I. Introduction
In the metropolitan cities of developing countries like India due to heavy growth of vehicular population specially private and intermediate transport services and limited carriageway capacity has affected mobility of individuals. As a result traffic congestion, reduced travel speed, poor level of service and environmental pollution are prevailing. Under such worst conditions it is required to increase the efficiency of public transport system so that use of intermediate transport be reduced to a lesser degree and private transport is discouraged. Therefore it is required to plan and design urban public transit in the most efficient manner with existing constraints. Usage of mass public transportation in four major metropolitan cities of India is above 70% in which the share of bus transport is dominant. Thus public transportation has become an integral and essential feature of cities especially in developing countries. In populous and developing countries like India, where urbanisation trend is on the increase, mass transit systems like commuter rails and bus transportation are mostly unavoidable.

Among various means of public transportation, the bus transport has dominated because of its door to door accessibility and flexibility in operation. The efficiency of bus transport system depends on routes and schedules. In past various attempts are made to design routes and frequencies. In most of the approaches development of routes and selection of frequencies of buses is done separately to avoid complications and computational burden. In actual practice both go hand in hand. If routing and scheduling go together the routes generated will support the defined schedule. Hasselstrom (1981) and Marwah et al (1984) used a complex two level optimization model, which first reduces the network by eliminating links that are seldom or never used by passengers. A large set of possible routes is then generated from the remaining links. Finally the network routes are selected by assigning frequencies using a linear programming model which maximizes the number of transfers saved by changing from a link network to a public transport network. In the proposed study single optimization model is developed, which finds optimal routes and schedules together and optimizes the objective function, which is the minimisation of user and operator's costs with the real life constraints. Thus from literature review reveals that limited attempts are made for simultaneous design of routes and schedules. In other words the developed routes should support a set of schedules.

2. Model for simultaneous routing and scheduling
In this research work a model is developed for simultaneous routing and scheduling of public buses. The model determines optimal fleet size for optimally designed routes and corresponding set of frequencies of public buses. User cost is taken as total travel time, which includes in-vehicle travel time, waiting time and the transfer time. The operator cost is taken as running time cost of the buses. The objective function is minimisation of summation of both costs. Constraint for load factor is taken for better level of service and to ensure a minimum number of passengers for economical operation. Fleet size constraint is adopted to keep the number of buses under certain limit and link-overloading constraint ensures that there can not be buses over certain specified limit on any link. The mathematical formulation of
the model is as follows.

**Objective Function**

Minimize \( \left\{ C_p \left[ \sum_{i,j} d_{ij} \times t_{ij} \right] \right\} \times C_p \left[ \sum_{i,j} d_{ij} \times t_{at1j} \right] \times C_p \left[ \sum_{i,j} d_{ij} \times t_{at2j} \right] \times C_p \left[ \sum_{i,j} d_{ij} \times t_{at3j} \right] \right\}

Subject to

Load factor constraints:

\[
LF_k = \frac{(Q_k)_{\text{max}}}{f_k \times \text{CAP}} \leq LF_{\text{min}} \quad \text{for all } k \in \text{SR}
\]

\[
LF_k = \frac{(Q_k)_{\text{max}}}{f_k \times \text{CAP}} \geq LF_{\text{max}} \quad \text{for all } k \in \text{SR}
\]

Fleet size constraint:

\[
\sum_{i} N_k \sum_{j} f_{kj} \times T_j \leq W
\]

Link overloading constraint:

\[
f_{ij} \leq F_{\text{max}}
\]

Where,

- \( dj \) = demand between nodes i and j
- \( \text{CAP} \) = Seating Capacity of buses operating on the network’s routes
- \( \text{SR} \) = Set of transit routes
- \( C1 \) = In - vehicle time cost per minute,
- \( C2 \) = Waiting time cost per minute
- \( C3 \) = Transfer time cost per minute
- \( C4 \) = Bus operating cost per minute

**2.1 Steps Involved for Model Development**

Step 1:

Inputs are identified for the model like:

- Number of nodes in the network and their connectivity along with travel time details between nodes.
- Demand matrix for above network.
- Design parameters like bus seating capacity, maximum and minimum load factors, fleet size etc.

