Soft Comput (2016) 20:3595–3609
DOI 10.1007/s00500-015-1718-1

METHODOLOGIES AND APPLICATION

A modified genetic algorithm applied to the elevator dispatching problem

M. Beamurgia[1] · R. Basagoiti[1] · I. Rodríguez[2] · V. Rodriguez[2]

Published online: 3 June 2015

Abstract Reduction of passenger waiting time in a multiple elevator system is an important goal in the lift industry. Genetic algorithms (GAs) have been applied to the dispatching problem in vertical transportation. In this paper, we present an approach based on a GA with several relevant adjustments to adapt this type of algorithm to this problem. The algorithm serves calls currently registered in the system to create a dispatch plan, under the assumption that just one passenger has made each call (i.e. without passenger forecasting). We develop and investigate various versions of the GA incorporating one or more adjustments in this research area. The algorithms were implemented and evaluated using ELEVATE, for two different building configurations, in terms of incoming, outgoing and interfloor profiles. To compare results, one-factor analysis of variance tests were applied to passenger waiting times. The performance of the basic GA was significantly improved upon by making these adjustments. These adjustments turn out to be essential for a successful implementation of a GA in the dispatching problem.

Communicated by V. Loia.

Practical application A genetic algorithm can be used to solve the elevator dispatching problem. Adjustments can optimize the solution. The current paper lists and describes possible adjustments, and evaluates their effects on performance in isolation and in combination.

✉ M. Beamurgia
mbeamurgia@uco-mondragon.edu.mx

1 Department of Electronics and Computing, University of Mondragon, Arrasate, Spain

2 Faculty of Economics and Business Science, University of Navarra, Pamplona, Spain

Keywords Elevator dispatching problem · Genetic algorithm · Adjustments

Abbreviations

ANN	Artificial neural networks
ANOVA	Analysis of variance
ATT	Average transit time
ATTD	Average time to destination
AWT	Average waiting time
$C1$	Building configuration [6 cars, 12 floors, average population/floors for populated floors, car capacity (Kg) 1600], 400 landing calls to be attended
$C2$	Building configuration [6 cars, 33 floors, average population/floors for populated floors, car capacity (Kg) 850], 350 landing calls to be attended
EGCS	Elevator group control system
ELEVATE	simulation software for vertical transportation http://www.elevateconsulting.co.uk/
ETA	Estimated time arrival. A rule based dispatching algorithm
EWT	Estimated waiting time
GA	Genetic algorithm
GA + ALL	The algorithm GA with all the next adjustments LbI + Sd + S + $P3$
GA + ALL	GA with all adjustments (GA + LBI + Sd + S + $P3$)

GA + LBI	The GA algorithm with the Last Best Individual adjustment
GA + P	The GA algorithm with the P penalty adjustment
GA + $P3$	The GA algorithm with the $P3$ penalty adjustment
GA + S	The GA algorithm with the stability adjustment
GA + S + LBI + Sd + P	The GA algorithm with the S, the LBI, the Sd and the P adjustments
GA + S +LBI + Sd + $P3$	The GA algorithm with the S, the LBI, the Sd and the $P3$ adjustments
GA + S + LBI + Sd	The GA algorithm with the S stability, the LBI and the Sd adjustments
GA + S + P	The GA algorithm with the S and the P adjustments
GA + S + $P3$	The GA algorithm with the S and the $P3$ adjustments
GA + Sd	The GA algorithm with the seeding adjustment
GC	Group collective, a rule Based dispatching algorithm
HC	Handling capacity, a percentage of the population of the building in a vertical transportation system that the system can move in up-peak mode in a 5 min period
LBI	Last best individual adjustment
LTT	Longest transit time
LTTD	Longest time to destination
LWT	Longest waiting time
NSF	Next stopping floor
P	Penalization
$P3$	$P3$ Penalization
RWT	Real waiting time. The average of the real waiting times for the passengers already in the system. The real waiting time of passengers in the system is the waiting time from the moment they press the button until the lift's arrival in the simulation
S	Stability adjustment
Sd	Seeding adjustment
STEP1	Mixed passenger profiles where 45 % of the landing calls were for incoming, 45 % were for outgoing and 10 % were for interfloor with increasing handling capacity from 11 % to 13 %
STEP2	Mixed passenger profiles where 0 % of the landing calls were for incoming, 100 % were for outgoing and 0 % were for interfloor with increasing handling capacity from 11 to 13 %
STEP3	Mixed passenger profiles where 80 % of the landing calls were for incoming, 15 % were for outgoing and 5 % were for interfloor, with increasing handling capacity from 11 to 13 %
TSP	Travelling salesman problem

1 Introduction

An important concern for lift producers is reduction of user waiting time when a system has more than one lift and in the recent years this problem has become an active research topic in both academia and industry. Most published studies aim to optimize the performance of the controller or elevator group control system (EGCS).

Genetic algorithms (GAs) have been used to minimize the average waiting time of passengers, while recognizing that long waiting times are not acceptable. Alander et al. (1994) use a GA and explains that from each floor there can be two calls, one upwards and the other downwards. Each call is weighted according to what type of traffic it involves. Cortés et al. (2003) compare the conventional algorithms with the GAs. Cortes uses a landing call strategy to identify the "chromosomes" of individuals in the population: each individual is associated with upwards landing calls, followed by downwards landing calls. The objective function used is focused in time, and returns the expected time. Cortés et al. (2004) use GAs to reallocate lifts to calls if necessary. Cortés et al. (2003, 2004) use a replacement rule in which the probability of replacement for a given individual is inversely proportional to the individual's fitness.

GAs have also been used to resolve one level of the elevator dispatching problem when the problem has been divided into two levels. Sorsa et al. (2009) use a two-level approach in which the first level is dealt by a GA and the second one is dealt by a heuristic algorithm. The heuristic algorithm selects the nearest call to the elevator with respect to its travelling direction. The GA is based on a database of the passenger information, which is used to improve subsequent solutions if that state of the system appears again. The objective function is to minimize travelling time, which is separated into three components: the call time, the waiting time and the journey time.

GAs can also be used to optimize two objectives simultaneously: the first objective being minimization of the passenger waiting time and the second being minimization of energy consumption. These two objectives work against each other and so the system has to decide when to prioritize one objective over the other. Tyni and Ylinen (2006) propose a way of prioritization of each objective depending on the different type of traffic at a given moment. Tyni uses a PI-controller (Proportional and time-Integral terms) to provide a specified service level in terms of average call time. Tartan et al. (2014) also use genetic algorithms but considers not only the waiting time but also the journey time of the passenger as one of the optimization criteria. Liu et al. (2014) apply a particle swarm optimization algorithm and a GA to obtain a combined control method.

The GA developed in this paper is based on the work done by Sorsa et al. (2009), it solves the first optimization level (the assignment of cars to landing calls), but it assumes that one landing call represents one passenger. Building traffic patterns change in real time and we decided to analyse a system where it is assumed that exactly one passenger is behind one landing call. Further versions will analyse the system with more complex passenger traffic modules. This assumption simplifies management of lift capacity. The algorithm reallocates landing calls each time it is run (typically in cycles of milliseconds), which allows the algorithm to improve the assignment in real time according to the current state of the system thereby making the GA more flexible and robust against abrupt changes in traffic patterns.

