PERGAMON

Transportation Research Part C 10 (2002) 19–34

TRANSPORTATION RESEARCH PART C

www.elsevier.com/locate/trc

Genetic algorithms in bus network optimization☆

Maurizio Bielli [a], Massimiliano Caramia [b,*], Pasquale Carotenuto [c]

[a] *Istituto di Analisi dei Sistemi ed Informatica, C.N.R., Viale Manzoni 30, 00185 Rome, Italy*
[b] *Department of Computer Science, Systems and Production, University of Rome "Tor Vergata", Via di Tor Vergata 110, 00133 Rome, Italy*
[c] *Centro Charles Babbage, Istituto delle Tecnologie Industriali e Automazione, C.N.R., c/o Department of Computer Science, Systems and Production, University of Rome "Tor Vergata", Via di "Tor Vergata"110, 00133 Rome, Italy*

Received 1 April 1998; accepted 27 June 2000

Abstract

This paper focuses on a new method to compute fitness function (*ff*) values in genetic algorithms for bus network optimization. In the proposed methodology, a genetic algorithm is used to generate iteratively new populations (sets of bus networks). Each member of the population is evaluated by computing a number of performance indicators obtained by the analysis of the assignment of the O/D demand associated to the considered networks. Thus, *ff* values are computed by means of a multicriteria analysis executed on the performance indicators so found. The goal is to design a heuristic that allows to achieve the best bus network satisfying both the demand and the offer of transport. © 2001 Elsevier Science Ltd. All rights reserved.

Keywords: Genetic algorithms; Network optimization; Transportation

1. Introduction

The urban development and the installation of new and manifold activities on a territory, with industrial, commercial or service character, cause an always increasing demand on mobility to which it is necessary to offer an efficient well-organized and well-distributed transportation system. The adjustment of networks to new needs must occur through a unitary and organic vision of

☆ An earlier version of this manuscript was presented at TRISTAN III (Triennial Symposium on Transportation Analysis), 17–23 June 1998 in San Juan, Puerto Rico.

* Corresponding author. Tel.: +39-06-7259-7361; fax: +39-06-7259-7305.

E-mail addresses: bielli@iasi.rm.cnr.it (M. Bielli), caramia@disp.uniroma2.it (M. Caramia), carotenuto@disp.uniroma2.it (P. Carotenuto).

PII: S0968-090X(00)00048-6

the problem in order to obtain the maximum economy for the resources employed and the maximum functionality for the users.

Network design is certainly the most recurrent problem to solve when facing with planning problems. In particular it involves the strategic level, when it is necessary to decide about the financing of large facilities such as the construction of large infrastructure, the tactical level, when it is necessary to reorganize the lines of an urban bus network, and in more general terms, when an urban traffic control with several transportation modes has to be performed.

The goal of this work is to define a heuristic, based on genetic operators, that allows to generate, from an initial set of bus networks, new bus networks in order to improve the performance of the old ones and to reduce the number of vehicles employed in the network, without penalizing the average travelling time (see Bielli et al., 1998).

Here we give a sketch of the algorithm. The starting point is an initial collection of bus networks, which are handled sequentially by two different algorithm frames. The first one is represented by a genetic algorithm (GA), which acts as an external envelope of an internal *assignment algorithm*, working separately on each one of the considered networks. The output of the latter is processed by an *aggregation function*, that takes in input the assignment values and gives, for each bus network, a set of performance indicators. These are finally used in a *multicriteria analysis* returning *fitness function* (*ff*) values for each bus network. The external envelope, i.e., the genetic operators, uses these values in order to obtain new bus networks.

The remainder of the paper is organized as follows. Section 2 contains notations and definitions of GAs. In Section 3 we survey existing GAs in the transportation literature. In Section 4 we describe our heuristic and in Section 5 we experiment with a case study. Finally some conclusions are given in Section 6.

2. Genetic algorithms

The first papers in the literature dealing with GAs are those of Holland (1975) and Schwefel (1981). More recently Goldberg (1989), Michalewicz (1992) and Chambers (1995) have discussed several applications of GAs in optimization problems. A genetic algorithm is a local search algorithm, which works starting from an initial collection of strings (a population) representing possible solutions of the problem. Each string of the population is called a *chromosome*, and has associated a value called *fitness function* (*ff*) that contributes in the generation of new populations by means of genetic operators (denoted *reproduction, crossover* and *mutation*, respectively). Every position in a chromosome is called a *gene* and its value is called *allelic* value. This value may vary on an assigned *allelic alphabet*; most commonly the allelic alphabet is $\{0, 1\}$. At each generation, the algorithm uses the fitness function values to evaluate the survival capacity of each string i of the population using simple operators in order to create a new set of artificial creatures (a new population) which try to improve on the current *ff* values by using pieces of the oldest ones. Fig. 1 shows a simple layout of a GA implementation.