Step 2:

Node pairs are identified and sorted with the following criterions:

- As per production & attraction of nodes - The nodes having more than above average production and attraction are selected and paths connecting these nodes are considered for further scrutiny.
- Route Length - From the above the paths having length between 15 minutes to 33 minutes are considered.
- Average Route Flow Values (ARFV) - After route length criteria ARFV a criterion is applied and paths having more than Average Route Flow Value are selected. Thus nodes present on the above paths are selected for further analysis.

Step 3:

K-Shortest Paths are generated between above identified node pairs.

Step 4:

Assignment on Routes - After identifying the direct node pairs each alternate is considered for assigning the load to the link, based on exponential function of frequency and distance. Thus for example as in Fig. 1, if there are three acceptable routes \( R_1, R_2, \) and \( R_3 \) between nodes i and j and frequencies are \( f_1, f_2, f_3 \) and distances \( d_1, d_2, d_3 \) respectively. Load on each route can be calculated as follows:

\[
\text{Route R1 demand} = \left( \frac{e^1}{e^1 + e^2 + e^3} \right) \times \text{dj}
\]

Where

- \( v1 = \{ f \times f_1 \times df \times d \} \)
- \( ff \) = Frequency distribution factor (percentage of trip makers prefer frequency based choice
- \( df \) = Distance distribution factor (percentage of trip makers prefer the shorter length Criterion as compare to frequency.
- \( d_{ij} \) = Demand between node i and j
- \( n \) = Number of route pass between node i and j
- \( f_1, f_2, f_3 \) = Frequency of route \( R_1, R_2, \) and \( R_3 \) respectively
- \( d_1, d_2, d_3 \) = Lengths of route \( R_1, R_2, \) and \( R_3 \) respectively

Thus load is assigned to links on the route. This flow incurs average waiting time (in minutes)

\[
= \left( \frac{60.0}{2.0(f+1+f_2+f_3)} \right)
\]

This means that for any of the alternative the waiting time incurred is the same as the average waiting time. This rule is based on the assumption that arriving trip makers’ board the first bus, which arrives at the stop. The waiting time calculation further ignores stochastic nature in bus headway. Further from all the available alternative routes by one transfer, shortest route is chosen using the distance criteria. The demand (\( d_{ij} \)) is assigned on the links of the path followed by the transfer. The waiting time at transfer point ( \( t_{wlij} \)) is calculated based on frequencies on routes before and after transfer.

Thus total travel time is calculated as

\[
t_{lij} = t_{ij} + t_{atij} + t_{at1j} + t_{at2j} + t_{at3j} + \text{Transfer penalty}
\]

Where,

- \( t_{atij} = \text{In - vehicle time from node i to transfer point through R}_1\) route
Step 5:
Application Of Genetic Algorithm - Genetic Algorithm is applied for different alternatives using random frequencies. Finally a set of optimal routes and schedules are selected.

Step 6:
Computation of number of buses - Required number of buses are calculated using following formula

2.2 Application of Genetic Algorithm for Model Development

2.2.1 Overview of Genetic Algorithm
Genetic Algorithms (GAs) are computerised search and optimization algorithms based upon the principles of Darwinian evolution. The concept of the survival of the fittest is used in a structured, yet randomised information exchanges to form a robust search algorithm. Genetic algorithms efficiently exploit historical information to locate search points with improved performance. The basic idea behind GAs is to generate an initial pool of solutions, represented as string structures, and then thorough continuous copying, swapping, and modifying of partial strings in a manner similar to natural genetic evolution, to allow the solution pool to evolve towards better and better solutions.