We develop a basic GA and four different adjustments: the first and the second adjustments help the GA in the generation of the first population of individuals (i.e. potential routes), the third one uses the best individual or solution of the previous cycle for the current cycle, and the fourth one adds penalties to individuals according to their waiting times. We also use an analysis of variance (ANOVA) to quantitatively evaluate the impact of these adjustments on the raw genetic. The modified version of the raw GA, that incorporates all these adjustments, outperforms the results for different traffic configurations.

The paper is organized as follows. Section 3 contains a description of the problem. Section 4 presents the mathematical model of the dispatching problem. Section 5 describes the genetic algorithm applied to solve the dispatching problem. Section 6 presents simulation results. Finally, conclusions are made and discussed in Sect. 7.

2 Related work

In this section, we summarize other authors' works using techniques from the artificial intelligence to solve different lift concerns. Artificial intelligence control methods can be used to find a reasonable solution to an optimization problem in an appropriate time. Other approaches applied are neural networks for the passenger arrival prediction and fuzzy logic for the implementation of fuzzy rule sets for recognizing traffic patterns.

An artificial neural network (ANN) in the elevator group control can be used to learn dynamically the behaviour of the elevator system and predict the next stopping floor (NSF) on the basis of the previous pattern of demand. Imrak and Özkirim (2004) determine the NSF with an ANN and shorten waiting time by forecasting car position and using call distribution laws. However, the ANN needs to know the number of passenger and also the traffic profile pattern has to be available. Imrak and Özkirim (2006) uses the NSF to shorten waiting time by assigning one of the lifts to a new landing call and by taking into account mean waiting time of waiting passengers, travel time, and the number of passengers waiting on each floor. This system has no requirement to predefine traffics events. Echavarria and Frenz (2009) improve the lift call time responsiveness by approximating the lift call pattern through association of time of day with specific call allocations.

Fuzzy logic has been used to analyse the passenger behaviour, and the controller adapts itself to the situation of the predominant traffic pattern (Cortés et al. 2012a). Siikonen (1997a) uses fuzzy logic for traffic pattern recognition. Kim et al. (1998) use a multiple objective system in which the assignment method is based on fuzzy theory, classification of passenger traffic and the system manager's requirements. Depending on traffic demand, the priority of objectives can be modified, assigning more importance to a certain criteria, Kim uses three criteria: the waiting time of the passengers, the percentage of passengers with a long wait and the run count used for the power consumption of the system. Rosso Mateus and Soriano (2008) take into account all available information: the fuzzy logic can be used to assign cars to the landing calls according to which criteria are given more importance.

Other authors have used bio-inspired techniques to deal with the dispatching problem. Cortés et al. (2012b) use a viral system algorithm that is based on a virus infection analogy. In his work, he compares this approach to genetic and tabu search algorithms.

3 Description of the problem

Lift buttons send passenger requests to the lift control system. The control system collects all the requests at every time interval and then decides which lift will pick up which passenger. The control system has to respond to all the requests, minimizing passenger waiting time. Following Siikonen (1997b), landing calls (that come from outside the lift) can be initially allocated to one lift but then reallocated to another. Only when a lift is stopping at a floor to receive a passenger, reallocation can be disabled for that call. Otherwise, as this is a dynamic and a real-time problem, reallocation is continuously applied.

The optimization problem is treated as a case of the travelling salesman problem (TSP), as suggested in Sorsa et al. (2009). The TSP typically considers a salesman who needs to travel around various cities, visiting each city only once with the shortest travelled distance. In the current work, we model the vertical transportation scenario as a multiple salesman problem where a lift (the salesman) visits each landing (city) to answer requests registered on buttons in two directions, up and down. In this implementation, up or down requests from the same landing or floor can be considered akin to different cities. A requested call is only assigned to a lift once. To create routes for a given lift, Closs rules (see below) are applied to rule out certain possibilities, and the load of the lift is continuously controlled. Therefore, a lift can skip one landing call if the lift is full, and return and answer it later.

The control system has been developed in two levels. At the upper level, the control system tells the different lifts what they have to do (go up or down), evaluating the overall performance of the whole lift system. At the lower level, the control system makes decisions taking into account all the landing calls of each lift.

There are two types of call: the landing call, which is made by a passenger on a floor *outside* the lift (in a conventional button panel, with up and down button), and the car call which is made by a passenger *inside* the lift (typically with a button for each destination floor).

Different traffic can be observed. An incoming flow occurs when most of the passengers are coming into the building. The outgoing flow is associated with net efflux. The interfloor flow is that related to people moving inside the building from one floor to another. Traditionally, landing call traffic is categorized into four basic types: the normal traffic, an up peak, a down peak, and peak of two directions.

The process of assignation must take into account some explicit constraints. A lift can only be assigned to a single request at a time. For this paper, some general conditions and assumptions have been made: behind each call there is only one passenger; car calls have priority over landing calls; current landing calls and car calls are registered from the moment, the time stamp, at which they are created; the source floor of landing calls is registered as is the required direction (up/down) of travel; the destination floor of car calls is registered; when a lift is stopping at a floor in order to receive a passenger, the corresponding call will disappear from the list of landing calls for allocation/reallocation, but any existing destination floor for that lift will be maintained. In addition, we adopt certain management criteria: a lift will attend the nearest call that fulfils all the requirements; and if a lift has stopped and has no assigned direction, it will attend to the passenger with the longest waiting time. Finally, our model applies certain protocols: for each landing call, a fictitious car call is created, which is then deleted when the landing call disappears: and if a lift is not full, only the first landing call of the route of the highest-priority individual is assigned to the lift.

The system also follows Closs rules (1970):

- A car may not stop at a floor where no passenger enters or exits
- A car may not pass a floor at which a passenger wishes to exit
- A passenger may not enter a car carrying passengers and travelling in the reverse direction to his required direction of travel
- The lift cannot change the direction if at least one passenger is inside.

The dispatcher algorithm needs the following constant information for each global lift:

- Maximum capacity
- Maximum velocity
- Maximum acceleration
- Maximum jerk
- Door open time (DOT)
- DOT while passengers entering and/or leaving the lift
- Door close time (DCT)
- Floors where the lift can move.

Similarly, the following variable information is registered for each lift each time the optimization procedure starts: the number of passengers currently being carried, velocity, acceleration, jerk, position, current floor, direction, destination floor, quickest stop floor position, stopping in a floor, door status, and travel status.

4 Mathematical model

The mathematical model in this section is oriented to the implementation of the genetic algorithm that we explain in Sect. 5. This model shows the procedure for the calculation of the main variable related to the fitness function of the individuals of the GA. The main variable considered for each individual is the sum of the estimated waiting times (EWTs) for every passenger in the system. The EWT of a passenger is the estimated time in seconds that the passenger will wait after he pushes the button in the floor. Waiting times larger than 30 s are usually considered as long and undesirable for this type of service (Sorsa et al. 2003).

