Engineering Optimization, 2002, Vol. 34(1), pp. 83–100

Taylor & Francis
Taylor & Francis Group

OPTIMAL ROUTE NETWORK DESIGN FOR TRANSIT SYSTEMS USING GENETIC ALGORITHMS

PARTHA CHAKROBORTY*† and TATHAGAT DWIVEDI

Department of Civil Engineering, Indian Institute of Technology, Kanpur – 208 016, India

(Received 14 November 2000)

The paper proposes a technique for the development of "optimal" transit route networks (for example, a bus route network) given the information on link travel times and transit demand. The proposed method, unlike previous techniques, primarily uses optimization tools for the development of the transit route network — the reliance on heuristics is minimal. In the proposed method, genetic algorithms, an evolutionary optimization technique, is used to develop the "optimal" set of routes. Results show that the proposed procedure performs better than the existing techniques.

Keywords: Optimal routing; Public transportation; Genetic algorithms

1 INTRODUCTION

In recent times it has been realized that a sustainable transportation system must include a strong public transportation system (or transit system) — building more roads for the ever increasing number of private transportation vehicles (automobiles) is not the solution. However, for a public transportation system to be successful in attracting passengers, the system must be efficient. A transit system (which typically consists of many transit routes, collectively referred to as the transit route network or the transit route set) will be efficient if the transit route network and the schedules of the routes are efficient. Hence, it is imperative that procedures for efficient route set and schedule design are available in order to achieve efficient transit systems.

In this paper an attempt is made to develop a procedure for designing efficient transit routes forming a transit route network (or route set) for a given road network. An efficient transit route network may be thought of as a set of routes which is "optimal" or "near-optimal" with respect to
certain criteria. These criteria together with a detailed description of the problem are presented in the next section. The second section also reviews past work done in the area of transit route network design.

* Corresponding author. E-mail: partha@iitk.ac.in

ISSN 0305-215X print; ISSN 1029-0273 online © 2002 Taylor & Francis Ltd
DOI: 10.1080/03052150290004649

The third section of the paper discusses the proposed optimization procedure which evolves "optimal" or "near-optimal" route sets from some starting route sets using the principles of genetic algorithms. Section four presents route sets obtained using the proposed methodology for a benchmark Swiss network initially used by Mandl [14] and later by Baaj and Mahmassani [1] and Kidwai [12]. The results show that the proposed methodology produces better route sets than the available methods. The fifth section concludes the paper with a discussion on the present work and scope for future extensions.

2 PROBLEM STATEMENT, LITERATURE REVIEW AND MOTIVATION

The paper attempts to develop an algorithm which will provide an "optimal" or "efficient" route network for transit system (for example, a bus system) operation on a road network given the following information: (i) the road network data (*i.e.* how the nodes of the network are connected), (ii) link travel time data (*i.e.* data on the travel time on road sections directly connecting any two nodes), and (iii) the transit demand matrix giving
demand between all pairs of nodes in the network in terms of number of trips per unit time (normally a day). In general, an efficient transit route
network (or route set) is supposed to have the following properties:

(i) The route network (given the maximum number of routes) should satisfy most, if not all, of the existing transit demand;
(ii) The route network should satisfy most of the demand without requiring passengers to transfer from one route to another;
(iii) The route network should offer low travel time (including the time spent by passengers in transferring) to its passengers.

This problem of designing an efficient route network for a transit system is a difficult optimization problem which does not lend itself readily to mathematical programming formulations and solutions using traditional techniques. Newell [16] observes that designing an efficient route network " . . . is generally a non-convex (even concave) optimization problem for which no simple procedure exists short of direct comparisons of the various local minima." Similar observations are also made by Baaj and Mahmassani [1]. Further, it must be realized here that the present problem is a discrete, NP-hard, combinatorial problem [1] with a difficult-to-calculate objective function — features which pose almost unsurmountable difficulties in obtaining a solution through traditional optimization techniques.

These reasons, perhaps, have limited the solution of this problem to either using heuristic algorithms or analytical techniques which optimize only parameters like route spacing, route length, etc. for simplistic idealized networks. The analytical techniques (for example, Holroyd [11], Byrne and Vuchic [3], and Byrne [4]), as also observed by Ceder and Wilson [5] and van Nes *et al.* [19], cannot be used for designing actual routes on any given road network. As mentioned earlier, most of the other studies in the area of route design like Lampkins and Saalmans [13], Rosello [17], Mandl [14, 15], Dubois *et al.* [9], Ceder and Wilson [5], Baaj and Mahmassani [1, 2] basically propose heuristic algorithms at various levels of sophistication. Recently, Kidwai [12] has made an attempt to use an optimization tool for solving the problem. However, a close look at the procedure proposed by Kidwai [12] shows that it still remains primarily a heuristic algorithm with use of the optimization tool only marginal.

The lack of an optimization procedure for the design of efficient route networks motivated the authors to look at the problem again and attempt to develop such a procedure. The diffi-

culty of solving the problem through traditional optimization techniques led the authors to use the non-traditional optimization technique of Genetic Algorithms (GA). The choice of GA, in particular, is due to the fact that GA perform well in solving large, discrete, combinatorial optimization problems with difficult-to-calculate objective functions (for example, see Chakroborty *et al.* [6, 7] and Deb and Chakroborty [8]).

The use of GA allows the development of a route design procedure which, given a group of initial route networks (*i.e.* a group of initial route sets), evolves better and better route networks in successive iterations till an "optimal" or "efficient" route network is obtained. The detailed description of the proposed procedure is provided in the next section.

3 PROPOSED METHODOLOGY

This section presents the principles and the details of the proposed methodology of route design. The purpose of the proposed methodology is to determine an "optimal" or "efficient" route set (comprising of a pre-specified (or fixed) number of routes) for a given road network and transit demand matrix. The properties of an efficient route set are as enunciated in Section 2.

The proposed methodology, the conceptual overview of which is shown in Figure 1, follows a three step iterative process. The important features of the proposed procedure are listed here.