Working Principle
The working principle of GAs is illustrated in the form of pseudocode as follows:

begin
Initialise population of strings
Compute fitness of population
Repeat:
    Reproduction,
    Crossover,
    Mutation,
    Compute fitness of population
Until termination criterion is achieved
end

Coding
An important characteristic of genetic algorithms is the coding of variables that describe the problem. The most common coding method is binary coding (Goldberg 1989). The decision variables are usually mapped and represented by a string (chromosome) of binary alphabets (genes). The length of the string is usually determined according to the desired solution accuracy. Once the coding of variables has been done, the corresponding point can be found using a fixed mapping rule, usually the following linear mapping rule is used (Goldberg 1989).

\[ X_i = X_{\text{min}} + \frac{X_{\text{max}} - X_{\text{min}}}{2^r-1} \cdot \text{decoded value (S)} \]

Where,
\( X_{\text{min}} \) = is lower bound on decision variable \( X_i \),
\( X_{\text{max}} \) = is upper bound on decision variable \( X_i \),
The variable \( X_i \) is coded in a substring \( S_i \) of length \( l \).
Decoded value \( (S_i) \) is \( \frac{X_{\text{max}} - X_{\text{min}}}{2^r-1} \) where \( S_i \in (0,1) \) and the string is represented as \( S_{1:i}, S_{1:2}, \ldots, S_{1:3}, S_3 \). When all the decision variables are decoded using above mapping rule, the function value can also be calculated by substituting the variables in the given objective function \( f(x) \). The obtainable accuracy of the variable for a \( l \)-bit coding is

\[ \frac{X_{\text{max}} - X_{\text{min}}}{2^r-1} \]

Fitness function
Since GAs mimic the survival of the fittest principal of nature to make a search process. Therefore, GAs are naturally suitable for solving maximisation problems. Minimisation problems are usually transformed into maximisation problems by some suitable transformation.

After all the values of variables are obtained, they can be used to calculate the objective function value. In general, a fitness function \( F(x) \) is first derived from the objective function and used in successive genetic operations. For maximisation problems, the fitness function can be considered to be the same as objective function i.e. \( F(x) = f(x) \). For minimisation problems the fitness function is an equivalent maximisation problem chosen such that the optimum points remains unchanged. The following fitness function is usually used (Deb 1995).

\[ F(x) = \frac{1}{1 + |f(x)|} \]

This transformation does not alter the location of minima but converts a minimisation problem to an equivalent maximisation problem. The fitness function value of a string is known as the string’s fitness.

The operation of GAs begins with population of random strings representing design of decision variables. Thereafter, each string is evaluated to find the fitness value. Three main operators then operate the population: reproduction, crossover and mutation to create a new population of points. The new population is further iteratively operated by above three operators and evaluated. This procedure is continued and tested for termination. One cycle of these operations and subsequent evaluation procedure is known as a generation.

GA Operators
Reproduction (or selection) is an operator that makes more copies of better strings in a population. The commonly used reproduction operator is the proportionate reproduction operator where a string is selected for mating pool with a probability proportional to its fitness. Thus, the \( i^\text{th} \) string in the population is selected with a probability proportional to \( F_i \). The probability for selecting the \( i^\text{th} \) string is
\[ p_i = \frac{F_i}{\sum F_j} \]

Where \( n \) is population size and \( F_i \) is fitness of \( i^{th} \) string. After the reproduction phase is over, the population is enriched with good strings. Reproduction makes clones of good strings, but does not create any new string. A crossover operator is used to recombine two strings with a hope of creating a better string. In the crossover operator, exchanging information among strings of mating pool creates new strings. In most crossover operators, two strings are picked from the mating pool at random and some portion of the strings is exchanged between the strings. A single-point crossover operator is performed by randomly choosing a crossing site along the string and by exchanging all bits on the right side of the crossing site as shown in Fig. 2.