Before describing the generic operators in detail, we list a number of useful observations on the differences between this method and the other local search techniques:

- GA operates with codes of the parameter set and not with the parameters themselves;
- GA searches for a population of points and not a single point;

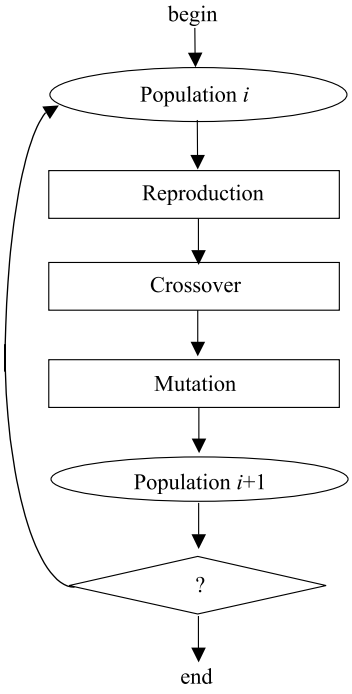

Fig. 1. Layout of a GA implementation.

- GA uses objective function information and not derived or auxiliary knowledge;
- GA uses probabilistic transition rules and not deterministic ones.

These particular aspects make this method applicable in a very general way, without the limitations imposed by other local search methods (i.e., continuity, derivative existence, and unimodality). Moreover it makes possible exploiting consequent information from more points in the dominion of the solutions, reducing the probability of finding false peaks, i.e., traps or local optima.

The working method of genetic algorithms is very simple and involves nothing more than copying strings or swapping partial strings. The simplicity of the operations and the power effect are two characteristics which make this method very attractive. Next the operators *reproduction*, *crossover* and *mutation* are discussed.

2.1. Reproduction

Reproduction (see e.g., Goldberg, 1989; Michalewicz, 1992) is a process in which individual strings are copied according to their objective function (*ff*) values. This represents a measure of the utility or goodness related to what we want to maximize. Copying strings according to their fitness function values means that strings with a high value have a higher probability of contribution to one or more offsprings in the next generation. The easiest way to implement the reproduction

operator is to create a biased roulette wheel where each string in the current population has a slot sized proportionally to its *ff* value.

Given a population with n items, in order to calculate the slot size of the roulette wheel, corresponding to the reproduction probability $p_r(i)$ of the string i, the following formula can be used:

$$p_r(i) = \frac{ff_i}{\sum_{i=1,\ldots,n} ff_i} \quad (n \text{ is the number of the total members of the population}).$$

At each simple spin of the weighted wheel we can select a candidate and enter its exact copy into a mating pool for further genetic operator actions (Fig. 2).

2.2. Crossover

The second operator denoted *simple crossover* (see e.g., Chambers, 1995; Goldberg, 1989; Michalewicz, 1992) works as follows: the members reproduced in the new mating pool are mated randomly and afterward each pair of strings undergoes a cross-change. In order to do this, an integer position k (*cut point*) is selected uniformly at random among the strings between position 1 and the string length l less one. Two new strings are created swapping all genes (with the corresponding allelic values) between the selected position k and the end of the string. Fig. 3 shows a simple crossover operator.

An improvement on the simple crossover operator, which enhances the exchange of information between the old population members in order to obtain the new ones, is to consider two

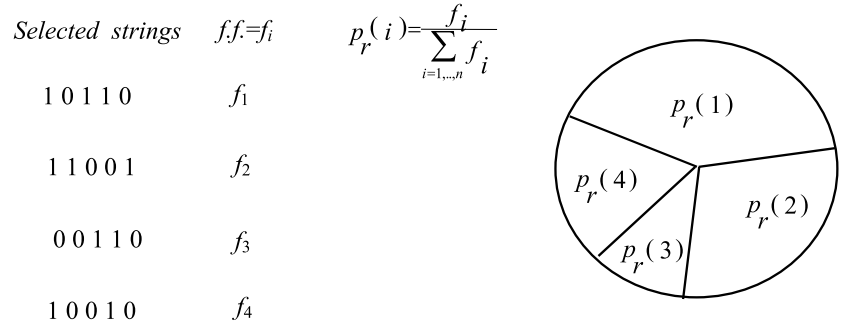

Fig. 2. Reproduction operator.

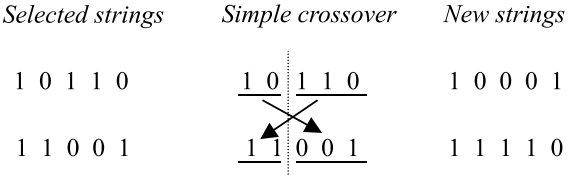

Fig. 3. Simple crossover operator.

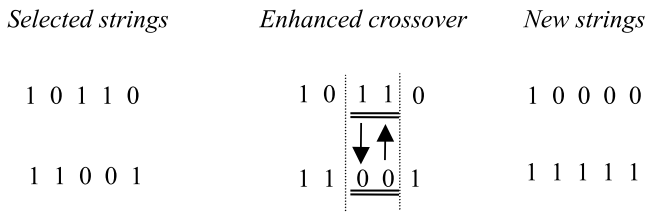

Fig. 4. Crossover operator improvement.

integer positions (cut points) uniformly selected at random among the string, and swapping the genes (with the corresponding allelic values) between the selected positions. Fig. 4 shows the crossover operator improvement.