The individuals of the GA are assignments of landing calls to the lifts. Many assignments involve the generation of routes for the lifts. The routes are used to estimate the waiting time of the passengers assigned to a lift, and involve the estimated time that the elevators need to arrive to the passengers' floors and open the doors.

4.1 Description of the model

We state the model using the following assumptions, I is the number of lifts and m the number of floors. Floors are connected in a direct graph and a landing call L will be associated with only one lift after the optimization finishes. A state of the system (the position of the lifts, the lift direction, the velocity and the acceleration, the weight of the lifts, the landing calls and the cab calls) is read and any time it changes, a set of individuals H representing an assignment is randomly created and evolved until finding an optimal individual during the available computing time. Parameters as runtime or DOT, DCT and DOT while passengers are entering and/or leaving the lift (DTT) introduce additional information for a proper calculation of the estimated waiting time EWT.

Indices:

I	set of lifts $I \in \{1, 2, \ldots, n\}$
i	index of lifts $i \in I$
m	number of floors
L	set of landing calls $L \subseteq \{1, 2, \ldots, m\}$
Lc,i	Set of landing calls associated with lift i $\mathrm{Lc} \subseteq L$ $\sum_{i=1}^{I} \mathrm{Lc}, i = L$
j	index of landing calls $j \in \mathrm{Lc}$
H	set of individuals (Note that the term individual refers to a GA entity, not a passenger, as is explained below.)
h	index of individual $h \in H$
k	index of the winner individual $k \in H$

Parameters:

$\mathrm{runTime}_{h,i,j}$	Time needed to move the lift i from one landing call j to landing call $j+1$ considering the time associated with the car call of individual h
$\mathrm{DOT}_{h,i,j}$	Door open time for lift i in the jth landing call of individual h
$\mathrm{DTT}_{h,i,j}$	Door open time while passengers entering and/or leaving for the ith lift in the jth landing call of individual h
$\mathrm{DCT}_{h,i,j}$	Door close time for the ith lift in the jth landing call of individual h

Variables:

$\mathrm{doorStatus}_i$	The door status (open or closed) for the lift i, at the time when the algorithm is called to obtain a route.

4.2 The model

The main objective of the model is to minimize the EWT of all passengers. To estimate the waiting time for an individual, an optimal route for each lift is estimated. Once a route has been decided upon, Eq. (1) is used to calculate $\mathrm{EWT}_{h,i,j}$, the EWT of each landing call j, attended by the ith lift according to the solution for the individual h. Note that to attend the landing call j, first the ith lift needs to close the doors if they are open ($\mathrm{DCT}_{h,i,j}$); second, to cover the distance from its position to the floor of the passenger ($\mathrm{runTime}_{h,i,j}$); third, to open the doors when arrives to the floor ($\mathrm{DOT}_{h,i,j}$) and finally, the passenger might wait some passengers to leave before entering into the lift ($\mathrm{DTT}_{h,i,j}$).

To calculate the $\mathrm{runTime}_{h,i,j}$ of the ith lift, we have applied some equations related to the lift kinematics, described in Peters (1995). These equations use the maximum acceleration and the jerk (rate of change of acceleration) values constrained by human comfort criteria.

$$\mathrm{EWT}_{h,i,j} = \mathrm{DCT}_{h,i,j} + \mathrm{runTime}_{h,i,j} + \mathrm{DOT}_{h,i,j} + \mathrm{DTT}_{h,i,j} \tag{1}$$

With Eq. 2, the EWT of all the landing calls of the system is taken into account according to the scheduling of the individual h of the GA.

$$\mathrm{EWT}_h = \sum_{i=1}^{I} \sum_{j=1}^{\mathrm{Lc},i-1} \mathrm{EWT}_{h,i,j} \tag{2}$$

In Eq. 3, the $\mathrm{individual}_k$ represents the GA winner individual, i.e. the individual whose request is associated with the best calculated solution that will determine the next set of lift movements.

$$\mathrm{individual}_k = \min(\mathrm{EWT}_h) \tag{3}$$

When j is equal to zero, the initial state of the ith lift with the following values of the parameters is assumed.

$$\begin{aligned}
\mathrm{DCT}_{h,i,0} &= \begin{Bmatrix} \mathrm{DCT} & \text{if } \mathrm{doorStatus}_i = \text{open} \\ 0 & \text{otherwise} \end{Bmatrix} \\
\mathrm{DTT}_{h,i,0} &= 0 \\
\mathrm{DOT}_{h,i,0} &= \text{door open time of lift } i \\
\mathrm{runtime}_{h,i,0} &= \text{trip time from the initial position of lift } i \\
&\quad\ \text{until the first destination}
\end{aligned}$$

5 Genetic algorithm

The Elevator Group Control algorithm has been implemented using GAs. The loop is closed after each run. In this way, the tool is dynamic but does not involve reconsideration of how to improve the dynamic call allocation. Lift states are simulated during allocation and future costs of the system are estimated. The algorithm assumes that there is only one passenger for each button push. This will not reflect reality when, for example, two or more passengers arrive simultaneously at a landing and call the lift.

The GA is implemented to solve the dispatching problem as a two-level optimization problem. The first level deals with the assignment of all passenger requests to various lifts. The GA is used to optimize this assignment. The second level is carried out as a TSP in which the routes of each lift are optimized using a greedy algorithm that takes into account the position of the lifts and uses a heuristic function to create the routes using the rule of assigning the first call to attend as the closest one to the lift position. Once we have assigned the first closest call to the lift, the resting calls are ordered according to its direction. If there is a cab call for the lift, the direction is already set to attend that cab call. If it is necessary the direction of the lift is reversed to attend calls in the opposite direction

5.1 First level

A GA (Goldberg 1989) is used with the following steps: initialization, evaluation of the population, selection, and reproduction (crossover and mutation).

- *Initialization*

Individuals are coded considering all the requested landing calls when the system calls the algorithm. Note that, in the context of the GA, the term *individual* refers not to a passenger but rather to a possible scheduling by which to respond to landing calls. Thus, the length of the array holding an individual will be the number of landing calls that have to be attended. The landing calls are put in ascending order with respect to the floor where they have been requested. For example, for a building with six floors and two lifts in which four different passengers have called a lift on floors 1, 3 and 5, the individuals of the GA should be arrays of four elements (one for each call). Figure 1 shows this process of coding the individuals of the GA. The first individual is formed assigning the first call of floor 1 to the lift A, the second call of floor 3 to the lift B, the third call of floor 5 upwards to the lift A and the fourth downwards to lift B. In each floor, we consider there is a two button panel for the passenger to decide his travelling direction, upwards or downwards. In case both

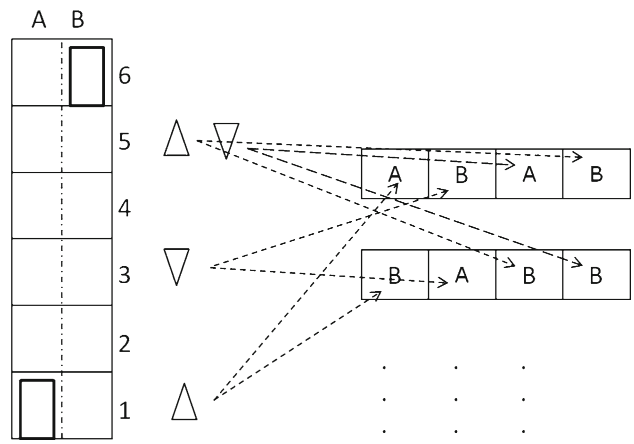


Fig. 1 Coding of the individuals of the GA based on the request calls

buttons are pushed, two different landing calls are registered from the same floor, and two different lifts could be assigned. Therefore, in this example, two different genes are created in the individual.