- First, various reasonable route sets for the given road network and demand matrix, are determined using a novel stochastic procedure, IRSG. The procedure is described in detail in Section 3.1. It may be pointed out here that this heuristic procedure, which only

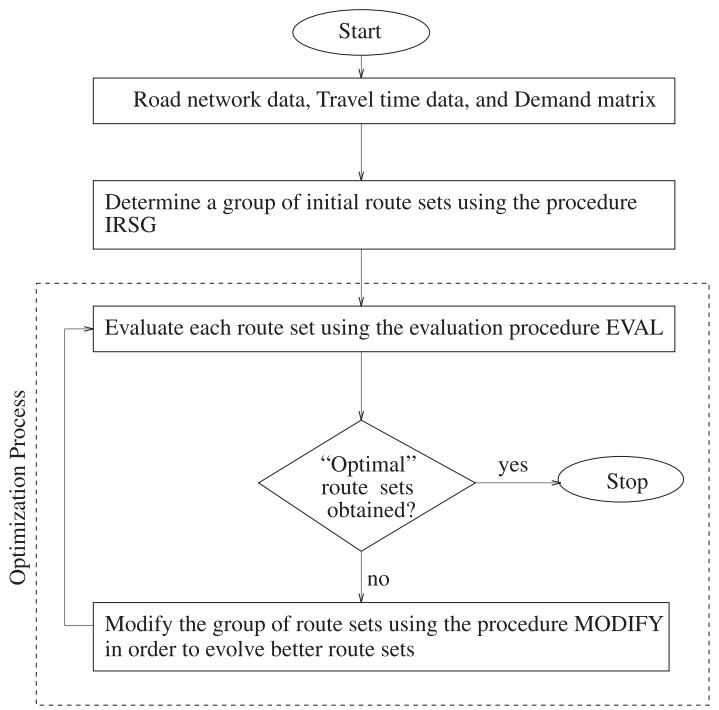


FIGURE 1 Flowchart giving an overall view of the proposed algorithm.

provides a starting point for the proposed optimization procedure, is also developed as a part of this study.

- After the initial one-time use (See Figure 1) of IRSG, the evaluation step, and the route modification step are executed repeatedly one after the other until a route set is obtained which satisfies the basic properties of an efficient route set maximally; *i.e.* until an "efficient" or "optimal" route set obtained.
- The goodness of a route set as a whole is determined in the evaluation step using the evaluation scheme, EVAL. The evaluation procedure is described in detail in Section 3.2.
- Given a group of route sets and their goodness, the route sets are modified using the proposed modification procedure, MODIFY, in order to obtain a better group of route sets. The modification is done using the evolutionary principles of genetic algorithms. Although guided by the concepts of genetic algorithms, MODIFY uses operators which are typical to the proposed procedure. Section 3.3 describes MODIFY in details.

It may be pointed out here that use of the principles of genetic algorithms implies that in the proposed procedure one has to work with a multitude of possible "solutions." Note that a "solution" in the present case means an entire route set and not a single route. Thus, in the proposed procedure, more than one initial route set is developed using IRSG and the MODIFY algorithm launches itself on this group of initial route sets in order to obtain "optimal" or "efficient" route sets (or "solutions").

3.1 Initial Route Set Generation Procedure, IRSG

Ideally, an optimization algorithm should be able to find optimal route sets irrespective of the initial set of routes. However, determination of good and efficient route sets is a difficult optimization problem, as highlighted in Section 2, and hence in order to make the proposed procedure efficient it is important that the initial route sets on which the optimization procedure launches itself are not arbitrary and are obtained using simple yet logical guidelines.

Here, each initial route set (consisting of a pre-specified number of routes, say, R) is obtained by determining each of the R routes. Each route is determined by first selecting the starting node (referred to as the *first node*) and then selecting all the other nodes sequentially till some termination criterion is satisfied. The underlying principles which motivate the selection procedure of the nodes are: (i) the basic properties of good or "efficient" routes stated earlier, and (ii) computational simplicity. The latter principle is important because unnecessary complications in the heuristics used here will reduce the computational efficiency of the overall procedure and also because in the proposed procedure the only requirement from the IRSG is that the initial route sets generated should not be completely arbitrary. The process of selecting the first node and all other nodes of a route are described next.

Procedure to select the first node of a route

1. Determine the activity level, $\alpha_i = A(i) + P(i)$, for each node, i, in the network. Here, $A(i)$ and $P(i)$ are the number of transit trips culminating in node i and originating from node i, respectively.
2. Arrange the nodes in a descending order of activity levels.
3. Select the first K nodes from the ordered list of nodes (obtained from Step 2) to form the Initial Node Set, INS. The value of K is a user specified parameter.

4. Based on the activity level of a node j in the INS, assign a probability, $p_j^{(I)}$ to node j. This probability, $p_j^{(I)}$, is the probability of selecting node j as the first node of a route; $p_j^{(I)}$ is obtained as follows:

$$p_j^{(I)} = \frac{\alpha_j}{\sum_{\forall j \in INS} \alpha_j} \tag{1}$$

5. Obtain the first node of a route by randomly selecting a node from INS using the probabilities $p_j^{(I)}$.

Once a node is selected it is removed from INS and the $p_j^{(I)}$'s are recalculated using Eq. (1) before selecting the first node of another route in the same route set. If, however, the cardinality of INS falls below a certain number then, before selecting the first node of another route in the same route set, more nodes from the previously unselected nodes of the ordered list are included in the set INS and the probabilities recalculated.

Procedure to select any other node of a route

In the following step-by-step description of the procedure the phrase "previous node (PN)" means the last node which has been included in the route. For example, when selecting the second node of the route, PN is the first node of the route.

1. For a given PN determine the acceptable nodes in the vicinity of PN. These nodes are placed in a set called the Vicinity Node Set (VNS).

   Determination of the vicinity nodes requires two checks to be made. First, a node will qualify as a vicinity node of PN if a single (road) link joins PN to that node. Second, a node will qualify as an acceptable vicinity node if that node has not been already included in the route being currently generated.
2. Based on the activity level of a node k in the VNS, assign a probability $p_k^{(N)}$. The probability, $p_k^{(N)}$, is the probability of selecting node k as the next node; $p_k^{(N)}$, is obtained as follows:

$$p_k^{(N)} = \frac{d_k \alpha_k}{\sum_{\forall k \in VNS} d_k \alpha_k} \tag{2}$$

   where α_k is the activity level of node k and d_k is a user specified biasing term for the selection of node k. This term may be used in order to avoid cycles and frequent backtracking in the initial route. For example, if the choice of a node from the VNS implies reversal of the current predominant direction of the route then the biasing term for that node can be kept small so that its probability of selection as the next node becomes small.
3. Obtain the next node of the route by randomly selecting a node from the VNS using the probabilities $p_k^{(N)}$.

Termination of route and route set generation

A route is generated, as discussed earlier, by selecting the first node for the route and then selecting the other nodes for the route. The process of generating a route continues till at least one of the following criteria is met: (i) the number of nodes in the route equals a pre-specified maximum number of nodes, M, or (ii) the route length (either in travel time or travel distance units) reaches a pre-specified maximum route length, L, or (iii) the cardinality of the VNS (for the particular PN) falls to zero.