2.3. Mutation

The *mutation* operator (see e.g., Goldberg, 1989; Michalewicz, 1992) plays a secondary role with respect to reproduction and crossover operators. Nevertheless mutation is needed to prevent an irrecoverable loss of potentially useful information which occasionally reproduction and crossover can cause. This operator is an occasional random alteration, with small probability, of the allelic value of a gene. Mutation is practically a random walk through the string space, thus guarantees the possibility of exploring the whole search space independently from the specific initial population and reduces the probability of finding a false peak.

3. Review of GAs in transportation systems

In this section, we survey some representative papers which use genetic algorithms in transportation problems. These choices are made according to the contest of this paper. Thus, there are a number of high quality papers in the literature which use natural algorithms (see e.g., Baaj and Mahmassani, 1995) and are used to solve transportation problems, but we do not mention due to the reason stated.

3.1. A multicriteria analysis for urban network design and parking location

Cantarella and Vitetta (1994) show that GAs can be efficiently used to solve the *urban network design* problem. A multi-level heuristic is used in order to find non-dominated approximate solutions to the urban network design problem (Fig. 5). At the outer level a new configuration of networks is evaluated through a genetic procedure. At the inner level the traffic signal setting and link flow assignment are carried out by an iterative method inside the traffic assignment procedure, where the signal setting, the delays on the network and the temporary flows are cyclically computed until two successive flow patterns are closed within a specified tolerance.

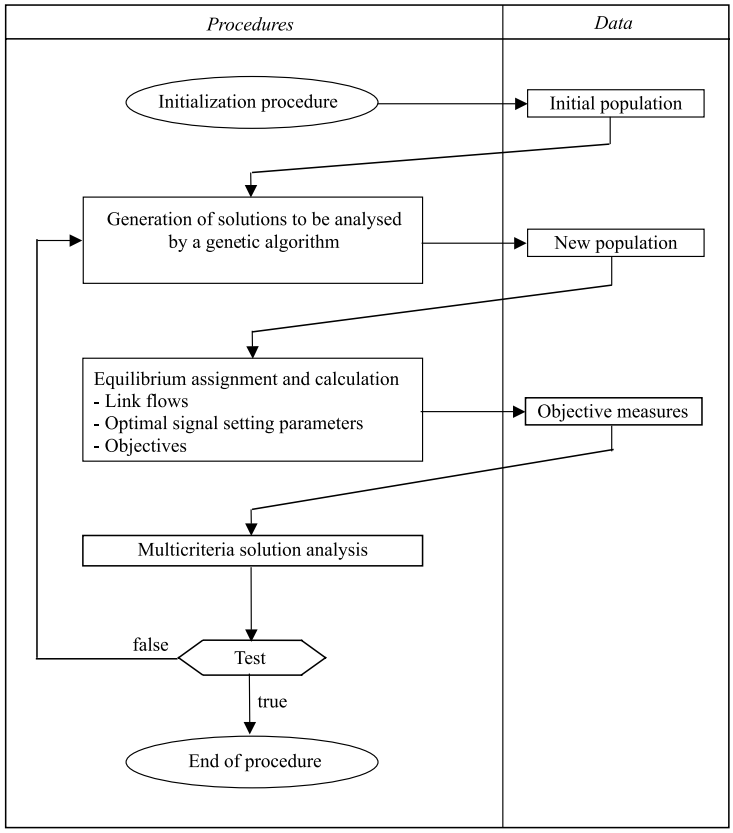

Fig. 5. A solution procedure for the *urban network* design problem – Cantarella and Vitetta (1994).

3.2. The continuous equilibrium optimal problem: a genetic approach

Cree et al. (1996) have implemented a computer program, namely GANDES, which is a genetic approach to find solutions for *network design* problems. GANDES takes in input networks and parameters, controlling the frequency and the type of the genetic operators. The program generates a random population of chromosomes, which are evaluated and then manipulated by the GA operators in order to produce better chromosomes.

3.3. Transportation network design using a cumulative algorithm and neural network

Xiong and Schneider (1993) introduced an improved version of a GA, denoted the cumulative genetic algorithm (CGA), and applied it to the *transportation network design* problem, where classical GAs do not work well, as good solutions can be lost in new population generations. In CGA all the population members with high *ff* are saved and used together with new population members as input for reproduction. A genetic algorithm that uses the trained neural network is represented in Fig. 6.

