required parameters directly into the Python environment, followed by which the input parameters enter into the rest of the program flow.

The purpose of processing data here is extract data from the BEST database file, typecast raw data into NumPy arrays or Pandas data frames and convert it into input parameter matrices. The Route Matrix (See Table no. 4.1), Demand Matrix (Table no. 4.2) and Time Travel Matrix (Table no. 4.3) and the entire time scheduling model can be extracted using particular datasets of ticket data: Route combinations (4.3.6 routeNo), Node combinations (4.3.14 fromStopcode, toStopCode) for RouteMatrix, Number of passengers (4.3.5 FULL_PAX) for Demand Matrix and Timestamps (4.3.13 ticketTime) for Time Travel Matrices and time scheduling of the model.

For processing data follow the initial procedure for parsing data (Step 1 to Step 3) and omit Step 4.

Step 4: (Taking a variable “data” which carries the raw form of database information)

For conversion to NumPy arrays (Reading directly from the database)

numpy_data=np.asarray(data)

For conversion to Pandas data frames (Reading from excel files)

Route_Mat1=pd.read_excel(r"C:\Users\Jyotsna\Python\Python37\Project1\RouteMatrix.xlsx")

RouteMat=Route_Mat1.values

4.5 Complete Program Flow:

Post all data extraction and getting all input parameters, a second phase of parameters need to be evaluated which goes into what is called as the **Objective or Fitness function**.

