A viral system algorithm to optimize the car dispatching in elevator group control systems of tall buildings

Pablo Cortés[1*], Luis Onieva[2], Jesús Muñuzuri[3], José Guadix[4]

University of Seville, Escuela Técnica Superior de Ingeniería, Ingeniería Organización
Camino de los Descubrimientos s/n, Sevilla 41092, Spain

Email: [1]pca@esi.us.es, (+34954486153); [2]onieva@esi.us.es; [3]munuzuri@esi.us.es; [4]guadix@esi.us.es
[*] *Corresponding author*

Abstract.- Nowadays is very common the presence of tall buildings in the business centres of the main cities of the world. Such buildings require the installation of numerous lifts that are coordinated and managed under a unique control system. Population working in the buildings follows a similar traffic pattern generating situations of traffic congestion. The problem arises when a passenger makes a hall call wishing to travel to another floor of the building. The dispatching of the most suitable car is the optimization problem we are tackling in this paper. We develop a viral system algorithm which is based on a bio-inspired virus infection analogy to deal with it. The viral system algorithm is compared to genetic algorithms, and tabu search approaches that have proven efficiency in the vertical transportation literature. The experiments undertaken in tall buildings from 10 to 24 floors, and several car configurations from 2 to 6 cars, provide valuable results and show how viral system outperforms such soft computing algorithms.

Keywords: Elevator; lift; viral system; bio-inspired algorithms; elevator group control system; vertical transportation

Acknowledgments

The authors acknowledge the financial support provided by the Spanish Plan for Research and Development Technology and Innovation Plan through its Programme Industrial Production and Design, project Ref. DPI2010-15352.

A viral system algorithm to optimize the car dispatching in elevator group control systems of tall buildings

Abstract.- Nowadays is very common the presence of tall buildings in the business centres of the main cities of the world. Such buildings require the installation of numerous lifts that are coordinated and managed under a unique control system. Population working in the buildings follows a similar traffic pattern generating situations of traffic congestion. The problem arises when a passenger makes a hall call wishing to travel to another floor of the building. The dispatching of the most suitable car is the optimization problem we are tackling in this paper. We develop a viral system algorithm which is based on a bio-inspired virus infection analogy to deal with it. The viral system algorithm is compared to genetic algorithms, and tabu search approaches that have proven efficiency in the vertical transportation literature. The experiments undertaken in tall buildings from 10 to 24 floors, and several car configurations from 2 to 6 cars, provide valuable results and show how viral system outperforms such soft computing algorithms.

Keywords: Elevator; lift; viral system; bio-inspired algorithms; elevator group control system; vertical transportation

1. Introduction

Transportation is a classic discipline with a long tradition in the industrial engineering research (see for example Xie and Jia, 2012; Yan et al. 2012; Yin and Kim, 2012; Keshavarz and Khorram, 2011; or Muñuzuri et al. 2010 amongst others as recent contributions). A particular case of transportation is vertical transportation which considers the vertical transportation of passengers or freight in a building through its floors. In fact, in the late years, the rapid growth of the building industry and associated technologies has been demanding parallel growth in the field of vertical transportation. The progressive price increase in the urban centres of the larger cities makes the necessary intensive ground exploitation by means of the construction of tall buildings (see figure 1 depicting an image of the Manhattan business centre). To manage with the vertical transportation system of such buildings, the installation of synchronized elevator groups in is a usual practice.

Figure 1. A landscape of the upper Manhattan skyscrapers from the Empire State Building

The main objective of an optimal elevator group control system (EGCS) is to provide a good quality of service to its passengers (see CIBSE Guide, 2005). The problem tackled by every EGCS is to solve the assignment process of a car of the elevator group to a hall call made by a passenger wanting to travel from a floor to other different floor in the building. When a passenger presses a landing call button of the panel in the hall, he/she expects that a car of the EGCS will arrive in a few seconds. This assignment process must be done analysing the different options and selecting the most suitable car to serve the person having issued the call. This assignment must be done optimizing the waiting times that the passenger will experiment. The most common optimization criterion (Cortés et al., 2006) for quality of service in elevator group controllers is the reduction of the average journey time (AJT) which consists of the waiting time queuing in the hall (average waiting time, AWT) plus the travel time to destination inside the car (average travel time, ATT).

The average waiting time (AWT) is the actual time a prospective passenger waits after registering a hall call (or entering the waiting queue if a call has already been registered) until the responding elevator doors begin to open. Following the CIBSE Guide, for car loads less than 50%, AWT can be approximated by (1), while for car loads over 50%, it is approximated by (2)

$$AWT = 0.4INT \text{ , for car loads < 50 \%} \quad (1)$$

$$AWT = \left[0.4 + \frac{1.8P}{CC} - 0.77^2 \right] INT \text{ , for car loads > 50 \%} \quad (2)$$

Where *INT* is the interval (the main floor arrival average time), *P* is the number of passengers, and *CC* is the rated car capacity.

The average travel time (ATT) is the time the responding elevator doors begin to open to the time the doors begin to open again at the passengers' destination. Following the CIBSE Guide again, it is calculated as (3).

$$ATT = t_v \frac{H}{2S}(S + 1) + t_s \frac{S+1}{2} + t_p \cdot P \quad (3)$$

Where *H* is the highest reversal floor, *S* the expected number of stops, t_v the single floor transit time (in seconds), t_S the stopping time (in seconds), t_P the passenger transfer time (in seconds), and *L* the number of cars within the elevator group.

Therefore, the problem that an EGCS must tackle consists of the allocation of a car of the group to a hall call made by a passenger based on a criterion such as the AJT, and it is a NP-Hard problem that presents n^k possible combinations for *n* floors and *k* lifts, and therefore it is an extreme computational complexity problem.

