Mondragon Unibertsitatea Biblioteka Biblioteca

biblioteka@mondragon.edu

Beamurgia, M., Basagoiti, R., Rodríguez, I. et al. Improving waiting time and energy consumption performance of a bi-objective genetic algorithm embedded in an elevator group control system through passenger flow estimation. Soft Comput (2022).

This version of the article has been accepted for publication, after peer review (when applicable) and is subject to Springer Nature's AM terms of use, but is not the Version of Record and does not reflect post-acceptance improvements, or any corrections. The Version of Record is available online at:

https://doi.org/10.1007/s00500-022-07358-4

Improving Waiting Time and Energy Consumption Performance of a Bi-objective Genetic Algorithm embedded in an Elevator Group Control System through passenger flow estimation

Authors and affiliations:

M Beamurgia[1], R. Basagoiti[2], I Rodríguez[3,4], V Rodríguez[3]

(1) Fagor AOTEK, R&D&i Department

(2) University of Mondragon, Department of Electronics and Computing

(3) University of Navarra, Department of Business Administration

(4) University of Navarra, Institute of Data Science and Artificial Intelligence (DATAI)

Abstract

Passenger waiting time is a significant issue related to the quality of service of a multiple lift system; however, energy consumption reduction is also an important concern in the lift industry. In this paper, we evaluate different versions of a genetic algorithm (GA) published previously by the authors with several relevant adjustments for the lift dispatching problem to minimize passenger waiting time and/or energy consumption. To the raw GA with adjustments (that works under the assumption one call-one passenger) we incorporated several elements: a passenger-counting module using origin-destination (OD) matrices, and the activation of certain policies (zoning and/or parking) under different detected traffic profiles (up-peak, interfloor or down-peak profiles). Besides, we added a proportional integral controller (PI) to assign different weights to passenger waiting time and energy consumption to evaluate the performance of our GA. Different versions of this GA, minimizing passenger waiting time and/or energy consumption, were compared among them and to a conventional control algorithm using three different types of simulated profiles: a mixed one, three well-known full day office profiles, and three different step profiles. The results showed that the bi-objective GA version with the estimation of the number of passengers behind a call, i.e., the passenger forecasting, together with the parking policy for up-peak or down peak conditions significantly improved performance of passenger waiting time, and in some cases in energy consumption as well. The addition of the PI controller to the GA proved to be especially useful when the system was under a high intensity traffic demand. The advantages of all these elements to forecast the passenger flow and detect the traffic profile to help the controller shows unquestionable benefits to minimize passenger waiting time and energy consumption.

Keywords: Genetic algorithm, Elevator Dispatching Problem, Passenger Flow Patterns, Transportation.

1. Introduction

The lift dispatching problem is a real time optimization problem similar to high-rack warehouses and dispatching service of vehicles where the assignment changes dynamically and the decision must be made before all the information is known (Hiller 2011; Bolat and Cortés 2011; Ruokokoski et al 2015). The decision adopted by this type of system can be considered revocable, since assignments can be changed at the point at which a lift starts to decelerate to serve a passenger call. As a new passenger request arrives, the system computes a new schedule for the current set of requests, replacing the old schedule with the new one, following this schedule until it is finished or replaced. With conventional control, the lift dispatching problem suffers from a lack of information, due to conventional button panel characteristics. Outside the lift, only upwards and/or downwards landing calls can be made and, inside the lift, a car call can be made indicating the destination floor of the passenger. When a passenger makes a landing call, the destination floor is unknown, and therefore the system will only know the direction the passenger wishes to travel. In the literature, it is commonly accepted that a Poisson process can be used to model the arrival of passengers (Kuusinen et al. 2012; Liu et al 2011; Peters et al. 1996). Later research suggests that people move around and use lifts in batches. As well as Poisson, Artificial Neural Networks (ANN) have been widely used for predicting incoming passengers and the next stopping floor (Imrak and Özkirim 2004; Imrak and Özkirim 2006).

Passenger flow estimation processes are important to provide inputs to the Elevator Group control System (EGCS), improving the work done by the controller at the time of assigning passenger requests to cabs. This forecasting of the number of passengers helps the controller to obtain a better plan, minimizing the waiting time and moving more people per time unit. Passenger flow is also helpful when minimizing energy consumption. As fast response times are required from a lift dispatching algorithm, the system providing the estimated passenger flow to the controller needs to be designed as a real time algorithm.

For conventional button panels, once a passenger makes a call, there is no option for additional passengers to make any new call. In this work, this is the type of button panels we have worked with, the number of passengers behind a call cannot be precisely detected. Problems arise when a car arrives to answer a call and it has insufficient capacity to transport all the passengers there. The remaining passengers need to re-register their call after the cab leaves the floor. However, the information missing during this process can be estimated using the assumption that continuously operating systems should exhibit some level of repeatability. Assuming that passengers travelling through specific buildings follow approximately the same pattern day after day, we can take advantage of this stability. This repeatability provides data that could be

preserved and analysed to learn a passenger flow pattern. A system that shows this repeatability can be analysed, and information about completed trips obtained, looking for a passenger flow pattern (Bera and Rao 2011; Ji et al. 2011; Sherali and Park 2001). Using the destination control button panel, the elevator dispatching problem with uncertain passenger arrivals has also been modelled (Sorsa et al. 2018; Ruokokoski et al. 2016).

This analysis processes and summarises data about passengers involved in many trips happening at the same time frame. The information desired as an output of this estimation process is the number of passengers behind each landing call as well as their individual destination. This is valuable information in order to obtain better routes and balance the relevance between different objective functions in case a bi-objective optimization problem (time, energy) is used (Tyni and Ylinen 2006).

Several bio-inspired algorithms have been applied to deal with the lift dispatching problem. Genetic algorithms have been successfully applied alone (Sorsa et al. 2003; Sorsa et al. 2009; Tartan et al. 2014) or combined with other algorithms, for instance with a particle swarm optimization algorithm (Liu et al. 2014). Cortés et al. (2012b) compares an algorithm based on a virus infection analogy to genetic and tabu search algorithms.

Other optimization algorithms that have been compared to GAs and particle swarm optimization algorithms such as cuckoo search algorithms should be taken into account in developing new approaches to deal with the lift dispatching problems. Cuckoo search algorithm finds optimum solutions in fewer iterations and yields real optima in complex functions (Rajabioun R, 2011; Abed-alguni and Paul, 2018; Alawad and Abed_alguni, 2021). Another new approach of GAs, as continuous GAs were developed to solve optimization problems where the parameters to be optimized are correlated and smooth (Momani et al. 2016; Abo-Hammour et al. 2014).

Our GA implemented in this paper is connected to the continuous GAs in several aspects: the individuals of our algorithm are solution at a higher level, providing the whole dispatching schedule of the system; it gives a smooth dispatching plan for the system anytime new land calls are received; and, is applied to variables that are correlated, as passenger waiting time and energy consumption.

Our GA is used to optimize more than one objective simultaneously: i) passenger waiting time and ii) energy consumption. The minimization of energy consumption is a challenge when a good Quality of Service (QoS) is required. These two objectives represent an essential trade-off for lift companies. It is worth noting that passenger flow estimations could give more relevance to energy consumption when the installation is relaxed or to the passengers waiting time, if the installation is stressed.

In this research, we try to determine the impact of the passenger flow estimation module on the performance of the controller using GAs.

An ECGS based on GA can assign landing to cabs and, for each cab, calculate the best routes to follow in order to attend all the pending requests for that cab. Tyni (2006) proposed a way of balancing the minimization of time and energy consumption depending on the different types of traffic in the building. They used a PI controller (Proportional and time-Integral terms) to provide a specified service level in terms of average waiting time. Other authors (Liu et al. 2011) applied a particle swarm optimization algorithm and a GA to obtain a combined control method. Usually, the controller reallocates landing calls each time it is run (typically in cycles of milliseconds). This 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 during abrupt changes in traffic patterns.

In our previous work (Beamurgia et al. 2015), we addressed the problem of designing a GA based controller as a two-level optimization problem, where i) the first level of the GA dealt with the assignment of all passenger requests to lifts and ii) the second level was solved using the Travelling Salesman Problem (TSP) approach, obtaining optimized routes for each lift. We followed a simple assumption: one landing call corresponds to only one passenger and the destination of this passenger could be any floor. The control system had to properly find the best plan and surprisingly, the GA based controller showed the capacity to give good response for intense passenger flow profiles.

For the passenger flow estimation or forecasting, load weight sensors installed in cabs registered data that was put together with data generated by the ECGS (Beamurgia and Basagoiti 2011; Beamurgia et al. 2011). Not all lifts have a weight sensor installed, but we assumed it was available. Origin destination matrices (OD) were also estimated using the historical data of the system. Some previous contributions in (Caggiani et al. 2013; Cortés et al. 2012; Basagoti et al. 2013; Kuusinen et. al. 2015) were also considered for this work. The developed system worked towards a complete information system, recording, processing and providing all the information related to the problems to those who will use it (Hu et al. 2010).

For the present work, we evaluate the performance of different versions of the GA with adjustments, presented in (Beamurgia et al. 2015), when we use the aggregation method to minimize the passengers' waiting time and/or the energy consumption (the objective function is a linear combination of these variables). We vary the coefficients of the objective function of the GA using a PI controller, as in (Tyni and Ylinen 2006), but with the difference of using it with the forecast of the passenger data flow of the system. Besides, we evaluate the performance of the

GA when we add a passenger estimation module and specific policies like zoning and/or parking, useful to detect and handle different traffic profiles (up-peak, interfloor or down-peak profiles).

This comparison of the performance of these different versions of this GA considers many objective functions related to: i) passenger waiting time, ii) energy consumption, or iii) both.

Moreover, we provide a comparison of three different versions of the GA compared to a conventional control algorithm (CGC) under three different full-day office profiles and three different step traffic profiles.

2. Material

2.1. Simulations

To simulate different real building configurations and obtain performance indicators to test the system, we used Elevate, the well-known vertical transport simulation software from Peters Research Ltd. (a detailed description can be found at https://www.peters-research.com). The building characteristics used for testing are explained in Table 1. For this building, the theoretical Handling Capacity (HC) is 10.1%, so when the profile increases up to 13% of demand intensity, the building will be saturated, working under high demand conditions.