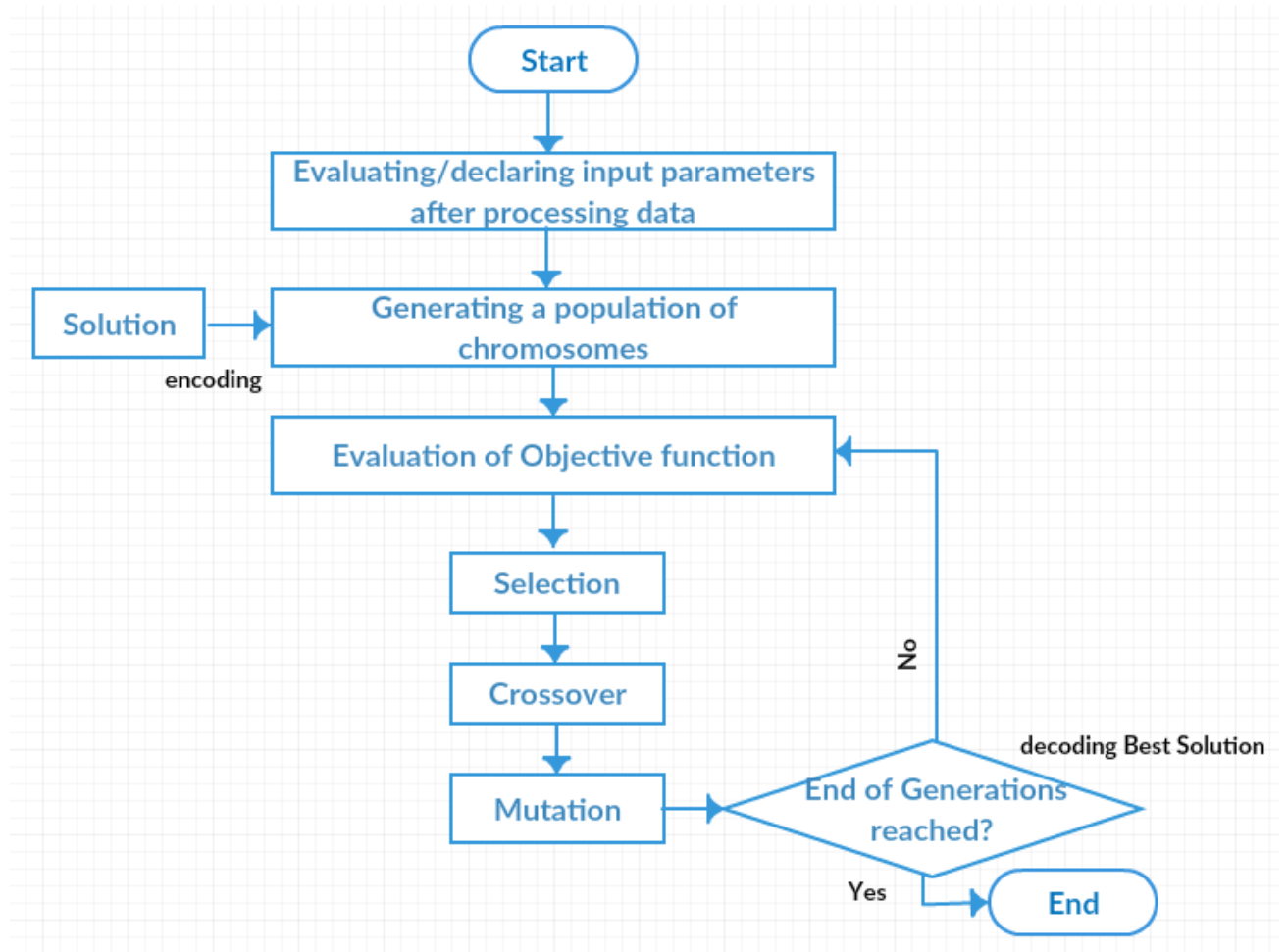


Fig 4.8 Main structure of the Time scheduling model/Complete Program Flow

The Objective function is composed of the following kind of arguments:

1. Revenue generated by the BEST after the bus is run and passengers in the network have been catered to.
2. Cost incurred to BEST including all starting costs of infrastructure and capital required to run the bus.

3. Number of satisfied passengers in the network that have boarded the bus to travel from a source to destination.

The arguments will be placed in the Objective function in such a manner that the revenue generated and the number of satisfied passengers is increased and cost incurred to BEST after bus run is decreased. A very general notion of Genetic algorithm is that it can produce better genomes by maximizing a function. Genetic algorithm can however be used for both minimization and maximization of functions.

$$f = 1 * \frac{Revenue}{Cost} + 1 * \frac{Satisfied Demand}{Expected Demand} \quad --(1)$$

$$f = 1 * \frac{1}{1 + Unsatisfied} + 1 * \frac{Revenue}{Cost} \quad --(2)$$

As per our requirements, the above equation (1) and (2) can be represented as objective functions and our goal is to maximize them.

