View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by idUS. Depósito de Investigación Universidad de Sevilla

# A Genetic Algorithm for Controlling Elevator Group Systems

P. Cortes[1], J. Larrañeta and L.Onieva

Ingeniería Organización. Escuela Superior Ingenieros. Seville University, Camino de los Descubrimientos s/n. Sevilla 41092. SPAIN
[1]pca@esi.us.es

**Abstract.** The efficient performance of elevator group system controllers becomes a first order necessity when the buildings have a high utilisation ratio of the elevators, such as in professional buildings. We present a genetic algorithm that is compared with traditional controller algorithms in industry applications. An ARENA simulation scenario is created during heavy lunchpeak traffic conditions. The results allow us to affirm that our genetic algorithm reaches a better performance attending to the system waiting times than THV algorithm.

## 1 Introduction

The installation of synchronized elevator groups in professional use buildings (offices, hospitals or hotels) is an usual practice. The large utilisation ratio of the elevators makes necessary the implementation of such systems in order to give quality to the users-passengers and energetic efficiency to the building managers.

In this situation, it is usual to select the system waiting time as the goal to attain an efficient system performance. Also, the maximum waiting time has to be had in account as an additional limitation. The system waiting time includes the waiting time for the lift in the hall plus the trip time inside the lift.

When dealing with elevator systems is a usual practice to consider the following assumptions (where most of them are evident assumptions). Each hall call is attended by only one cabin. The maximum number of passengers being transported in the cabin is bounded by its capacity. The lifts can stop at a floor only if it exists a hall call or a cabin call in that floor. The cabin calls are sequentially served in accordance with the lift trip direction. A lift carrying passengers cannot change the trip direction.

The traditional type of controllers implemented in the industry follows simple dispatch rules that make use of an IF-ELSE logical commands set. Among these dispatch rules, the THV is one of the most habitual algorithms. The THV assigns the hall call to the nearest lift in the adequate trip direction [1].

Recently, advanced methods have been proposed showing better performance. Examples of them are the *Optimal Routing algorithm*, the *Dynamically Adaptive Call Allocation* (DACA) and the *Adaptive Call Allocation* (ACA) [2] that are based on Dynamic Programming. Also Fuzzy Logic has been proved as a valuable alternative when evaluating a large amount of criteria in a flexible manner. The fuzzy elevator group control system [3] and the Fuzzy Elevator Group Controller with Linear Context Adaptation [4] are some examples. Bio-inspired systems [5] and [6] have been revealed successful capacities. Here we propose a genetic algorithm based on a hall call allocation strategy to identify the chromosomes of the population individuals.

The rest of the paper deals with the simulation model in section 2, the genetic algorithm characteristics in section 3, the main results of the simulations showed in section 4 and the conclusions in the final section.

## 2 Simulation Model

We have made use of the ARENA v.5.0 software to simulate the possible event set. ARENA is a powerful interactive visual modelling system that makes use of the SIMAN language. The initial model consists of an animation zone and a module logical zone that can be divided into one controller zone, one passenger zone and two elevator zone for each of the cabins.

The ***animation zone*** is defined by the `Arrive` and `Depart` modules, which regulate the arrivals and departures of the passengers at the system.

In the ***controller zone***, one entity is created by lift to travel around the logical zone. When the passengers come into the lift, the passengers are joined to the `controller` entity shaping one only entity at the same time as holding all the particular individual entities attributes.

The ***passenger zone*** consists of the allocation of the `UpDown` attribute (1 if the passenger goes up and 2 otherwise) that is stated as function of the `Origin` and `Destination` attributes. So, the passenger is sent to the waiting queue if it exists. Otherwise the hall call allocation procedure is done by means of the correspondent optimisation algorithm (our genetic algorithm by the case).

For each one of the ***elevator zones***, when the lift arrives at a floor the subsequent actions must be checked and done if necessary: lift waits for calls, passengers leaves the lift, passengers come into the lift, lift allocation in case of full capacity, cabin call allocation and call evaluation.