No. Of Lifts	Capacity	Speed	Acceleration	Jerk	No. Of Floors
6	580kg	2.5m/s	0.8m/s^2	1m/s^3	31

Table 1. Building general characteristic

Three different types of passenger flow profiles were used to test the different GA versions/variants: a) one mixed profile, b) three well-known full day office traffic design profiles and c) three different step profiles.

The mixed profile used can be seen in Table 2. This profile was generated mixing all the different types of traffic profiles. The number of passengers of this profile is 7650. The first four periods are up-peak traffic profiles, where the demand reaches 13% (% pop. per 5 minutes) in terms of demand intensity. The next two periods (5th and 6th) and the periods 10th and 11th are interfloor traffic profiles. The 7th, 8th and the 9th represent lunch-peak traffic profiles where demand also reaches 13% of demand intensity. Finally, the periods 12th, 13th, 14th and 15th are outgoing traffic profiles.

The three different profiles used for testing were the following: CIBSE full day office profile (CIBSE) (CIBSE 2010), containing about 9,234 passengers, Strakosh full day office profile, with about 9,210 passengers (STRAK) (Sorsa et al. 2009) and Siikonen full day office profile, about 7,329 passengers (SIIK) (Siikonen 1997).

The three different step profiles used to test the system swells in traffic intensity, from 11% to 16%:

- 45% incoming, 45% outgoing and 10% interfloor traffic, about 734 passengers STEP1 (CIBSE 2010)
- 0% incoming, 100% outgoing and 0% interfloor traffic, about 731 passengers STEP2 (CIBSE 2010)
- 80% incoming, 15% outgoing and 5% interfloor traffic, about 726 passengers STEP3 (CIBSE 2010)

Period	%Pop. per 5 mins	%Incoming	%Outgoing	%Interfloor
1	5	85	10	5
2	8	85	10	5
3	13	85	10	5
4	5	85	10	5
5	2	10	10	80
6	2	10	10	80
7	1	50	50	0
8	13	50	50	0
9	1	50	50	0
10	2	10	10	80
11	2	10	10	80
12	5	10	85	5

13	13	10	85	5
14	8	10	85	5
15	5	10	85	5

Table 2. Mixed passenger flow profile

Figure 1 shows the passenger flow profiles of the different scenarios, showing the accumulated number of passengers per 5 minutes in upwards and downwards trips.

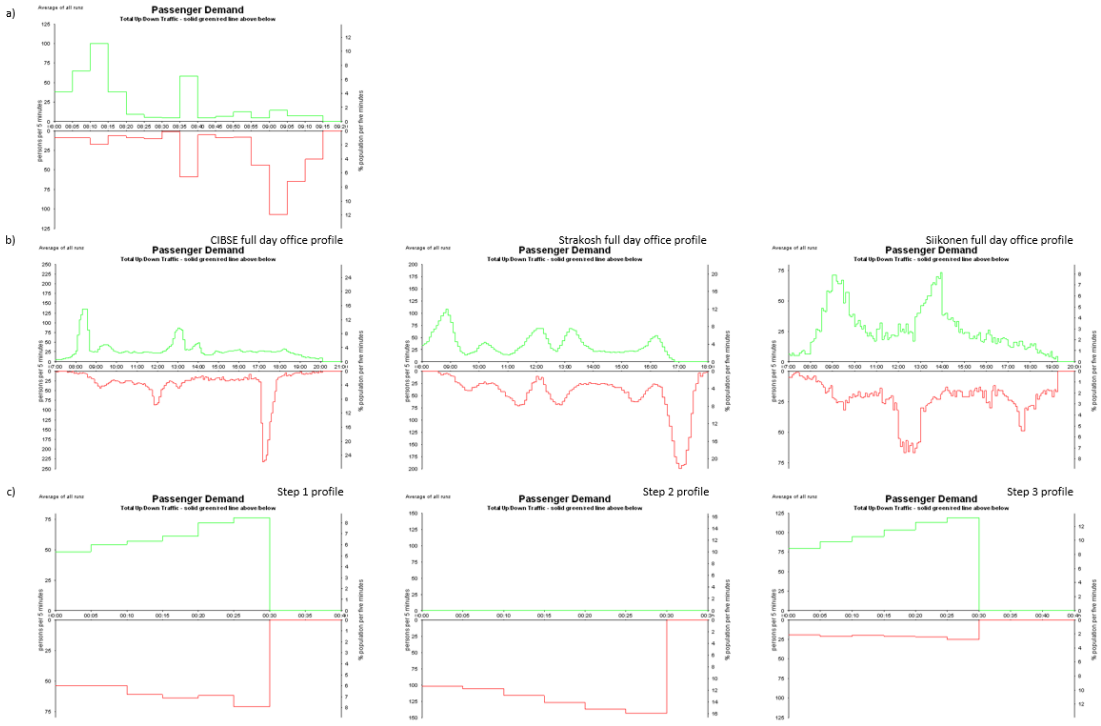


Figure 1. Passenger flow profiles showing the accumulated number of passengers per 5 minutes. Upwards trips in green and downwards trips in red a) a mixed profile b) full day office traffic profiles and c) step profiles

3. Methodology

3.1. Genetic Algorithm

The GA developed in (Beamurgia et al. 2015) addressed the assignment of cars to landing calls. The algorithm reallocates landing calls each time it is run, typically in less than half a second (Siikonen 1997; Sorsa et al. 2009; Tyni and Ylinen 2006). This improves the assignment according to the current state of the system, thereby making the GA more flexible and robust during abrupt changes in traffic patterns.

In our GA, individuals contain the relation of lifts that are going to serve the pending landing calls any time the system calls the algorithm, similar to the GA described in (Tyni and Ylinen 2006). The main difference in our approach is that we did not consider another gene specifying the initial

direction (up or down) of the empty lift as they do. Thus, the term individual refers to a possible scheduling or assignment of lifts to unattended landing calls. The length of the array holding an individual will therefore be the number of landing calls to be attended.

Figure 2 shows an example of how it works. In that figure, a six-floor building and two lifts form the system. Four different passengers make a call on floors one, three and five. Thus, the chromosomes of our GA are arrays of four elements, one for each call. We form the chromosomes considering the calls in ascending order.

In a typical GA, the individuals are randomly created; the lifts are randomly assigned to the calls in the system. In our implementation, we use a different approach in the initial population, we seed the population by creating an individual using topological criteria by assigning landing calls to the nearest lifts.

For instance, the first individual of the population assigns the call of the first floor to the lift A, the calls of the fourth and the fifth floors to the lift B, and the last call of the sixth floor to the lift A (Fig.2.a). As we said before, in each floor there is a two-button panel used by the passenger to decide his travelling direction. In this example, we show three different individuals created for the GA starting population (Fig.2.a).

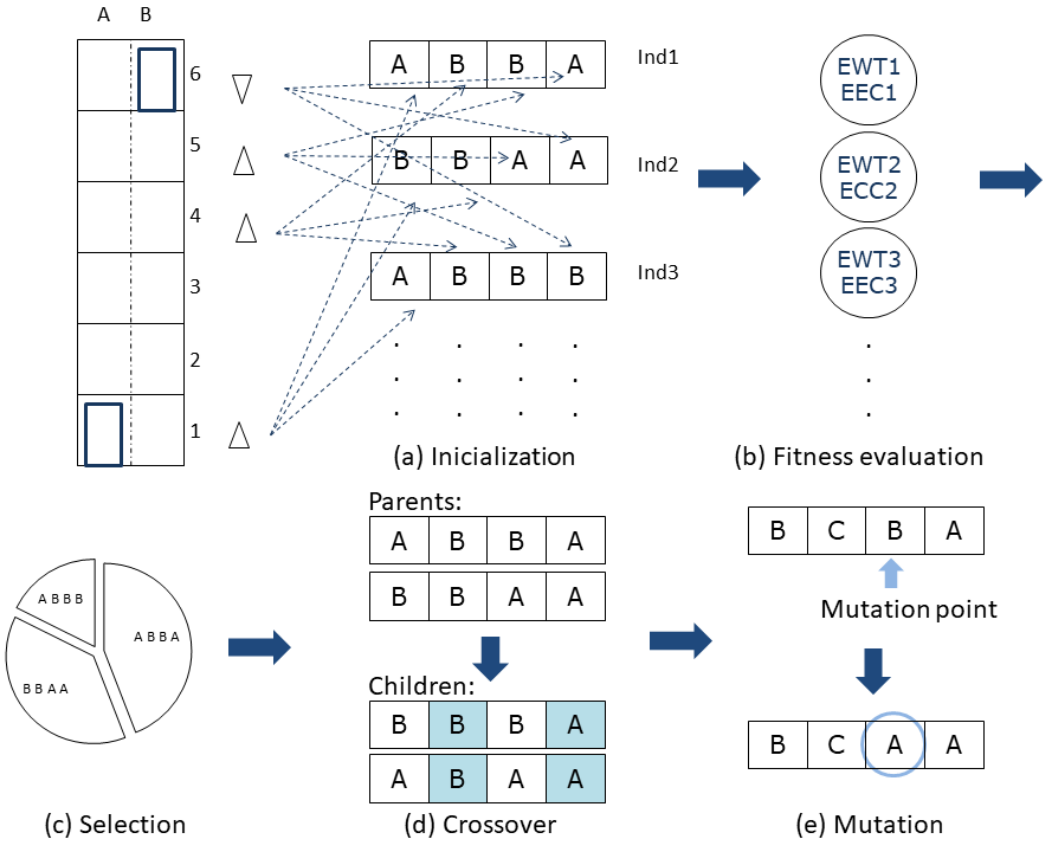


Figure 2. An example of the assignment process of requests to cabs

Once we have the population, we have to evaluate the fitness of each individual. To this aim, we can minimize the global estimated passenger waiting time (EWT) or the estimated energy consumption (EEC) of each individual in the population. For this task, we will find the optimum feasible routes of all the lifts involved in the dispatching plan created by each individual in the population (Fig.2.b). This is the second level of the optimization algorithm, based on the travelling salesman problem (TSP). Once the algorithm knows the location of the lifts and the landing calls, it forms the optimum route of each lift taking into account some restrictions. If there is a car call, the direction of the lift cannot change, as the car call has more priority than a landing call. From all the possible calls that each lift has to attend in the dispatching plan, the lift will attend the closest one in terms of distance (this is the improvement implemented in our GA version called seeding). This process will continue until there are not more calls to attend.
After the fitness evaluation, the selection for reproduction works as a roulette wheel selection, in which the fittest individual of the population has a greater probability of being selected (Fig.2.c) for the reproduction or crossover process.