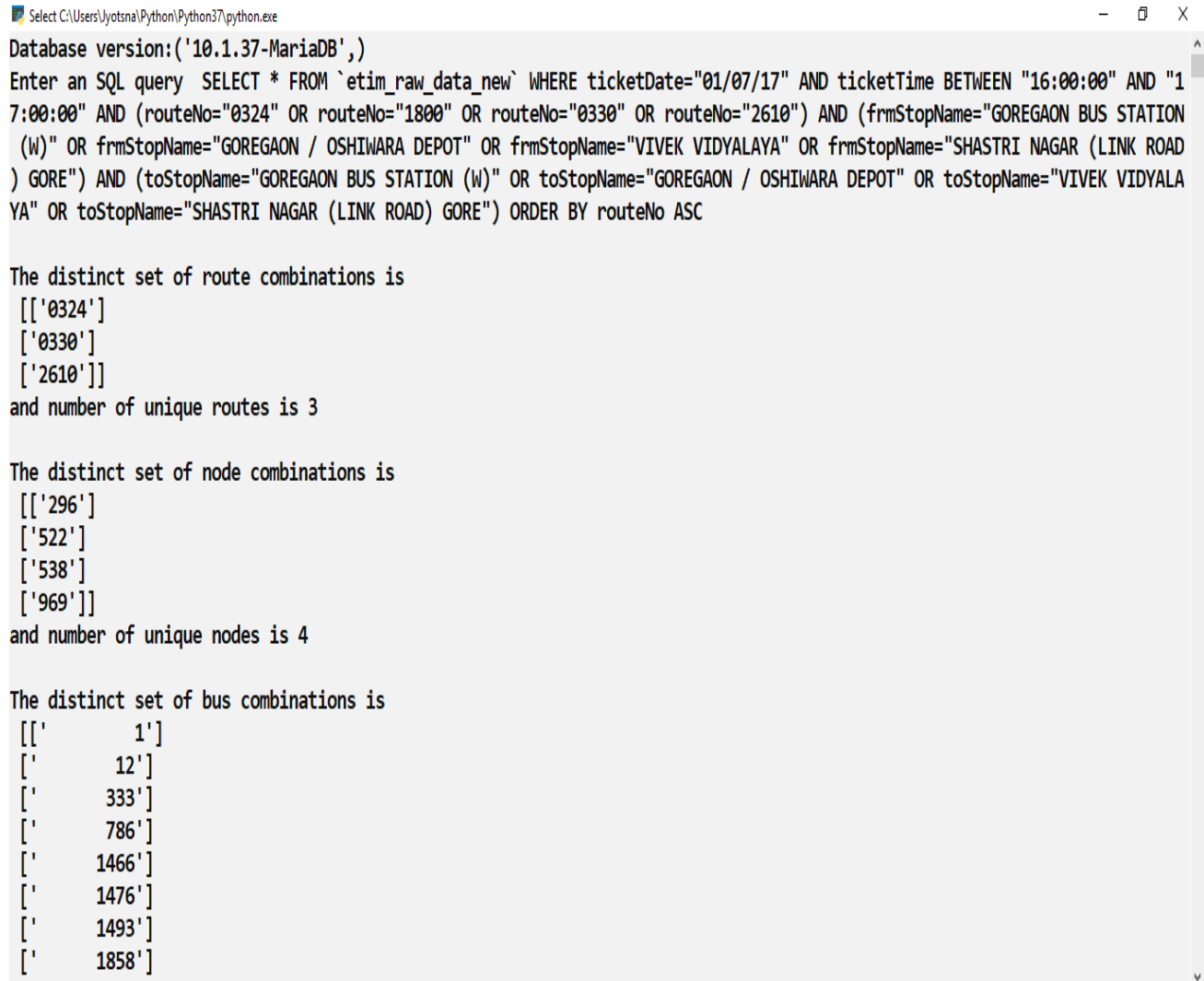
After encoding Genetic algorithm, we expect to compute and generate the following:

1. For particular time instances, generate demand and time taken for the bus to traverse from a source to one destination. This will take into account the demand and traffic variation at different variations of time slots through the day.
2. For a particular time instance, generate the optimized set of routes for the buses to be deployed for a given number of routes and number of nodes.

Chapter 5

Results and Discussions

5.1 Data Extraction and Evaluation output



The screenshot shows a Python terminal window with the following content:

```
Select C:\Users\Jyotsna\Python\Python37\python.exe
Database version:('10.1.37-MariaDB',)
Enter an SQL query SELECT * FROM `etim_raw_data_new` WHERE ticketDate="01/07/17" AND ticketTime BETWEEN "16:00:00" AND "17:00:00" AND (routeNo="0324" OR routeNo="1800" OR routeNo="0330" OR routeNo="2610") AND (frmStopName="GOREGAON BUS STATION (W)" OR frmStopName="GOREGAON / OSHIWARA DEPOT" OR frmStopName="VIVEK VIDYALAYA" OR frmStopName="SHASTRI NAGAR (LINK ROAD) GORE") AND (toStopName="GOREGAON BUS STATION (W)" OR toStopName="GOREGAON / OSHIWARA DEPOT" OR toStopName="VIVEK VIDYALAYA" OR toStopName="SHASTRI NAGAR (LINK ROAD) GORE") ORDER BY routeNo ASC

The distinct set of route combinations is
[['0324']
 ['0330']
 ['2610']]
and number of unique routes is 3

The distinct set of node combinations is
[['296']
 ['522']
 ['538']
 ['969']]
and number of unique nodes is 4

The distinct set of bus combinations is
[['1']
 ['12']
 ['333']
 ['786']
 ['1466']
 ['1476']
 ['1493']
 ['1858']]
```