Generation of different routes in a route set is continued till the number of routes generated for the route set equals R, the pre-specified number of routes in a route set.

3.2 Evaluation Procedure, EVAL

Evaluation of a route in isolation is meaningless since the design of one route depends, as it should, on the other routes in the network. Hence, all routes in the network should be evaluated as a whole; that is, the entire route set must be evaluated and not a single route. This section discusses the procedure proposed here to evaluate a route set.

The evaluation is done using a set of criteria which gives a measure of the goodness of a route set. These criteria are:

1. The average (or total) in-vehicle travel time (including transfer time) experienced by each (or all) of the users of the transit network.
2. The percentage of users who can go directly (*i.e.* without a single transfer) from their origin to their destination.
3. The percentage of users who can go from their origin to their destination by making a single transfer.
4. The percentage of users who can go from their origin to their destination by transferring twice.
5. The percentage of users who cannot use the transit network to go from their origin to their destination. It is assumed here that no user will make more than two transfers to travel from their origin to their destination and hence such users will simply not use the transit network.

Although, the criteria stated are reasonably standard (for example, similar criteria are used by Mandl [14], and Baaj and Mahmassani [1] among others) the way in which they have been used here to evaluate a route set is novel. In the proposed procedure, for every route set, r, a single number, $TOTFIT(r)$, indicating the route set's goodness (or fitness) is assigned to the route set. The value $TOTFIT(r)$ is obtained as follows:

$$TOTFIT(r) = \omega_1 F_1(r) + \omega_2 F_2(r) + \omega_3 F_3(r) \tag{3}$$

where, $F_1(r)$ is a score obtained by evaluating the route set, r, using the first criterion only; $F_2(r)$ is a score obtained by evaluating the route set, r, using the second, third, and fourth criteria; and $F_3(r)$ is a score obtained by evaluating the route set, r, using the fifth criterion. ω_1, ω_2, and ω_3 are user specified weights for $F_1(r)$, $F_2(r)$, and $F_3(r)$, respectively.

In the following three sections the procedures for obtaining $F_1(r)$, $F_2(r)$, and $F_3(r)$ are described in detail.

Procedure for obtaining $F_1(r)$

The purpose of $F_1(r)$ is to determine a score which reflects the average in-vehicle travel times of users. Certainly, it should give a low score if the time is "large" and a high score if the time is "small." This means that one not only has to evaluate what the average in-vehicle travel time is but also has to determine whether that should be considered as "large" or "small" before assigning a score. The following steps are followed to obtain the score $F_1(r)$.

Step 1 For every node pair, $\{i, j\}$, and a given route set, r, the in-vehicle travel time $IVT_{i,j}(r)$ is computed by determining the smallest time in which a user can go from i to j using the routes of the rth transit route set. The following points should be noted here: (i) the travel

time, $T_p(r)$ on a path, p, is obtained using $T_p(r) = \sum_{\forall a \in p} t_a + nU$, where t_a is the travel time on link a, n is the number of transfers involved in path p and U is the time penalty (used as a surrogate for transfer time which is, in reality, dependent on the exact schedule) for each transfer; (ii) the smallest travel time path, p^*, is such that $T_{p^*}(r) = \min_p T_p(r)$; and (iii) $IVT_{i,j}(r) = T_{p^*}(r)$.

Step 2 The time calculated in Step 1 is obviously dependent on the routes. However, one could also determine the absolute minimum travel time between any node pair $\{i, j\}$, which is only dependent on the road network and not on the transit route network. This absolute minimum travel time, $T^{\min}_{i,j}$, is calculated using Floyd's minimum path algorithm (see Ref. [18] for a good description of the algorithm) in this work.

Step 3 $T^{\min}_{i,j}$ is used as a benchmark against which the $IVT_{i,j}(r)$ offered by the particular route set, r, is compared, in order to determine the index $f_{i,j}(r)$. This index indicates whether the $IVT_{i,j}(r)$ is "large," "small," etc. The index can be obtained using any monotonic decreasing function of $IVT_{i,j}(r) - T^{\min}_{i,j}$; the form proposed here is:

$$f_{i,j}(r) = \begin{cases} -\left(\dfrac{\beta_1}{x_m} + \dfrac{K_1}{x_m^2}\right)x^2 + \beta_1 x + K_1 & 0 \le x \le x_m \\ 0 & x > x_m \end{cases} \tag{4}$$

where, $x = IVT_{i,j}(r) - T^{\min}_{i,j}$, x_m is an assumed acceptable upper limit of x, K_1 is a user-defined positive parameter indicating the maximum value that $f_{i,j}(r)$ can take, and $-K_1/x_m \le \beta_1 \le 0$. The index is then used to calculate a combined score $F_1(r)$ thus:

$$F_1(r) = \frac{\sum_{\forall i,j \in S(r)} d_{i,j} f_{i,j}(r)}{\sum_{\forall (i,j) \in S(r)} d_{i,j}} \tag{5}$$

where, $S(r)$ is the set of $\{i, j\}$ pairs for which the demand, $d_{i,j}$, can be satisfied by the route set, r.

Procedure for obtaining $F_2(r)$

$F_2(r)$ is a score which depends on the total demand satisfied by a route set either directly or through one or two transfers. $F_2(r)$ is determined through a monotonically increasing scaling function defined as follows:

$$F_2(r) = \frac{K_2 - \beta_2 a}{a^2} d_T^2(r) + \beta_2 d_T(r) \tag{6}$$

where K_2 is a user-defined positive parameter indicating the maximum value of $F_2(r)$, $K_2/a \le \beta_2 \le 2K_2/a$, a is the maximum value which $d_T(r)$ can take, and $d_T(r)$ is a weighted sum obtained as:

$$d_T(r) = a d_0(r) + b d_1(r) + c d_2(r) \tag{7}$$

where, $d_0(r)$ is the fraction of the total transit demand which is satisfied directly by the route set, r, $d_1(r)$ is the fraction satisfied by one transfer, and $d_2(r)$ is the fraction satisfied by two transfers; and a, b, and c are user specified weights satisfying the inequality, $a \ge b \ge c$. It should be noted here that through the weights one could specify the relative importance of the ways in which the demand should be satisfied in a good route set. For example, keeping

a much larger than the other two weights would imply that a good route set is one which satisfies demand by providing direct connections.