In the process of crossover, with a certain crossover probability, the assignment of landing calls to lifts that are common to both parents are inherited by children, while other elements are randomly adopted (Fig.2.d). This crossover operator ensures the feasibility of all the individuals in the population.

Next, with a certain probability of mutation, a gene is randomly changed (Fig.2.e). Then, a new generation is created and evaluated again. In addition, elitist selection is used, i.e. the winning individual from the previous population evaluated is added to the new population created. This process is iteratively repeated until 10 iterations, an approximation of the 250 milliseconds period that the controller uses to reallocate landing calls in the system.

. The implementation of this GA is described with detail in (Beamurgia et al. 2015). A pseudocode of the process appears in figures 3 and 4.

1st level Algorithm

1: Initialize the population H of h individuals assigning randomly a lift i to each landing call Lc,i
2: while (not stop condition) do
3: for each individual h do
4: evaluate the individual $EWT_{h,i,j}$ using 2nd level algorithm
5: end for
6: make a selection with the Roulette wheel selection
7: crossover with a certain probability, take 2 individuals (parents) to obtain 2 new individuals (children)
8: mutation with a certain probability, select chromosomes to change randomly its value

9: make an elitist performing, introducing the best individual in the next generation

10: end while

11: Return best individual

Figure 3. Pseudocode of the first level of the GA explained.

Indices and terms are explained in section 3.2

2nd level algorithm

1: Initialize a call list *Lc,i* , with landing calls associated with car calls *c* in lift *i*

2: if (car call in i) do

3: *d* = read direction

4: else

5: *nc* = find nearest call from the lift i (in distance)

6: *ncd* = read direction of the call nc

7: end if

8: while (call list in *Lc,i*) do

9: add to the route *Rc,i* the nearest call in ncd direction

10: update load of the lift i, taking into account estimationj

11: remove the nearest call from the call list *Lc,i*

12: if (no calls in *ncd* direction) or (not load in lift) do

13: *cnd* = change direction

14: end if

15: end while

15: EWT = 0 (estimated waiting time)

16: EEC = 0 (estimated energy consumption)

17: for (route stops in Rc,i) do

18: $EWT_{h,i,j} = estimation_j * \left(EWT_{h,i,j} + DoorOpenTime_{h,i,j} + Runtime_{h,i,j} + DoorTransitTime_{h,i,j} + DoorCloseTime_{h,i,j} \right)$

19: $EEC_{h,i,j} = estimation_j * \left(EEC_{h,i,j} + DoorOpenEnergyConsumption_{h,i,j} + RideEnergyConsumption_{h,i,j} + DoorTransitEnergyCconumption_{h,i,j} + DoorCloseEnergyConsumption_{h,i,j} \right)$

20: end for

21: Return EWT, EEC

Figure 4. Pseudocode of the second level of the GA explained.

Indices, terms and equations are explained in section 3.2

All the individuals of the GA population are feasible, as lift routes always exist for any assignment of a set of calls to a set of lifts. The routes exist regardless of the number of changes of direction of the lifts or the number of stops the lifts have to make.

For our GA we developed four different adjustments to run online to adapt the dynamics of these kind of systems:

- Stability. We developed two different stability adjustments. Both adjustments avoid the reassignment of calls to another lift in two different situations: when a lift is stopping to attend that call or if an empty lift has started moving to attend that call. Those adjustments prevent solutions with high waiting times due to reassignments, preventing empty lifts from changing their direction.
- Seeding. In this adjustment, one solution was formed using topological criteria by assigning landing calls to the nearest lifts. In this way, the adjustment provided the GA with a reasonable initial solution so that the algorithm can evolve to deliver better solutions faster.
- Last best individual. Through this adjustment, each time the system calls the algorithm, a modified version of the best individual that was obtained in the previous system call (deleting any attended calls and adding assignments for any new landing calls randomly) is included in the solution space of the actual system call. This adjustment allows the GA to reach convergence to a better solution faster.
- Penalization. Solutions with excessively long passenger waiting times were prevented by adding a penalty to solutions exceeding certain waiting times.

The crossover probability was quite high (0.8) as we wanted the GA to evolve fast taking into account the decision time available. The mutation probability was 0.1. In the 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 number of ten iterations is an approximation of the 250 milliseconds period that the controller uses to reallocate landing calls in the system.

The version of the GA that incorporates all these adjustments (stability, seeding, last best individual and penalization) is used hereinafter and is referred as GA+ALL.

3.2. Optimization concepts: minimizing energy consumption and waiting time, a biobjective approach

A system that adapts its behaviour to the changing conditions of the building will reduce energy consumption when the installation is under low passenger demand condition but will automatically react and change behaviour once high demand profiles were detected. The fitness function could be implemented as a weighted linear combination of these two estimations, the weights being, W1 and W2. This is not a real multi-objective optimization, which would be very time-consuming, in the $O(MGN^2)$, where M is the number of objective functions, G is the number of generations and N is the population size (Jensen 2003). This time is inadmissible for an EGCS, as the computing time would exceed the time acceptable for obtaining a solution (nearly 250 milliseconds). Therefore, a Weighted Aggregation (WA) method is used to convert the problem into a single objective optimization method, as in (Strakosh 2007). The fitness function to minimize for x, a flexible solution, becomes then (eq. 1):

$$F(x) = W1 * EWT(x)^2 + W2 * EEC(x)^2 \quad (1)$$

where $EWT(x)$ and $EEC(x)$ for a given feasible solution from an individual h of the GA population will be calculated according to equation 15 in appendix A and equation 3 respectively.

Figure 5 shows an example about the different solutions obtained if for the objective function we use the estimated passenger waiting time and/or the energy consumption. Fig. 5.a shows a system with a twenty-floor building and six lifts. There are five landings calls waiting to be attended. In the thirteenth floor, there are two passengers, going to the seventh and sixth floor respectively.

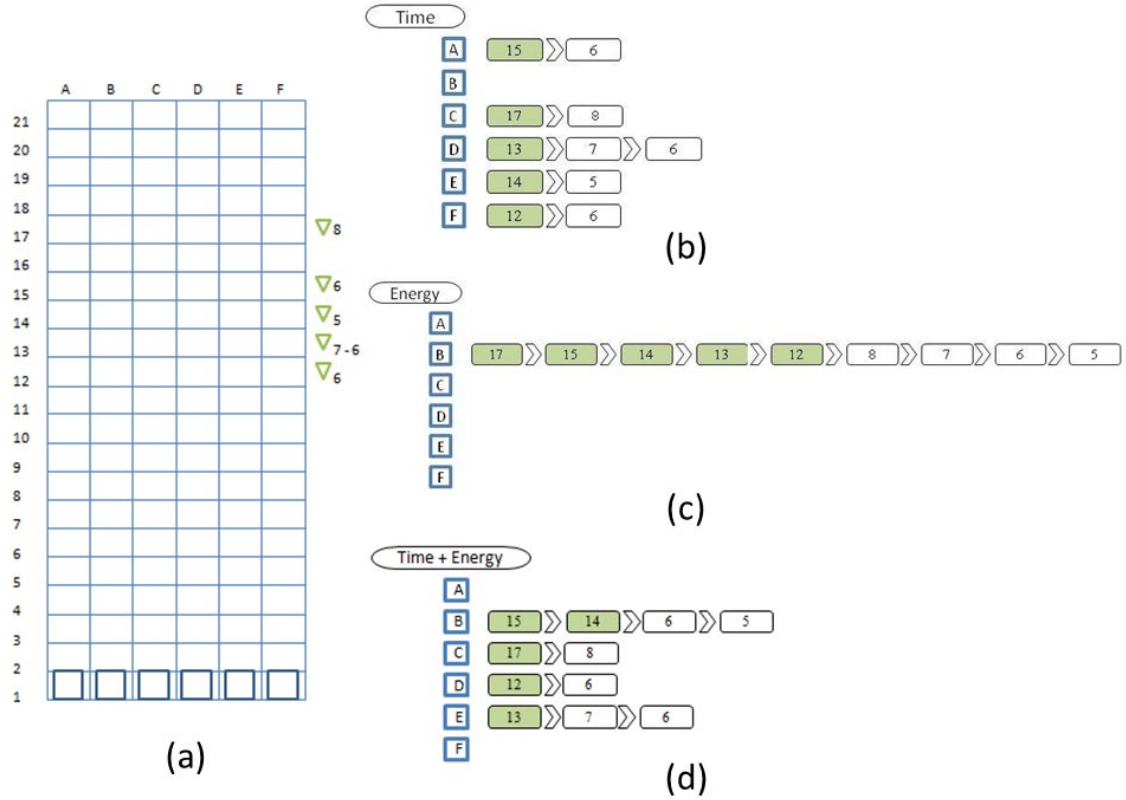


Fig 5: An example of the different dispatching plans when we consider the GA minimizing passenger waiting times (b), energy consumption (c) and both passenger waiting time and energy consumption (d).

Figure 5.b shows the dispatching plan generated by the GA when we minimize the passenger waiting times. The calls are assigned to a number of lifts without limitations, following the strategy of attending the closer calls first. Figure 5.c shows the dispatching plan when the GA minimizes the energy consumption. In this case, we try to minimize the number of lifts in the dispatching plan, attending the furthest call first. In the example, the lift B that is empty, attends first the call of the seventeenth floor and attends the rest of the passenger calls until it is full. Finally, figure 5.c shows a version of the GA where W1 and W2 are equal to 0.5. In this case, the GA is minimizing a linear combination of both objectives. In this solution, four lifts attend the five passenger calls.

Regarding $EEC(x)$, the energy consumption for a cab trip is modelled in terms of the energy consumption of the motor that moves the cab, the most consuming part. This consumption depends on the velocity of the movement, number of people inside and length of the trip. It is not easy to model the mechanical system of a cab using regenerative systems. The counterweight makes the system consume energy when the movement is on the opposite direction and return energy when it goes in the direction of the counterweight. Additional energy consumption elements such as the door opening and closing, losses due to friction, motor losses and electrical losses were not considered. For this research work, only the consumption of the drive/motor (W)

for car loads discretized up to 0%, 25%, 50%, 75%, 100% for both up and down directions were considered, as well as the door opening and lighting consumption.