In the recent years, research on EGCS for vertical transportation in buildings is gaining a considerable interest from the scientific community. Although mathematical approaches have been sometimes considered (see Zhang et al., 2010; Mulvaney et al., 2010; or Utgoff and Connell, 2012), advanced approaches including soft computing or artificial intelligence algorithms are prevailing to provide optimal car-call allocation for EGCS in nowadays. In this line, genetic algorithms have been widely used providing good and valuable results since a long time (Fujino et al., 1997; Cortés et al., 2004; Tyni and Ylinen, 2006) and research continues being undertaken in this field (Hirasawa et al., 2008; Bolat et al., 2010). Also, other soft computing techniques such as particle swarm optimization have been also applied (Li et al., 2007), as well as immune systems algorithms (Li et al., 2007). Tabu search has attracted less

attention than genetic algorithms, although recently two algorithms based on deterministic and probabilistic approaches have been presented to deal with the problem (Bolat et al., 2011). Lastly, control systems based on fuzzy logic are being enthusiastically tried recently (Jamaludin et al., 2010; Rashid et al., 2011; or Cortés et al., 2012) too.

Recently, viral systems which are a novel bio-inspired technique has been introduced in the literature devoted to applied bio-inspired algorithms (Cortés et al., 2008) providing successful applications to the Steiner problem in networks (Cortés et al., 2010), knapsack problem (Suryadi and Kartika, 2011), vehicle routing problem (Pérez-Martínez et al., 2011), or scheduling problems (Cortés et al., 2012). It is interesting to note that an application of the biological virus concept was developed by Kubota et al. (1996) as a genetic operator of a genetic algorithm applied to a self-organizing manufacturing system.

Here, we have applied this novel technique to deal with the optimization of the hall call-car allocation system of the EGCS. Following the method, cells are defined as the feasible solutions that are encoded following the Cortés et al. (2004) strategy that has been widely accepted by the vertical transportation scientific community. In addition, we follow the Bolat et al. (2011) methodology to make a quick evaluation of the cell fitness.

The remaining of the paper deals with the presentation of the problem and the adaptation of the viral system algorithm to solve the car-call allocation optimization in EGCS, which is shown in section 2. Then in section 3, experimental results are provided for tall buildings from 10 to 24 floors, and several car configurations from 2 to 6 cars. Finally, the main conclusions are drawn in the final section.

2. The viral system approach to optimize the elevator group control system

The elevator group control system of a building includes a set of microchips in controllers to determine which car of the group should serve each hall call. When passengers arrive to the hall of the building and press an up or down button the EGCS must assign a car of the group to such call.

Passengers arrive to a building following specific traffic patterns. Four main patterns are traditionally catalogued (see CIBSE Guide, 2005 and Benmakhlouf and Khatorfor, 1993 amongst others). Figure 2 shows usual patterns in an office building. Uppeak traffic characterizes a larger than average number of up landing calls; it typically appears when workers arrive to the building in the early morning to start their working day. Downpeak traffic characterizes a larger than average number of down landing calls; it takes place when the workers leave the building to go back home. A lunchpeak traffic pattern takes place in the middle of the day, and it is due to the appearance of up and down peaks; it appears usually for lunching at midday. Finally interfloor traffic corresponds to the rest of the day; this pattern is characterized for a low demand (usually around a 4% of the population) in both directions.

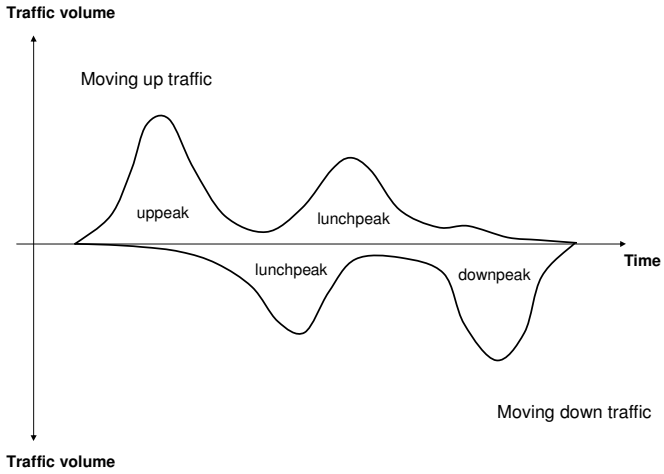

Figure 2. Traffic pattern in an office building

This section presents an algorithm that assigns the most suitable car to a hall call made by a passenger arriving to the hall of a building. The next subsections show the solution encoding (cell genome in terms of the viral system); the evaluation of the solution (that is, the fitness evaluation of such genome when infected by a virus); and the details of the viral system implementation.

2.1. Cell genome definition

The possible solutions for the allocation of a hall call to a car are encoded according to binary array (see Cortés et al., 2004 for a first proposal). So, one array is associated to each car in the elevator group, and is defined by representing the up and down calls at each floor that are assigned to this specific car. So the length of each array is equal to 2×(Number_of_Floors-1).

So, 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 3 depicts the solution encoding.

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

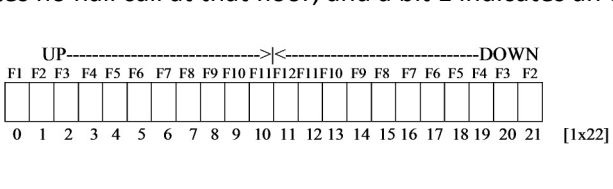

Figure 3. Solution encoding for a twelve story building corresponding to one car of the group and its associated physical button box

2.2. Evaluation of the cell fitness

Each possible car-hall call allocation (cell) is evaluated following the methodology described in Bolat et al. (2011), which has been proved to be an easy-to-implement and fast-to-compute technique.