When the lift arrives at a floor, the `state` of the lift is evaluated. If the lift `state` is set to zero, the lift is stopped and will have access to the `Waiting_for_Calls` submodule. If the lift is not stopped and it is carrying passengers, it inputs into the `Leaving_the_Lift` submodule. If it is not carrying passengers, it inputs into the `Taking_Passengers` submodule after a `Delay` to simulate the opening doors time (we use the delay variable `Time_Doors` (2.5 seconds).

When the lift arrives at a floor, the `Arrival_Evaluation` submodule presents three options: the lift continues up, the lift continues down or the lifts starts the deceleration process (preparing to stop). We use the `LDX (Transporter ID, unit number)` as an ARENA proprietary variable allowing to know the floor in which the lift is. We load this data in the variable `Level`.

After updating `Level`, if the lift is going up and the lift is in the ground floor, the lift is sent to the first floor; if the lift is going down and the lift is in the highest floor, the lift is sent to the last but one floor. Otherwise the simulation model checks if the lift has stopped in the floor that `Level` indicates (this data can be checked by means of the variable `Last_Visited_Floor`), in this case the lift is sent up or down depending on the trip direction. Otherwise the lift stops at the floor if there exists a cabin call or a hall call and the capacity is not full. If the lift is full capacity and it does not exist cabin calls, the lift is sent up or down depending on the trip direction.

## 3 Genetic Algorithm for the Controller

We propose a genetic algorithm that makes use of a hall call allocation strategy to perform the elevator group controller. For each time, *t*, the hall calls and the cabin calls of the system are evaluated, allocating the hall calls to one specific lift. Each time, *t*, the set of hall calls are reallocated allowing the subsequent modification if the system performance improves. Each time the set of decisions is taken managing all the available information (planning for the long term) but only carrying the immediate action out for each lift of the group: stop, upwards or downwards displacement.

So, each time, *t*, the simulation model makes a call to the controller optimisation module (the genetic algorithm) that returns the overall call allocation. The genetic algorithm is defined by the following characteristics.

### 3.1 Individuals and population

Two arrays of size [2·`Number_of_Floors`-2] define the individual chromosome. Each of the arrays defines the system `state` for each one of the lifts. The array is divided into two parts; the first refers to the up traffic and the second one to the down traffic.

The first `Number_of_Floors`-1 integers correspond to the hall calls in the upward direction from the ground floor to the highest floor. The second `Number_of_Floors`-1 integers correspond to the hall calls in the downward direction from the highest floor to the ground floor. Figure 1 depicts the chromosome individuals:

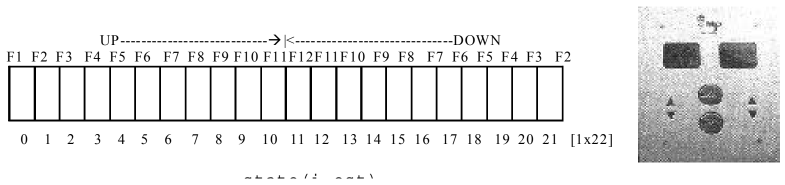
UP→|←DOWN
F1 F2 F3 F4 F5 F6 F7 F8 F9 F10 F11 F12 F11 F10 F9 F8 F7 F6 F5 F4 F3 F2
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 [1x22]

**Fig. 1.** Individual chromosome for a twelve floors case building corresponding to one specific elevator of the group and its associated physical button box

The array holds the information referring to the hall calls by means of a binary codification. The bit 0 indicates no hall call at the floor, and the bit 1 indicates an existing hall call at the floor.

In relation to the population size, our experiments show that increasing the population size beyond 20, although increasing the computational effort, is not rewarded by a corresponding increase of performance. So, we have maintained a population size of 20 individuals for our tests.

## 3.2 Fitness

We have used an approximate function (in seconds) to estimate the individual fitness. The fitness function returns the expected time in which the elevator group would serve the entire allocated hall calls and cabin calls. Obviously, it will be estimation because of the incapability of predicting the passenger future behaviour. The passenger arrival to the floor is random and their destinations are unknown.

The fitness estimation procedure depends on the elevator state (going up, down or stopped). However in every case it can be calculated by means of four peak values that we will note as P1, P2, P3 y P4.