A simplified model, evaluated the same elements considered for time estimations in terms of energy consumption:

- $DOEC_{h,i,j}$: Energy consumed for door open time for lift *i* in the *j-th* landing call of individual *h*
- $DTEC_{h,i,j}$: Energy consumed for lighting during door opening time while passengers entering and/or leaving for the *i-th* lift in the *j-th* landing call of individual *h*
- $DCEC_{h,i,j}$: Energy consumed for lighting during door closing time for the *i-th* lift in the *j-th* landing call of individual *h*
- $RideEC_{h,i,j}$: Energy consumed to move the lift *i* from one landing call *j* to landing call *j*+1 considering the percentage of the cab load of individual *h* for this journey.

where:

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*	$Lc \subseteq L$
$\sum_{i=1}^{I} Lc, i = L$	
j: index of landing calls	$j \in Lc$
H: set of individuals (the term individual refers to a GA entity, is not a passenger)	
h: index of individual	$h \in H$
k: index of the winner individual	$k \in H$

The expected energy consumed, EEC, for h, j, i was thus, calculated as:

$$EEC_{h,i,j} = DOEC_{h,i,j} + RideEC_{h,i,j} + DTEC_{h,i,j} + DCEC_{h,i,j} \quad (2)$$

EEC(h,i,j) (eq.2) and *EWT(h,i,j)* were both standardized before integrating them in the fitness equation. The expected energy consumption of all the landing calls of the system was estimated using the following equation (eq. 3):

$$EEC_h = \sum_{i=1}^{I} \sum_{j=1}^{L_{c,j-1}} EEC_{h,i,j} \quad (3)$$

Where *I* is the set of lifts, and $L_{c,i}$ is the set of landing calls associated with lift *i*.

Concepts previously mentioned for GA+ALL such as penalization or seeding were not applicable for energy consumption estimates.

3.3. Description of the Proportional Integral controller (PI) to set the values of the weights of the objective function of the GA

The WA method can be classified as either a posteriori or a priori method, depending on how it is applied. We consider it as a priori method as the Decision Maker (DM) balances the importance of each objective function in terms of weight coefficients. The WA method tries to minimize both $EWT(x)$ and $EEC(x)$ within the feasible region for the solutions (Tyni and Ylinen 2006). In (Tyni and Ylinen 2006), W1 and W2 must be empirically calculated through a PI controller that, for the integral part, works according to a previously established threshold for the Average Waiting Time of the passengers below which we would want to maintain in the service. The PI controller continuously calculates the area below the curve for the sum of the differences from the Average Waiting time to this threshold for a given time horizon in the past. For the proportional part, it is not so easy to set the values properly so, the most conservative alternative is to give more relevance to the integral part and let the proportional part just smooth the changes generated. The controller sets the value for W1 and then, W2 will be calculated as W2=1-W1.
In our work, we set the values of the PI controller through the value of the transported passenger data calculated from the system (detailed in section 3.4.3) to obtain the W1 and W2 values. Figure 6 shows the weights in relation to the percentage of passengers transported for intervals of five minutes. As this percentage goes closer or above 10% of the total number of passengers in the installation (the lift guidelines recommend that a lift system should be able to transport 12–15% of the building's population within a 5-minute period), the weight of W1, corresponding to time, increases to reach the value of 1, while W2 goes towards 0.

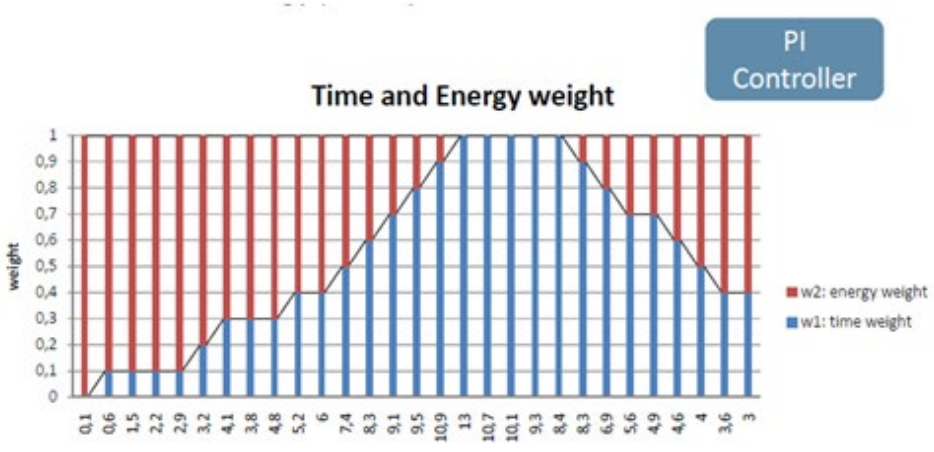


Figure 6. Weights for the PI controller related to the percentage of passengers transported for the last five minutes.

3.4. Using passenger information to change the EGCS

Usually, the data provided by the EGCS (no matter how it is implemented) refers only to the current state of the vertical transportation system. Regarding the passengers, it can provide the pending requests, the weight associated with each cab and the destination of the passengers inside it. It also provides the velocity, door status, cab position or moving direction (if it is moving).

The system does not know how many passengers could be behind a call. When a passenger makes a call, the button cannot be pushed again to make a new call from the same button until the lift attends the call and enables the button again. However, the information about the estimated number of passengers behind a call could enhance the performance of the EGCS (the GA in this work) to design optimum routes. Besides, the number of transported passengers by the system (boarded or alighted) in this work are used to help the bi-objective GA to priorize minimizing passengers waiting time or energy consumption, as commented in the previous section, using the PI.

To learn the passengers flow in the building (Fig. 7), we considered three different time frames: i) present data, the current requests provided by the controller every 100 milliseconds representing what is happening right now, ii) recent past data, preserved information in historical data about the previous 50 seconds of the system (destination or origin data), and iii) long-term data, historical data for what happened in the previous 2.5 or 5 minutes represented in OD matrices.

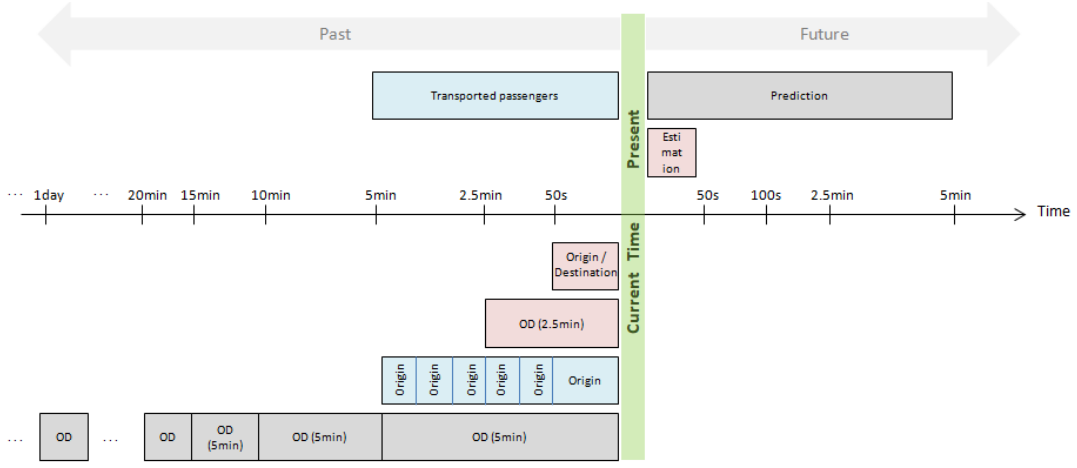


Figure 7. Data type concepts related with passenger information and time.

Together with one of these three time frames, raw data corresponding to the current passenger requests and the state of the systems provided by the controller were also gathered.

The OD matrix is an expression of the passenger trips where the cells are the number of people travelling from an origin floor to the destination floor for a certain time interval.

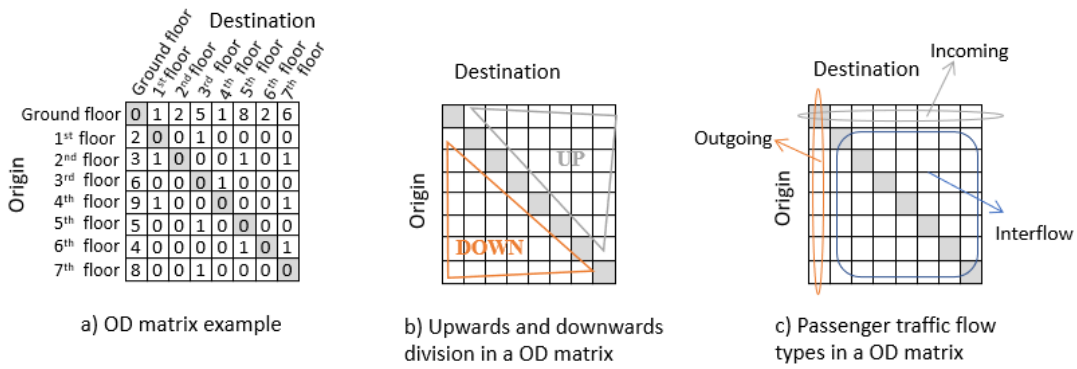


Figure 8. Information obtained from an OD matrix.

Figure 8 shows the information that can be obtained from the OD matrix: a) shows how the information is saved in cells, where each cell contains the information of the passenger that has been moved from an origin floor to a destination floor, b) shows the matrix representing the cells that are down direction trips and up direction trips, and c) shows the matrix describing the cells that creates the different traffic flow types.

In the next sections, we explain the methodology applied in this work to estimate the number of passengers behind a call, the number of passengers boarded or alighted, and the number of transported passengers in the system.

3.4.1. Estimation of the number of passengers behind a call

In this work, we developed two ways for estimating the number of passengers behind a call, using the present and the recent past data and using the present, recent past and long term data.

Using present and recent past data.

In this method, the number of passengers entered into the system at each floor is recorded in a variable called Origin (Figure 7). The variable origin has a counting of the passengers that have been attended per floor in the last 50 seconds by any lift. This origin data is separated according to the trip direction (up, down) that is given by the landing call, figure 9 shows the structure. For this counting, the number of lifts stops are found for the interval time and, at each stop, the passenger transfers are counted, as shown in equation 4 for Origin up (number of passengers entered into the system at each floor in an upwards lift in a time interval) and equation 5 for Origin down (number of passengers entered into the system at each floor in a downwards lift in a time interval).