So, given the following parameters:

- L_1: ground floor level
- L_2: highest down hall call level
- L_3: number of down hall calls between L_1 and L_2.
- L_4: highest up hall call level
- L_5: number of up hall calls between L_1 and L_4.
- L_6: lowest down hall call level
- t: door opening and closing time
- t_p: passenger transfer time
- *Hct*: Highest car trip time
- *Lct*: lowest car trip time

The cell fitness, f, is calculated depending on the type of passengers' movements. So, a fitness value is firstly calculated for each car, i, in the group taking into account the three different cases of table 1. Each formula relates in which floor the passenger is taken and in which floor the passenger is transferred. So, as a general definition for L, it shows the floors where passengers are get on and off. In case of no hall calls $f_i = 0$.

Table 1. Fitness evaluation as function of the traffic pattern in the building

Uppeak traffic pattern	Downpeak traffic pattern	Lunchpeak or interfloor traffic patterns
$f_i = t(L_4-L_1) + t_p(L_5-L_1)$	$f_i = t(L_2-L_1) + t_p(L_3-L_1)$	$f_i = t(L_4-L_1) + t(L_2-L_4) + t(L_2-L_6) + t_p(L_3+L_5-L_1)$

Finally the group fitness evaluation is given by (4), and the final fitness is evaluated for the proposed assignment and given in (5). Parameters k_1 and k_2 are chosen equal to 1.5 and 2 following the recommendations given in Bolat et al. (2011).

$$f_{group} = \frac{\sum_{i=1}^{n} f_i}{n}, \text{being } n \text{ the number of cars in the group} \tag{4}$$

$$f = k_1 \cdot f_{group} + k_2 \cdot (Hct - Lct) \tag{5}$$

2.3. Viral system algorithm

A viral system algorithm (VS) is inspired in the virus behaviour. Viruses are intracellular parasites shaped by nucleic acids, such as DNA or RNA, and proteins. The protein generates a capsule, called a capsid, where the nucleic acid is located. The capsid plus the nucleic acid shape the nucleus-capsid, defining the virus. Although there are many different types of viruses developing with different behaviours when infected cells, here we follow the specific case of phages. A phage is any one of a number of viruses that infect bacteria. They do this by injecting genetic material, which they carry enclosed in an outer protein capsid.

One of the main characteristics of phages is the replication mechanism which follows a lytic replication process. A lytic replication process follows the next steps: (i) the virus is adhered to the border of the bacterium. After that, the virus penetrates the border being injected inside this one; (ii) the infected cell stops the production of its proteins, beginning to produce the phage proteins, starting to replicate copies of the virus nucleus-capsids; and (iii) after replicating a number of nucleus-capsids, the bacterium border is broken, and new viruses are released, which can infect near cells.

As an alternative to lytic replication, some viruses are capable of lodging in cells giving rise to the lysogenic replication. A lysogenic replication follows the next steps: (i) the virus infects the host cell, being lodged in its genome, where a pro-phage (mutation) can arise; (ii) the virus remains hidden inside the cell during a while until it is activated; and (iii) the replication of cells altered, with proteins from the virus, starts. So, lysogenic replication produces the genome alteration of the cell leading to a procedure similar to a mutation process.

On the other hand, phages have the property of leading an antigenic response in the infected organism. In these situations an immune response is originated causing the creation of antibodies. So, in such case the infection is stopped for that cell.

So in computational terms, VS follows an exploration process that combines lytic replication to search the neighbourhood of the existing solutions together with a mutation process, including some randomness due to the probability of developing antibodies. VS has proved effectiveness for massive and selective infections (Cortés et al., 2008; Cortés et al., 2010). However, here we opted for a selective infection due to its lighter computational load required by a real time problem as the EGCS. Furthermore, here we have developed the evolution of a virus through the organism instead of a population of virus trying to make faster the computation of the algorithm too.

Once a selective infection takes place and viruses are liberated inside the organism, the virus selects a cell with a low value of fitness, $f(x)$, in the neighbourhood. This infected cell is stored in the clinical picture (accordingly with viral system notation). The clinical picture includes the encoding of the solution that is being explored (the genome of the cell that is infected, in biological terms) and the number of nucleus-capsids being replicated, NR, (for lytic replications) or the number of hidden generations, IT, (for lysogenic replications). Thus the state of each virus is given by the three-tuple "cell genome-NR-IT".

Every cell infected by a virus develops a lytic or a lysogenic replication according to a probability p_{lt} (for lytic replication) or p_{lg} otherwise, where $p_{lt} + p_{lg} = 1$.

In case of lysogenic replications, the activation of the mutation process takes place after a limit of iterations has been reached (LIT). The value of LIT depends on the cell's health conditions, so a healthy cell (high value of the objective function being minimised, $f(x)$) will have a low infection probability, i.e. the value of LIT will be higher. An unhealthy cell, on the contrary, will have a lower value of LIT. See Cortés et al. (2008) for adjusting the value of LIT.

In case of lytic replications, a number of virus replications (NR) is calculated for each iteration as a function of a binomial variable, Z, adding its value to the current NR in the clinical picture. Z is calculated using a Binomial distribution given by the maximum level of nucleus-capsids replicated, LNR, and the single probability of one replication, p_r. So, Z = Bin (LNR , p_r). LNR represents the limit to break the cell border and to release the lodged viruses. As in the lysogenic cycle, the value of LNR is set depending on the value of the objective function being minimised, $f(x)$. Thus cells with higher $f(x)$ have lower probability of getting infected, and therefore the value of LNR will be higher. See Cortés et al. (2008) for adjusting the value of LNR.

However, the virus will not be able to infect those cells that have developed antigens. Given $p_{an}(x)$ the probability of generating antibodies for a cell x, the antibody response is defined by the random variable $A(x)$ as a Bernoulli: $A(x) = \text{Ber}(p_{an}(x))$. If cell x generates antibodies, the cell is not infected, but starts a lysogenic process.