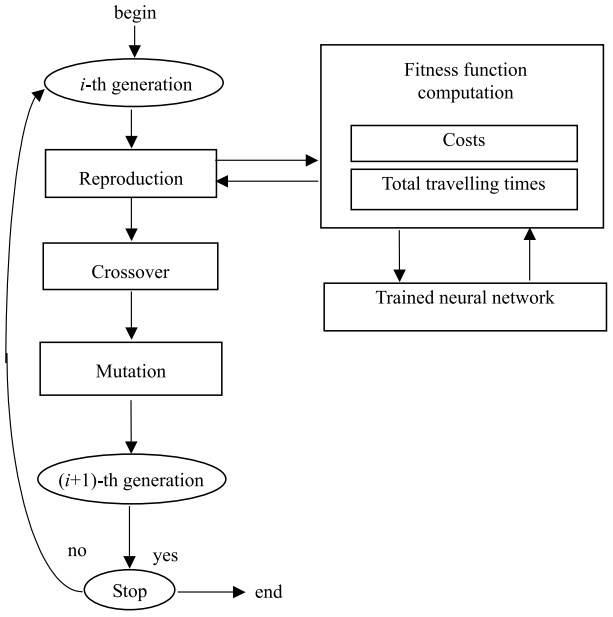

Fig. 6. The simple genetic algorithm – Xiong and Schneider (1993).

Each solution is represented as a binary string with a fixed length, which is the same as the input variable format of the neural network. Each solution has a construction cost and a total travelling time, but in the genetic algorithm each solution can have only one objective (*ff* value) to represent its performance. In order to use these two values to define a unique *ff* value, a domination comparison method is used. For each solution in a generation, a count of the number of times that it is dominated by the other solutions in the same generation is made. More times it is dominated, lower will be its *ff* value. Therefore, each *ff* value is calculated as $ff = (C - \text{number of times dominated})$, where C is the largest number of times dominated among all the solutions in that generation. Trivially non-dominated solutions will have $ff = C$. In CGA (Fig. 7) the reproduction operation was modified so that it picks up solutions randomly not only from the previous generation, but also from the historical non-dominated solution set denoted *HNDSS*.

3.4. Hybrid genetic algorithms for bus driver scheduling

Kwan and Wren (1994) describe a hybrid model for the *bus driver scheduling* problem, which incorporates a GA, a rule-based driver duties estimator, and an integer programming method called IMPACS. IMPACS produces near-optimal schedules. GA is quick when produces driver schedules, but so far it tends to converge at sub-optimal solutions. In the hybrid approach, the GA takes on an indirect role. It does not form a schedule, but IMPACS is applied using the elite population members produced to yield the optimal schedule almost trivially.

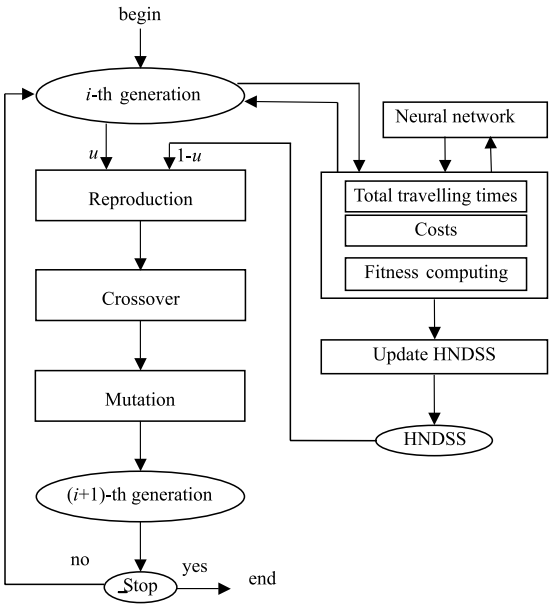

Fig. 7. The cumulative genetic algorithm – Xiong and Schneider (1993).

4. A new methodology involving gas in bus network optimization problems

Referring to the framework proposed by Ceder and Wilson (1986) on the transit planning process (Fig. 8) this work tries to unify the planning activities at level A and level B. In fact,

Independent Inputs	*Planning Activity*	*Output*
Demand data Supply data Route performance indicators	Level A Network design	Route changes New routes Operating strategies
Subsidy available Buses available Service policies Current patronage	Level B Setting frequencies	Service frequencies
Demand by time of day Times for first and last trips Running times	Level C Timetable Development	Trip departure times Trip arrival times
Deadhead times Recovery times Schedule constraints Cost structure	Level D Bus scheduling	Bus schedules
Driver work rules Run cost structure	Level E Driver Scheduling	Driver schedules

Fig. 8. The transit planning process – Ceder and Wilson (1986).

starting with predefined bus networks with fixed bus lines and frequencies, the proposed scheme tries to obtain new bus networks with better performance and more suitable line frequencies.

The first problem we deal with is to represent a bus network through a genetic representation. Fig. 9 shows how each chromosome is the hypothetical arrangement of a bus network, while Fig. 10 shows that one gene, formed by two alleles, is used to represent each (bus) line in a chromosome. The former allele represents the assigned frequency of the line, and the latter represents an on/off switch that able or disable the use of that line in the corresponding network. Thus, for a specific network there is a set of active lines and a remaining subset that is set off. In any case, we have to respect the cut point when applying crossover and mutation operators. Figs. 11 and 12 show how the crossover and mutation operators work in this particular implementation.

It has to be mentioned that the mutation operator works on both the on/off switch of a line and on its frequency. Indeed, when a line is set on, its frequency is randomly changed among a range of values.

The algorithm starts loading the initial population representing the initial set of networks. At each generation the algorithm defines a fitness function value for each network initially assigned, executing three successive phases next described.