Fig 5.1 Data extraction using SQL Query as input

Total unique fromStageNo combinations is

```
['001']  
['004']  
['005']  
['007']  
['019']  
['027']  
['029']
```

and number of unique stages is 7

The ROUTE MATRIX is

Node 1	Node 2	Node 3	Node 4
1	2	0	0
1	0	2	0
1	3	0	2

The number of timeslots is 2

The minimum timeslot in the data is 16:08:24

The maximum timeslot in the data is 16:59:40

The average time between each timeslot in minutes is 25.633333333333333

The array of timeslots is

```
16:08:24 | 16:34:02 | 16:59:40 |
```

Fig 5.2 Data evaluation

C:\Users\Jyotsna\Python\Python37\python.exe

The demand for time range 16:08:24 and 16:34:02 is

Node 1	Node 2	Node 3	Node 4
0	12	0	0
24	0	0	3
0	0	0	0
0	2	0	0

The demand for time range 16:34:02 and 16:59:40 is

Node 1	Node 2	Node 3	Node 4
0	10	1	0
20	0	0	9
0	2	0	0
0	0	0	0

Concatenated DEMAND MATRIX is

0	12	0	0	0	10	1	0
24	0	0	3	20	0	0	9
0	0	0	0	0	2	0	0
0	2	0	0	0	0	0	0

Fig 5.3 Data evaluation

5.2 Time based module evaluation using Genetic algorithm

C:\Users\Jyotsna\Python\Python37\python.exe

Initial Data

The routes followed by the buses are given as: ROUTE MATRIX

Node 1	Node 2	Node 3	Node 4
3	1	2	0
3	1	0	2
1	3	2	0
1	3	0	2

The demand at each node is given as: DEMAND MATRIX

Node 1	Node 2	Node 3	Node 4	Node 1	Node 2	Node 3	Node 4
0	2	4	2	0	3	2	0
1	0	3	2	2	0	1	1
4	1	0	0	2	5	0	0
3	1	0	0	1	0	0	0

Fig 5.4 Time based module output

The time taken for a bus to go from each node is: TIME TRAVEL MATRIX

Node 1	Node 2	Node 3	Node 4	Node 1	Node 2	Node 3	Node 4
0	4	2	1	0	3	1	1
2	0	3	2	1	0	2	1
3	6	0	0	2	5	0	0
5	1	0	0	4	1	0	0

The route sequence of the bus deployed is given as:

2	3	1	0
2	4	1	0
1	3	2	0
1	4	2	0

Total demand in each time slot is given as:

23	17
----	----

Fig 5.5 Time based module output

Parameters Matrix

Time	Satisfied	Cost	Revenue	Unsatisfied
14	15	132	150	25
6	8	68	80	32
15	13	140	130	27
10	12	100	120	28
8	13	84	130	27
17	18	156	180	22
0	0	20	0	40
7	6	76	60	34
14	15	132	150	25
23	26	204	260	14
2	5	36	50	35
14	15	132	150	25
9	11	92	110	29
8	13	84	130	27
8	7	84	70	33

Fig 5.6 Time based module output

The initial route combination is given as:

Bus 1	Bus 2	Bus 3	Bus 4
1	0	1	0
1	0	0	0
0	1	1	0
0	0	1	1
1	0	0	1
0	1	1	1
0	0	0	0
0	1	0	0
1	0	1	0
1	1	1	1
0	0	0	1
1	0	1	0
0	1	0	1
1	0	0	1
0	0	1	0

Fig 5.7 Initial input combination

Final Solution Route Combination

Bus 1	Bus 2	Bus 3	Bus 4
1	0	0	1
1	0	0	1
1	0	0	1
1	0	0	1
1	0	0	1
1	0	0	1
1	0	0	1
1	0	0	1
1	0	0	1
1	0	0	1
1	0	0	1
1	0	0	1
1	0	0	1
1	0	0	1
1	0	0	1

Fig 5.8 Final output combination

Chapter 6

Conclusions and Scope

6.1 Conclusion

With the current scenario of transportation and scarcity of spaces to do an infrastructure overhaul it is necessary to remodel what we have if we can't create new. If remodeled, then city planning could help us in a lot of problems that commuters face on a daily basis. The problem of positive feedback i.e. cabs and autos increasing in nature and adding to network congestion arises only due to "not" scarcity of buses but mismanagement of buses and this if looked into by the proposed design could work in favour of the city planning and more importantly make the BEST into a profit making unit but at the same time a responsible and reliable mode of transport for all commuters.