Figure 9. Origin data structure. It contains the number of passengers that have entered any lift at each floor in the last 50 seconds. In this case, it would be a building of 12 floors.

$$Origin_{up} = \sum_{i=1}^{\#lifts} \sum_{t=1}^{50} q_{it}^{up} \quad (4)$$

$$Origin_{down} = \sum_{i=1}^{\#lifts} \sum_{t=1}^{50} q_{it}^{down} \quad (5)$$

Where q_{it}^{up}: number of passengers entered in an upwards lift i at a certain moment t, which later is transferred into number of passengers and q_{it}^{down}: number of passengers entered in a downwards lift i at a certain moment t. The direction of these trips is known because it was preserved in the landing calls. The counting was performed over every lift and then, summarized and recalculated at each interval time.

The last three origin data (2,5 minutes) were used to look at the tendency in each floor of the passengers' movements. If there is an increasing tendency, an extra passenger was added to the estimation of the number of passengers in that floor ($estimation_j$, estimation of the number of passengers behind a landing call, in eq. 6). On the other hand, if the tendency was decreasing, a passenger was removed from the estimation.

The information about the estimated number of passengers behind a call is implemented in the GA when it has to calculate the optimum routes for a certain individual, using its assignment of landing calls to the existing lifts. If there is an estimation of the number of passengers behind a call in a certain floor and the lift assigned to attend it has enough room inside, it will have to stop only once in that floor avoiding to stop again. On the other hand, if there is not enough room inside the lift, that solution increases the estimated waiting time of the passengers (EWT) in the system, as the lift has to stop again in that floor. That is the reason why the GA selects the solutions of the individuals in which the lifts can attend mostly all the number of passengers behind a call. To modify the estimated waiting time of a solution in the GA (from eq. 15 of appendix A) with the estimation of the number of passengers we use the following equation:

$$EWT_{p,h} = \sum_{i=1}^{I} \sum_{j=1}^{Lc,i-1} estimation_j \cdot EWT_{h,i,j} \quad (6)$$

Where I is the set of lifts, and $L_{c,i}$ is the set of landing calls associated with lift i.

Estimation of the passengers behind a call using the present, recent past and long term data.

The origin data used in the previous section is mixed with the long term data, the historical data of passenger movements in the system, creating origin-destination (OD) matrices for an interval time of 2.5 minutes (Fig. 4).

Long term data was used to calculate OD matrices from raw data. An explanation of how those OD matrices are extracted is given in Appendix B. As the basis for this research, we assumed that a similar passenger pattern was repeated daily. We generated the OD matrices selecting as a pattern one random day.

To mix the long term data with the origin data of equations 4 and 5, we splitted the OD matrices in up and down passenger movements as shown in Equations 7 and 8:

$$OD_{up} = \sum_{i=1}^{\#floors} \sum_{j=i+1}^{\#floors} OD_{ij} \quad (7)$$

$$OD_{down} = \sum_{i=1}^{\#floors} \sum_{j=i+1}^{\#floors} OD_{ji} \quad (8)$$

Then, we added a third of the OD matrices to the Origin data for both directions. As real time data corresponding to new requests arrives, the number of remaining expected passengers updates according to the following formulas where Estimation up and Estimation down are the corrected expected number of passengers upwards and downwards:

$$Estimation_{up} = 0 \quad (9)$$

$$Estimation_{down} = 0 \quad (10)$$

$$Estimation_{up} = Estimation_{up} + (OD_{up}/3 - origin_{up}) \quad (11)$$

$$Estimation_{down} = Estimation_{down} + (OD_{up}/3 - origin_{up}) \quad (12)$$

Then, as before, this corrected estimation of the number of expected passengers is included in the GA as in eq. 6.

3.4.2. Estimation of the number of boarded /alighted passengers through recent-past data

Using the origin data following equations 4 and 5, only when a lift opens its doors the counting for the boarded passengers starts. To estimate the number of alighted passengers, we used the destination data, similar to the origin data.

To estimate the number of passengers boarding or alighting at each stop, previous approaches were used (Cascetta 1984). Our own previous work (Basagoiti et al. 2012; Basagoiti et al. 2013) explains in depth the method used to calculate these numbers.

3.4.3. Estimation of the number of transported passengers through recent-past data

The number of the transported passengers was calculated using the last recent 5 minutes data (Fig. 7). Firstly, all the passengers' movements in the 5 minutes were counted. The incoming, outgoing and interflow traffic passenger profiles were taken into account for this work. Once the number of passengers' movements was known, the percentage of the population transported is calculated. If we compare the handling capacity (HC; defined as the percentage of passengers that the system can move in 5 minutes) of a building to the percentage of transported passengers, we can know the level of saturation of the building. This way, the obtained value can be compared to the HC.

3.5. Combining learned passenger flow and current flow with the GA+ALL

Up-peak is formally defined in (Siikonen 1997) based on the proportion of incoming (high), outgoing (low) and interfloor (low) traffic components. Down-peak is also defined according to the proportion of incoming (low), outgoing (high) and interfloor (low) traffic components. Detection of these traffic conditions is necessary to manage effectively the control strategy.

For this work, we identify an up-peak traffic in the system if a lift leaves the ground floor with at least two passengers inside and immediately after there is a new landing call from the ground floor. In this situation the parking policy is applied (empty resources are moved to the ground floor).

On the other hand, we identify a down-peak traffic if the number of transported passengers is at least half of the HC of the system and under two possible conditions:

- If more than, 80% of the passengers leave the lifts on the ground floor in the recent-past interval time (last 50 seconds in the system) and more than 80% of the landing calls are down-landing calls.
- If more than, 80% of the passengers inside the cars want to leave the lift on the ground floor and more than 80% of the landing calls are down-landing calls.

Under a down peak detection, the zoning and parking policies are applied. By the zoning policy, the building is divided into zones; each lift serves only one zone. The parking policy applied in this case allows sending empty lifts at the top of their corresponding zones (CIBSE 2010; Park 2013; Siikonen 1997).

Both actions were integrated in the basic algorithm and correspond to the label policies attached to the GA+ALL.

3.6. Statistical analysis and comparison of different GA+ALL versions

To validate how passenger information can help the controller, 6 different configurations of the GA+ALL were analyzed. In the system, the controller reallocates landing calls every 250 milliseconds.

The first three configurations have the **Average Expected Waiting time** as objective function.

1. GA+ALL: This GA+ALL is based on our model (Beamurgia et al. 2015) and assumes that one landing call always corresponds to only one passenger.
2. GA+ALL +policies+ewtp: The origin data is used to estimate how many passengers are behind a call. The traffic detection and its policies were used.
3. GA+ALL +policies+ewtp+OD: The long-term, OD data is used here. The traffic detection and its policies were used.

In the following configurations, the objective function is the **Average Expected Energy Consumption.**

4. GA+ALL _E: This GA+ALL: is based on our model (Beamurgia et al. 2015) and assumes that one landing call always corresponds to only one passenger.
5. GA+ALL +policies+ewtp_E: The origin data is used to estimate how many passengers are behind a call. The traffic detection and its policies were used.

The 6th configuration has the **Weighted aggregation** of the **Average Expected Waiting Time** and **Average Expected Energy Consumption** and weights adjusted using the PI controller.

6. GA+ALL +policies+ewtp+pi: The origin value is used to estimate how many passengers are behind a call. The traffic detection and its policies were used.

As explained above, three different types of passenger flow profiles were used to test the different GA versions: one mixed profile, three well-known full day office traffic design profiles and, three different step profiles. In the last two experiments, we added the conventional control algorithm CGC algorithm to be compared to three GA versions with good performance in the first scenario with the mixed passenger profile.

To compare the algorithms in the different scenarios, we applied the rank-based nonparametric Kruskal-Wallis test, to determine if there were statistically significant differences in their

passengers waiting times distributions. Moreover, for pairwise comparisons (comparing the algorithms by pairs), we applied Wilcoxon tests to determine statistically significant differences in their passengers waiting time performance.

4. Results

4.1. Mixed passenger profile

Figure 10 shows the mean and standard deviations of the passenger waiting times for the six different versions of the GA+ALL applied to the mixed passenger profile explained previously. The p-value of the Kruskal-Wallis test in the figure indicates that there were statistically significant differences among the algorithms.

As can be observed, the versions of the GA+ALL including energy as a part or as a whole of their objective function (GA+ALL +policies+ewtp+pi, GA+ALL+policies+ewtp_E and GA+ALL _E) obtained significantly (Wilcoxon p-value <0.01) higher means and standard deviations for waiting time than the versions including the estimated waiting time as an objective function (GA+ALL, GA+ALL +policies+ewtp and GA+ALL +policies+ewtp+OD).