![Fig. 2.0: crossover operation](image)

The two strings participating in the crossover operation are known as parent strings and the resulting strings are known as children strings. It is intuitive from this construction that good sub strings from parent strings can be combined to form a better child string, if an appropriate site is chosen. Since the knowledge of an appropriate site is usually not known beforehand, a random site is often chosen. Reproduction operator takes this random site selection, because if good strings are created by crossover, there will be more copies of them in the next mating pool, otherwise they will not survive beyond next generation. The total number of strings participating in the mating pool can be controlled by specifying crossover probability, \( p_c \). Another operator, mutation is used sparingly. Under the action of this operator this operator, a 1 will change to a 0 and vice versa with a small probability. Mutation also creates a new string, but its effect is considered secondary. It introduces diversity in the population whenever the population tends to become homogeneous due to iterative use of selection and crossover operators. Furthermore, for local improvement of a solution, mutation may be found useful. Since this operator disrupts a string, the probability of mutation \( p_m \) is kept very low.

Termination Criterion
When the average fitness of all strings in a population is equal to the best fitness, the population is said to have converged. When population is converged, the GA is terminated. The same can be done by fixing maximum number of generations, the number of generations at which population will converge. In GAs maximum number of generations is generally used as the termination criteria.

2.2.2 Implementation of Genetic Algorithm for Simultaneous Routing and Scheduling
Genetic Algorithm is found to be most suitable for this problem due to multi objective nature of objective function, large number of variables, non linearity involved in objective function and in constraints.

Coding:
Routes and the frequencies of each pair are coded into a single string with the desired precision. For example Fig. 3 shows binary digits coding for route no. 5 and route no.3 with frequencies 6 and 21. First four bits show route and last six bits show corresponding frequency in a string.

![Table: Pair no 1 and Pair no 2](image)

Fitness function:
Fitness function is taken as minimisation of objective function i.e. summation of users and operator’s costs. Constraints are considered by imposing penalties for their violation. These penalties are added in objective function and thus penalised objective function is calculated. The value of penalised objective is used for optimisation. Weights to the penalties are given in terms of function of objective function value and as per their relative importance.

Therefore
Fitness function = Minimize (Objective function + penalties 1 to 6 ) Where,
Penalty 1 = penalty for the unsatisfied demand,
Penalty 2 = penalty for the one transfer demand satisfied,
Penalty 3 = penalty for the Minimum load factor,
Penalty 4 = penalty for the Maximum load factor,
Penalty 5 = penalty for the Fleet size,
Penalty 6 = penalty for the link over loading.

The program for fitness is developed in object-oriented environment in C++. Fig. 4 gives flow chart for implementation of GA for our problem.

3 Application of model for case study
The Model developed here is tested on the Mandi’s Swiss network of fifteen nodes. Mandi (1980) originally reported this benchmark network. The total demand is 15570 transit trips. The highest node pair demand being 880 transit trips. In this matrix, 82% of the node pairs have non-zero demand. The same network also used by Baaj et al (1990), Kidwai (1999) and Muralidhar (1999). The results of the proposed algorithm are compared with the previous researcher’s results. The network is small and dense; it comprises only 15 nodes within a 33 minutes shortest travel time between the two farthest nodes. Although this network may not be very representative of many real-world urban bus transit networks, it is still useful possibly as regional sub network. Mandi’s network is based on a real network in Switzerland and the demand matrix shows the
### Table 1: demand matrix for mandi's network

