Genetic and tabu search approaches for optimizing the hall call – car allocation problem in elevator group systems

BERNA BOLAT
Yildiz Technical University, Faculty of Mechanical Engineering,
Mechanical Engineering Department, Yildiz, TR-34349, Istanbul, Turkey.
Email: balpan@yildiz.edu.tr

PABLO CORTÉS
Escuela Técnica Superior Ingenieros, Ingeniería Organización, Seville University,
Camino de los Descubrimientos s/n, Sevilla 41092, Spain.
Email: pca@esi.us.es

Abstract

The most common problem in vertical transportation using elevator group appears when a passenger wants to travel from a floor to other different floor in a building. The passenger makes a hall call by pressing a landing call button installed at the floor and located near the cars of the elevator group. After that, the elevator controller receives the call and identifies which one of the elevators in the group is most suitable to serve the person having issued the call. In this paper, we have developed different elevator group controllers based on genetic and tabu search algorithms. Even though genetic algorithm has been previously considered in vertical transportation problems, the use of tabu search approaches is a novelty in vertical transportation and has not been considered previously. Tests have been carried out for high-rise buildings considering diverse sizes in the group of cars. Results indicate that the waiting time and journey time of passengers were significantly improved when dealing with such soft computing approaches. Also, a quickly evaluable solution quality function in the algorithms allows suitable computational times for industry implementation.

Keywords: Elevator; Lift; Tabu search; genetic algorithm; elevator group system; vertical transportation; car dispatching

1. INTRODUCTION

Building traffic design must be undertaken taking into account that passengers expect efficient service within acceptable time period from vertical transportation systems such as elevators and escalators. It is the aim of an optimal elevator control system to provide a good quality service to its passengers. The situation arises when a passenger wants to travel from a floor to other different floor in a building. The passenger makes a hall call of an elevator by pressing a landing call button installed at the floor and located near the cars of the elevator group. After that, the elevator controller receives the call and identifies which one of the elevators in the group is most suitable to serve the person having issued the call. This phenomenon is also called "dispatching". It is the task of the dispatcher to monitor the hall and car calls and to control the movements of the elevators to ensure that the passengers are collected promptly and transported rapidly to their destinations. So, the problem to be solved is to select for each hall call an elevator that will minimize a preselected cost function. The most common optimization criterion for quality of service in elevator group control is to reduce the average journey time which consists of waiting time and travel time of passengers.

Even though many years research on vertical transportation has showed a narrow interest, recently it is attracting an increasing interest from the research community. Now, there are a number of applicable optimization techniques to the car dispatching problem, which aim to reduce the average journey time of passengers. Examples of it are optimal control, in this field one of the most relevant contributions was Closs (1970); knowledge based systems e.g. Prowse *et al.* (1992); fuzzy logic e.g. Ho and Robertson (1994); dynamic programming (So and Chan, 1996); expert systems (Qu *et al.*, 2001), neural networks (Imrak and Barney, 2001), and genetic algorithm (Miravete, 1999; Siikonen *et al.* 2001; Cortés *et al.* 2004; and

Bolat *et al.*, 2010) with an increasing interest and showing an appreciated suitability to the problem. Even papers considering simulation studies in vertical transportation have arised over the last years (Cortés *et al*., 2006). However, no research has been carried out using tabu search algorithms in the field of elevator control systems.

In this paper, we deal with research on genetic and tabu search algorithms. Both approaches search for maximizing the hall call allocation efficiency and for reducing the average journey, travel and waiting time. We define a hall call allocation strategy to define the solution encoding and a new solution quality estimation function. Results are provided for high-rise buildings from 10 to 24-floors, and several car configurations are considered from 2 to 6 cars.

The rest of the paper deals with the description of the main performance indicators for elevator group systems in section 2. Section 3 includes the presentation of the soft computing approaches: genetic and tabu search algorithms, as well as the solution encoding specification and the detailed description of the quality function that is used to assess each solution. Section 4 presents the obtained results and simulations for a variety of real cases showing the suitability of the proposed methods. Finally, section 5 draws the main considerations and conclusions of the paper.