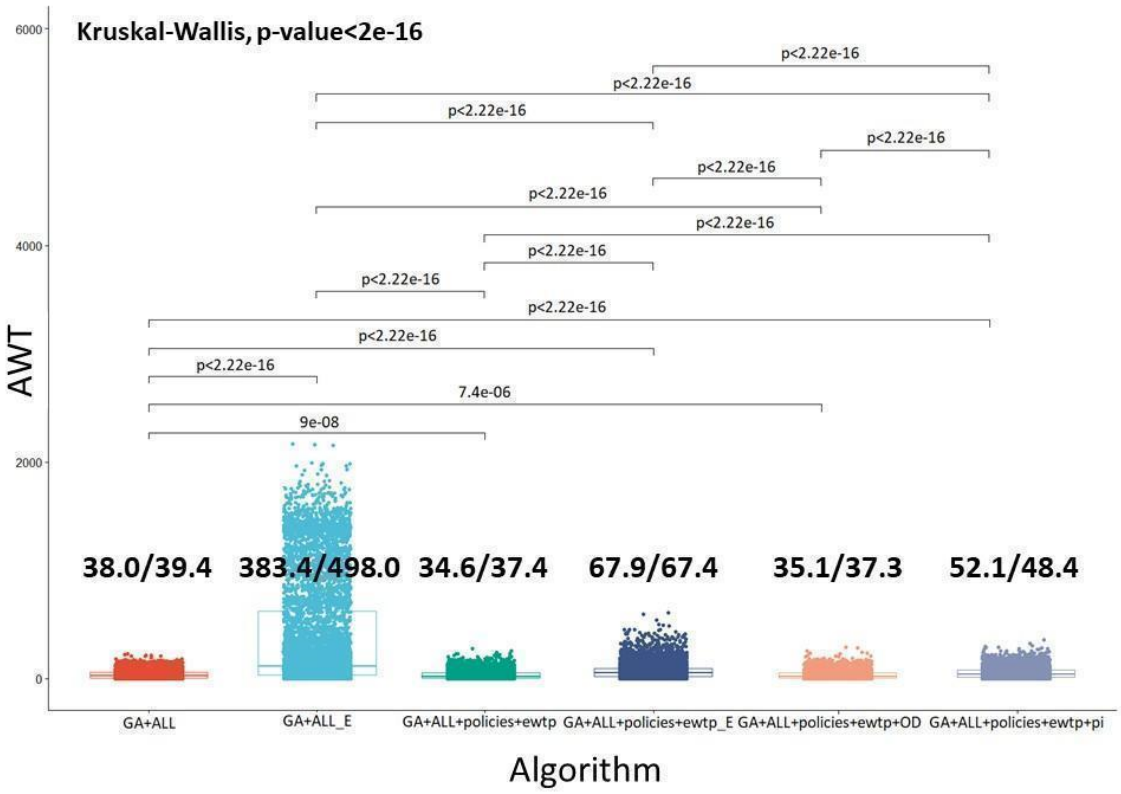


Figure 10. Mean/standard deviation values in bold (seconds) of the waiting time for the mixed passenger profile for the different six GA versions. Two horizontal lines joined together with a horizontal line and the p-values of pairwise Wilcoxon tests between two methods are shown (N=7650).

Therefore, the addition of the estimation of the number of passengers behind a call, the parking policies and the PI controller for the traffic detection did significantly decrease the waiting time for GA+ALL _E providing acceptable versions (GA+ALL +policies+ewtp+pi and GA+ALL +policies+ewtp_E).

Moreover, GA+ALL+policies+ewtp+pi had the lowest means and standard deviations for the waiting time among the versions considering energy. This is to be expected, since it is a hybrid algorithm including waiting time and energy as an aggregated objective function.

The use of passenger information (origin) does improve the results of the algorithm in terms of the waiting time. On the other hand, no significant differences were found between GA+ALL +policies+ewtp and GA+ALL +policies+ewtp+OD.

Table 3 extends the information of Figure 10. The average values for ten different simulations of the mixed passenger profile are shown.

	1 GA+ALL	2 GA+ALL +policies +ewtp	3 GA+ALL +policies +ewtp +OD	4 GA+ALL _E	5 GA+ALL +policies +ewtp_E	6 GA+ALL +policies +ewtp +pi
Average Waiting Time (s)=**AWT**	38,0	34,6	35,1	350,0	68,0	52,1
Longest Waiting Time (s)=**LWT**	190,4	211,8	200,8	1475,0	425,7	283,6
Average Transit Time (s)=**ATT**	61,0	59,3	59,6	69,7	64,7	63,5
Longest Transit Time (s)=**LTT**	177,7	162,9	166,8	190,3	172,2	168,8
Average Time to Destination (s)=**ATD**	99,0	93,9	94,7	109,0	419,8	115,7
Longest Time to Destination (s) =**LTD**	295,3	293,0	292,8	425,5	1596,3	343,0
Total Energy consumption (kWh)=**E**	13,2	13,9	13,8	9,6	11,7	12,8

Table 3. Average values for 10 different simulations of the mixed passenger. The table clearly shows that improvements in waiting time will make the energy consumption higher but acceptable passenger service could be obtained using a hybrid or bi-objective algorithm.

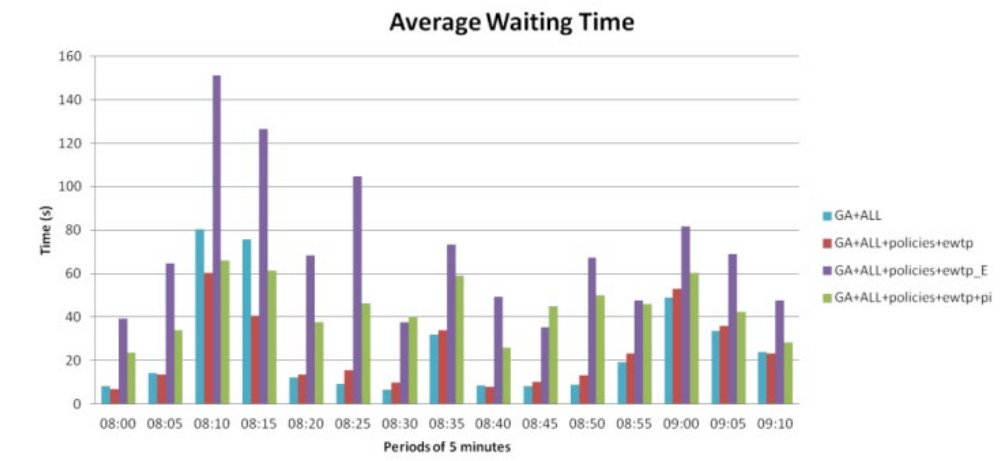


Figure 11. A histogram for passenger waiting time for time frames of 5 minutes for different GA versions of the control algorithm.

Figure 11 shows an histogram for the passenger waiting times grouped in frames of 5 minutes for GA versions 1, 2, 5 and 6. As expected, the energy aware algorithm generates longer waiting times than the other versions.

4.2. Full day office and step profiles

Three commonly used full day office profiles were used to test three different versions selected (GA+ALL, GA+ALL+policies+ewtp, GA+ALL+policies+ewtp+pi) of the algorithm. The results are shown in table 4.

Passenger profile	**CIBSE**				**STRAK**				**SIIK**			
Algorithms	**1** GA+ALL	**2** GA+ALL +policies +ewtp	**6** GA+ALL +policies +ewtp +pi	CGC	**1** GA+ALL	**2** GA+ALL + policies + ewtp	**6** GA+ALL + policies + ewtp + pi	CGC	**1** GA+ALL	**2** GA+ALL + policies + ewtp	**6** GA+ALL + policies + ewtp + pi	CGC
AWT(s)	180,7	175,2	**173**	189,3	132.9	104,5	**104,2**	114,2	30,8	**26,2**	39,4	35,6
LWT (s)	**2290,4**	2765	2433,9	3096,1	1474.8	2023,9	**2205,3**	3189,8	462,9	400,2	**354,2**	397,5
ATT (s)	55,6	**55,3**	57,7	57,0	59.6	**59,3**	59,8	61,6	**55,2**	55,5	61,2	59,3
LTT (s)	171,6	171,2	**171,2**	175,7	174.3	**171,2**	175,1	176,5	211,1	216,2	**206,8**	240,8
ATD (s)	236,3	**230,5**	230,7	246,3	192.5	**163,8**	164	175,9	86	**81,7**	100,7	94,8
LTD (s)	**2316**	2790,3	2458,2	3121,3	1498.8	**2047,9**	2229,3	3224,4	486,5	515,8	**398,6**	454,4
E(kWh)	150,8	161,0	151,8	**145,7**	118,6	125,0	121,9	**114,4**	154,9	155,9	**136,8**	145,2

Table 4. Simulation results of the different GA+ALL versions (1, 2, 6) and CGC for the three full day office profiles.

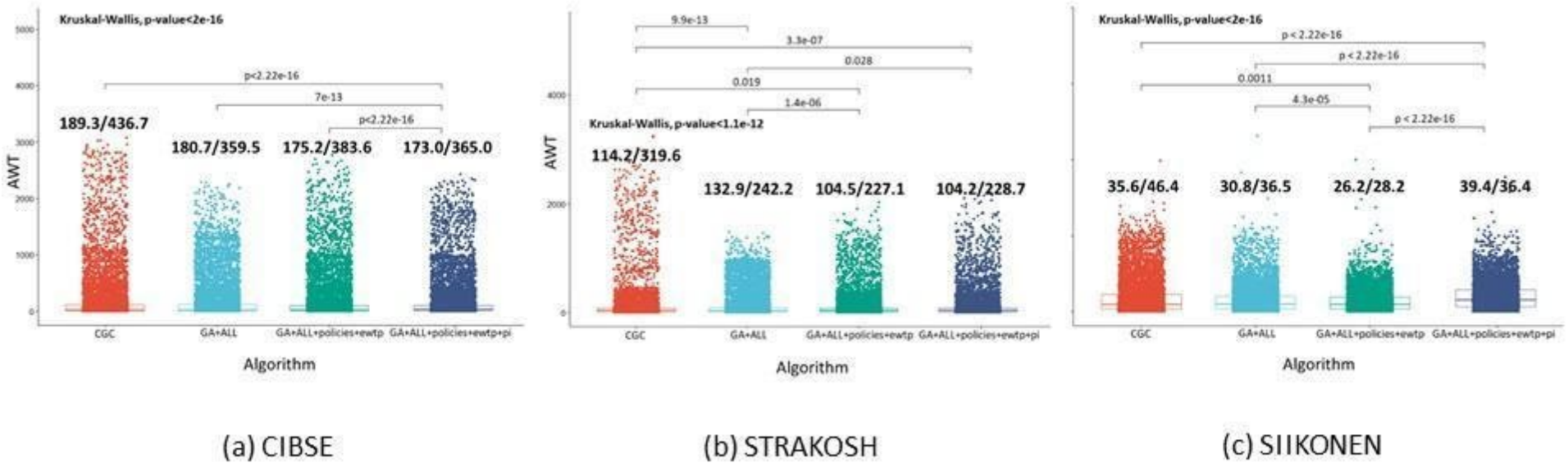


Figure 12. Mean/standard deviation values in bold (seconds) of the waiting time for the three full office profiles for the CGC and the versions 1, 2, 6 of GA+ALL. Two horizontal lines joined together with a horizontal line and the p-values of pairwise Wilcoxon tests between two methods are shown (N=9234 (a), N=7329 (b), N=9210 (c)).