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>400</th>
<th>200</th>
<th>60</th>
<th>80</th>
<th>150</th>
<th>75</th>
<th>75</th>
<th>30</th>
<th>160</th>
<th>30</th>
<th>25</th>
<th>35</th>
<th>0</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>50</td>
<td>120</td>
<td>20</td>
<td>180</td>
<td>90</td>
<td>90</td>
<td>15</td>
<td>130</td>
<td>20</td>
<td>10</td>
<td>10</td>
<td>5</td>
<td>0</td>
</tr>
<tr>
<td>400</td>
<td>0</td>
<td>50</td>
<td>0</td>
<td>40</td>
<td>60</td>
<td>180</td>
<td>90</td>
<td>90</td>
<td>15</td>
<td>45</td>
<td>20</td>
<td>10</td>
<td>10</td>
<td>5</td>
<td>0</td>
</tr>
<tr>
<td>200</td>
<td>50</td>
<td>0</td>
<td>40</td>
<td>60</td>
<td>180</td>
<td>90</td>
<td>90</td>
<td>15</td>
<td>45</td>
<td>20</td>
<td>10</td>
<td>10</td>
<td>5</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>60</td>
<td>120</td>
<td>40</td>
<td>0</td>
<td>50</td>
<td>50</td>
<td>15</td>
<td>130</td>
<td>20</td>
<td>10</td>
<td>10</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>80</td>
<td>20</td>
<td>60</td>
<td>50</td>
<td>0</td>
<td>50</td>
<td>25</td>
<td>25</td>
<td>10</td>
<td>120</td>
<td>20</td>
<td>15</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>150</td>
<td>180</td>
<td>180</td>
<td>100</td>
<td>50</td>
<td>0</td>
<td>100</td>
<td>100</td>
<td>30</td>
<td>880</td>
<td>60</td>
<td>15</td>
<td>15</td>
<td>10</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>75</td>
<td>90</td>
<td>90</td>
<td>50</td>
<td>25</td>
<td>100</td>
<td>0</td>
<td>50</td>
<td>15</td>
<td>440</td>
<td>35</td>
<td>10</td>
<td>10</td>
<td>5</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>75</td>
<td>90</td>
<td>90</td>
<td>50</td>
<td>25</td>
<td>100</td>
<td>50</td>
<td>0</td>
<td>15</td>
<td>440</td>
<td>35</td>
<td>10</td>
<td>10</td>
<td>5</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>15</td>
<td>15</td>
<td>15</td>
<td>10</td>
<td>30</td>
<td>15</td>
<td>15</td>
<td>0</td>
<td>140</td>
<td>20</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>160</td>
<td>130</td>
<td>45</td>
<td>240</td>
<td>120</td>
<td>880</td>
<td>440</td>
<td>440</td>
<td>140</td>
<td>0</td>
<td>600</td>
<td>250</td>
<td>500</td>
<td>200</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>20</td>
<td>20</td>
<td>40</td>
<td>20</td>
<td>60</td>
<td>35</td>
<td>35</td>
<td>20</td>
<td>600</td>
<td>0</td>
<td>75</td>
<td>95</td>
<td>15</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>10</td>
<td>10</td>
<td>25</td>
<td>15</td>
<td>15</td>
<td>10</td>
<td>10</td>
<td>5</td>
<td>250</td>
<td>75</td>
<td>0</td>
<td>70</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>35</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>5</td>
<td>15</td>
<td>10</td>
<td>10</td>
<td>0</td>
<td>0</td>
<td>500</td>
<td>95</td>
<td>70</td>
<td>0</td>
<td>45</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>0</td>
<td>10</td>
<td>5</td>
<td>5</td>
<td>0</td>
<td>200</td>
<td>15</td>
<td>0</td>
<td>45</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

---

**Figure 4: flow chart for developing optimal routes and number of buses on each route using Genetic Algorithm**
average number of passengers per hours. The table 1 shows the demand matrix of the network and figure 5 shows the mandl's network with time taken between each node in minutes.

![Fig. 3: binary digit coding](image)

3.1 Values of different Parameters
For calculating the fitness function all the penalties are assigned to the objective function. The various penalties weights are decided after several runs and observing the effect of the penalty on the search of fitness value for the model. For present case following penalties are considered.

- Unsatisfied demand = 0.8 times of objective function per trip
- One transfer = 0.7 times of objective function per trip
- More than one transfer = 0.8 times of objective function per trip
- Over loaded link constraints = 0.01 times of objective function per link
- Load Factor MAX constraints = 0.6 times of objective function per route
- Load Factor MIN constraints = 0.6 times of objective function per route
- Penalty for the Fleet Size constraints = 0.1 times of objective function.