Higher values of $f(x)$ imply healthy cells and therefore cells that have a higher probability of developing antigenic responses. On the contrary, cells with low value of $f(x)$ imply unhealthy cells with lower probability of developing antigenic responses. This effect is represented by an hypergeometric function, where the cell with an inverse objective function evaluation, $1/f(x)$, in ranking position-i, has a probability of generating antibodies, $p_{an}(x)$, that is given by $q(1\text{-}q)^i$, with q equal to the probability of generating antibodies for the worst healthy cell. Finally, a residual probability remains, which is added to the worst cell.

Figure 4 defines the algorithm evolution for the lytic replication. The initial state is on the left-hand side: the virus process starts with the virus breaking the border and starting the infection of new cells in its neighbourhood. The virus selects the most promising cell, which is the least healthy cell. The Organism process is characterized by the probability of antigenic response in the least healthy cell. Those cells developing antibodies are not infected. Finally, the right hand side of the figure defines the new clinical picture, with new infected cells lodging viruses. The cells generating antibodies follow a new lysogenic replication in order to gain computational efficiency.

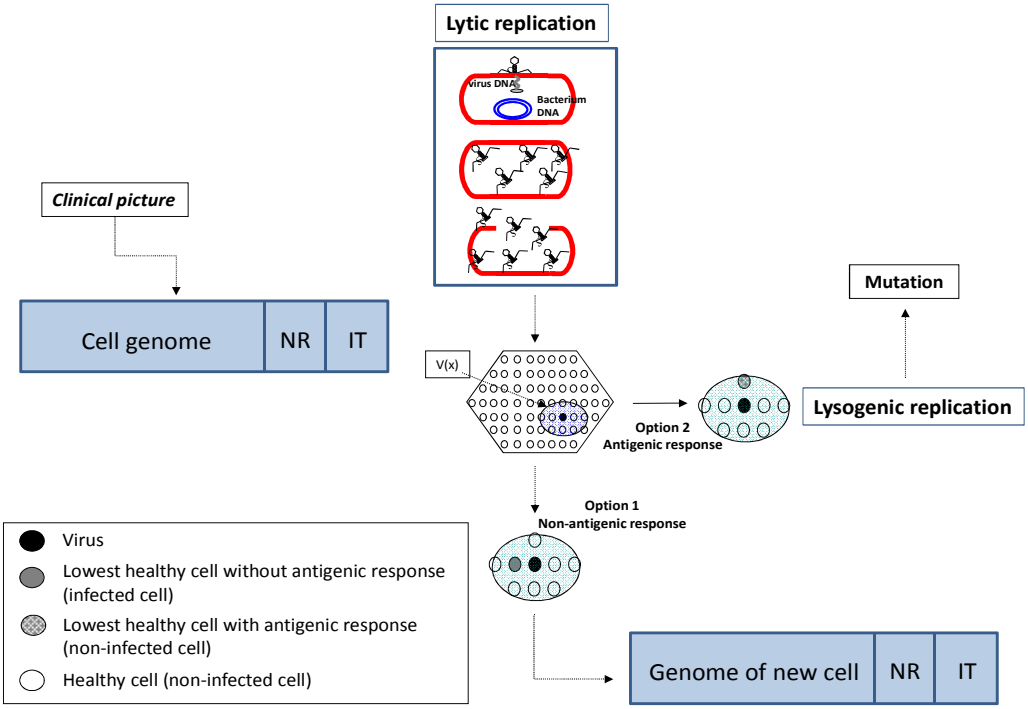

Figure 4. Viral system lytic replication evolution concept

The flowchart of the algorithm is presented in figure 5.