Every time, the procedure has in account the overall allocated hall calls stating each new hall call to be allocated. Figure 2 shows the options depending on the up or down traffic.

***The elevator is stopped or going up***

- P1. Current floor.
- P2. Highest floor to take passengers up. In figure 2: displacement *a*.
- P3. Lowest floor to take passengers down. In figure 2: displacement *b*.
- P4. The highest floor among the floors lower than P1 to take passengers up, always P4<P1. In figure 2: displacement *c*.

$$\mathit{Fitness} = [(P2-P1)+(P2-P3)+(P4-P3)]\times[\text{estimated interfloor trip time}] \qquad (1)$$

Fitness (from equation 1) includes the maximum known upward trip plus the maximum known downward trip plus the subsequent maximum known not-served upward trip in the first up traffic because P4<P1. We have to note that the mathematical expression do not include the passenger destination trips because we unknown it until the passengers come into the cabin.

***The elevator is going down***

- P1. Current floor
- P2. Lowest floor to take passengers down. In figure 2: displacement *a.*
- P3. Highest floor to take passengers up. In figure 2: displacement *b.*
- P4. The lowest floor among the floors higher than P1 to take passengers down, always P4>P1. In figure 2: displacement *c.*

$$\mathit{Fitness} = [(P1-P2)+(P3-P2)+(P3-P4)]\times[\text{estimated interfloor trip time}] \qquad (2)$$

Fitness (from equation 2) includes the maximum known downward trip plus the maximum known upward trip plus the subsequent maximum known not-served downward trip in the first down traffic because of P4>P1.

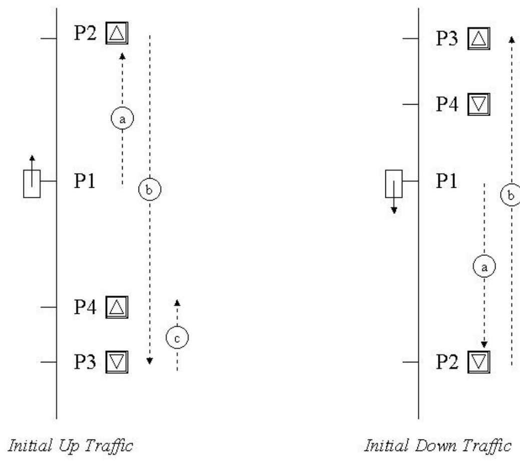

**Fig. 2.** Possible elevator streams to estimate the fitness

## 3.3 Operators

The genetic operators used are crossover and mutation. We have used an uniform crossover operator that randomly selects two individuals (parents) from the population and generates the offspring by crossing the individual genes. The offspring inherits an exact copy of those genes that are equal in the parents' chromosome and, in other case; it inherits each gene with probability of 50%. Although the parents' selection is random, the algorithm includes an incest prevention control when parents differ in less than a gene pair. The mutation operator replaces a hall call allocation from the individual chromosome by changing the genes from 01 to 10 or viceverse. The selection of the individual is random. We used a value of 85% for crossover and 15% for mutation.

## 3.4. Replacement rule

We propose the use of a hypergeometric function allowing more probability of replacement for individuals with worse fitness and less probability of replacement for individuals with better fitness. So, the individual in ranking position-$i$, have a replacement probability equal to $q(1\text{-}q)^i$, being $q$ the replacement probability of the worst individual. We obtained the better performances setting a value for $q$ between 55-65%. The main tests are run with a value of 60%.

Additionally to the replacement rule, we incorporate an individual duplicity control in the population generation.

## 4 Simulation Results

We have tested the algorithms in a twelve floors building. There are 30 workers in each of the building floors excepting the 7th floor (the administration department with 60 workers) and the 12th floor (the manager department with 15 workers). There are two 20 persons capacity elevators in the hall. The interfloor travel probabilities are defined within a lunchpeak traffic situation. It is important to note that lunchpeak traffic is the most critical situation in vertical traffic, because it includes the uppeak and downpeak traffic effects. So, from the ground floor to the 7th floor 15%, to the 12th floor 4% and to the rest of the floors 9%. And from the rest of the floors to the ground floor 95%, and to the rest of the floors 5%.