2. ELEVATOR GROUP CONTROL SYSTEM GENERAL PRINCIPLES

A single elevator car is not able to cope with the passenger traffic in high-rise buildings. Thus, a number of cars should be installed so that the handling capacity can be increased by a common group control that delivers the hall calls to the elevators. Elevator group control systems respond to the necessity of providing efficient control for an elevator group that provides service to a set of hall (or landing) calls (Barney, 1987).

In a real building, passengers arrive randomly to several floors, even at the same time, wanting to be transported from a floor to other different floor. When a passenger arrives at a landing floor and makes a hall call, the elevator group control system must take a decision allocating a car to the hall call in a short time. The performance of an elevator control system can be assessed by means of different measurement parameters. Mainly it is assessed analysing the passenger Average Waiting Time (AWT), the passenger Average Travel Time (ATT) and the passenger Average Journey Time (AJT) that is calculated as the sum of the other two. In fact, the average journey time is one of the most significant factors to define the performance of the group. It is defined as the time between the instant a passenger registers a hall call at the main terminal floor, and the leaving instant at the destination floor (Barney, 1987). The relation among the performance criterions are illustrated in figure 1.

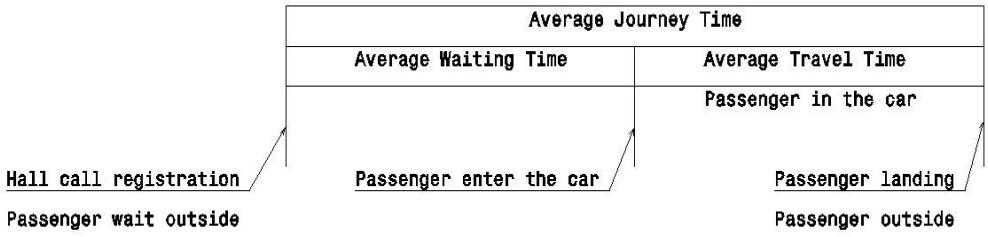

Figure 1. Definitions for Average Journey Time calculation

The optimization of the passenger journey time produces a reduction in the number of elevator stops, and at the same time the handling capacity is augmented (Cortés *et al.*, 2004), what are desirable consequences.

Average journey time consists of the total average travel time and average waiting time (1).

$$AJT = ATT + AWT \tag{1}$$

where the average waiting time 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 Barney (1987), for car loads less than 50%, it is possible to develop an approximate equation for AWT as (2),

$$AWT = 0.4INT, \qquad \text{for car loads} < 50\ \% \tag{2}$$

Also, for car loads more than 50%, it is possible to develop an approximate equation for AWT as (3),

$$AWT = \left[0.4 + \left(\frac{1.8P}{CC} - 0.77^2 \right) INT \right], \qquad \text{for car loads} > 50\ \% \tag{3}$$

being *INT* the interval (the main floor arrival average time), *P* the number of passengers, and *CC* the rated car capacity. The average waiting time is used to attain an efficient system performance and also dominant in elevator service quality.

The passenger average travel time is the time the responding elevator doors begin to open to the time the doors begin to open again at the passenger's destination. Following Siikonen and Korhonen (1993), we can calculate (4),

$$ATT = t_v \frac{H}{2S}(S+1) + t_s \frac{S+1}{2} + t_p \cdot P. \tag{4}$$

being, *H* 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.

3. SOFT COMPUTING APPROACHES

Elevator group control systems include controllers to determine which car should serve a hall call. This is also known as the car dispatching problem. Industry usually implements conventional algorithms that are based on the collective principle. That is, if the car is going down and the car finds a down hall call, the car will stop to collect passengers in that hall. The same reasoning happens for up hall calls and cars going up. Such conventional algorithms are mainly based on the rules expressed in the THV algorithm, (Barney, 1987), that was implemented in the computer-aided design suite LSD (Lift Simulation and Design) at UMIST (University of Manchester Institute of Science and Technology). The THV algorithm assigns the hall call to the nearest lift in the adequate trip direction. Note that this method is usually underperformance because the effect of a possible car displacing in the direction of a collection of successive calls in the same direction would lead to serve all the calls by the same car impacting in a negative way in the waiting times.