Procedure for obtaining $F_3(r)$

$F_3(r)$ is a score which depends on the demand which is unsatisfied by route set, r. It is obtained using a monotonically decreasing scaling function as follows:

$$F_3(r) = -(\beta_3 + K_3)d_{un}^2(r) + \beta_3 d_{un}(r) + K_3 \tag{8}$$

where, K_3 is a user-defined positive parameter indicating the maximum value of $F_3(r)$, $-K_3 \leq \beta_3 \leq 0$, and $d_{un}(r)$ is the fraction of total transit demand which is unsatisfied.

3.3 Modification Procedure, MODIFY

In this section, the proposed modification procedure, MODIFY, which forms the kernel of the optimization process, is discussed. MODIFY, is based on the principles of genetic algorithms (GA), *i.e.* MODIFY is an evolutionary procedure in which good route sets of a given iteration are modified to obtain better route sets in the next iteration. A separate introduction to GA is avoided here for the sake of brevity and the uninitiated reader may refer to Goldberg [10] for an excellent introduction to GA. Here, only the GA terminology (used in the ensuing text) is explained briefly.

GA work in an iterative manner, making improvements in each successive iteration. In GA, an iteration is referred to as a **generation**. Further, in every generation, GA work with a multitude of "solutions" referred to as the **population**. Each "solution" in GA is represented as a **string**. Improvements, from one generation to the next, are obtained using a set of three operators, namely, **crossover, mutation**, and **reproduction**. Each of these operators serve a very definite purpose.

The crossover operator exchanges features (somewhat randomly) between two "solutions" (referred to as **parents**) and creates two new solutions (referred to as **offsprings** or **children**). The purpose here is to try and obtain better "solutions" by mixing the good features of two existing "solutions." As is apparent, however, this mixing may sometime produce offsprings which are worse than their parents. One may view this operation as a search operation.

The mutation operator tries to maintain diversity in the population by randomly changing some of the features in some of the "solutions." This may sometime create a "mutated solution" which is much better than the original "un-mutated solution." This operator is used sparingly as excessive use of this operation will create instability in the convergence of the evolutionary process.

The reproduction operator may be thought of as an operator which provides direction to the evolutionary optimization process. The operator basically chooses some of the "solutions," selects the better amongst them, makes multiple copies of these "solutions," and places these copies in the population of the next generation. Since, in GA, the population size is generally kept constant from one generation to the next, making multiple copies of the good "solutions" of the present generation implies discarding the bad ones of the present generation.

Having briefly described the GA terminology, the rest of the discussion concentrates on the proposed procedure, MODIFY. The discussion proceeds as follows: first, the term "solution" in the present context and its string representation are described, next, an overview of MODIFY is presented, and finally the GA operators (which are typical to the proposed procedure) are described.

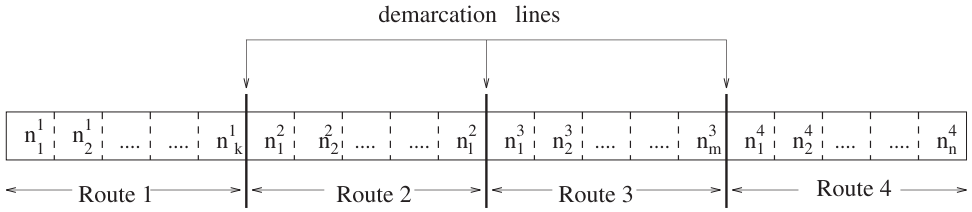


FIGURE 2 String representation of a route set ("solution").

"Solution" and its string representation

In the present context of route and route set design, a "solution" means a description of all the routes in a particular route set; and a description of a route means a statement of the nodes (of the road network) which compose the route. In the MODIFY procedure a "solution" is represented as a string of nodes with demarcation lines indicating the end of a route and the start of another route. Figure 2 shows a typical "solution" represented as a string. The example shown in the figure is for a route set containing four routes. The first route starts from node n_1^1 then goes to node n_2^1, and so on; the route terminates with node n_k^1. Similarly, the second route starts from node n_1^2 and terminates in node n_l^2, and so on for the third and fourth routes. Note that the number of nodes in any route is not an externally controlled factor (except in the initial route sets) but evolves with time. Hence the positions of the demarcation lines and the string lengths of the "solution" change from one generation to the next.

Overview of MODIFY

Figure 3 gives an overview of the entire MODIFY procedure. Note that the processes indicated outside the big dotted box are not a part of MODIFY as indicated in the figure. They are, however, included in the diagram for the sake of completeness. The figure shows how crossover, mutation and reproduction operators work sequentially to evolve the route sets of the next generation from the route sets of the present generation. Further, each generation is assumed to have a population size (*i.e.* the number of route sets or strings) of N. The next three sections describe each of the operators in greater detail.

Crossover operator

As was stated earlier, the basic purpose of crossover is to exchange different features of good strings with the hope of obtaining better strings. In the present scenario, features of strings (route sets) are nothing but routes and route sections. Hence, the proposed crossover operator aims at (i) exchanging routes from two different parent route sets to form two new offspring route sets, and (ii) exchanging sections of routes from two different routes of the same parent route set to form two new offspring routes (note that this exchange of route sections also changes the character of the route set and hence could be thought of as creating an offspring route set from one parent route set). The first type of exchange is termed "inter-string crossover," and the second type of exchange is called "intra-string crossover."

Figures 4 and 5 show the two types of crossover operators used here. Figure 4 shows a case where the ith and the $(i + 1)$th strings (route sets) are chosen from the present population as parent strings for inter-string crossover. The first parent has routes 1, 2, 3 and 4, while the second parent has routes A, B, C and D. After crossover, the first offspring has routes 1,