Figure 12 shows the passengers' waiting time distributions of the different algorithms, with their mean and standard deviation values. The bi-objective version GA+ALL+policies+ewtp+pi obtained the lowest AWT values for the CIBSE and STRAK profiles, showing statistically significant differences (Wilcoxon test with p-values<0.01) with respect to the other algorithms. In addition, it obtained the lowest values for LWT in STRAK and SIIK profiles. Those results are surprising as we expected to obtain better results for the other algorithms that are exclusively using an objective function that involves the waiting time.

On the other hand, regarding energy consumption, the bi-objective version has the lowest value for the SIIK profile. Besides, the CGC has the lowest values of energy consumption for CIBSE and STRAK profiles. The main issue in our hybrid version GA+ALL+policies+ewtp+pi is to adjust the weights of the PI so that the algorithm can increase the average passenger waiting time while simultaneously decreasing energy consumption; thresholding the maximum value of the weight related to the passenger waiting time would relax the timing and increase the energy saving.

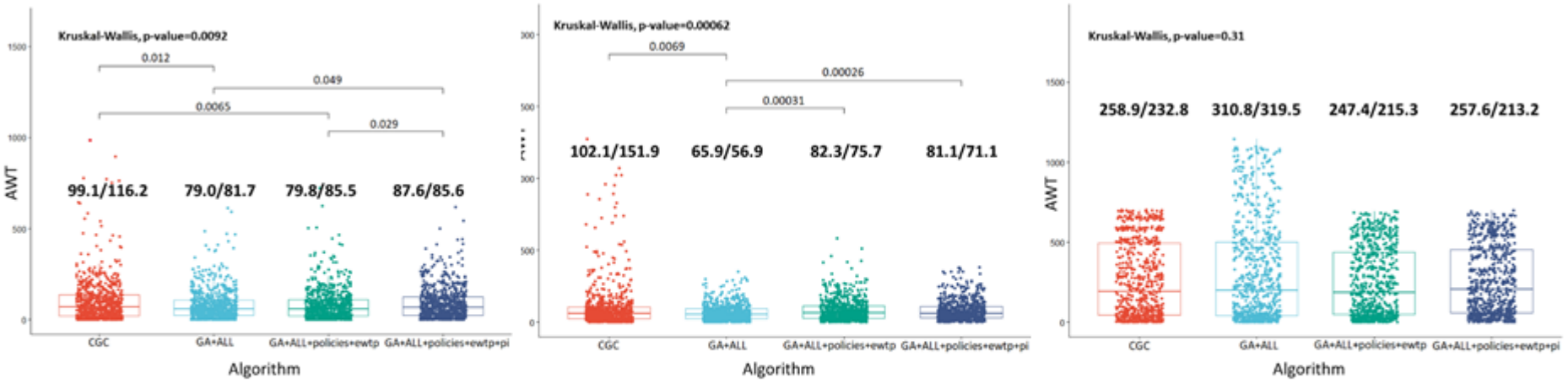
Table 5 shows the results of the three GA versions and the CGC for the three different step profiles from intensity varying from 11 to 16%. The lowest AWTs for the first (45-45-10) and the second (0-100-0) profiles were achieved by the GA+ALL. The lowest AWT for the third profile was achieved by the GA+ALL+policies+ewtp. The bi-objective algorithm obtained the lowest energy consumption in the first step profile, but in all the cases lower than the CGC.

In figure 13, the superior performance of the GA versions with statistically significant differences against the CGC can be observed.

Our hybrid GA+ALL+policies+ewtp+pi and the GA+ALL+policies+ewtp versions, have shown a good performance in terms of stability, as the steps profiles applied are stress tests in which the HC of the building is overcome, the building will be saturated, working under high demand conditions. Therefore, the energy consumption has to be put aside to miminize the passenger waiting time.

(11% to 16%)												
Passenger profiles	**STEP1** (45% Incoming-45% Outgoing-10% interfloor)				**STEP2** (0% Incoming-100% Outgoing-0% interfloor)				**STEP3** (80% Incoming-15% Outgoing-5% interfloor)			
Algorithms	GA+ALL	GA+ALL+ policies + ewtp	GA+ALL+ policies+ ewtp + pi	CGC	GA+ALL	GA+ALL + policies + ewtp	GA+ALL+ policies+ ewtp + pi	CGC	GA+ALL	GA+ALL+ policies + ewtp	GA+ALL+ policies+ ewtp + pi	CGC
AWT (s)	**79**	79,8	87,6	99,1	**65,9**	82,3	81,1	102,1	310,8	**247,4**	257,6	258,9
LWT (s)	**611,7**	721,9	618,8	986,3	**351,9**	582,4	378,8	1274,3	1145,6	**691,6**	699,3	703,2
ATT (s)	**75**	76,2	75,8	79,6	54,2	52,9	**52,9**	53	**80,5**	81,4	81,1	82,5
LTT (s)	229,5	241,4	**227,8**	231,7	134,4	133,6	**122,4**	124,6	187,4	**178,4**	210,1	197,7
ATD (s)	**154,1**	156	163,4	178,8	**120,1**	135,2	133,9	155,1	391,3	**328,7**	338,7	341,4
LTD (s)	649,6	761,1	**633,7**	1006,6	**370,7**	615,2	400,2	1308,9	1238,4	835,7	**831,9**	858,8
E (kWh)	6,7	6,727	**6,498**	6,673	**1,2**	1,613	1,645	1,615	**13,4**	13,401	13,441	13,561

Table 5. Simulation results of the 3 different GA+ALL versions, for the 3 different step profiles with increasing HC (theoretical is 10%)



(a) Step 45-45-10 (b) Step 0-100-0 (c) Step 80-15-5

Figure 13. Mean/standard deviation values in bold (seconds) of the waiting time for the three step profiles for the CGC and the versions 1, 2, 6 of GA+ALL. Two horizontal lines joined together with a horizontal line and the p-values of pairwise Wilcoxon tests between two methods are shown (N=9234 (a), N=7329 (b), N=9210 (c)).

4.3. Improvements of GA versions against CGC

The CGC was used to compare the performance of GA+ALL, GA+ALL+policies+ewtp and, GA+ALL+policies+ewtp+pi. The results of the AWT and the rest of the parameters were averaged for each algorithm (CGC and the three GA+ALL versions) for the three full day office profiles and the three step profiles (Table 6). Then, the difference in percentage with respect to the CGC was calculated for the three GA+ALL versions. If we take the CGC reference value, CGCv, in a certain time measure (AWT, LWT,etc) and the real value of a certain algorithm A in the same time measure, tm_realA, the changes in % with respect to that value are calculated as:

$$\%\ Change\ with\ respect\ CGC = 100\left(\frac{tm\_real_A - CGC_v}{CGC_v}\right) \qquad (13)$$

The results show that, on average, the GA+ALL obtains significant benefits reducing the LWT and in consequence the Longest Time to Destination (LTD). When the GA+ALL focuses on time and uses the passenger information to help the system, it achieves lower values in the AWT for both types of profiles. Related to energy, as the system was in a high traffic demand, no reduction was obtained compared to the CGC. Nevertheless, all three GA+ALL versions did show benefits in time.

Passenger profiles	**FULL DAY OFFICE PROFILES (CIBSE-STRAK-SIIK)**			**STEPS**		
GA version	GA+ALL	GA+ALL+ policies+ewtp	GA+ALL+ policies+ewtp+pi	GA+ALL	GA+ALL+ policies+ewtp	GA+ALL+ policies+ewtp+pi
AWT (s)	1%	14%	2%	12%	14%	11%
LWT (s)	21%	16%	21%	16%	28%	36%
ATT (s)	4%	4%	-1%	2%	2%	2%
LTT (s)	5%	5%	6%	-1%	-1%	-1%
ATD (s)	1%	9%	2%	7%	10%	8%
LTD (s)	24%	11%	21%	21%	27%	37%
E (kWh)	-5%	-9%	-2%	9%	0%	1%

Table 6. The three GA+ALL versions compared in percentage of variability with CGC (positive numbers imply better performance, negative numbers imply worse performance).

5. Discussion and conclusions

The current work evaluates the successful implementation of a GA+ALL in the dispatching problem for vertical transportation using the passenger module. Passenger information and different policies were added to evaluate its performance. The system was tested using different passenger profiles under high traffic demand, with waiting time and/or energy as objective functions.

The results show that passenger information together with the parking policy achieve significant improvement in the performance of the GA+ALL, considering either waiting time or energy consumption or both. In contrast, the use of OD matrices did not show significant improvement with respect to the version without this technique, when compared to the configuration that only used the recent-past data in combination with the origin.

Analysing a full day office profile, the best result was achieved by the GA+ALL+policies+ewtp+pi configuration, where the AWT was the lowest in the CIBSE and Strakosch profiles. In the Siikonen profile, it did not achieve the best AWT, but the average was not very high. The value was less than 40 seconds, and the energy consumption was about 12% less than the GA+ALL+policies+ewtp and GA+ALL. To increase the energy saving of the GA+ALL+policies+ewtp+pi, the weights of the PI should be thresholded.

When the traffic intensity was high and increased, the different traffic profiles gave different results. Under a lunch-peak (45-45-10) the best configuration was GA+ALL+policies+ewtp. The AWT value was not the best however, it was very close to the value for GA+ALL. A further advantage was that a greater number of passengers were moved before their waiting time increased to 60 seconds. In addition, the energy consumption of this configuration was the best although the GA+ALL produced similar results.

In the case of a high intensive down profile (0-100-0), the GA+ALL achieved the best results. In contrast the GA+ALL+policies+ewtp configuration performed the best for the up profile (80-15-05), achieving the best AWT and energy consumption value.

These findings provide two avenues for future research: the improvement of the PI weighting system, and the better integration of the information estimated in the OD matrices into the controller.

The real application of our approaches relies on the calculation and subsequent use of building specific passenger profiles which are recorded in OD matrices through historial data of the system, but also in the use of lift cars equipped with weight sensors. The advantages of using those elements to estimate the passenger flow shows unquestionable benefits. Updating the forecast of the passenger flow patterns in a building using the OD matrices will help the controller of the system to keep a good performance in the QoS.

6. References

Abed-alguni B.H. and Paul D.J. (2018). Hybridizing the Cuckoo Search Algorithm with Different Mutation Operators for Numerical Optimization Problems. J. Intell. Syst. 2020; 29(1): 1043–1062https://doi.org/10.1515/jisys-2018-0331.