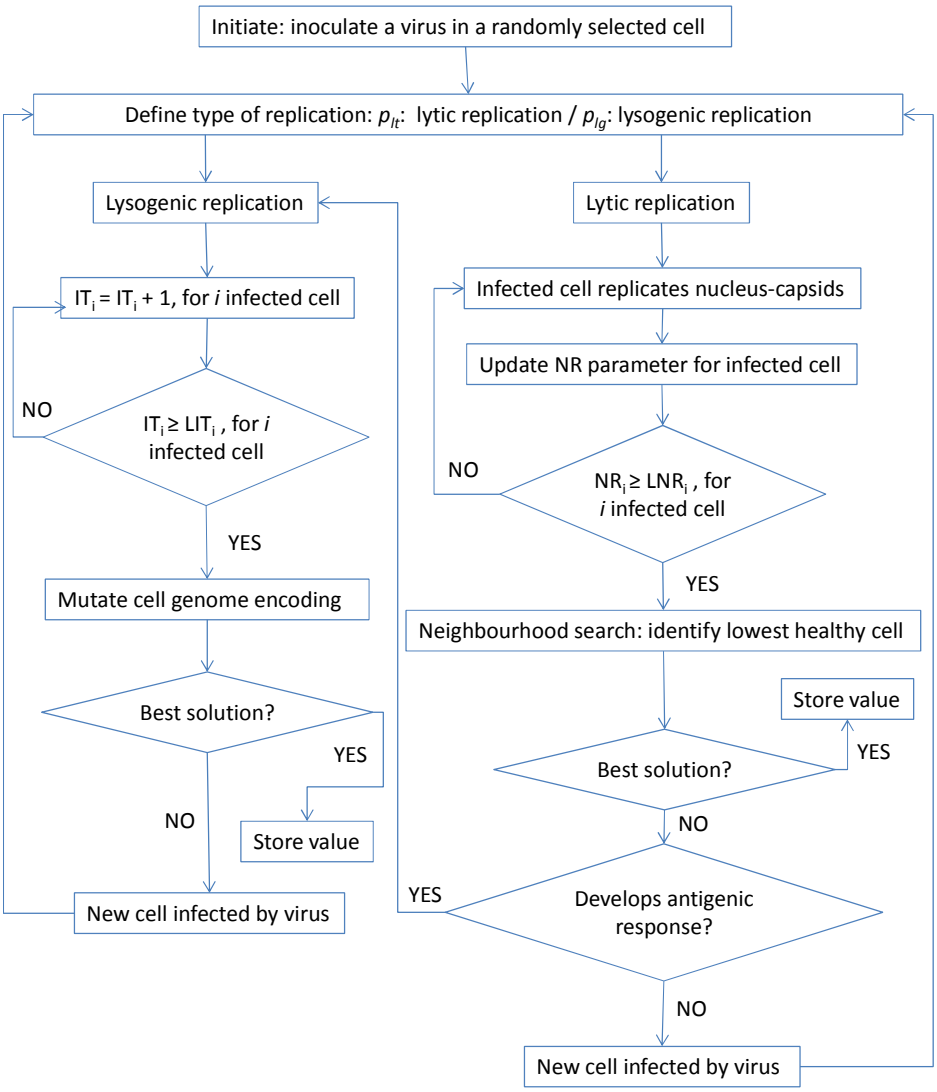

Figure 5. Viral system flowchart

3. Experimental results

The experimentation was carried out using an Intel Pentium M Processor 740 at 1.73 GHz with 2 GB RAM, and algorithms were encoded using software Matlab version 7.0.

Tests were carried out for tall buildings from 10 to 24 floors at 3.3 metres of distance. We also considered different car configurations for the EGCS: 2, 3, 4 and 6 cars. Table 2 gives the main specifications of the building and elevators respectively. Total time of a car stopping is given by $t_s = t_{open}+t_{close}+t_{p,}$ where t_{open} is the time for opening door, t_{close} is the time for closing door, and t_p is the time for passenger transfer.

Parameters of the viral system algorithm were calibrated to the following values (table 3) after testing and trying with different combinations. The algorithm was 10,000 times iterated.

Table 2. Specifications of the elevator system

Items for elevator system	Value
Car capacity (people)	8
Time of travel between floors	4.5 (s)
Time for opening door	2.5 (s)
Time for closing door	3 (s)
Time for passenger transfer	3 (s)

Table 3. Specifications of the elevator system

Parameter	Value
Lytic probability / Lysogenic probability	$p_{lt} = 0.7$ / $p_{lg} = 0.3$
LNR (limit of nucleus-capsids)	15
LIT (limit of iterations)	10
Antigenic response probability	$p_{an} = 0.053$
Single replication probability	$p_r = 0.7$

We compared the viral system algorithm with the genetic algorithm, and the tabu and probabilistic tabu search implementations addressed in Bolat et al (2010), and Bolat et al. (2011). Average journey time analysis is provided in table 4 for each approach for buildings that have 10 to 24 floors and 2 to 6 cars.

Table 4. Average Journey Time comparison

Floors	2 cars				3 cars				4 cars				6 cars			
	GA[1]	TS[2]	PTS[3]	VS[4]	GA	TS	PTS	VS	GA	TS	PTS	VS	GA	TS	PTS	VS
10	58.5(SPC) 61.5(TPC) 58.5(UC)	58.5	58.5	45.5	41(SPC) 40(TPC) 39(UC)	42.0	41.0	31.4	37.5(SPC) 35.25(TPC) 37.5(UC)	41.3	31.5	26.8	32.5(SPC) 31.5(TPC) 31(UC)	33.0	27.5	17.7
12	60(SPC) 60(TPC) 60(UC)	63.0	64.5	57.9	46(SPC) 47(TPC) 47(UC)	46.0	45.0	34.3	42(SPC) 38.25(TPC) 39.75(UC)	40.5	42.0	32.5	36(SPC) 32.5(TPC) 39(UC)	37.0	31.5	22.4
14	82.5(SPC) 75(TPC) 76.5(UC)	82.5	76.5	71.9	54(SPC) 53(TPC) 54(UC)	56.0	51.0	47.7	46.5(SPC) 45(TPC) 45.75(UC)	46.5	46.5	39.3	40(SPC) 40(TPC) 40(UC)	48.5	40.5	27.0
16	72(SPC) 75(TPC) 72(UC)	72.0	72.0	86.8	56(SPC) 52(TPC) 61(UC)	60.0	52.0	58.7	48.75(SPC) 46.5(TPC) 48(UC)	48.8	48.8	45.0	49.5(SPC) 45.5(TPC) 46(UC)	49.5	47.5	33.8
18	90(SPC) 90(TPC) 84(UC)	87.0	87.0	66.6	66(SPC) 65(TPC) 66(UC)	65.0	64.0	59.8	60(SPC) 63.75(TPC) 61.5(UC)	60.8	59.3	57.0	49(SPC) 53.5(TPC) 61(UC)	67.5	50.0	37.7
20	93(SPC) 93(TPC) 99(UC)	108.0	99.0	99.6	89(SPC) 74(TPC) 84(UC)	84.0	76.0	60.3	75.75(SPC) 69(TPC) 66(UC)	74.3	59.8	64.6	66(SPC) 64.5(TPC) 67.5(UC)	70.0	66.5	43.4
22	105(SPC) 120(TPC) 105(UC)	105.0	109.5	96.6	87(SPC) 85(TPC) 81(UC)	87.0	89.0	82.4	78.75(SPC) 77.25(TPC) 76.5(UC)	81.0	75.8	60.4	68.5(SPC) 69.5(TPC) 66.5(UC)	71.0	64.5	47.9
24	115.5(SPC) 115.5(TPC) 115.5(UC)	115.5	111.0	117.3	95(SPC) 89(TPC) 90(UC)	90.0	90.0	68.1	80.25(SPC) 78.75(TPC) 73.5(UC)	87.8	74.3	72.7	64(SPC) 67(TPC) 65.5(UC)	66.5	63.0	53.4

[1] Genetic algorithm (Bolat et al., 2010); SPC: single point crossover; TPC: two point crossover; UC: uniform crossover
[2] Tabu search (Bolat et al., 2011)
[3] Probabilistic tabu search (Bolat et al., 2011)
[4] Viral system

The results confirm that the viral system approach outperforms the genetic and tabu approaches practically in every configuration. Results provided for VS are only slightly worse for specific configurations given by two cars. Just, these cases represent those situations where an EGCS is less necessary and soft computing approaches can be avoided by using traditional heuristic dispatching rules. When the complexity of the EGCS increases, as well as the height of the building, VS provides very interesting results outperforming the AJT provided by other methods.