(1) The first one is an assignment phase where the O/D demand is spread on the single networks. In Fig. 13 is reported the network structure on which the assignment is performed. There are two levels. The former is the *pedestrian level*, where the arcs refer only to the pedestrian mode; the latter is the *bus level*, defined by the bus line nodes and board arcs representing disjoint paths or cycles each for a single bus line (see Caramia and Storchi, 1997; Fernandez et al., 1994).

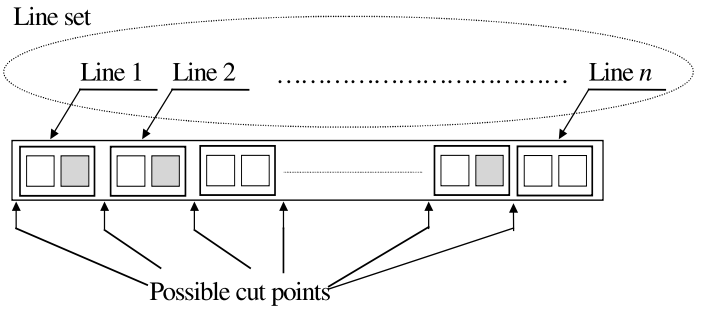

Fig. 9. Genetic representation: chromosome and cut points (network).

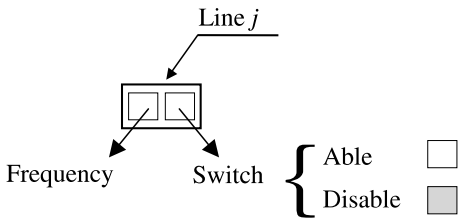

Fig. 10. Genetic representation: gene (line).

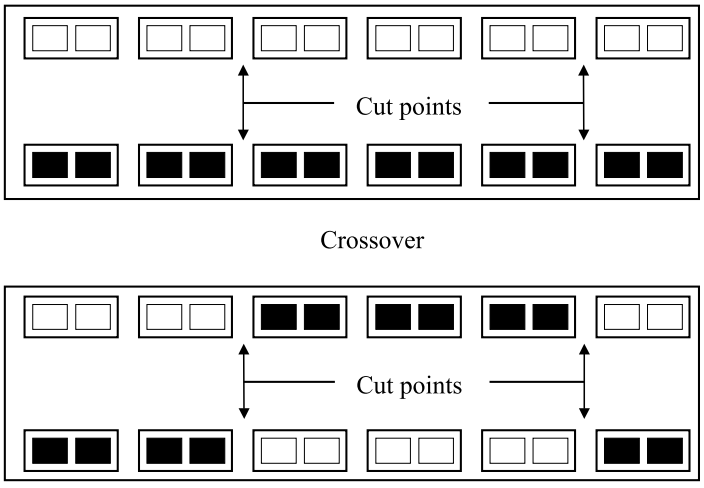

Fig. 11. Crossover operator implementation.

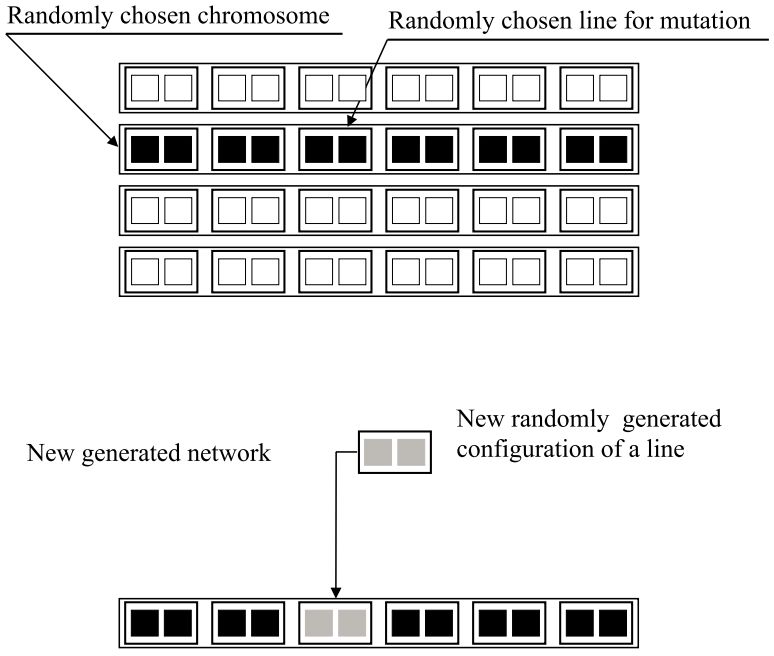

Fig. 12. Mutation operator implementation.