Here we consider the application of soft computing approaches such as genetic algorithms (what has been previously applied to vertical transportation successfully, and tabu search algorithm what has not been previously tested in vertical transportation problems). Both algorithms make use of common ideas which we are explaining in next two sections.

3.1 Solution encoding

It is assumed that a potential solution to a problem may be represented as a set of parameters. In this study, we follow an encoding strategy that was initially suggested in Cortés *et al.* (2004). 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 $2(N-1)$, where N is the total number of floors in

the building. Figure 2 depicts a possible solution encoding for an array representing a specific car in the group. A solution encoding consists of an array as shown in figure 2 for each car of the group.

Upwards Landing Calls																						
F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	F15	F16	F17	F18	F19	F20	F21	F22	F23
0	1	0	1	1	1	0	1	0	0	0	0	1	0	1	1	1	0	0	0	0	1	1

Downwards Landing Calls																						
F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	F15	F16	F17	F18	F19	F20	F21	F22	F23	F24
1	1	1	1	0	0	0	0	0	0	1	0	1	0	0	0	1	0	1	1	0	1	1

Figure 2. Solution encoding for a car of the group

In this encoding, "1" means that the hall call of that specific floor is assigned to the car, and "0" means that no hall call is assigned in that floor to the car. Binary encoding provides flexibility, simplicity and has proven usefulness in a variety of problems.

3.2. Solution quality assessment function

Every feasible solution encoded as previously stated is assessed to determine the quality (or fitness) of such solution.

Here we present a novel fitness estimation procedure that requires the definition of the following parameters:

- Ψ_1: ground floor level
- Ψ_2: highest down hall call level
- Ψ_3: number of down hall calls between Ψ_1 and Ψ_2.
- Ψ_4: highest up hall call level
- Ψ_5: number of up hall calls between Ψ_1 and Ψ_4.
- Ψ_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 solution 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 by considering four different cases that are shown in equations (5-8). Note that each formula relates in which floor the passenger is taken and in which floor the passenger is transferred. So, as a general definition for Ψ, it shows the floors where passengers are get on and off.

Case I: there is not any hall call in the system

$$f_i = 0 \tag{5}$$

Case II: there are only down hall calls

$$f_i = \left[t\left(\Psi_2 - \Psi_1\right) + t_p\left(\Psi_3 - \Psi_1\right) \right] \tag{6}$$

Note that case II takes place during downpeak traffic.

Case III: there are only up hall calls

$$f_i = \left[t\left(\Psi_4 - \Psi_1\right) + t_p\left(\Psi_5 - \Psi_1\right) \right] \tag{7}$$

Note that case III takes place during uppeak traffic.

Case IV: there are up and down hall calls

$$f_i = \left[t\left(\Psi_4 - \Psi_1\right) + t\left|\left(\Psi_2 - \Psi_4\right)\right| + t\left|\left(\Psi_2 - \Psi_6\right)\right| + t_p\left|\left(\left(\Psi_3 + \Psi_5\right) - \Psi_1\right)\right| \right] \tag{8}$$

Finally, case IV takes place during lunchpeak traffic or interfloor.

Then the average value is calculated as (9):

$$f_{group} = \frac{\sum_{i=1}^{n} f_i}{n}, \text{being } n \text{ the number of cars in the group} \tag{9}$$

The final fitness value is computed as (10):

$$f = k_1 \cdot f_{group} + k_2 \cdot \left(Hct - Lct\right) \tag{10}$$

So, the total fitness for the system is calculated as a two-part function where the first part collects the concern related to passengers waiting in the halls, and second part the concern related to an unbalanced performance in the cars of the group. That is, the second part tries to level the use of each car. Let consider the following example: car no. 1 is working during 100 seconds and car no. 2 during 11 seconds, so both cars would be working during 111 seconds but in a very unlevelled manner. On the other hand if car no. 1 works during 57 seconds and car no. 2 during 54 seconds the two cars work during 111 seconds but in a much more levelled manner. This action will have repercussion on the life cycle of the cars. Additionally a levelled use of cars does impact in a positive way in the waiting times of passengers too. Parameters k_1 and k_2 are weights for each part and after testing several tests we found that better results were obtained for values equal to $k_1 = 1.5$ and $k_2 = 2$.