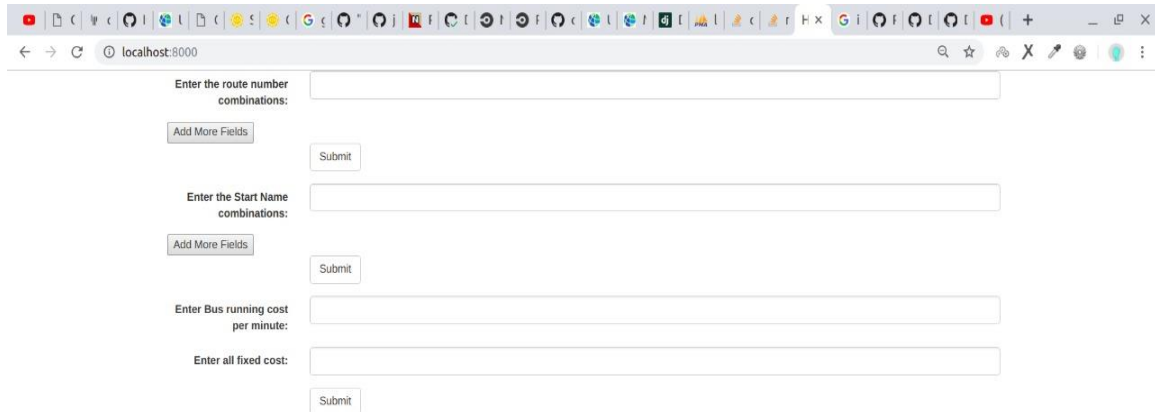
The system when successfully implemented can lead to new possibilities which can massively improve the state of public transportation in cities. Some of these include and are not limited to:

1. **Dynamic Route Network:** Routes can be made flexible such that the Route Network i.e. the bus stops (nodes) and their interconnection can themselves change based on the demand/connectivity to and from particular nodes.
2. **Citywide Unified Ticketing System:** A unified system can help commuters change the mode of transport effortlessly and hassle-free and hence, depend less on auto-rickshaws/taxis for places remote and inaccessible otherwise.
3. **Remodeling for different cities:** This new system can be modified to implement in different cities and also can be used to grow alongside burgeoning new cities.

An action plan can be put forward which can start off with data collection and mapping to understand routes, peak load capacity timings and profitable and non-profitable routes, congested network points and peak hour requirements. The data analysis will help us in creating a network flow optimization diagram which will lead us to the design methodology and execution of modelling a failsafe network with alternative route management.

6.2 Scope

The future scope of this project is to encompass all the backend algorithm development under a frontend website to get it as a final product to be used by the BEST for scheduling buses over combination of routes and nodes on time basis over the entire day span.



The screenshot shows a web browser window with the address bar displaying 'localhost:8000'. The page contains four input sections, each with a text input field, a 'Submit' button, and an 'Add More Fields' button. The sections are labeled as follows:

- Enter the route number combinations:
- Enter the Start Name combinations:
- Enter Bus running cost per minute:
- Enter all fixed cost:

Fig 6.1 Preview of the website

The website is built on Django interface, a Python frontend frame work which connects to the MySQL BEST database file uploaded on PhpMyAdmin and the functions on the Python backend. The intention of building this product is to choose a random set of routes and nodes from a total of 514 routes and 958 nodes for a time slot and for a day or a week's schedule and display a table of optimized routes at the output end.