The next figure 3 depicts the arrival rate during lunchpeak traffic. Most of the workers go out for lunch during the interval [14:00,15:00] hours, returning to the building during [15:20,16:00] hours.

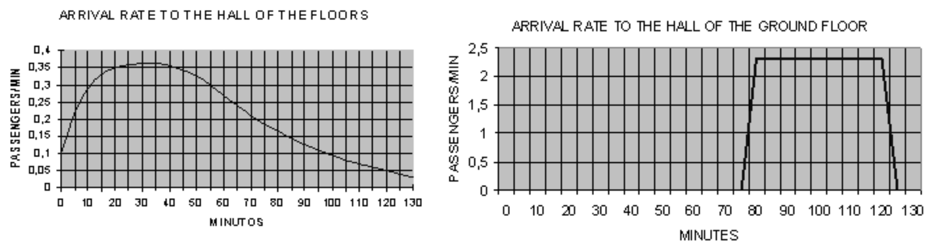

**Fig. 3.** Arrival rate to the halls.

### 4.1 Comparison of algorithms

Our genetic algorithm has been put in competition against the well-known THV Duplex algorithm. We have simulated 20 replications, table 1 summarises the results.

**Table 1.** THV and Genetic Algorithm system waiting time statisitical results

```
ARENA Simulation Results
PCA - License #8910593
Summary for Replication 20 of 20

Project:  THV DUPLEX lunchpeak             Run execution date:    16/07/2002
Analyst:  PCA                               Model revision date:    16/07/2002
Replication ended at time: 7800.0
TALLY VARIABLES
        Identifier        Average   Half Width   Minimum   Maximum   Observations
   Time_System    195.60     11.157      20.194     2899.3       44239

Project:  GENETIC ALGORITHM lunchpeak   Run execution date:     16/07/2002
Analyst:  PCA                                  Model revision date:     16/07/2002
Replication ended at time: 7800.0
TALLY VARIABLES
          Identifier        Average  Half Width   Minimum    Maximum   Observations
    Time_System    149.98      (Corr)      20.333      662.75       44237
```

The average system waiting time is 195.60 seconds for the THV algorithm. Every replication holds the minimum waiting time between 20 and 25 seconds. However it is worthwhile to highlight the maximum waiting time reaching 2899.3 seconds, which cannot be considered an isolated peak moreover. This is a heavy value and represents an approximation to the worst-case eventuality for the case: a building with 375 workers and two only lifts during the lunchpeak traffic.

The inadequate THV behaviour is due to several reasons. Firstly the THV takes the higher floors passengers down, after that the algorithm takes the rest of passengers in the other floors downwards. This phenomenon increases the waiting in the lower floors heavily. Moreover, the lunckpeak traffic refocuses this characteristic. During the lunchpeak traffic, the lifts saturate their capacity in the upper floors being not capable of taking additional passengers in lower floors. Afterwards, the lifts would come back to the top to take new passengers and the same phenomenon is repeated.

The genetic algorithm reduces the system waiting times in a considerable form. The average waiting time is reduced from 195.60 seconds to 149.98 seconds. It corresponds to the 23.32% reduction. In other line, the maximum waiting time is drastically reduced to 662.75 seconds. See figure 6.