The quality of such solution is calculated for every feasible encoded solution following this previous procedure. This will condition the searching trajectories in the feasibility region according to the TS algorithm, or the possibilities of a successful offspring in GA.

The advantages of using such fitness function are associated to the capability for distributing hall calls according to the travel direction of the car, the suitability for dynamic systems, the effectiveness in the search space, and its fast computation reducing computation times.

Let consider an example with a 2 cars group in a ten floors building. The up hall calls and down hall calls are assigned as in figure 3.

	Up hall calls									Down hall calls							
Car	F1	F2	F3	F4	F5	F6	F7	F8	F9	F2	F3	F4	F5	F6	F7	F8	F9	F10
No.1	0	0	0	0	0	0	0	0	0	1	0	0	1	1	0	0	0	0

	Up hall calls									Down hall calls							
Car	F1	F2	F3	F4	F5	F6	F7	F8	F9	F2	F3	F4	F5	F6	F7	F8	F9	F10
No. 2	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0

Figure 3. Solution encoding for the group

Supossing values for t = 5.5, t_p = 3, and computing 57 seconds running Car No. 1 and 54 seconds running Car No. 2:

- The fitness for Car No. 1 is calculated using equation (6) that corresponds to the case without up hall calls. Setting the values,
    - Ψ_1= 1 (ground floor)
    - Ψ_2 = 6 (the highest of all downwards landing calls)
    - Ψ_3= 3 (because corresponds to the number of down hall calls between Ψ_1 and Ψ_2.; that is the down calls in second, fifth and sixth floors)

    The fitness is then calculated as:

$$f_1 = \left[t\left(\Psi_2 - \Psi_1\right) + t_p\left(\Psi_3 - \Psi_1\right)\right] = \left[5.5\left(6-1\right) + 3\left(3-1\right)\right] = 33.5$$

- The fitness for Car No. 2 is calculated using equation (7) that corresponds to the case without down hall calls. Setting the values,
    - Ψ_1= 1 (ground floor)
    - Ψ_4 = 7 (the highest of all upwards landing calls)
    - Ψ_5= 2 (because corresponds to the number of up hall calls between Ψ_1 and Ψ_4; that is the up calls in third and seventh floors)

    The fitness is then calculated as:

$$f_2 = \left[t\left(\Psi_4 - \Psi_1\right) + t_p\left(\Psi_5 - \Psi_1\right)\right] = \left[5.5\left(7-1\right) + 3\left(2-1\right)\right] = 36$$

So, the final fitness is given by:

$$f = k_1 \cdot f_{group} + k_2 \cdot \left(Hct - Lct\right) = 1.5 \cdot \frac{\left(33.5 + 36\right)}{2} + 2 \cdot \left(57 - 54\right) = 58.125$$

3.3. Genetic algorithm

Genetic algorithms are inspired in the natural selection principle, whose main idea is that new powerful offspring forms are expected from old generations (Goldberg, 1989). The individuals of the population represent feasible solutions and are described by means of its

chromosome corresponding to the solution encoding presented in section 3.1. So, the genotypes are binary encoded. Bit 0 indicates no hall call allocation for the car, and bit 1 indicates that the registered hall call of that floor is allocated to the car.

We implemented a genetic algorithm whose main ideas were introduced in Bolat *et al.* (2010). However, here we are introducing a new fitness concept which is described in previous section 3.2. So, an initial randomly generated population is considered according to a hall call allocation based on a 5-minute period. The size of population which is one of the most critical parameters is chosen between 20 and 30. In fact, in large populations, the effectiveness of searching increases, because of a good exemplification of solution areas, but related to this, searching takes time. On the contrary, in small populations, the solution area is not exemplified sufficiently and untimely might occur.

Genetic operators such as selection, crossover and mutation are performed on the population chromosomes to produce offspring. The selection operator simulates the survival competition mechanics in the natural world and chooses genotypes to propagate their genetic information according to the fitness of the individuals in the populations. There are a number of selection methods, although our tests lead us to choose the roulette wheel selection method where the slot size for each individual is proportional to its fitness.