Chapter 7

REFERENCES & BIBLIOGRAPHY

REFERENCES & BIBLIOGRAPHY

1. Goldberg, D. (1989). *Genetic Algorithm in Search, Optimization and Machine Learning*.
2. Dr. V. Ramesh and Parulekar, M. (2016). *BEST R: Buses for Enhanced State Transport- Roadmap to Urban Transport Modelling*. Research and Innovation Center, Dwarkadas J. Sanghvi College of Engineering, University of Mumbai
3. Sheppard, C. (2016). *Genetic Algorithm with Python*. 5th ed. [ebook] Clinton Sheppard, 2018. Available at: https://books.google.co.in/books/about/Genetic_Algorithms_with_Python.html?id=3jNqtAEACAAJ&source=kp_book_description&redir_esc=y [Accessed 22 Apr. 2018]..
4. BEST. (2019). / *BEST*. [online] Available at: <https://www.bestundertaking.com/in/main.asp?m=Transport> [Accessed 22 Apr. 2019].
5. Towards Data Science. (2019). *How to define a Fitness Function in a Genetic Algorithm?* [online] Available at: <https://towardsdatascience.com/how-to-define-a-fitness-function-in-a-genetic-algorithm-be572b9ea3b4> [Accessed 22 Apr. 2019].
6. Onefivenine.com. (2018). *India. All states, Districts, Villages, Schools, Colleges, Maps, Pin Codes of India*. [online] Available at: <https://www.onefivenine.com/> [Accessed 25 Oct. 2018].
7. En.wikipedia.org. (2018). *Brihanmumbai Electric Supply and Transport*. [online] Available at: https://en.wikipedia.org/wiki/Brihanmumbai_Electric_Supply_and_Transport [Accessed 26 Oct. 2018].

Acknowledgements

It feels great to take this opportunity and write about our journey and the work carried out so far. Our plan was straight i.e. learning as much as we can and contributing towards a project that has great potential with objectives that aim to have a positive impact and a change for the society.

Firstly, we would like to thank **Prof. Mayur Parulekar**, a person who is a constant support throughout the project. Prior to our involvement in this project, the BEST conclave was arranged by him after which the BEST real time data was retrieved, which centers on all the current system implementation that we are presently working with. Along with providing us key points on how to follow up the project, he has given his full support and has shown enthusiasm in the project work. How to get something done with limited leads or in difficult situations for a project at such a level was one of the virtues we learnt from him.

We would like to thank **Angelo Francis**, Final Year Chemical Engineering student (2018 pass out), for his helping attitude and unparalleled interest towards the project. This project has gone through initial phases of surveys and large amounts of data collection and computation of the same; without which we would not be able to get an opportunity to observe the potential that it has to take it to the next level.

We would also like to thank **Jatin Dalvi** and **Chirag Gada**, Third Year Electronics Engineering students, for helping us to make the website, a yet ongoing task which will help to establish the project in a full-fledged product.

Finally, we would like to thank **HOD Prof. Prasad Joshi** for extending his support towards the completion of this project and for the timely encouragement.

(Signature)

(Name of student and SAP no.)

Date:

BE project report

ORIGINALITY REPORT

18%

SIMILARITY INDEX

20%

INTERNET SOURCES

9%

PUBLICATIONS

12%

STUDENT PAPERS

PRIMARY SOURCES

1	en.wikipedia.org Internet Source	7%
2	etd.adm.unipi.it Internet Source	3%
3	Tzafestas, S. G., M.-P. Saltouros, and M. Markaki. "A Tutorial Overview of Genetic Algorithms and Their Applications", World Scientific Series in Robotics and Intelligent Systems, 1999. Publication	2%
4	www.terna.org Internet Source	1%
5	www.digitalsignaturemart.com Internet Source	1%
6	www.dtic.mil Internet Source	1%
7	Submitted to Amity University Student Paper	1%

Submitted to Mahidol University

8

Student Paper

1%

9

personal-injury-lawyer-usa.info

Internet Source

1%

10

Jun Li. "Aerodynamic optimum design of transonic turbine cascades using Genetic Algorithms", Journal of Thermal Science, 06/1997

Publication

1%

11

www.mobiloitte.com

Internet Source

<1%

12

Submitted to University of Mauritius

Student Paper

<1%

13

arun-aiml.blogspot.com

Internet Source

<1%

14

prezi.com

Internet Source

<1%

Exclude quotes On

Exclude matches < 40 words

Exclude bibliography On