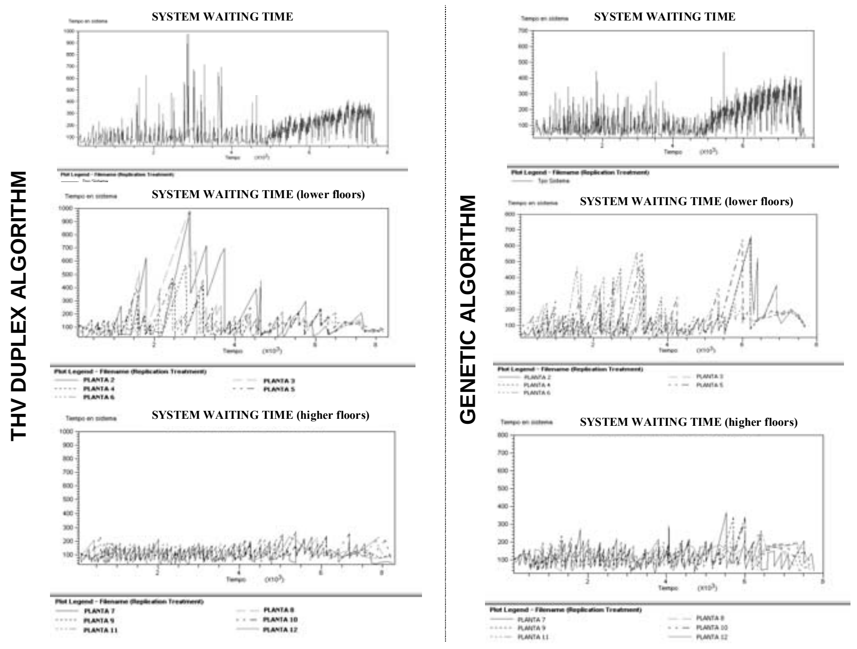

**Fig. 6.** Comparison of system waiting times by floors. The graphics depict the waiting times by floors as well as the average value. Note how the THV results are distributed between 0 and 1000 seconds, and the genetic algorithm results are distributed between 0 and 700 seconds.

The most important peaks appear around the 14:50 (after 3000 seconds). At this moment a lot of passengers are accumulating due to the lunchpeak effect. Another peculiar effect is observed after 4800 seconds (15:20), the graphic have a monotonic increasing tendency with less significant peaks. This change is due to the arrival of passengers to the ground floor hall after lunching. In every case both of the algorithms tend to benefit the passengers in the top of the building. The reason is that these passengers would be capable of using the stairs of the building with less probability than the passengers in lower floors (see waiting time for floor number 2 and 3). Note that the stairs effect has not been simulated.

## 5 Conclusions

We have proposed a genetic algorithm to control the elevator group in a professional building. The results allow us to affirm that our genetic algorithm reaches a better performance attending to the system waiting times than THV algorithm (the universal controller algorithm in industry applications). The reduction has been almost the 25%. In this situation the passengers are supposed to experiment a system time reduction from 3min15sec to 2min30sec. The analysis has been done under heavy lunchpeak traffic.

The results obtained allow us to affirm that genetic algorithms are valuable tools with a great potential in the control of elevator systems. However, the implementation of such type of algorithms in real controllers has to be done carefully in order to maintain bounded the response time of the algorithm. Genetic algorithms are iterative and therefore they could take very much time of execution. An alternative for real cases can be stopping the algorithm previously to reach the next critical event. Of course all this kind of decisions are very dependent on the computation speed of the electronic microchips installed by the company.

**Acknowledgements**
The authors acknowledge the financial support given by the Ministerio de Ciencia y Tecnologia, through the R+D National Program (project ref. DPI2002-01264), Spain.

## References

1. Barney, G.C. and S.M. dos Santos. Elevator Traffic Analysis, Design and Control (Peter Peregrinus Ltd, 2nd edition, London, 1985).
2. Siikonen, M-L.. Planning and control models for elevators in high-rise buildings, Ph.D. Thesis, Helsinki University of Technology, 1997.
3. Kim, C., K.A. Seong and H. Lee-kwang. Design and implementation of a fuzzy elevator group control system, en: Proceedings of the IEEE Transactions on systems, man and Cybernetics (1998), vol. 28, No. 3, 277-287.
4. Gudwin, R., F. Gomide and M.A. Netto. A Fuzzy Elevator Group Controller with Linear Context Adaptation, in: Proceedings of FUZZ-IEEE98, WCCI'98 - IEEE World Congress on Computational Intelligence, Anchorage, Alaska, USA (1998) 481-486.
5. Gudwin, R. and F. Gomide. Genetic Algorithms and Discrete Event Systems: An Application, en: Proceedings of The First IEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence (1994), vol II, 742-745.
6. Alander, J.T., J. Ylinen and T. Tyni. Elevator Group Control Using Distributed Genetic Algorithm, en: Proceedings of the International Conference. Springer-Verlag, Vienna, Austria (1995), 400-403.