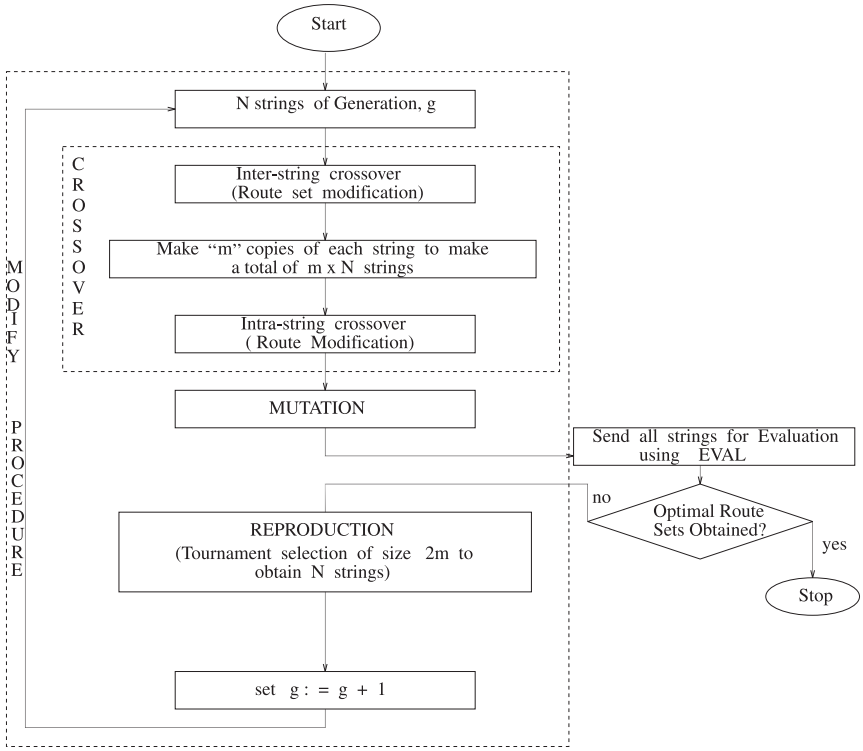


FIGURE 3 Overview of the MODIFY procedure.

2, C and D, and the second offspring has routes 3, 4, A and B; *i.e.* the offsprings are formed by exchanging routes 3 and 4, and routes C and D between the parents. Two points are to be noted here: (i) the choice of parent strings are random; further, the choice of a string as a parent is guided by a user-defined probability value referred to as the *inter-string crossover probability* and (ii) the choice of the crossover site (in this case any of the demarcation lines in the string) is also random; further, it is assumed that each of the demarcation lines are equally likely to become a crossover site.

Figure 5 shows a schematic of the intra-string crossover operator. In this operation two routes from the same route set are chosen at random (in the figure routes 1 and 3 are chosen). If feasible crossover sites exist then sections of the routes are exchanged to form two new routes (as shown in the figure) otherwise a new pair of routes is selected for intra-string crossover. A few points are to be noted here: (i) a feasible crossover site exists if the two routes chosen for crossover have one or more common nodes (like node A in the figure), (ii) if more than one feasible crossover site exists (*i.e.* the number of common nodes is greater than one) then each of them is equally likely to be selected and any one is selected at random, (iii) the crossover site (which is always a node in the routes) is assumed to divide each of the routes into two sections (like route 1 in the figure is divided into the section to the left of A and the section above A), and (iv) the two new offspring routes are formed by exchanging sections between the two parents (as seen in the figure).

As can be seen in Figure 3, the intra-string crossover operation is preceded by an operation where *m* copies of the same string are made. This is done in order to allow routes of the same route set to mate differently. Mating differently means intra-string crossover amongst different route pairs of the same route set as well as intra-string crossover of the same route pair at different crossover sites.

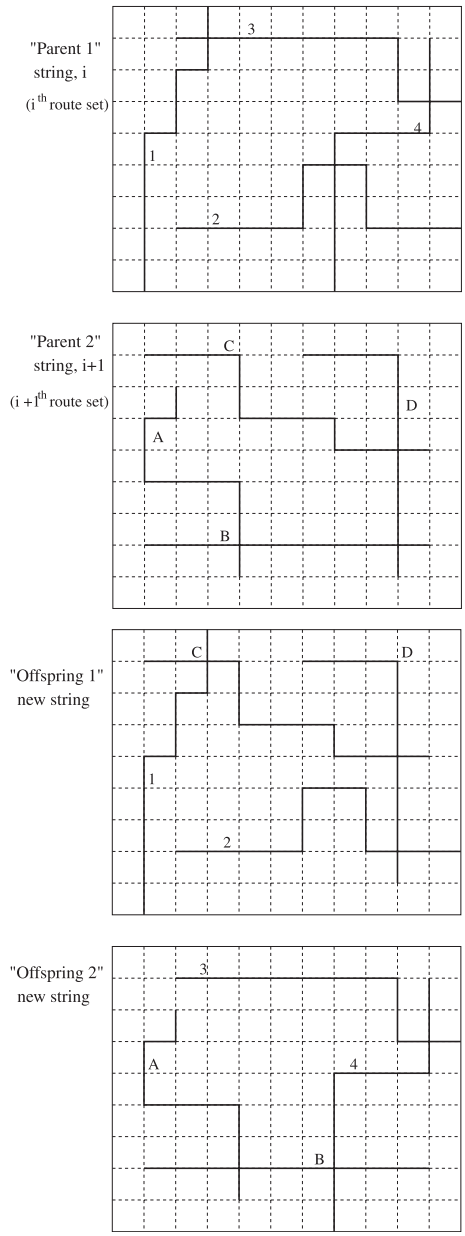
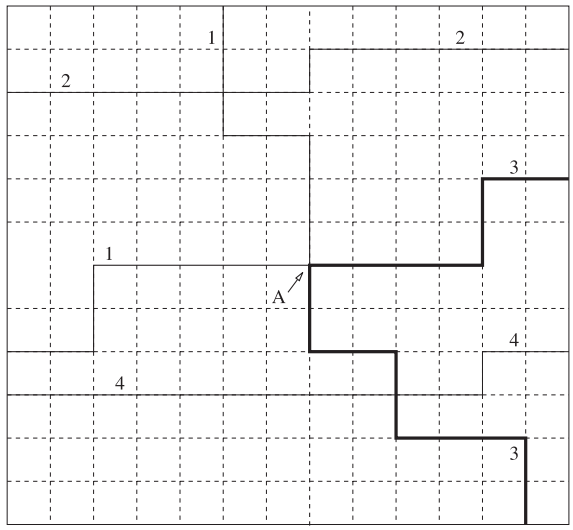


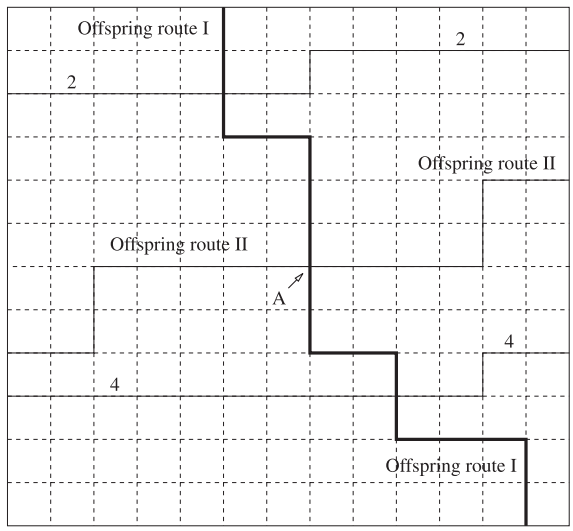
FIGURE 4 Schematic showing inter-string crossover operation.