Which lift is assigned to which request is random for a given individual; however, each request will be assigned once and once only for one individual. In this way, all possible assignments (polymorphisms or genes) are made available in the starting population.

- *Evaluation*

The fitness of each individual is calculated as the inverse of the square of the sum of all the expected waiting times of all the landing calls of the system, as shown in Eq. 2. For optimization with a GA, maximization, as opposed to minimization, of a fitness function is recommended by Goldberg (1989). Then, the fitness of the individual h is calculated as in Eq. 4:

$$\text{Fitness}_h = \frac{1}{\text{EWT}_h^2} = \frac{1}{\left(\sum_{i=1}^{I}\sum_{j=1}^{\text{Lc},i-1}\text{EWT}_{h,i,j}\right)^2} \quad (4)$$

- *Selection*

During successive generations, a proportion of the existing population is used to breed a new generation. Individual solutions are selected through a fitness-based process, where fitter solutions are more likely to be selected. But all the selection methods are always designed so that a small portion of less fit solutions are selected. This helps to keep the diversity of the population, preventing premature convergence on poor solutions.

The selection algorithm implemented in this work is based on the roulette wheel selection.

First, the sum of the fitness of all the individuals is calculated. Then, a random quantity of the total fitness is obtained. The algorithm sequentially adds the individuals fitness, and stops when the partial sum of finesses of the individuals overload this random value. Through this process the probability of being selected ps_h of an individual h in the population formed by N individuals is computed according to the relation of its fitness to the overall population fitness:

$$ps_h = \frac{\text{Fitness}_h}{\sum_{j=1}^{N} \text{Fitness}_j} \tag{5}$$

where, $0 \leq ps_h \leq 1$.

This is similar to a roulette wheel selection, in which more probability exists to be selected for the slots that cover more proportion of the total fitness.

Using Eq. 5, a proportion (ps_h) of the wheel (the overall population fitness) is assigned to each of the possible individuals. The higher the fitness of an individual, the larger its “segment” of the “roulette wheel”, and thus the greater the probability ps_h it has of being selected. The selection process is analogous to spinning a roulette wheel and seeing where the ball rests. The implementation of this process is completely described in Goldberg (1989).

For GA, we are using elitist selection, i.e. the winner individual from the previous population evaluated is added to the new population created. Therefore, the fittest individual of the population has a great probability to be selected in the next generation. Then, the fitness of the best individual of the new population will always be the same as or better than that in the old population.

- *Reproduction*

Once selection has been made, a new population resulting from crossover and mutation applied to the selected individuals is created. From two different selected individuals, according to a certain crossover probability another two new individuals are created.

Crossover is what makes the GA succeed, and in this sense, is the most critical part of the GA.

Crossover operation Certain elements (landing calls in individual arrays) that are common to both parents are inherited by children, while other elements are randomly adopted. In Fig. 2, there are 5 landing calls, for 3 lifts, A, B and C; the parent individuals have two lifts in common (one in the 2nd call and the other in the 4th call) which the children inherit; the other elements are selected randomly.

Mutation operation After crossover, selected individuals are subjected to mutation according to a certain mutation probability. An element (a landing call) of an individual is selected and changed randomly, i.e. the lift assigned to the call is changed. In Fig. 3, there are 5 landing calls, for 3 lifts, A, B and C; a mutation occurs in the 3rd call, where lift C replaces lift B.

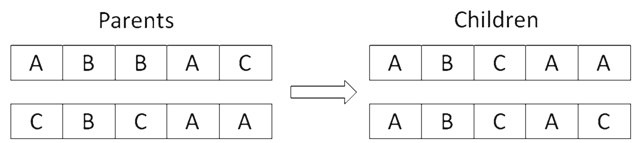


Fig. 2 Crossover operation example

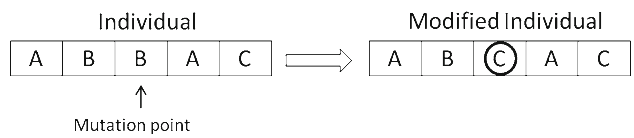


Fig. 3 Mutation operation example

By crossing and mutation, a new population with new individuals is created. Then, the same processes of selection and reproduction are repeated until a target value of fitness is achieved or until the algorithm has undertaken a given maximum limit of iterations. In the lift dispatching problem, an optimal individual has to be found, and thus a decision reached before the next time the system calls the algorithm.

In addition, all the individuals of the GA population are feasible, as there always exist lift routes for any assignment of a set of landing calls to a set of lifts. This exists regardless of the number of stops or number of changes of direction the lifts have to complete.

5.2 Second level

This level optimizes the route of each lift. At the first level, the landing call assignment is made, and then at the second level, the optimum route that minimizes the average waiting time is created.

To minimize the average waiting time, all the possible routes are evaluated.

Possible routes are created on the basis of the landing calls assigned to the lift. It is necessary to take into consideration all the features of the lift: its situation, the number of passengers being carried, the car call, the landing call, its direction, etc. We considered and implemented some additional assumptions (apart from the Closs rules) for this dynamic system:

- A car call has higher priority than a landing call, if there is no call from a floor the lift does not have to stop on that floor.
- If the lift is moving when the algorithm is called, then the lift's destination floor at that moment can be regarded as defined and as a restriction.

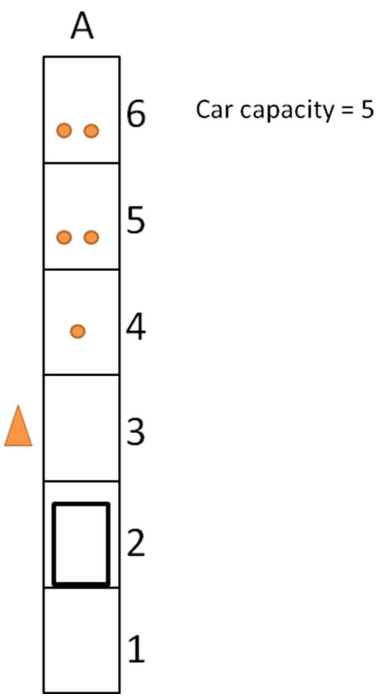


Fig. 4 Example of how to choose a call depending on the capacity of the lift car

- The time to the destination floor is calculated and then the possible floors to stop at subsequently are those where landing calls have been made and those indicated by car calls from within the lift.
- From the possible stops, the closest to the lift is identified. If there is enough space to carry a new passenger in the car, the closest call will be selected as the next destination. If there is not space, the car call with the shortest distance will be the selected (i.e. any landing calls are ignored). Figure 4 shows these possibilities for a car with capacity for five passengers. In the left building there is a lift with five passengers and another passenger has made a landing call from the third floor. The closest call is the landing call, but, because the car is full, the car call for the fourth floor is selected as the next destination. The lift on the right of Fig. 5 has only four passengers, and the third floor is selected as the next destination.
- Once a stop has been selected, all the calls for that floor are attached to the route. For example, two passengers may make the same car call and a landing call may coincide with a car call. In this way, one stop may serve more than one passenger.