Cost of in-vehicle time taken as Rs 0.22 per minute, cost of transfer Rs. 0.13 per minute and the cost of waiting time is adopted as Rs. 0.13 per minute (Draft Report of MMPG on Mumbai Metro Study, 1997). Running time cost of BEST buses is taken as Rs. 6.8 per minute (Monthly statistical review part-2 of August, 1999). An average speed of the bus 15 km/hr. is adopted for calculation.

4 Results and discussions
4.1 Selection of appropriate value of GA Parameters
The various genetic parameters are fixed for the fitness function by observing their effect on fitness value. Population size 40, Cross Over Probability 0.95, Mutation Probability 0.1 give better fitness values. Refer figs 6, 7 and 8. It is observed that considering above parameters the GA converges within 100 generation refer Fig. 9. However maximum generation was allowed up to 500 for the present study.

![Fig. 6: effect of population size on fitness value](image)
![Fig. 7: effect of crossover probability on fitness value](image)
![Fig. 8: effect of mutation probability on fitness value](image)
![Fig. 9: effect of various seeds on generation number and fitness value](image)
4.2 Comparison of Results

The results obtained by the model compared with the Mandl and Baaj (1980) solution as the same network is used for analysis. Developed Routes detail by Mandl and Baaj and from proposed model is given as follows. Table 2 shows that direct demand satisfied in all the cases are higher than the Mandl and Baaj’s solution. The reason is that the previous one is heuristic approach where as in the present study the Genetic Algorithm is used to get the optimal solutions. It

<table>
<thead>
<tr>
<th>Set of routes</th>
<th>%Demand satisfied through transfer</th>
<th>In-veh. Travel Time (min)</th>
<th>Transfer Time (min)</th>
<th>Waiting Time (min)</th>
<th>Total Travel Time (min)</th>
<th>Number Of Buses</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Mandl I</td>
<td>Zero 68.78 One 31.22 Two 0.0 Un</td>
<td>211210</td>
<td>24330</td>
<td>14672</td>
<td>250212</td>
<td>128</td>
</tr>
<tr>
<td>2. Mandl II</td>
<td>Zero 69.85 One 29.97 Two 0.17</td>
<td>177752</td>
<td>23590</td>
<td>17598</td>
<td>218940</td>
<td>111</td>
</tr>
<tr>
<td>3. Baaj I</td>
<td>Zero 78.3 One 21.69 Two 0.0 Un</td>
<td>169138</td>
<td>16970</td>
<td>20007</td>
<td>206115</td>
<td>103</td>
</tr>
<tr>
<td>4. Baaj II</td>
<td>Zero 79.65 One 20.35 Two 0.0 Un</td>
<td>166461</td>
<td>15915</td>
<td>26675</td>
<td>209051</td>
<td>93</td>
</tr>
<tr>
<td>5. Baaj III</td>
<td>Zero 80.88 One 19.12 Two 0.0 Un</td>
<td>180620</td>
<td>14910</td>
<td>22156</td>
<td>217686</td>
<td>100</td>
</tr>
<tr>
<td>6. Mandl II</td>
<td>Zero 72.95 One 26.91 Two 0.13</td>
<td>177006</td>
<td>21070</td>
<td>21070</td>
<td>216469</td>
<td>114</td>
</tr>
<tr>
<td>7. Baaj I</td>
<td>Zero 77.92 One 19.62 Two 2.3</td>
<td>166167</td>
<td>18690</td>
<td>18690</td>
<td>205220</td>
<td>110</td>
</tr>
<tr>
<td>8. Baaj II</td>
<td>Zero 85.67 One 14.32 Two 0.0 Un</td>
<td>161540</td>
<td>11150</td>
<td>11940</td>
<td>199000</td>
<td>92</td>
</tr>
<tr>
<td>9. Baaj III</td>
<td>Zero 85.54 One 14.45 Two 0.0 Un</td>
<td>162680</td>
<td>11250</td>
<td>12430</td>
<td>199911</td>
<td>84</td>
</tr>
</tbody>
</table>