Mutation operator

The purpose of the mutation operator is to slightly modify the routes (and therefore the route set). The modification is achieved by first selecting a node of a route randomly (the probability of selection of a node is a user-defined parameter called *mutation probability*) and then changing it to any of its acceptable vicinity nodes (the concept of acceptable vicinity nodes is described in Section 3.1). Note that an arbitrary change in the node may violate the feasibility of a route. For example, consider the following route: 1-2-7-9-11; *i.e.*, the route starts at node 1 then goes to node 2 and so on. Now, consider the case where node



String "j" before intra-string crossover



String "j"after intra-string crossover

FIGURE 5 Schematic showing intra-string crossover.

7 (which has, say, nodes 6 and 8 as its acceptable vicinity nodes) has been selected as the node for mutation. Before changing node 7 to node v (where v is either 6 or 8) one has to check whether a single link exists between nodes v and 9; if it exists then node 7 can be changed to node v. If, however, none exists then one or more extra nodes may have to be inserted (so as to have a feasible path between node v and node 9) into the route in order to allow node 7 to change to node v. Thus, after mutation, the route will be: 1 - 2 - v - (extra nodes, if required) - 9 - 11.

Reproduction operator

Selection of N (where N is the population size of a generation) good strings for the next generation from among the $N \times m$ strings. (There are $N \times m$ strings because, as explained

earlier, m copies of each string are made) is done using a standard GA operator called *Tournament selection*. In this operation, first, $2m$ strings from among the $N \times m$ strings are chosen at random; next, their $TOTFIT(r)$ values are compared and the string with the highest value is identified; then two copies of this string are made and stored as a part of the population of the next generation; finally all of these $2m$ strings are discarded. The process starts again with the remaining strings in the current set of $N \times m$ strings. The process stops when no string is left in the current set of strings. Note that after the reproduction operation is over the number of strings for the next generation is again N ($= 2 \times \{(N \times m)/2m\}$).

4 RESULTS

This section presents results obtained using the proposed optimal route design algorithm. In order to highlight the efficacy of the proposed algorithm the results presented here are for a benchmark road network (shown in Figure 6) used earlier by several authors to test their route design algorithms. This Swiss network was first used by Mandl [14] and later used by Baaj and Mahmassani [1] and Kidwai [12]. Mandl [14] also provided the link travel time data (the number on a link in the figure is the link's travel time) and the complete transit demand matrix for the network. Although the matrix is not reproduced here, it may be pointed out that (i) the total transit demand is 15570 trips per day, and (ii) only node 14 has a zero activity level, *i.e.*, it neither produces nor attracts any transit demand.

Four cases (I, II, III and IV) with different number of routes in the route set are presented. In Cases I, II, III and IV the number of routes in the route set are four, six, seven and eight, respectively. For each of the above cases, the results are presented using the following format. First, the route set obtained using the proposed algorithm is presented (only for Case I the route set is also presented using a diagram; this is not done for the other cases due to space restrictions). Then, a comparison between the performance of the route set obtained here with those obtained, for the same number of routes, by Mandl [14], Baaj and Mahmassani [1] and Kidwai [12] is provided. Mandl [14], Baaj and Mahmassani [1] and Kidwai [12] all use a transfer penalty of 5 min; the results from the proposed algorithm are also for a transfer penalty of 5 min. The comparison is done using the following measures of effectiveness:

- d_0^p, the percentage of demand satisfied directly by the route set,
- d_1^p, the percentage of demand satisfied with one transfer by the route set,
- d_2^p, the percentage of demand satisfied with two transfers by the route set,
- d_{un}^p, the percentage of demand unsatisfied by the route set,
- *ATT*, the average travel time (including transfer penalty) per user in minutes, and
- *TS*, the total man-hours saved per day by using the route set designed by the proposed algorithm instead of the route set reported in the literature.

However, before describing the results for the different cases obtained using the proposed method, the values of the parameters used in obtaining these results are presented.

4.1 Parameter Values

The following parameter values are used in obtaining the results from the proposed method.

- K, the cardinality of INS, $= 14$
- M, the maximum number of nodes in an initial route, $= 10$

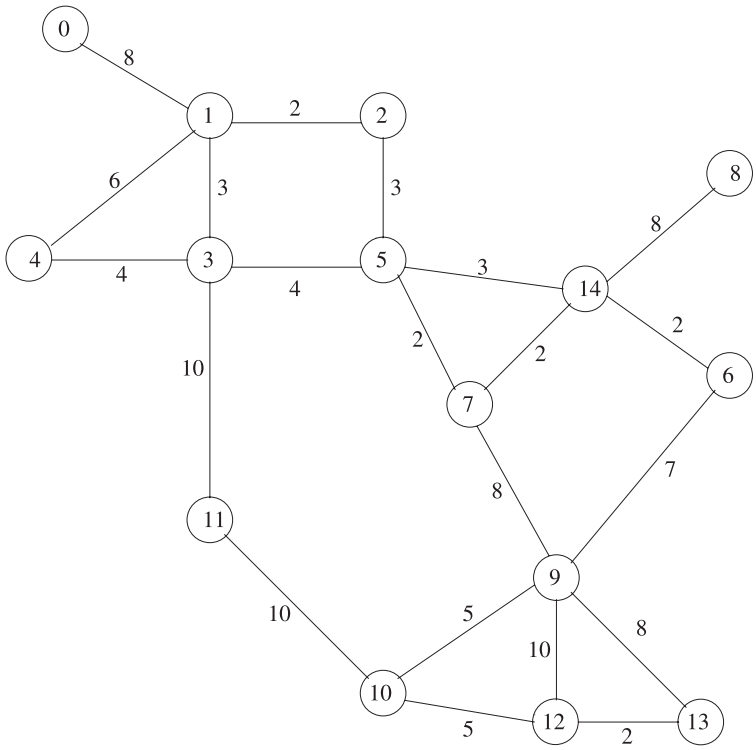


FIGURE 6 The benchmark Swiss road network first used by Mandl [14].