The crossover genetic operator is a structured information exchange process which chooses a random crossover point in a pair of parent genotypes and then swaps parts of bit strings between the two parents to produce two offspring. There have been implemented different types of crossover techniques which are single-point, two- point and uniform crossover techniques all of whom are applied in our study. The choice of the crossover operator has a significant impact in the yield of the GA.

Mutation is another important genetic operator that randomly changes a gene of an offspring. When using a binary representation, a mutation corresponds to change a 0 to 1 and vice versa. This operator allows the introduction of new chromosomes material to the population, assuring that given any population the entire search space is connected (Buckles and Petry, 1992).

The number of generations of the GA can be a critical parameter. One advantage of GA is that can be stopped at any time having the better solution at the moment (Bolat *et al.*, 2010). We tested with the number of generations between 30 and 100 iterations and best solutions were found for values between 30 and 50 without increasing too much time consumption.

Figure 4 defines the performance of the genetic algorithm.

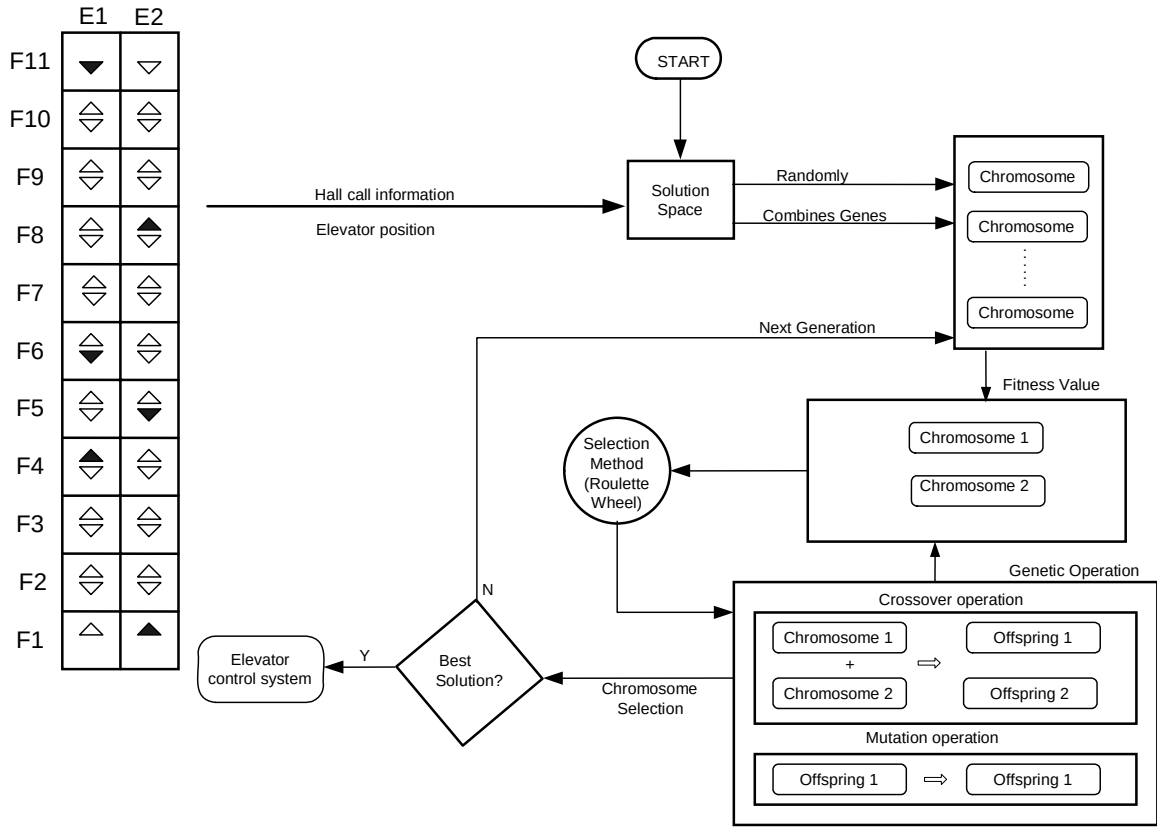

Figure 4.Genetic algorithm description

3.4. Tabu search algorithm

Tabu Search (TS) is to derive and exploit a collection of principles of intelligent problem solving. TS which is founded on ideas proposed by Glover (1989 and 1990) is based on selected concepts that join the fields of artificial intelligence and optimization. TS is a powerful algorithmic approach that has been applied with great success to many difficult combinatorial problems. The basic principle of TS is to pursue the search whenever a local optimum is encountered by allowing non-improving moves, cycling back to previously visited solutions is prevented by the use of memories, called tabu lists, that record the recent history of the search. In this problem we propose the use of TS to optimize the hall call allocation problem. To do so, we define the candidate solution encoding, the procedure to assess the quality of such solution in terms of waiting times, and the guidelines and pseudocode of the proposed tabu search algorithms.