The above process for selecting the next destination is iterated until there are no more calls to be attended. All the calls will then be in the route, and the sum of the waiting time of all the passengers for that route moving in that lift is calculated according to Eq. 2.

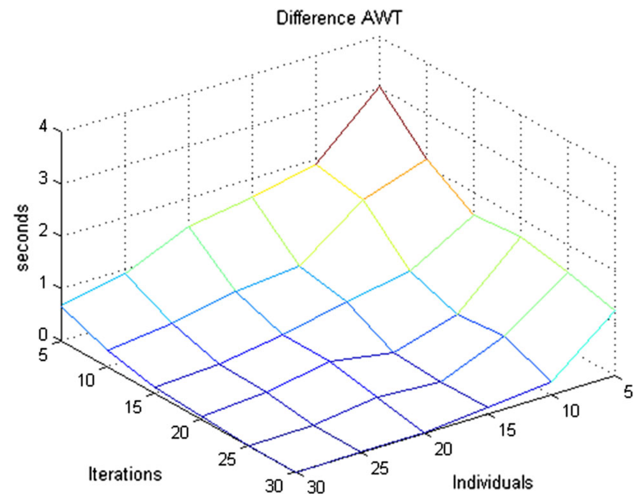


Fig. 5 Differences between the average waiting time of the optimal solution and the average waiting time of different GAs with 5–30 individuals and 5–30 iterations

6 Simulations

6.1 Static simulation with the raw GA

We analysed the performance of the raw GA without adjustments in a static situation using MATLAB version R2011a. The main goal of this simulation was to find the optimal number of individuals in the population and the optimal number of iterations for the GA. For only 1 set of landing calls, we ran the algorithm to solve the assignment trying to find the optimal solution. This analysis was done to adjust the number of iterations and number of individuals in the GA population. The simulation involved a building of 20 floors with 4 lifts, with 5 landing calls, 2 upwards and 3 downwards. In this situation, there are 4^5 possible solutions (i.e. 1024). We found the optimal exact solution and then we ran the GA with different combinations of number of individuals and iterations from 5 to 30, with a crossover probability of 0.8 and a mutation probability of 0.1. Figure 5 shows the differences between the average waiting time of the passengers for the different GA configurations and the real optimal solution. The more the individuals and iterations, the best the performance of the GA, as expected.

6.2 Simulations of GA with adjustments

To simulate different real configurations and obtain performance indicators, we used Elevate, the well-known vertical transport simulation software mainly based on Dr. Peter' developments, and released from Peters Research Ltd. (a detailed description can be found in Caporale (2000) and at https://www.peters-research.com). In these configurations, the GA has to manage the real dynamic dispatching prob-

Table 1 List of building configurations $C1$, $C2$

	No. of cars	No. of floors	Average population/floor for populated floors	Car capacity (kg)
$C1$	6	12	80	1600
$C2$	6	33	30	850

lem, in which the dispatcher has to be ran anytime the state of the systems changes having new landing calls.

We selected two different configurations shown in Table 1. The second configuration is the most challenging one due to the number of floors of the building and the limited capacity of the lifts that increases the number of stops and creates longer trips. The parameters for the simulations are the default ones in Elevate. The GA was implemented using Elevate 8's option to insert external code in C (according to the manual, http://www.peters-research.com/index.php?option=com_content&view=article&id=96&Itemid=91). Intensity was determined by a step profile with a minimum demand of 11 % and a maximum demand of 13 % of the population per 5 min. Profiles used were, outgoing, incoming and interfloor.

With these two different configurations, $C1$ and $C2$, we ran the GA with one or several adjustments explained in the following Sect. 6.3.

In these simulations using Elevate, due to the software and hardware constraints, we had to use ten individuals and ten iterations for the real GA implementation. The crossover and mutation probability was set as 0.8 and 0.1, respectively. The crossover probability was quite high as we wanted the GA to evolve fast taking to account for the decision time available. The issue of the dispatching problem has to be solved in less than half a second (Sorsa et al. 2009; Siikonen 1997b). As the time to solve the problem is limited, these values were experimentally determined by the decision time made available by the system processor constraints. Consequently, the values are strongly dependent on the hardware and software limitations of the system in which the GA could be implemented.

6.3 Adjustments to the GA

We evaluated several modifications to the basic GA described above. The adjustments, which can be activated and deactivated, are described below:

- *Stability*(S) Two adjustments involve the concept of stability(S). The first adjustment prevents reassignment of a landing call that is assigned to a lift that is already in the process of stopping to attend that call, as suggested in Siikonen (1997b). This modification avoids the creation of individuals with abnormally high waiting times due to such tardy reassignment. The second adjustment fixes the assignment of a landing call to an empty lift as soon as that lift has started moving to attend the call. Thus, such a landing call cannot be reassigned (i.e. this "gene" of the individual is fixed). This modification prevents empty lifts from changing direction. The adjustments are applied when car load is less than 80 % maximum capacity.
- *Seeding* (*Sd*) This adjustment improves creation of the initial population of individuals each time the system calls the GA algorithm. One of the individuals of the population is formed using topological criteria Elbaum and Sidi (1996), by assigning landing calls to the nearest lifts that satisfy all the requirements and assumptions explained in Sect. 4. In this way, the adjustment provides the GA with a reasonable preliminary solution so that the algorithm can evolve to better solutions faster.
- *Last best individual* (*LBI*) Through this adjustment, each time the system calls the algorithm, a modified version of the best individual that was obtained in the previous call is included in the initial population of individuals. The previous best individual is modified by deleting any attended calls and adding assignments for any new landing calls randomly. This adjustment allows the GA to reach convergence to a better solution faster.
- *Penalization* (*P and P3*) Excessively long passenger waiting times should be avoided. This problem can be tackled with constrained optimization Kuri-Morales et al. (2002). Equation 6 shows the modification, addition of a penalty term, to the evaluation of individuals. Equation 7 expresses the objective function that has to be minimized, the EWT with penalty (EWTP).

$$\mathrm{EWTP}_h = \mathrm{EWT}_h + \mathrm{Penalization}_h \tag{6}$$

$$\mathrm{individual}_k = \min(\mathrm{EWTP}_h) \tag{7}$$

Two different penalization strategies have been implemented. First, a death penalty, P, that adds a big term to the fitness function for any individuals with a passenger waiting time bigger than a fixed amount (30 s as a threshold), as shown in Eq. 8. When using P penalization, the Penalization$_h$ term of Eq. 6, takes the value of the P_h, as defined in Eq. 8:

$$P_h = \begin{Bmatrix} 1000 \times I & \text{if any } \mathrm{EWT}_{h,i,j>30} \\ 0 & \text{otherwise} \end{Bmatrix} \tag{8}$$

The value 1000 was selected as an arbitrary value higher than any of the passengers' longest waiting times observed in the simulation for any of the algorithms tested. As the EWT is the sum of the waiting times for the passengers for all the lifts, we penalize the GA individuals with EWTs bigger than 30 s proportionally to the number of cabs in the installation, I. Then the penalization term depends on the number of lifts in the buildings. In addition, 30 s is the threshold value that the average waiting time should not exceed (Sorsa et al. 2003).