- L, the maximum length of an initial route, $= 35$ min
- U, transfer penalty for each transfer, $= 5$ min
- ω_i, the weight associated with $F_i(r)$, $= 1$
- K_i, the parameter associated with the determination of $F_i(r)$, $= 10$
- a, b, and c, parameters associated with the determination of $d_T(r)$, $= 1$
- Inter-string crossover probability $= 0.5$
- Mutation Probability $= 0.01$
- m, the number of copies for a string, $= 5$
- N, the population size, $= 20$

4.2 Case I: Route Set Design with Four Routes

The route set obtained using the proposed algorithm is as follows:

Route 1: 5-7-9-6-14
Route 2: 13-12-10-9-7-5-2-1-4-3-11
Route 3: 8-14-5-3-1
Route 4: 0-1-2-5-7-9-12-10-11

Figure 7 shows the same routes on the road network diagram. Note that, although node 14 has zero activity level there are routes which touch this node; this is expected because node 8 has a positive activity level and the only way to reach node 8 is through node 14.

Table I shows the comparison of the performances of the route sets designed using the proposed algorithm and those reported in the literature. In the table, *NR* in a cell indicates that the result for that particular cell was not reported by the author. For example, Baaj and Mahmassani[1] did not report the results for four routes and hence the row corresponding to Baaj and Mahmassani has *NR* in all its cells.

It can be seen from the table that the route set obtained using the proposed algorithm offers a substantially lesser average travel time (*ATT*) than the route networks proposed by Mandl [14] and Kidwai [12]. It may also be noted that since the total demand is 15570 trips per day, using the proposed route set instead of the one suggested by Mandl [14] produces a total saving (see last column of Table I) of 259.5 man-hours per day. Similarly, use of the proposed route set instead of the one suggested by Kidwai [12] produces a total saving of 213 man-hours per day. Further, note that the proposed route set satisfies a much larger percentage of demand without any transfers (d_0^p in the table)—a feature desirable in any route network design. These observations indicate the superiority of the proposed route network design algorithm.

4.3 Case II: Route Set Design with Six Routes

The route set obtained using the proposed algorithm is as follows:

Route 1: 12-10-11-13
Route 2: 9-13-12-10-9-6-14-7-5-3-1-0

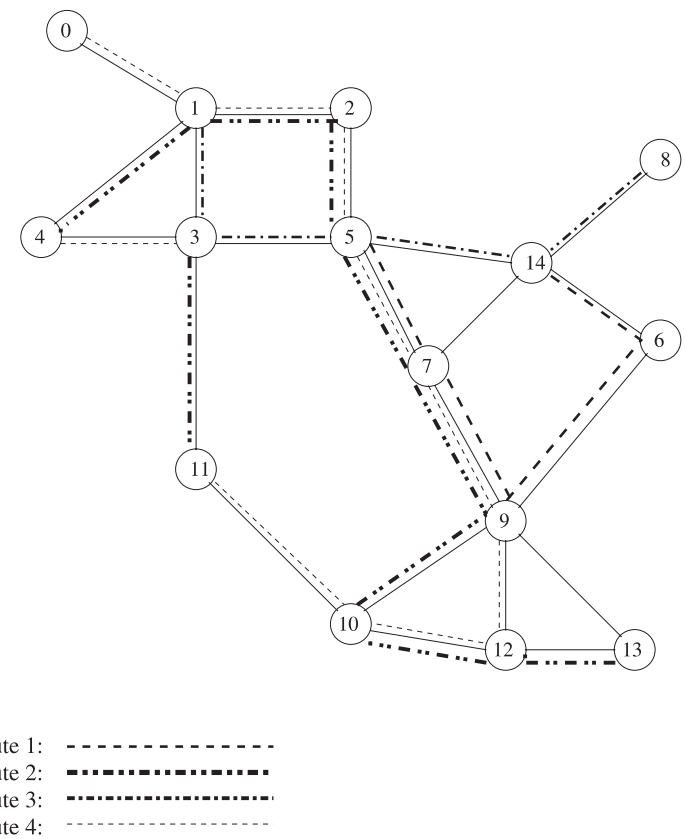


FIGURE 7 The "optimal" route network with four routes in the route set.

TABLE I Comparison of route sets with four routes

Models	d_0^p *(%)*	d_1^p *(%)*	d_2^p *(%)*	d_{un}^p *(%)*	*ATT* *(mpu†)*	*TS* *(mhd‡)*
Mandl [14]	69.94	29.93	0.13	0.0	12.90	259.5
Baaj and Mah.[1]	NR	*NR*	*NR*	*NR*	*NR*	NR
Kidwai [12]	72.95	26.91	0.13	0.0	12.72	213
Proposed Algorithm	86.86	12.0	1.14	0.0	11.90	—

† mpu: minutes per user
‡ mhd: man-hours per day

Route 3: 13-12-10-9-6-14-7-5-3
Route 4: 11-10-9-7-5-2-1-4
Route 5: 0-1-4-3-5-7
Route 6: 9-10-11-3-5-14-8

Table II shows the comparison of the performances of the route sets designed using the proposed algorithm and those reported in the literature. As seen in Case I, the average travel time per user (*ATT*), the total time savings (see last column of Table II) as well as the percentage of demand satisfied without transfers (d_0^p in the table) show that the proposed algorithm performs substantially better than the existing methodologies.

4.4 Case III: Route Set Design with Seven Routes

The route set obtained using the proposed algorithm is as follows:

Route 1: 12-10-9-6
Route 2: 9-6-14-7-5-3
Route 3: 4-1-2-5-7-9-10-12-13
Route 4: 10-9-13-12-10-11-3
Route 5: 8-14-5-3-11-10-9
Route 6: 5-3-4-1-0
Route 7: 12-10-9-6-14-7-5-2-1-0

Table III shows the comparison of the performances of the route sets designed using the proposed algorithm and those reported in the literature. As in the previous cases, the average travel time per user (*ATT*), and the total time savings (see last column of Table III) show that the proposed algorithm performs substantially better than the existing methodologies. It may be noted that although Kidwai's model [12] provides a higher percentage of demand satisfied without transfers (d_0^p in the table), this, however, comes at the cost of a higher average travel

TABLE II Comparison of route sets with six routes

Models	d_0^p *(%)*	d_1^p *(%)*	d_2^p *(%)*	d_{un}^p *(%)*	*ATT* *(mpu†)*	*TS* *(mhd‡)*
Mandl [14]	*NR*	*NR*	*NR*	*NR*	*NR*	*NR*
Baaj and Mah. [1]	78.61	21.39	0.0	0.0	11.86	405
Kidwai [12]	77.92	19.62	2.4	0.0	11.87	407
Proposed Algorithm	86.04	13.96	0.0	0.0	10.30	—

† mpu: minutes per user
‡ mhd: man-hours per day

TABLE III Comparison of route sets with seven routes

Models	d_0^p (%)	d_1^p (%)	d_2^p (%)	d_{un}^p (%)	*ATT* (mpu†)	*TS* (mhd‡)
Mandl [14]	*NR*	*NR*	*NR*	*NR*	*NR*	*NR*
Baaj and Mah. [1]	80.99	19.01	0.0	0.0	12.5	610
Kidwai [12]	93.91	6.09	0.0	0.0	10.70	143
Proposed Algorithm	89.15	10.85	0.0	0.0	10.15	—

† mpu: minutes per user
‡ mhd: man-hours per day

time per user (*ATT* in the table). It is felt that a 4% improvement in demand satisfied without transfers at the cost of 143 man-hours per day is not justifiable.