**RESULTS OF PROPOSED MODEL**

| 10. ARFV 300** | 93.38 | 6.62 | 0.0 | 0.0 | 160521 | 5150 | 18154 | 183825 | 75 |
| 13. ARFV 400** | 89.92 | 10.08 | 0.0 | 0.0 | 173394 | 7850 | 16446 | 197690 | 88 |
| 14. ARFV 500** | 88.89 | 11.11 | 0.0 | 0.0 | 173616 | 8650 | 21049 | 203315 | 88 |

**Capacity of buses is 50
ARFV : Average Route Flow Value**
shows that Genetic Algorithm gives the better results moreover the developed model find optimal routes and schedules simultaneously in contrast to previous approaches. It is also evident that the total travel time and the number of buses are considerably reduced. Number of buses required on each route for different ARFV is given in table 3. It is observed that increasing in ARFV the total travel time also increases as shown in table 4.

<table>
<thead>
<tr>
<th>ARFV</th>
<th>Route No.</th>
<th>LENGTH (Min.)</th>
<th>TRIPS</th>
<th>MAXIMUM LINK FLOW</th>
<th>LOAD FACTOR</th>
<th>BUSSES ON ROUTE</th>
<th>TOTAL BUSES</th>
</tr>
</thead>
<tbody>
<tr>
<td>300</td>
<td>1</td>
<td>33</td>
<td>1530</td>
<td>500</td>
<td>1.11</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>30</td>
<td>2431</td>
<td>658</td>
<td>1.2</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>28</td>
<td>2442</td>
<td>613</td>
<td>1.12</td>
<td>11</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>27</td>
<td>2370</td>
<td>655</td>
<td>1.19</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>19</td>
<td>3938</td>
<td>1072</td>
<td>0.98</td>
<td>14</td>
<td></td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>25</td>
<td>1311</td>
<td>422</td>
<td>0.94</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>24</td>
<td>1548</td>
<td>604</td>
<td>1.00</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>400</td>
<td>1</td>
<td>33</td>
<td>3038</td>
<td>853</td>
<td>0.95</td>
<td>19</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>30</td>
<td>5145</td>
<td>1362</td>
<td>1.18</td>
<td>23</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>28</td>
<td>3175</td>
<td>994</td>
<td>1.1</td>
<td>16</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>27</td>
<td>3252</td>
<td>968</td>
<td>0.74</td>
<td>23</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>25</td>
<td>915</td>
<td>517</td>
<td>1.15</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>500</td>
<td>1</td>
<td>33</td>
<td>7138</td>
<td>1320</td>
<td>1.2</td>
<td>28</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>28</td>
<td>2688</td>
<td>795</td>
<td>0.88</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>29</td>
<td>585</td>
<td>189</td>
<td>0.75</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>22</td>
<td>3751</td>
<td>1132</td>
<td>0.87</td>
<td>24</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>25</td>
<td>1408</td>
<td>517</td>
<td>1.15</td>
<td>10</td>
<td></td>
</tr>
</tbody>
</table>

Table 3: scheduling of buses with different arfv.