We have developed conventional and probabilistic tabu search algorithms (TS and PTS). The principle of both algorithms is the same only differentiating in the neighbourhood mechanism. Figure 4 depicts the general structure for the tabu search algorithms. The detailed description of the main aspects for both algorithms follows next:

a) Neighbourhood structure: a neighbour solution is defined as any other solution that is obtained by a pair wise interchange of any two nodes in the solution. In this study, we used two different swap mechanisms for each algorithm (TS and PTS). The swap mechanism in TS is carried out after mapping the entire neighbourhood and selecting that neighbour with a minimum fitness value (best neighbour). On the contrary, the swap mechanism in PTS works randomly. Each iteration provides a new solution that is found by making a

definitive local movement over the current solution within the neighbourhood. Then a solution is selected depending on the type of algorithm (conventional or probabilistic), and the tabu list and aspiration criterion that are taken into consideration in both algorithms and are explained next.

b) Tabu list, recency and frequency: tabu movements are stored in a list that is called tabu list. The basic role of the tabu list is to prevent cycling by including some attribute of recently visited solutions in a tabu list. Only solutions that are not in tabu list would be feasible solutions for a definitive movement (excepting those solutions satisfying the aspiration criterion that is detailed next).

An important factor of tabu search algorithms attends to the length of the tabu list and the replacement of solutions in such list. Generally, the strategy is applying a first in/first out rule. After reaching the last free register of the list, the new solution enters the list and replaces the oldest solution. If the tabu list is too long, possible movements will be restricted, because most of the actions will be tabu. On the contrary in cases of too short tabu lists, designs may be around the same cycle, near the beginning of the design, so the searching procedure may converge to a local optimum. In this study, the length of tabu list was chosen as an integer constant, which is 10 times the number of cars in the group because it provided the better results attending to the fitness function.

To state if a solution attribute is tabu or not, we make use of the recency and frequency tabu. Every time a solution is selected from the neighbourhood, the tabu list is modified. The solution is recorded and its frequency is accordingly increased. The tabu frequency of a solution is a parameter stating the number of times that can appear previously to be stated as tabu. If some solution surpasses the threshold such solution is labelled as tabu and it is

included in the tabu list. The tabu recency of a solution is a parameter stating the number of iterations that the solution is maintained in the tabu list. When recency is equal to zero the solution is erased from the tabu status.

These two factors are carefully selected for putting in a fast and effective algorithm. Recency and frequency memory were selected [0, 1] in this study.

c) Aspiration Criterion: an aspiration criterion is a factor allowing movements to tabu solutions in certain conditions. The better visited solutions during the searching process area are stored in the elite list. The aspiration criterion determines when the tabu status of a solution can be ignored and the algorithm allows the move to that solution.

The idea is to provide flexibility in the searching procedure. So, movements to tabu solutions are ignored temporally, and the movement is allowed even if it has been made very recently or frequently.

d) Short Term Memory (STM): STM carries out the generation of a neighbourhood shaped by a set of similar solutions and its quality analysis, as well as the identification of the new generated solution shaping a specific trajectory. STM also checks the validity of such solution moving to it in case of not belonging to the Tabu List. In case of not validation, a next neighbour is selected (selecting the next better neighbour for the TS, or selecting another candidate in a probabilistic way for the PTS).

e) Long Term Memory (LTM): LTM checks if the new searching process is evolving good, allowing the return to a previous stage of the searching procedure. So that, a better design, can be found in the previous region, by re-evaluating again. Long-term memory is used as a local call to strengthen, and to ensure the diversification of the global process.

f) Chain of solutions: a chain of solutions is constituted by all the enchained adjacent solutions. STM is responsible for the optimal enchaining of these solutions, depicting a trajectory. The trajectory is consequence of the quality of the solutions and a tabu filter depicting an intelligent exploration, meanwhile the LTM periodically analyses whether the trajectory is acceptable or the process should not follow this path and the algorithm should move back to take another more promising path to configure a new trajectory.