(2) The second phase is a routine that aggregates the output of the assignment phase in order to obtain a set of performance indicators (Bielli et al., 1996) reported in Table 1. The first column in the table reports indicators and the second their abbreviations. The third column contains two symbols: ▼ means that the lowest the value, the best the indicator, while ▲ means that the highest the value, the best the indicator. The fourth column defines the type of the indicator: effectiveness

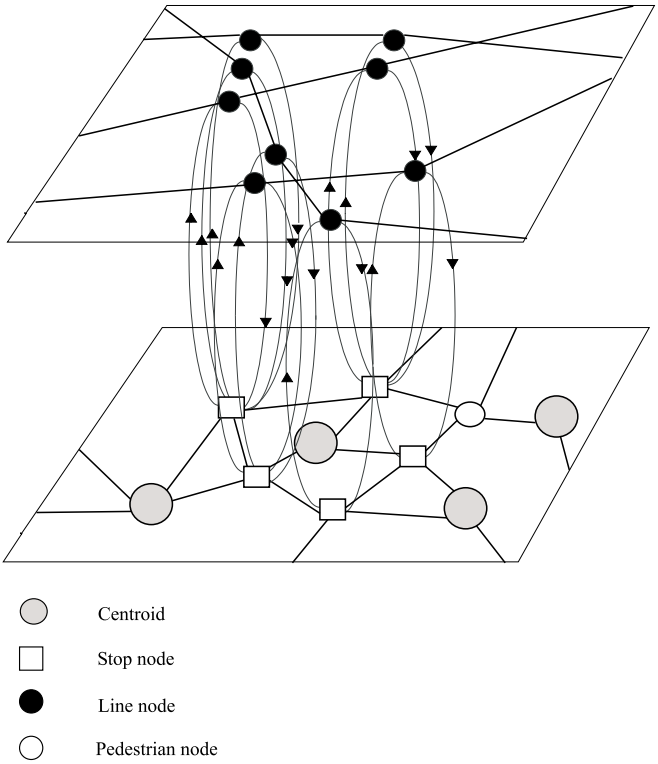

Fig. 13. The network representation for the flow assignment phase.

indicators are individuated by the letter E, efficacy indicators by the letter e, and quality indicators by the letter Q.

(3) Once the first and the second phases have been executed for each one of the networks forming the current population, the third phase starts. The latter is a multicriteria analysis performed in order to compute the *ff* values of each network (see e.g., Drive Project, 1991). In fact, when the assignment and the aggregation function have obtained the performance indicators for each network in the current population, it can be defined a matrix *indicator-network* as reported in Fig. 14. The multicriteria analysis applied to this matrix defines a classification of the networks, i.e., it orders networks according to the weighted sum of the indicators, where the weights represent the importance associated to the indicators and are predetermined before the computation starts. The values assigned to the single networks in the general classification obtained, represent their *ff* values which are used by the GA in the reproduction phase in order to find the survival networks in the next generation. Hence, the genetic operators are applied to the survived networks and the process described iterates.

If stopping rules are not verified, then a new generation is found, otherwise the algorithm stops. In general the algorithm stops after a fixed number of generations, or when no improvement arises after a fixed interval of time. Fig. 15 shows this process in detail. The best solution found by the algorithm, i.e., the chromosome having the highest *ff*, is a bus network formed by a set of lines

Table 1
The performance indicators

Vehicle number	Vn	▼	*E*
Vehicle capacity per km/Vehicle number	CKM/Vn	▲	*E*
Vehicle number/Frequency	Vn/F	▼	*E*
Users	U	▲	*e*
Users/Vehicle capacity per km	U/CKM	▲	*E*
Users/Network extension	U/L	▲	*e*
Users/Vehicle number	U/Vn	▲	*E*
Users per km	UKM	▲	*e*
Users per Km/Vehicle capacity per km	UKM/CKM	▲	*E*
Users per km/Network extension	UKM/L	▲	*e*
Users per km/ Vehicle number	UKM/Vn	▲	*E*
Number of stops/Number of links	Ns/Nlk	▲	*e*
Number of links	Nlk	▼	*E*
Number of lines	Nln	▼	*E*
Average number of transfers	Nt	▼	*Q*
Average waiting time	Tw	▼	*Q*
Average travelling time	Tt	▼	*Q*
Average walking time	Tm	▼	*Q*
Number of pedestrian relations	Npr	▼	*e*
Number of pedestrians	Np	▼	*e*
Crowd index (passengers $\leqslant 20$)	CI1	▲	*Q*
Crowd index ($20 <$ passengers < 75)	CI2	▲	*Q*
Crowd index (passengers $\geqslant 75$)	CI3	▼	*Q*
Equivalent pollution index	EPI	▼	*Q*

	Network 1	*Network* 2	*Network* 3	*Network* 4	*Network* 5	.....
Indicator 1	w_{11}	w_{12}	w_{13}	w_{14}	w_{15}	.....
Indicator 2	w_{21}	w_{22}	w_{23}	w_{24}	w_{25}	.....
.....	.....	.....	.....	.....	.....	.....
.....	.....	.....	.....	.....	.....	.....
.....	.....	.....	.....	.....	.....	.....
.....	.....	.....	.....	.....	.....	.....
Indicator k	w_{k1}	w_{k2}	w_{k3}	w_{k4}	w_{k5}	.....

Fig. 14. The matrix indicator-network.

corresponding to the genes having the switch set on. Moreover, each line has associated a frequency represented by the value of the other allele in that gene.