<table>
<thead>
<tr>
<th>ARFV</th>
<th>Total travel time per person in minutes</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARFV 300</td>
<td>11.80</td>
</tr>
<tr>
<td>ARFV 400</td>
<td>12.70</td>
</tr>
<tr>
<td>ARFV 500</td>
<td>13.05</td>
</tr>
</tbody>
</table>

Table 4: relation of total travel time per person with ARFV.
4.2.1 Comparisons of routing results

Mandl’s Solution:
(a) Initial set of routes (before improvement)
   RI: 10 - 12 - 13 - 9 - 7 - 14 - 5 - 2 - 1 - 0
   RII: 6 - 14 - 5 - 3 - 4
   RIII: 11 - 3 - 5 - 14 - 8

(b) Final set of routes (after improvement)
   RI: 0 - 1 - 2 - 5 - 7 - 9 - 10
   RII: 4 - 3 - 5 - 7 - 14 - 6
   RIII: 11 - 3 - 5 - 14 - 8
   RV: 12 - 13 - 9

Baaj’s Solution:
(a) First set of routes generated by Baaj
   RI: 6 - 14 - 7 - 9 - 10 - 11
   RII: 6 - 14 - 5 - 7 - 9 - 13 - 12
   RIII: 0 - 1 - 2 - 5 - 7
   RV: 8 - 14 - 6 - 9
   RV: 4 - 3 - 5 - 7 - 9
   RV: 0 - 1 - 2 - 5 - 14 - 8
   Minimum Demand Satisfied directly = 50%

(b) Second set of routes generated by Baaj
   RI: 0 - 1 - 3 - 11 - 10 - 13
   RII: 2 - 5 - 7 - 14 - 6 - 9
   RIII: 9 - 10 - 12
   RV: 9 - 10 - 11
   RV: 7 - 9 - 13
   RV: 0 - 1 - 3 - 5 - 7
   RVII: 8 - 14 - 5 - 7 - 9
   RVIII: 4 - 1 - 2 - 5 - 14 - 6 - 9
   Minimum Demand Satisfied directly = 50%

(C) Third routes generated by Baaj
   RI: 9 - 12
   RII: 9 - 10 - 11
   RIII: 9 - 13
   RV: 0 - 1 - 2 - 5 - 7 - 9
   RV: 8 - 14 - 6 - 9
   RV: 4 - 3 - 5 - 7 - 9
   RVII: 0 - 1 - 3 - 5 - 4
   Minimum Demand Satisfied directly = 70%

Proposed algorithm solution
(a) First set of routes for ARFV (Average Route Flow Value) of 300
   RI: 0 - 1 - 2 - 5 - 7 - 9 - 13 - 12
   RII: 0 - 1 - 3 - 5 - 7 - 9 - 10
   RIII: 4 - 3 - 5 - 14 - 6 - 9

RIV: 8 - 14 - 6 - 9 - 10
RV: 0 - 1 - 2 - 5 - 7 - 14 - 6
RVI: 9 - 13 - 12 - 10 - 11
(b) Second set of routes for ARFV of 400
   RI: 0 - 1 - 2 - 5 - 7 - 9 - 13 - 12
   RII: 0 - 1 - 5 - 14 - 6 - 9 - 10
   RIII: 4 - 3 - 5 - 7 - 14 - 6 - 9 - 10
   RIV: 8 - 14 - 5 - 7 - 9 - 10
   RV: 9 - 13 - 12 - 10 - 11
   (b) Third set of routes for ARFV of 500
   RI: 0 - 1 - 2 - 5 - 7 - 9 - 13 - 12
   RII: 4 - 3 - 5 - 7 - 14 - 6 - 9 - 10
   RIII: 8 - 14 - 5 - 7 - 9 - 10
   RIV: 9 - 13 - 12 - 10 - 11

5 Conclusions
The results show improvements over previous researcher works for the same network problem and demand matrix. It is also observed that Genetic Algorithm gives reasonably good values even with lesser pool size. However if pool size is increased results can be refined but it increases the computational time. Selection of ARFV value is affecting the total travel time per person. As it is observed that higher the value of ARFV higher the total travel time per person. However it is to be noted that time taken for solving the problem is not compared for the given network. For smaller network as considered in study there may not be much difference but computational burden increases with larger networks so computational time will also increase. The results have proved that simultaneous routing and scheduling using Genetic Algorithm for optimization has better edge over other existing approaches for routing and scheduling problems specially in the domain of Public Transportation.

REFERENCES