4.5 Case IV: Route Set Design with Eight Routes

The route set obtained using the proposed algorithm is as follows:

Route 1: 3-5-2-5-14-8
Route 2: 5-7-9-6
Route 3: 6-9-10-12-13
Route 4: 2-5-7-9-10-12-13
Route 5: 4-1-2-5-3-1-0
Route 6: 6-9-7-5-2-1-0
Route 7: 11-10-9-12
Route 8: 0-1-2

Table IV shows the comparison of the performances of the route sets designed using the proposed algorithm and those reported in the literature. Here again, it is seen that the route network designed using the proposed algorithm not only provides a lesser average travel time per user and substantial time savings (see last column of Table IV), it also satisfies a larger percentage of demand without transfers (d_0^p in the table). These observations again indicate the superiority of the proposed route network design algorithm.

5 CONCLUSION

In this paper an optimization procedure which evolves "optimal" or "efficient" transit route sets (or transit route networks) for a given road network and transit demand data from an initial set of routes is developed. The optimization strategies used in the proposed procedure

TABLE IV Comparison of route sets with eight routes

Models	d_0^p (%)	d_1^p (%)	d_2^p (%)	d_{un}^p (%)	*ATT* (mpu†)	*TS* (mhd‡)
Mandl [14]	*NR*	*NR*	*NR*	*NR*	*NR*	*NR*
Baaj and Mah. [1]	79.96	20.04	0.0	0.0	11.86	363
Kidwai [12]	84.73	15.27	0.0	0.0	11.22	197
Proposed Algorithm	90.38	9.58	0.0	0.0	10.46	—

† mpu: minutes per user
‡ mhd: man-hours per day

are based on the principles of genetic algorithms. The procedure is used to determine "optimal" route sets for a real-world network used as a benchmark by several other authors. The results show that the proposed method performs substantially better than the existing procedures.

The primary purpose of the results presented in this paper was to explore how the proposed genetic algorithm based procedure compares with existing techniques. Having shown that the proposed procedure is superior to the existing route network design methods, and given the fact that it is possibly the first route network design method which attempts to use non-heuristic techniques in the optimization process, the authors feel that the proposed method holds a lot of promise as an aid to transit planners and designers. However, in order for the proposed methodology to realize its full potential as a good route network design algorithm some more analysis, especially relating to the sensitivity of the outcome to the parameter values (of the proposed algorithm), should be carried out. Such analysis will help in developing guidelines for the selection of the parameter values for different problem scenarios.

References

[1] Baaj, M. H. and Mahmassani, H. (1991). An AI-based approach for transit route system planning and design. *Journal of Advance Transportation*, **25**(2), 187–210.
[2] Baaj, M. H. and Mahmassani, H. (1992). TRUST: A LISP program for the analysis of transit route configurations. *Transportation Research Record*, **1283**, 125–135.
[3] Byrne, B. F. and Vuchic, V. (1972). Public transportation line positions and headways for minimum cost. *Proceedings of the Fifth International Symposium on Traffic Flow and Transportation*, New York, 347–360.
[4] Byrne, B. F. (1975). Public transportation line positions and headways for minimum user and system cost in a radial case. *Transportation Research*, **9**, 97–102.
[5] Ceder, A. and Wilson, N. H. M. (1986). Bus network design. *Transportation Research -B*, **20B**(4), 331–344.
[6] Chakroborty, P., Deb, K. and Subrahmanyam, P. S. (1995). Optimal scheduling of urban transit system using genetic algorithms. *ASCE Journal of Transportation Engineering*, **121**(6), 544–553.
[7] Chakroborty, P., Deb, K. and Srinivas, B. (1998). Network-wide optimal scheduling of urban transit networks using genetic algorithms. *Journal of Computer Aided Civil and Infrastructure Engineering*, **13**, 363–376.
[8] Deb, K. and Chakroborty, P. (1998). Time scheduling of transit systems with transfer considerations using genetic algorithms. *Journal of Evolutionary Computation*, **6**(1), 1–24.
[9] Dubois, D., Bel G. and Libre, M. (1979). A set of methods in transportation network synthesis and analysis. *Journal of Operational Research Society*, **30**(9), 797–808.
[10] Goldberg, D. E. (1989). *Genetic Algorithm in Search, Optimization and Machine Learning*. Addison Wesley, Reading, Mass.
[11] Holroyd, E. M. (1967). The optimal bus service: A theoretical model for a large uniform urban area. *Proceedings of the Third International Symposium on the Theory of Traffic Flow*, New York, 308–328.
[12] Kidwai, F. A. (1998). Optimal design of bus transit network: A genetic algorithm based approach. *Ph.D. Dissertation*, Indian Institute of Technology Kanpur, India.
[13] Lampkin, W. and Saalmans, P. D. (1967). The design of routes, service frequencies, and schedules for a municipal bus undertaking: A case study. *Operation Research Quarterly*, **18**(4), 375–397.
[14] Mandl, C. E. (1979). Evaluation and optimization of urban public transportation networks. *Third European Congress on Operations Research*, Amsterdam, Netherlands.
[15] Mandl, C. E. (1980). Evaluation and optimization of urban public transportation networks, *European Journal of Operational Research*, **5**, 396–404.
[16] Newell, G. F. (1979). Some issues relating to the optimal design of bus routes. *Transportation Science*, **13**(1), 20–35.
[17] Rosello, X. (1977). An heuristic algorithm to generation an urban buses networks. *Proceedings of the Second European Congress on Operations Research*, Stockholm, Ed. M. Roubens, 409–419.
[18] Teodorovic, D. (1986). *Transportation Networks, A Quantitative Treatment*. Gordon and Breach Science Publishers.
[19] van Nes, R., Hamerslag, R. and Immerse, B. H. (1998). Design of public transportation networks. *Transportation Research Record*, **1202**, 74–82.