The second penalization strategy uses a dynamic penalty involving consideration of the average real waiting time (RWT). RWT is the average waiting time for current passengers calculated from the time that they pressed a call button. If the EWT of an individual is smaller than a certain reference value, the penalty $P3$ is zero, see Eq. 9. Otherwise, $P3$ is calculated as the square of the difference between the individual's EWT and the reference value. The reference value is computed as the higher of RWT and a threshold value (30 s in this study). The penalty is calculated for each lift i and for all landing calls, and so, under this penalization strategy, the Penalization$_h$ term of Eq. 6 is given by:

$$P3_h = \sum_{i=1}^{I} \sum_{j=1}^{\mathrm{Lc},i} \max(0, (\mathrm{EWT}_{h,i,j} - \max(\mathrm{RWT}, 30)))^2 \qquad (9)$$

The inclusion of these penalties in GA fitness calculation precludes evolution of individuals with long EWTs.

6.4 Statistical analysis

We analysed the GA and the effects of the adjustments in two phases:

1. *Individual effects over the GA* First, to compare the effects over the GA planning tool of the adjustments commented before, six different versions of the GA were considered: GA without any adjustment, GA using the last best individual technique (GA + LBI), GA with seeding (GA + Sd), GA with dynamic penalization (GA + $P3$), GA with stability (GA + S) and GA with all adjustments (GA + LBI + Sd + S + $P3$ = GA + ALL). These algorithms were applied to the building configuration 1, $C1$ (6 cars and 12 floors) under three different passenger profiles: outgoing, incoming and interfloor. Each tested profile had 400 landing calls to attend to. The resulting waiting times of the different GA versions were compared using a one-factor ANOVA test. Distributions of the passengers waiting times were also present in histograms for visual comparison. Preliminary tests performed for P and $P3$ penalties showed that for $C1$ building configuration, the range for the long waiting times for the passengers was more than double for P than for $P3$ penalization. Given this drawback, P was discarded from subsequent testing.
2. *Combined effects over the GA* Six different combinations of adjustments were also evaluated with a more challenging building configuration ($C2$). The traffic profile applied was interfloor so that the raw planning algorithm would not be influenced by its lack of a module to predict the number of passengers behind each call. The adjustment combinations were: GA + S, GA + S + P, GA + S + $P3$, GA + S + LBI + Sd, GA + S + LBI + Sd + P, and, GA + S + LBI + Sd + $P3$. We ran ten different simulations on the same configuration for each of the different versions of the GA. The number of landing calls was 350 per simulation. The resulting waiting times of the different GA versions over the 10 simulations were compared using a one-factor ANOVA test.

6.5 Comparison with other algorithms

We compared the best version of the GA with combined effects to two well-known commercial algorithms available on ELEVATE based on rules called group collective (GC) and estimated time of arrival (ETA).

The pre-stage EGCS presented in this paper (i.e. the modified version of the GA) could improve using additional passenger traffic modules to handle high demanding system configurations as up-peak (i.e. most of the passengers entering the building) or down-peak (i.e. most of the passengers leaving the building). Under the assumption that one request corresponds to exactly one passenger, we have compared the GA with all the adjustments to GC and ETA algorithms for 3 different step profiles with increasing handling capacity (HC) from 11 to 13, being the theoretical HC 10 %. The tested profiles were STEP1 (45 % Incoming–45 % Outgoing–10 % interfloor), STEP2 (0 % Incoming–100 % Outgoing–0 % interfloor) and STEP3 (80 % Incoming–15 % Outgoing–5 % interfloor). All of them were for building configuration $C2$.

7 Results

7.1 Individual effects

Tables 2, 3 and 4 show for outgoing, incoming and interfloor passenger profiles, respectively, the mean and standard deviation of the passenger waiting times for basic GA, GA with one adjustment, and GA + ALL. Relative to the other algorithms, GA + S and GA + ALL had significantly lower means and standard deviations ($p < 0.05$) (Table 2). The stability adjustment was the most effective single adjustment for decreasing the average waiting time of passengers.

Table 2 Mean/standard deviation values (seconds) of the waiting time for the outgoing passenger profile in configuration 1

GA	GA+LBI	GA+Sd	GA+P3	GA+S	GA+ALL
82,9/91,0	76,1/72,2	78,1/81,6	69,8/67,4	50,5/48,6	24,8/21,7

Two vertical lines joined together with a horizontal line and an asterisk above or below the horizontal line in the midpoint between two methods indicate significant differences between them

* $p < 0.05$ (one-way ANOVA). $n = 400$

Table 3 Mean/standard deviation values (seconds) of the waiting time for incoming passenger profile in configuration 1

GA	GA+LBI	GA+Sd	GA+P3	GA+S	GA+ALL
33.9/44.2	36.6/45.6	33.9/81,6	19.9/24.7	18.5/24.0	17.5/21,0

Two vertical lines joined together with a horizontal line and an asterisk above or below the horizontal line in the midpoint between two methods indicate significant differences between them

* $p < 0.05$ (one-way ANOVA). $n = 400$

GA + ALL had the lowest means and standard deviations with all passenger profiles and was significantly better than even GA + S ($p < 0.05$) with the outgoing passenger profile (Table 2).

Results for outgoing, incoming and interfloor traffic profiles were similar (Tables 2, 3 and 4): GA + $P3$, GA + S and GA + ALL performed better than the other algorithms. The stability and the penalization factor $P3$ adjustments were those that most effectively reduced waiting time.

To visualize the impact of each of the adjustments on passenger waiting time, Figs. 6, 7 and 8 show the results of one specific execution of each adjustment in a histogram. With the GA + ALL algorithm waiting times are more condensed around shorter waits.