STM and LTM are the most important procedures of the algorithm. Both procedures are necessarily combined to gain equilibrium between exploration of the feasibility region (STM procedure) and exploitation of a promising region (LTM procedure). In this line STM constitutes a form of aggressive exploration (exploitation) that seeks to make the best possible movement within the neighbourhood. The use of STM together with recency and frequency tabu, and the aspiration criterion allows a very efficient and quick exploration of the neighbourhood. Meanwhile, LTM allows moving back to the best stored solutions to continue searching in those promising areas once the number of visited solutions reaches certain value (this is called the length of the chain of solutions). This mechanism tries to maintain the quality of solutions.

g) Stopping criteria: the algorithms utilize a maximum number of iterations as stopping criterion. After various experiments, a value equal to ten times the number of cars was selected.

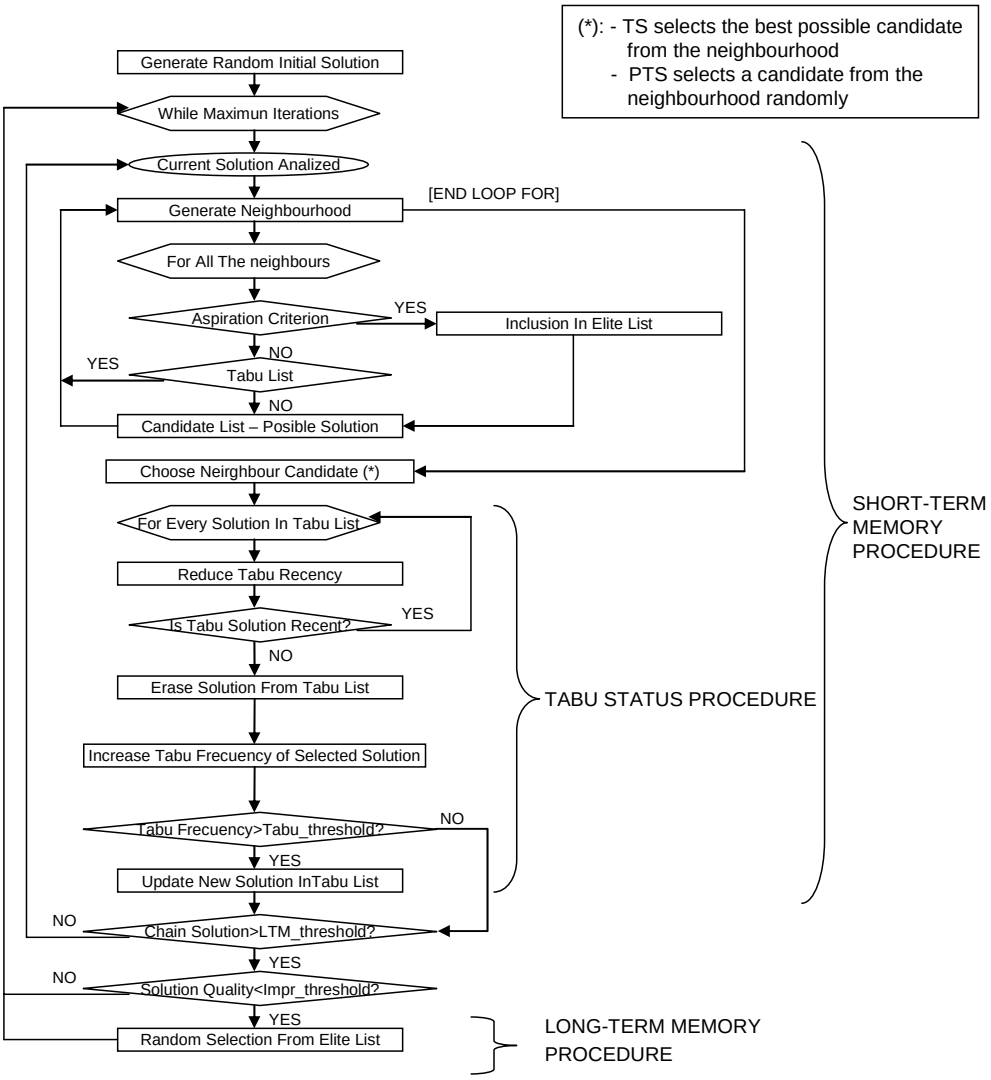

Figure 5. The flowchart of Tabu Search Algorithm

4. COMPUTER SIMULATIONS

4.1. Simulation scenario

We used several target buildings to test our algorithms. We select tall buildings from 10 floors to high rise building with 24 floors. We have also considered different possibilities for the elevator group configuration: 2,3,4,6 cars. Table 1 defines the specifications of the building.

Table 1. Vertical transportation system specifications for simulations

Items	Value
Number of floors	10; 12; 14; 16; 18; 20; 22; 24
Number of cars	2;3;4;6
Floor Distance (m)	4.5
Car capacity (persons)	8
Time spent on:	
Time for opening door (s)	2.5
Time for closing door (s)	3
Time for passenger transfer (s)	3