Finally two further considerations can be done (Fig. 16). The first one is related to the introduction of a *n*-best solutions container, that during the whole process stores a predefined number of good solutions generated. This container is very useful to perform an off-line sensitivity analysis on the weights used in the multicriteria analysis, in order to accept the results obtained or perform a new run with a new set of weights.

The second consideration is related to the possibility to consider a neural network implementation. The neural network can be used in the *ff* evaluation in substitution of the three phases previously described (assignment, aggregation and multicriteria analysis). This worth that, after a

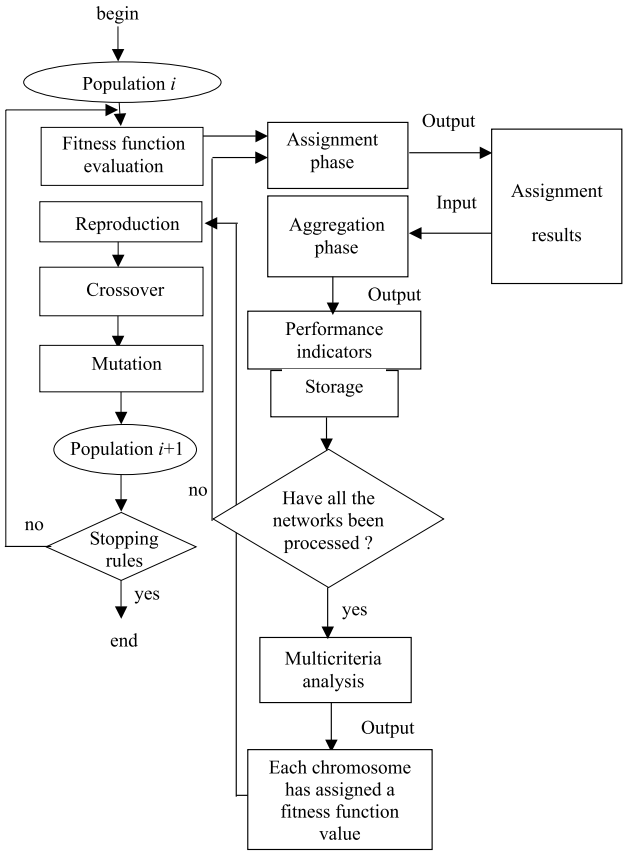

Fig. 15. Details of the fitness function evaluation.

learning phase, giving in input a chromosome at a generic generation, the neural network returns in output a vector whose components represent the *ff* values.

5. An experimental study

The proposed heuristic has been applied to a small city in the middle-north of Italy, namely *Parma*. We considered four different real projects of bus networks. The first one is the current existing Parma bus network, denoted *Parma*0: it has 1134 nodes, 3016 arcs, 459 stops and 22 lines. The second bus network considered is oriented to the O/D demand and is denoted *Parma*1. Given the O/D matrix containing the requests of movement between couples of centroids, this bus network is designed by considering the couple of origin–destination nodes having the highest query, and by inserting a direct line between them. The successive couples are considered in non-increasing values of the demand and are satisfied by exploiting existing bus lines, or building new ones when an overload of a line occurs. Parma1 is formed by 1075 nodes, 2843 arcs, 459 stops and 18 lines. The last two bus networks, denoted *Parma*2 and *Parma*3, respectively, are based on

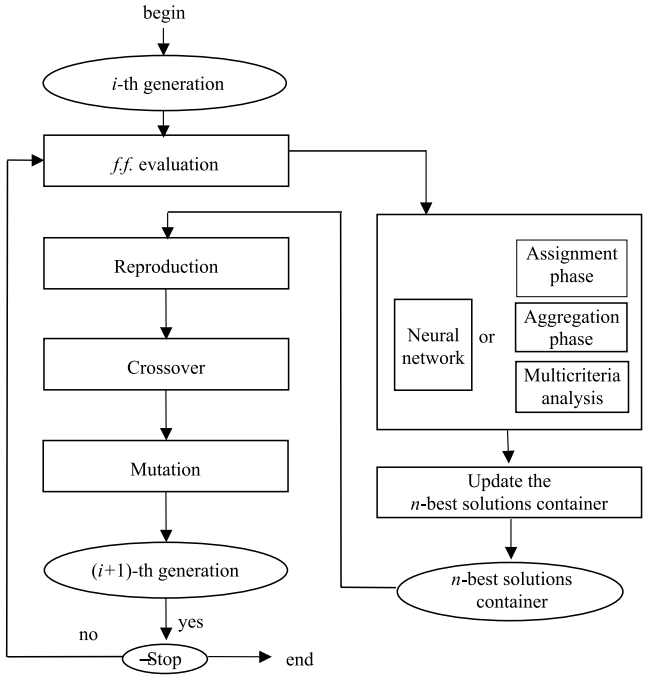

Fig. 16. A cumulative genetic algorithm.

geometrical structures. Parma2 is formed by 20 circular lines, involving 1130 nodes, 3006 arcs and 459 stops. Parma3 is constituted by 20 semicircular lines, where each line starts from a suburban node, goes through the centre of the city, where it first makes a semicircular route, and then gets to another suburban node. Parma3 is formed by 1092 nodes, 2892 arcs and 459 stops. Each one of the considered bus networks has 99 centroids and 685 pedestrian nodes which allow to complete the two-level representation given in the previous section.