Figure 6 depicts the evolution of AJT when the number of floors grows and the number of cars decreases. The increase of AJT is especially significant for the case of 2 cars and more than 16 floors.

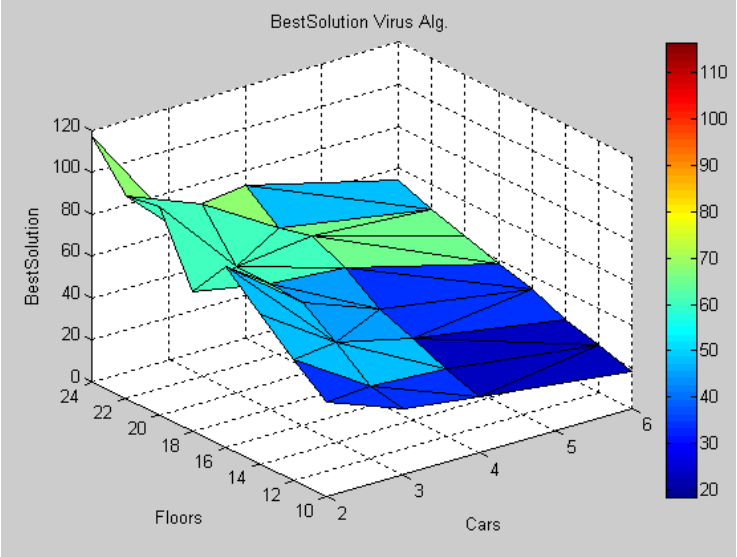

Figure 6. VS AJT evolution with respect to the number of cars in the EGCS, and floors in the building

The computational required for each approach is provided in table 5 for the considered configurations.

PTS produced the quickest results for low car configurations (2 and 3 cars) although its computational time increases significantly for higher configurations. In general terms, GA provided the best results from a computational perspective, specifically when the size of the group grows (cases with 4, and especially 6 cars). The computational time provided by the viral system algorithm was better than the results provided by the tabu approaches especially for those complex cases (that is, 4 to 6 cars in the group, and tall buildings). Results provided by the VS were bounded in general terms and did not rise as significantly as the tabu approaches following a more moderate increase as the genetic approaches.

Table 5. Computational time comparison

Floors	2 cars				3 cars				4 cars				6 cars			
	GA[1]	TS[2]	PTS[3]	VS[4]	GA	TS	PTS	VS	GA	TS	PTS	VS	GA	TS	PTS	VS
10	1.6 (SPC) 3.0 (TPC) 1.6 (UC)	3.2	0.5	6.5	2.6 (SPC) 2.5 (TPC) 2.4 (UC)	4.4	0.8	8.1	2.9 (SPC) 2.2 (TPC) 1.8 (UC)	9.3	6.2	9.7	1.9 (SPC) 1.9 (TPC) 2.0 (UC)	43.6	27.3	12.6
12	2.0 (SPC) 1.9 (TPC) 2.2 (UC)	6.5	0.4	6.6	2.3 (SPC) 2.5 (TPC) 2.5 (UC)	4.8	0.8	8.1	1.7 (SPC) 1.7 (TPC) 1.7 (UC)	22.0	2.9	9.7	2.0 (SPC) 2.5 (TPC) 2.0 (UC)	57.2	22.3	13.0
14	2.2 (SPC) 2.0 (TPC) 2.0 (UC)	5.0	0.5	6.5	2.2 (SPC) 2.2 (TPC) 2.6 (UC)	9.2	0.8	8.1	1.8 (SPC) 1.7 (TPC) 1.5 (UC)	30.7	4.7	9.8	2.0 (SPC) 2.0 (TPC) 2.0 (UC)	81.6	29.6	12.9
16	2.1 (SPC) 2.3 (TPC) 2.1 (UC)	6.8	0.5	7.0	2.2 (SPC) 2.4 (TPC) 2.2 (UC)	9.9	0.7	8.3	1.8 (SPC) 1.7 (TPC) 1.7 (UC)	28.9	5.1	9.8	1.8 (SPC) 1.9 (TPC) 1.7 (UC)	104.1	16.9	13.0
18	2.3 (SPC) 2.3 (TPC) 2.0 (UC)	4.8	0.6	6.5	2.4 (SPC) 2.3 (TPC) 2.2 (UC)	13.5	1.5	8.3	1.7 (SPC) 1.7 (TPC) 1.8 (UC)	67.5	8.4	9.9	1.9 (SPC) 1.9 (TPC) 2.1 (UC)	183.5	46.2	13.0
20	2.0 (SPC) 2.3 (TPC) 2.1 (UC)	4.6	0.6	6.8	4.8 (SPC) 2.4 (TPC) 2.4 (UC)	13.0	1.9	8.4	1.7 (SPC) 1.9 (TPC) 1.9 (UC)	57.0	6.3	9.8	2.1 (SPC) 2.1 (TPC) 1.9 (UC)	167.8	57.5	13.1
22	2.2 (SPC) 2.2 (TPC) 2.0 (UC)	5.4	0.6	6.6	2.3 (SPC) 2.5 (TPC) 2.4 (UC)	9.4	2.0	8.3	1.9 (SPC) 1.8 (TPC) 1.9 (UC)	50.4	14.0	9.9	1.8 (SPC) 1.9 (TPC) 2.1 (UC)	173.0	60.8	13.2
24	2.3 (SPC) 2.1 (TPC) 2.0 (UC)	6.1	0.7	7.0	2.7 (SPC) 2.8 (TPC) 2.5 (UC)	16.3	2.1	8.3	1.8 (SPC) 1.8 (TPC) 1.8 (UC)	89.4	12.8	10.0	1.8 (SPC) 1.9 (TPC) 2.5 (UC)	134.4	47.8	13.3