7.2 Combined effects

Ten different simulations on the same configuration were run for each of the different versions of the GA. Table 5 shows the mean and standard deviation of waiting time for the six different algorithms considered. Significant differences in paired test were found between GA + S + LBI + Sd, GA + S + LBI + Sd + P, GA + S + LBI + Sd + $P3$ on the lower performance side and GA + S, GA + S + P and GA + S +

Table 4 Mean/standard deviation values (seconds) of the waiting times for interfloor passenger profile in configuration 1

GA	GA+LBI	GA+Sd	GA+P3	GA+S	GA+ALL
54.8/50.2	56.7/53.8	60.3/54.24	71.4/71.8	43.6/42.3	41.5/37.5

Two vertical lines joined together with a horizontal line and an asterisk above or below the horizontal line in the midpoint between two methods indicate significant differences between them

* $p < 0.05$ (one-way ANOVA). $n = 400$

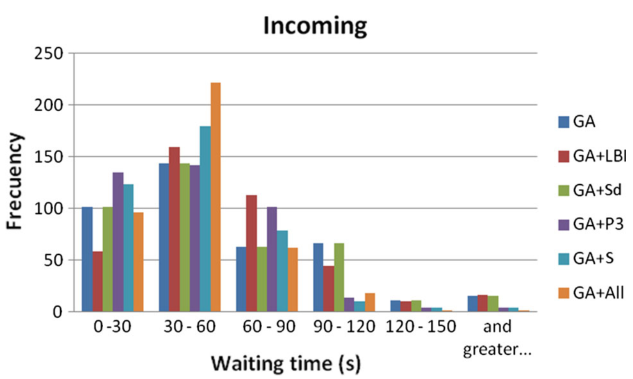


Fig. 6 Histogram of passenger waiting time for the outgoing passenger profile in $C1$ configuration for different adjustments

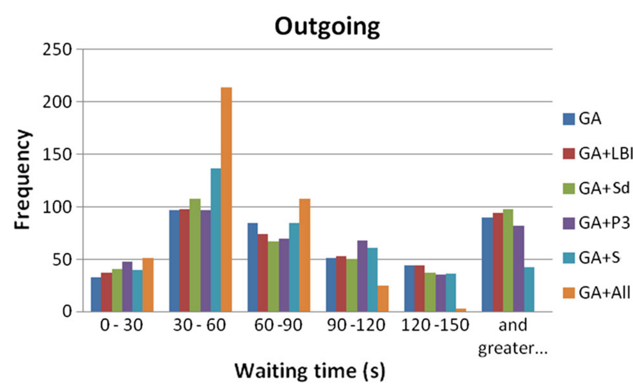


Fig. 8 Histogram of passenger waiting time for the interfloor traffic profile in $C1$ configuration for different adjustments

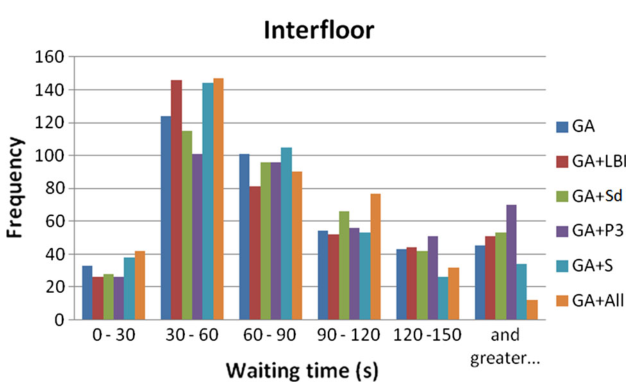


Fig. 7 Histogram of passenger waiting time for the incoming passenger profile in $C1$ configuration for different adjustments

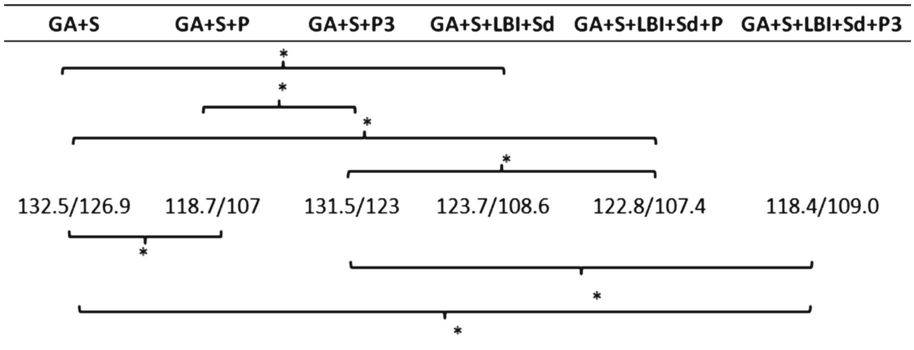
$P3$ on the higher performance side. These differences can be considered to define, in terms of performance at minimizing waiting times, two groups of algorithms for this interfloor traffic. GA + S + P and GA + S + LBI + Sd + $P3$ presented the lowest values of mean and standard deviation of waiting times.

Table 6 shows the results for the ten different executions of each of the six versions of the algorithm; GA + S + LBI + Sd + $P3$ achieved lower minimum and maximum waiting times.

7.3 Comparison with other algorithms

Table 7 shows the results of the comparison of the algorithms GA + ALL, GC and ETA. The table shows the changes in % (with respect to arbitrary values) of the mean and the range of the interval of 10 trials for the average and longest values of several time measures related to the simulation of the performance of the system (waiting time, transit time and time to destination). If we take an arbitrary reference value, Ref_v, and the real value of a certain algorithm A in a certain time measure, $\mathrm{tm\_real}_A$, the changes in % with respect to that value are calculated as:

Table 5 Mean/standard deviation values (seconds) of the waiting times for interfloor passenger profile in configuration 2



Two vertical lines joined together with a horizontal line and an asterisk above or below the horizontal line in the midpoint between two methods indicate significant differences between them

* $p < 0.05$ (one-way ANOVA). $n = 3500$

Table 6 Detailed results for the average waiting time (AWT) in seconds for 10 individual trials of different adjustments for $C2$ configuration

	AWT										Max	Min
GA + S	123	131.1	134.7	119.6	135.9	143.7	131.4	134.2	142.2	128.8	143.7	119.6
GA + S + P	115.5	137.7	129.8	141.3	128.5	132.8	137.6	122	140.5	129.8	141.3	115.5
GA + S + $P3$	129.5	126.4	115.5	117.3	123.5	121.3	131.5	119	123.8	119.9	131.5	115.5
GA + S + LBI + Sd	124.4	114.5	115.8	113.7	123.3	119.3	115.3	116.6	127.3	116.6	127.3	113.7
GA + S + LBI + Sd + P	123.2	112.6	122.5	114.1	125	121.2	121.7	144.9	129.6	122	144.9	112.6
GA + S + LBI + Sd + $P3$	113.3	107.5	115.7	123.6	106.8	119.8	125.5	122.4	131.8	122	131.8	106.8

$$\% \text{ Change in tm}_A = 100 \cdot \left(\frac{\text{tm\_real}_A - \text{Ref}_v}{\text{Ref}_v} \right) \tag{10}$$

On the one hand, negative % values of the table signify improvements in the mean or range of the measurements with respect to the reference values. On the other hand, positive % values denote worse performance of the methods. The most positive, the best performance of a method. The GA + ALL shows the best results in terms of transit time (TT) and most of the times for the longest waiting times (LWT), highlighting the power of genetic algorithm as elevator car routers.

8 Discussion and conclusions

The current work evaluates, for different building and traffic flow configurations, several techniques relevant to a successful implementation of a genetic algorithm in the dispatching problem for vertical transportation.

The different techniques or adjustments have been investigated in isolation of each other and in various combinations. The stability adjustment was the most effective single adjustment for decreasing the average waiting time of passengers, while the $P3$ penalization adjustment also gave significant benefits. The best performance was obtained when all the presented adjustments were applied to the basic GA.