Abo-Hammour Z. S., Abu Arqub O., Momani S. and Shawagfeh N (2014). Optimization Solution of Troesch's and Bratu's Problems of Ordinary Type Using Novel Continuous Genetic Algorithm. Hindawi Publishing Corporation. Discrete Dynamics in Nature and Society. Volume 2014, Article ID 401696. http://dx.doi.org/10.1155/2014/401696

Alawad N.A. and Abed-alguni B.H. (2021). Discrete Island-Based Cuckoo Search with Highly Disruptive Polynomial Mutation and Opposition-Based Learning Strategy for Scheduling of Workflow Applications in Cloud Environments. Arabian Journal for Science and Engineering (2021) 46:3213–3233. https://doi.org/10.1007/s13369-020-05141-x

Basagoiti R., Beamurgia M., Peters R., Kaczmarczyk S. (2012) Origin Destination Matrix Estimation and Prediction in Vertical Transportation. In 2nd Symposium on Lift and Escalator Technologies.

Basagoiti R., Beamurgia M., Peters R., Kaczmarczyk S. (2013) Passenger flow pattern learning based on trip counting in lift systems combined with real-time information. In 3rd Symposium on Lift and Escalator Technologies, 119.

Beamurgia M., Basagoiti R. (2011) Predicting the passenger request in the elevator dispatching problem. In Soft Computing Models in Industrial and Environmental Applications, 6th International Conference SOCO, 87:387-394. https://doi.org/10.1007/978-3-642-19644-7_41

Beamurgia M., Basagoiti R., Rodríguez I. (2011) A genetic algorithm with passenger arrival advanced information to solve the elevator dispatching problem. In 42nd Annual Conference of the Italian Operational Research Society, AIRO Conference.

Beamurgia M., Basagoiti R., Rodríguez I., Rodríguez V. (2015) A modified genetic algorithm applied to the elevator dispatching problem. Soft Computing 20:3595–3609. https://doi.org/10.1007/s00500-015-1718-1

Bera S., Rao K.V.K. (2011). Estimation of origin-destination matrix from traffic counts: the state of the art. European Transport \ Trasporti Europei, ISTIEE, Institute for the Study of Transport within the European Economic Integration, issue 49:2-23.

Bolat B., Cortés P. (2011). Genetic and tabu search approaches for optimizing the hall call-Car allocation problem in elevator group systems. Appl. Soft Comput. 11. 1792-1800. 10.1016/j.asoc.2010.05.023.

CIBSE. (2010) Transportation Systems in Buildings: CIBSE Guide D. Chartered Institution of Building Services Engineers.

Caggiani L., Ottomanelli M., Sassanelli D. (2013) A fixed point approach to origin-destination matrices estimation using uncertain data and fuzzy programming on congested networks. Transportation Research Part C: Emerging Technologies, 28:130- 141. https://doi.org/10.1016/j.trc.2010.12.005

Cascetta E. (1984) Estimation of trip matrices from traffic counts and survey data: a generalized least squares approach. Transportation Research Part B: Methodological, 18B (4/5) 289-299. https://doi.org/10.1016/0191-2615(84)90012-2

Cortés P, Fernández JR, Guadix J, Muñuzuri J. (2012) Fuzzy Logic Based Controller for Peak Traffic Detection in Elevator Systems. Journal of Computational and Theoretical Nanoscience, 9(2):310-318. https://doi.org/10.1166/jctn.2012.2025

Hiller B. (2011) Online optimization: Probabilistic analysis and algorithm engineering. Operations Research Proceedings, 647-652. https://doi.org/10.1007/978-3-642-20009-0_102

Hu Z., Liu Y., Su Q., Huo J. (2010) A multi-objective genetic algorithm designed for energy saving of the elevator system with complete information. Energy Conference and Exhibition (EnergyCon), 2010 IEEE International, 126-130. https://doi.org/10.1109/ENERGYCON.2010.5771661

Imrak C.E., Özkirim M. (2004) Neural Networks application in the next stopping floor problem of elevator systems. Journal of Engineering and Natural Sciences.

Imrak C.E., Özkirim M. (2006) Determination of the next stopping floor in elevator traffic control by means of neural networks. Journal of Electrical and Electronics Engineering 6, 1:27-33.

Jensen M.T. (2003) Reducing the run-time complexity of multiobjective EAs: The NSGA-II and other algorithms. IEEE Transactions on Evolutionary Computation 7, 5:503-515.

Ji Y., Mishalani R.G., McCord M.R., Goel P.K. (2011) Identifying homogeneous periods in bus route origin-destination passenger flow patterns from automatic passenger counter data. Transportation Research Record: Journal of the Transportation Research Board. 2216. https://doi.org/10.3141/2216-05

Kuusinen J.M., Sorsa J., Susi T., Siikonen M.L., Ehtamo, H. (2010) A new model for vertical building traffic. Transportation Research Part B.

Kuusinen J.M., Sorsa J., Siikonen M.L., Ehtamo H. (2012) A study on the arrival process of lift passengers in a multi-storey office building. Building Services Engineering Research and Technology, 33:437-449. https://doi.org/10.1177/0143624411427459

Kuusinen J.M., Sorsa J., Siikonen M.L. (2015) The Elevator Trip Origin-Destination Matrix Extimation Problem. Transportation Science, 49, 3:559-576. https://doi.org/10.1287/trsc.2013.0509

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

Liu T.D., Chen J., Jiang H. (2011) Passenger volume estimation based on the relational model of visual density for elevator group-controlled system. https://doi.org/10.4028/www.scientific.net/AMM.127.338

Momani S., Abo-Hammour Z. S. and Alsmadi O. MK (2016). Solution of Inverse Kinematics Problem using Genetic Algorithms. Appl. Math. Inf. Sci. 10, No. 1, 1-9. http://dx.doi.org/10.12785/amis/Solution*of*inverse*kinematics*problem

Park M., Ha H., Lee HS., Choi Y., Kim H., Han S. (2013) Lifting demand-based zoning for minimizing worker vertical transportation time in high-rise building construction. Automation in Construction, 32:88-95. https://doi.org/10.1016/j.autcon.2013.01.010

Peters R., Mehta P., Haddon J. (1996) Lift passenger trac patterns: Applications, current knowledge and measurement.

Rajabioun R (2011). Cuckoo Optimization Algorithm. Applied Soft Computing 11 (2011) 5508–5518

Ruokokoski M., Ehtamo H., Pardalo P.M. (2015) Elevator dispatching problem: a mixed integer linear programming formulation and polyhedral results. J Comb Optim 29, 750–780. https://doi.org/10.1007/s10878-013-9620-1

Ruokokoski M., Sorsa J., Siikonen M.L., Ehtamo H. (2016) Assignment formulation for the Elevator Dispatching Problem with destination control and its performance analysis. European Journal of Operational Research, 397-406. https://doi.org/10.1016/j.ejor.2016.01.019

Sherali H., Park T. (2001) Estimation of dynamic origin-destination trip tables for a general network. Transportation Research Part B: Methodological, 35:217- 235. https://doi.org/10.1016/S0191-2615(99)00048-X

Siikonen M.L. (1997) Elevator Group Control with Artificial Intelligence.

Siikonen M.L. (1997) Planning and Control Models for Elevators in High-Rise Buildings. Helsinki University of Technology, Systems Analysis Laboratory, Research Reports A68.

Sorsa J.S., Ehtamo H., Kuusinen J.M., Ruokokoski M., Siikonen M.L. (2018) Modeling uncertain passenger arrivals in the elevator dispatching problem with destination control. Optim Lett 12, 171–185. https://doi.org/10.1007/s11590-017-1130-0

Sorsa J.S., Ehtamo H., Siikonen M.l., Tyni T., Ylinen J. (2009) The Elevator Dispatching Problem. Transportation Science.

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

Strakosh G.R. (2007) The vertical Transportation Handbook, Third edition.

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. European Journal of Operational Research, 169:960-977. https://doi.org/10.1016/j.ejor.2004.08.027

7. Appendices

APPENDIX A

The mathematical model used when we want to minimize the estimated passengers average waiting time of all the different dispatching plans generated by each individual h of the GA population, EWT_h

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	$Lc \subseteq L$
$\sum_{i=1}^{I} Lc, i=L$	
j: index of landing calls	$j \in Lc$
H: set of individuals (the term individual refers to a GA entity, is not a passenger)	
h: index of individual	$h \in H$
k: index of the winner individual	$k \in H$

Parameters:

$runTime_{h,i,j}$: 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
$DOT_{h,i,j}$: Door open time for lift i in the *j-th* landing call of individual h
$DTT_{h,i,j}$: Door open time while passengers entering and/or leaving for the *i-th* lift in the *j-th* landing call of individual h
$DCT_{h,i,j}$: Door close time for the *i-th* lift in the *j-th* landing call of individual h

Variables:

$doorStatus_i$: The door status (open or closed) for the lift i, at the time when the algorithm is called to obtain a route.

The model

$$EWT_{h,i,j} = DCT_{h,i,j} + runTime_{h,i,j} + DOT_{h,i,j} + DTT_{h,i,j} \quad (14)$$

$$EWT_h = \sum_{i=1}^{I} \sum_{j=1}^{Lc,i-1} EWT_{h,i,j} \quad (15)$$

$$individual_k = (EWT_h) \quad (16)$$

APPENDIX B

Long term data is referred to the historical data in the system 2.5 before the current time. This data is used to calculate OD matrices from raw data. Using the flow conservation principle, "the number of entering passengers is equal to the number of leaving passengers in a lift trip", missing information is estimated as in (Kuusinen 2010).

These ODs have the same number of rows and columns as the number of floors in the building. Each cell of the matrix contains, aggregated for a given time interval, the number of passengers that have travelled from one floor (related to the row number that is the origin floor) to the destination floor (related to the column number). An OD matrix is specific for a building and a particular time interval. An OD matrix has three main parts, the incoming traffic flow, the outgoing traffic flow and the interflow passenger traffic.

The estimation process of this matrix requires separating each lift movement into trips for each period. A trip begins when a lift that stopped with the doors closed has to attend a new request and starts moving. The trip ends when it has no more requests to attend or has to change the direction.

Symbolic calculation and constraint programming tools were used to implement the following assumptions: a) at least one passenger boards or alights when the lift stops on a floor, b) when there is a car call, at least one passenger is in the lift, c) there are not more passengers than the capacity of the lift, and d) the passenger who boards on a floor does not alight in the same floor.