[1] Genetic algorithm (Bolat et al., 2010); SPC: single point crossover; TPC: two point crossover; UC: uniform crossover
[2] Tabu search (Bolat et al., 2011)
[3] Probabilistic tabu search (Bolat et al., 2011)
[4] Viral system

Figure 7 shows the computational time evolution with respect to the number of floors, and the number of cars. The higher values are obtained for cases of higher number of cars, and tallest buildings.

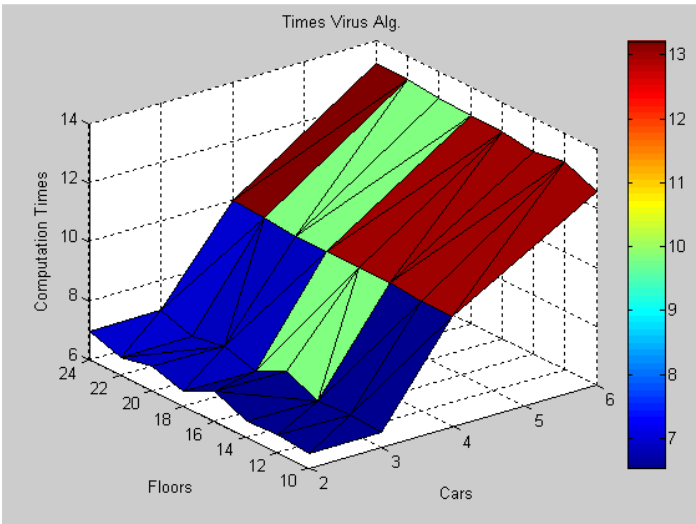

Figure 7. Computational time evolution with respect to the number of cars in the EGCS, and floors in the building

Another important aspect arises when analyzing the number of iterations of the viral system. If the number of iterations is reduced from 10,000 to 1,000 the quality of the solutions is practically the same (AJT is increased only in a 1 to 2% as average) and the computational times are drastically reduced to a value in the order of magnitude of genetic implementations. An example is provided for the case of a building with 24 floors and 6 cars in figure 8: the best value is reached at the 446th iteration providing a fitness value equal to 53.42 seconds. The computational time required to get such value was 2.5 seconds, very similar to the times required in the genetic implementations. In addition, another good fitness value (55.04 seconds) was reached at the 751st iteration. In general for every case, a maximum of 1,000 iterations in the VS can be considered enough, and it provides values around 2 seconds which is suitable time consumption for a real time control as an EGCS requires.

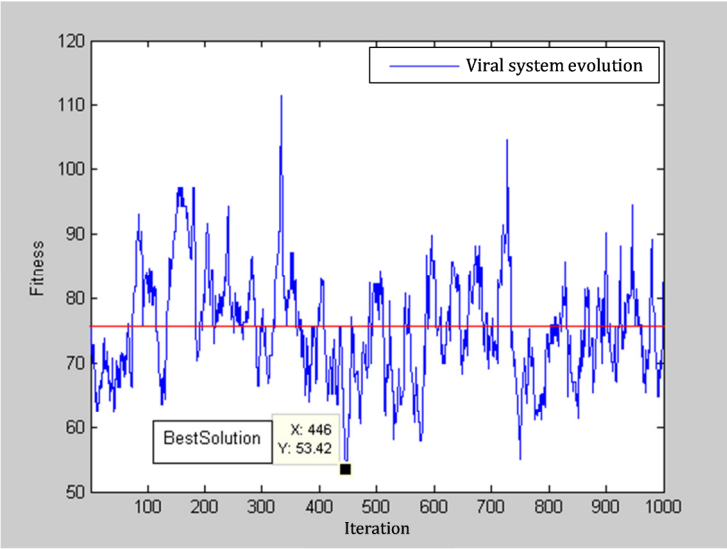

Figure 8. Learning curve of the algorithm for a building with 24 floors and 6 cars for 1,000 iterations

4. Conclusions

This paper presents the application of a novel bio-inspired algorithm called viral system to the car dispatching problem for multi-cars elevator group control systems. The problem arises when passengers make a landing call in the hall of the building wanting to travel from a floor to other floor of the building. The algorithm makes use of a binary encoding strategy to identify the cars being assigned to the landing calls, and of a fitness estimation allowing a quick evaluation. This estimation requires the identification of the type of traffic pattern in the building: uppeak, downpeak, lunchpeak or interfloor.

The viral system algorithm provided valuable figures for the average journey time (AJT) when was compared to genetic algorithms, and tabu search approaches that have proven efficiency in the vertical transportation literature. The experiments were undertaken in tall buildings from 10 to 24 floors, and several car configurations from 2 to 6 cars, and the better results were obtained for the more complex configuration implying larger car groups and taller buildings.

The computational time required by the VS to find the best solution was lower than the tabu approaches, and it was in the order of magnitude of the genetic implementations when using

1,000 iterations for the VS. It can be said that the required computational time was adequate when dealing with a real time problem such as the elevator group control system. In addition, the better results attending to the computational time were again obtained for the more complex configurations. Even more, it has to be noted that results obtained with a CPU can be limited with respect the real industry operation. In practice, real implementations are installed on proprietary microchips requiring lighter implementations. Bounding this fact, the real implementation of the VS algorithm in the industry appears to be possible. For example, in real cases an alternative can be calculating the fitness not every time but in a selective manner, or stopping the algorithm after a lower number of iterations. Of course all these decisions are very dependent on the computation speed of the electronic microchips installed by the company in the controller, and could affect the quality of the implemented algorithm.