In addition, this algorithm was compared to two commercial ones. Our algorithm led to better results in terms of longest waiting times and transit times, highlighting the power of genetic algorithms as elevator car routers.

Further research is required to develop, implement and evaluate a module to estimate the number of passengers per call. Such a module, which might be based on records of passenger movements, is needed confirm adequate performance of the GA with outgoing and incoming passenger profiles.

The study uses the one-factor ANOVA test to compare techniques; this test might be used to select which version of the algorithm is most appropriate. All the techniques proposed here can be readily implemented in commercial software to improve the performance of GAs.

Table 7 Changes in % in the mean/range of the different variables (average waiting time, longest waiting time, average transit time, longest transit time, average time to destination and longest time to destination) with respect to arbitrary values over 10 runs of three different step profiles (STEP1, STEP2 and, STEP3) for the algorithms GA + ALL, GC and ETA

	STEP1 (45 % Incoming–45 % Outgoing–10 % interfloor)			STEP2 (0 % Incoming–100 % Outgoing–0 % interfloor)			STEP3 (80 % Incoming–15 % Outgoing–5 % interfloor)		
	GA + ALL	GC	ETA	GA + ALL	GC	ETA	GA + ALL	GC	ETA
Average waiting time	11.3/0.4	**−6.3/7.2**	−3.7/−1.6	20.4/−8.7	**−13.4/8**	16.6/−19.3	**−20.7/24.7**	18.3/337.7	−18.6/112.3
Longest waiting time	28.4/5.6	**−2.3/−0.6**	33.4/18.5	**−12.8/−13**	5.2/108.8	51.1/45.3	**−15.9/−38.9**	14.1/35.5	−14.3/−12.6
Average transit time	**−4.0/38.3**	12.5/−5	3.5/−1.7	**−8.0/−60**	8.5/13.3	3.6/−20	**−1.2/6.7**	0.5/−13.3	−1.0/16.7
Longest transit time	**−1.2/19.5**	6.4/7.2	5.2/77.8	**−9.1/−0.2**	10.4/−28.3	4.3/2.5	**−3.7/−17.3**	3.2/10.3	0.1/−33.3
Average time to destination	2.2/6.8	**−1.0/10**	−3.2/−23.2	13.6/−8	**−1.7/13**	15.6/−24.3	**−2.4/−47.9**	31.3/71.6	−0.5/−16.4
Longest time to destination	24.9/−1.7	**−1.4/−0.1**	29.3/70.2	**−7.0/−24.3**	7.3/110.9	47.3/34.9	**−5.5/−18.7**	22.8/91.7	−4.8/14.5

STEP1 = 45 % Incoming–45 % Outgoing–10 % interfloor, STEP2 = 0 % Incoming–100 % Outgoing–0 % interfloor, STEP3 = 80 % Incoming–15 % Outgoing–5 % interfloor
The bold numbers are the best results of each configuration

References

Alander JT, Ylinen J, Tyni T (1994) Optimizing elevator control parameters. In: Proceedings of the second finnish workshop on genetic algorithms and their applications (FWGA)

Caporale RS (2000) Elevate traffic analysis software (eliminating the guesswork). Elevator World 48(6):118–124

Closs GD (1970) The computer control of passenger traffic in large lift systems. PhD Thesis. University of Manchester Institute of Science and Technology

Cortés P, Larrañeta J, Onieva L (2003) A genetic algorithm for controlling elevator group systems. In: Mira J, Álvarez JR (eds) Artificial neural nets problem solving methods. Lecture Notes in Computer Science, vol 2687. Springer, Berlin, Heidelberg, pp 313–320

Cortés P, Larrañeta J, Onieva L (2004) Genetic algorithm for controllers in elevator groups: analysis and simulation during lunchpeak traffic. Appl Soft Comput 4(2):159–174

Cortés P, Fernández JR, Guadix J, Muñuzuri J (2012a) Fuzzy logic based controller for peak traffic detection in elevator systems. J Comput Theor Nanosci 9(2):310–318

Cortés P, Onieva L, Muñuzuri J, Guadix J (2012b) A viral system algorithm to optimize the car dispatching in elevator group control system of tall buildings. Computers Industrial Engineering 64:403–411

Echavarria J, Frenz CM (2009) Improving elevator call time responsiveness via an artificial neural network mechanism. In: Systems, Applications and Technology Conference, 2009. LISAT '09. IEEE Long Island. IEEE, Farmingdale, pp 1–3

Elbaum R, Sidi M (1996) Topological design of local-area networks using genetic algorithms. EEE/ACM Trans Netw 4(5):766–778

Goldberg D (1989) Genetic algorithms in search, optimization, and machine learning. Addison-Wesley Professional

Imrak CE, Özkirim M (2004) Neural Networks application in the next stopping floor problem of elevator systems. J Eng Natural Sci

Imrak CE, Özkirim M (2006) Determination of the next stopping floor in elevator traffic control by means of neural networks. J Electr Electr Eng 6(1):27–33

Kim CB, Seong KA, Lee-Kwang H, Kim JO (1998) Design and implementation of a fuzzy elevator group control system. IEEE Trans Syst Man Cybern Part A Syst Hum 28(3):277–287

Kuri-Morales A, Gutiérrez-García J, Sucar L, Battistutti O (2002) Penalty function methods for constrained optimization with genetic algorithms: a statistical analysis. In: MICAI 2002: Advances in artificial intelligence. Lecture notes in computer science, vol 2313. Springer, Berlin, Heidelberg, pp 108–117

Liu J, Bai ZL, Gu MH, Zhang X, Zhang R (2014) The research of multicar elevator control method based on PSO-GA. Appl Mech Mater 556–562:2418–2421

Peters RD (1995) Ideal lift kinematics complete equations for plotting optimum motion. In: Proceedings of ELEVCON 1995. Elevator Technology, vol 6, pp 165

Rosso Mateus AE, Soriano JJ (2008) Sistema de control inteligente para un grupo de elevadores. Ciencia e Ingeniería Neogranadina 18(2). **ISSN e 0124-8170**

Siikonen M-L (1997a) Elevator group control with artificial intelligence Systems analysis laboratory research reports. KONE Corporation,Helsinki University of Technology, Helsinki

Siikonen M-L (1997b) Planning and control models for elevators in high-rise buildings. Helsinki University of Technology, Systems Analysis Laboratory, Research Reports A68

Sorsa J, Siikonen M-L, Ehtamo H (2003) Optimal control of double-deck elevator group using genetic algorithm. Int Trans Oper Res 10(2):103–114

Sorsa JS, Ehtamo H, Siikonen M, Tyni T, Ylinen J (2009) The elevator dispatching problem. Transp Sci

Tartan, EO, Erdem H, Berkol A (2014) Optimization of waiting and journey time in group elevator system using genetic algorithm. In: Proceedings of INISTA 2014—IEEE international symposium on innovations in intelligent systems and applications. Art. no. 6873645, pp 361–367

Tyni T, Ylinen J (2006) Evolutionary bi-objective optimisation in the elevator car routing problem. Eur J Oper Res 169(3):960–977