What we have done is to create four initial chromosomes having length 80, being the total number of bus lines corresponding to the four projects, where a gene in a chromosome has a line switch set on, whether the corresponding line is in the corresponding project and is set off otherwise. Each line has also associated a frequency.

In Table 2 we reported a set of experiments performed with different combinations of crossover and mutation probability. In particular we have implemented a crossover probability 0.8, 0.9, and

Table 2
Best fitness function values for different combinations of crossover and mutation probability

Mutation probability	Crossover probability		
	0.8	0.9	1.0
	Fitness function values		
0.2	0.66	0.75	0.85
0.1	0.68	0.78	0.89
0.0	0.65	0.74	0.78

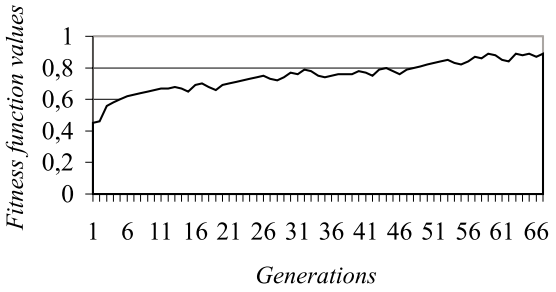

Fig. 17. Fitness function values for each generation.

1.0, and a mutation probability 0.0, 0.1, and 0.2. Results show that the best *ff* is obtained in correspondence to a 0.8 crossover probability and a 0.1 mutation probability. We remark that our computation was conducted on 100 generations and the best *ff* value was obtained in correspondence to the 66th generation. Moreover, the best initial *ff* value, i.e., the best *ff* value among the four initial bus networks, was 0.47.

In Fig. 17, we report the graphic representing the trend of the *ff* with respect to the number of generations, in the case of a crossover and mutation probability equal to 0.8 and 0.1, respectively.

6. Conclusions

In this paper, we have proposed a heuristic approach to solve transportation bus network optimization problems. The method involves generic operators and a number of additional ingredients which allows to compute *ff* values aggregating the values of a number of performance indicators. An experimental study is given where we tested our heuristic on a real transport network. Results obtained show an improvement on the initial *ff* value of about 90%.

Further researches will be devoted to investigate, as mentioned in the Section 4, the usage of a neural network or an *n*-best solutions container. In this latter case a sensitivity analysis can be performed on the results saved in the *n*-best solutions container, in order to verify the robustness of the solutions with respect to the pattern of weights assigned.

References

Baaj, M.H., Mahmassani, H.S., 1995. Hybrid route generation heuristic algorithm for the design of transit networks. Transportation Research C 3, 31–50.

Bielli, M., Caramia, M., Carotenuto, P., 1998. Genetic algorithms in bus network optimization. In: TRISTAN III Conference Proceedings. Puerto Rico.

Bielli, M., Carotenuto, P., Gastaldi, M., 1996. Multicriteria evaluation model of public transport networks. In: Bianco, L., Toth, P. (Eds.), Advanced Methods in Transportation Analysis. Springer, Berlin, pp. 135–156.

Cantarella, G.E., Vitetta, A., 1994. A multicriteria analysis for urban network design and parking location. In: TRISTAN II Conference Proceedings. Capri, pp. 839-852.

Caramia, M., Storchi, G., 1997. Multimodal shortest hyperpaths on transportation networks. Technical Paper 25, University of Rome La Sapienza.

Ceder, A., Wilson, N.H.M., 1986. Bus network design. Transportation Research B 20, 331–344.
Chambers, L., 1995. Practical Handbook of Genetic Algorithms: Applications. CRC Press, Boca Raton, FL.
Cree, N.D., Maher, J.H., Paechter, B., 1996. The continuous equilibrium optimal problem: a genetic approach. In: Fourth Meeting of Euro Working Group on Transportation, Newcastle, UK.
Drive Project V1036, 1991. EVA Manual – Evaluation Process for Road Transport Informatics. EVA Consortium (Eds.).
Fernandez, E., De Cea, J., Florian, M., Cabrera, E., 1994. Network equilibrium models with combined modes. Transportation Science 28, 182–192.
Goldberg, E.D., 1989. Genetic Algorithms in Search Optimization and Machine Learning. Addison-Wesley, Reading, MA.
Holland, J.H., 1975. Adaptation in Natural and Artificial Systems. The University of Michigan Press, Ann Arbor, MI.
Kwan, R.S.K., Wren, A., 1994. Hybrid genetic algorithms for bus driver scheduling. In: TRISTAN II Conference Proceedings, Capri.
Michalewicz, Z., 1992. Genetic Algorithms + Data Structures = Evolution Programs. Springer, Berlin/Heidelberg.
Xiong, Y., Schneider, J.B., 1993. Transportation network design using a cumulative algorithm and neural network. Transportation Research Record 1364.