References

Benmakhlouf, S.M., Khator, S.K., 1993. Smart lifts: Control design and performance evaluation. Computers & Industrial Engineering, 25(1-4), 175-178.

Bolat, B., Cortés, P., Yalcin, E., Alisverisci, M., 2010. Optimal car dispatching for elevator groups using genetic algorithms. International Journal of Intelligent Automation and Soft Computing, 16(1), 89-99.

Bolat, B., Cortés, P., 2011. Genetic and tabu search approaches for optimizing the hall call - car allocation problem in elevator group systems. Applied Soft Computing 11, 1792-1800.

Cortés, P., Fernández, J.R., 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.

Cortés, P., García, J.M., Muñuzuri, J., Guadix, J., 2010. A Viral System massive infection algorithm to solve the Steiner tree problem in graphs with medium terminal density. International Journal of Bio-Inspired Computation 2(2), 71-77.

Cortés, P., García, J.M., Muñuzuri, J., Guadix, J., 2012. Viral system algorithm: foundations and comparison between selective and massive infections. Transactions of the Institute of Measurement and Control 34 (6), 677-690.

Cortés, P., García, J.M., Muñuzuri, J., Onieva, L., 2008. Viral systems: A new bio-inspired optimisation approach. Computers and Operations Research 35(9), 2840-2860.

Cortés, P., Muñuzuri, J., Onieva, L., 2006. Design and analysis of a tool for planning and simulating dynamic vertical transport. Simulation 82 (4), 255-274.

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

Fujino, A., Tobita, T., Segawa, K., Yoneda, K., Togawa, A., 1997. An elevator group control system with floor-attribute control method and system optimization using genetic algorithms. IEEE Transactions on Industrial Electronics 44(4), 546-552.

Hirasawa, K., Eguchi, T., Zhou, J., Yu, L., Hu, J., Markon, S., 2008. A double-deck elevator group supervisory control system using genetic network programming. IEEE Transactions on Systems, Man, and Cybernetics Part C, Appl. Rev. 38(4), 535-550.

Jamaludin, J., Rahim, N.A., Hew, W.P., 2010. An elevator group control system with a self-tuning fuzzy logic group controller. IEEE Transactions on Industrial Electronics 57(12), 4188-4198.

Keshavarz, E., Khorram, E., 2011. A fuzzy bi-criteria transportation problem. Computers & Industrial Engineering, 61(4), 947-957.

Kubota, N., Fukuda, T., Shimojima, K., 1996. Virus-evolutionary genetic algorithm for a self-organizing manufacturing system. Computers & Industrial Engineering, 30(4), 1015-1026.

Li, Z., Tan, H.-Z., Zhang, Y., 2007. Particle swarm optimization applied to vertical traffic scheduling in buildings. Lecture Notes in Computer Science 4692 LNAI (PART 1), 831-838.

Li, Z., Tan, H.-Z., Zhang, Y.-N., Mao, Z.-Y., 2007. Dynamic optimization of elevator group control based on artificial immune algorithm for inter-floor peak traffic during lunch-time. Control Theory and Applications 24(2), 177-182.

Mulvaney, D., White, J., Hamdi, M., 2010. Elevator dispatching using heuristic search. Intelligent Automation and Soft Computing, 16(1), 77-87.

Muñuzuri, J., Cortés, P., Onieva, L., Guadix, J., 2010. Modelling peak-hour urban freight movements with limited data availability. Computers & Industrial Engineering, 59 (1), 34-44

Perez-Martinez, K.Y., Maury-Otero, S.R., López-Pereira, J.M., 2011. Viral system aplicado al problema de ruteo de vehículos con flota heterogénea y ventanas de tiempo (FSMVRPTW), in: XLIII Simposio Brasileiro de Pesquisa Operacional, 15-18 august 2011.

Rashid, M.M., Rashid, N.A., Farouq, A., Ataur Rahman, Md., 2011. Design and implementation of fuzzy based controller for modern elevator group. IEEE Symposium on Industrial Electronics and Applications, ISIEA 2011, pp. 63-68.

Suryadi, D., Kartika, E.K., 2011. Viral systems application for Knapsack problem, in: Proceedings of the 3rd International Conference on Computational Intelligence, Communication Systems and Networks, CICSyN, pp. 11-16.

The Chartered Institution of Building Services Engineers, 2005. Transportation systems in buildings Cibse Guide D. Cibse Pub: London.

Tyni, T., Ylinen, J., 2006. Evolutionary bi-objective optimisation in the elevator car routing problem. European Journal of Operational Research 169, 960-977.

Utgoff, P.E., Connell, M.E., 2012. Real-time combinatorial optimization for elevator group dispatching. IEEE Transactions on Systems, Man, and Cybernetics Part A: Systems and Humans 42(1), 130-146.

Xie, F., Jia, R., 2012. Nonlinear fixed charge transportation problem by minimum cost flow-based genetic algorithm. Computers & Industrial Engineering, 63(4), 763-778.

Yan, S., Wang, S-S., Wu, M-W., 2012. A model with a solution algorithm for the cash transportation vehicle routing and scheduling problem. Computers & Industrial Engineering, 63(2), 464-473.

Yin, M., Kim, K.H., 2012. Quantity discount pricing for container transportation services by shipping lines. Computers & Industrial Engineering, 63(1), 313-322.

Zhang, Y., Fan, Z-P., Liu, Y., 2010. A method based on stochastic dominance degrees for stochastic multiple criteria decision making. Computers & Industrial Engineering, 58(4), 544-552.