Lunch peak traffic has been recognized as one of the most complex traffic patterns in buildings because it includes uppeak and downpeak effects which require opposing responses from a controller (Cortés *et al.*, 2004). The lunch-peak period was simulated with a 40% up-peak, 40% down-peak flow and 20% inter-floor traffic following the CIBSE guide (2005) and generating 5-minutes arrival periods. Once these percentages were considered the arrivals were randomly generated.

4.2. Simulation Results

In this section, we provide results for different car group configurations as specified in table 3. Computational tests were carried out in a Pentium 4 CPU 3.00 GHz workstation.

The tabu search approaches (basic tabu search algorithm, TS, and probabilistic tabu search algorithm, PTS) were compared to three different genetic algorithm implementations based on different crossover operators (single point crossover, SPC, two point crossover, TPC, and uniform crossover, UC); and also results from a conventional algorithm implementation, (Cortés *et al.*, 2004), were provided.

Table 2 depicts the average journey time (AJT) for building from 10 to 24 floors, and for different cars' implementations in the elevator group (2 to 6 cars).

Table 2. AJT results in buildings from 10 to 24 floors for each algorithm, and elevator group under consideration (results are provided in seconds)

Building floors	2 cars				3 cars				4 cars				6 cars			
	CONV	GA	TS	PTS	CONV	GA	TS	PTS	CONV	GA	TS	PTS	CONV	GA	TS	PTS
10	98.6	58.5(SPC) 61.5(TPC) 58.5(UC)	58.5	58.5	79.7	41.0(SPC) 40.0(TPC) 39.0(UC)	42.0	41.0	70.3	37.5(SPC) 35.3(TPC) 37.5(UC)	41.3	31.5	60.8	32.5(SPC) 31.5(TPC) 31.0(UC)	33.0	27.5
12	103.5	60.0(SPC) 60.0(TPC) 60.0(UC)	63.0	64.5	83.6	46.0(SPC) 47.0(TPC) 47.0(UC)	46.0	45.0	73.6	42.0(SPC) 38.3(TPC) 39.8(UC)	40.5	42.0	63.6	36.0(SPC) 32.5(TPC) 39.0(UC)	37.0	31.5
14	108.6	82.5(SPC) 75.0(TPC) 76.5(UC)	82.5	76.5	87.6	54.0(SPC) 53.0(TPC) 54.0(UC)	56.0	51.0	77.0	46.5(SPC) 45.0(TPC) 45.8(UC)	46.5	46.5	66.5	40.0(SPC) 40.0(TPC) 40.0(UC)	48.5	40.5
16	112.6	72.0(SPC) 75.0(TPC) 72.0(UC)	72.0	72.0	90.7	56.0(SPC) 52.0(TPC) 61.0(UC)	60.0	52.0	79.7	48.8(SPC) 46.5(TPC) 48.0(UC)	48.8	48.8	68.8	49.5(SPC) 45.5(TPC) 46.0(UC)	49.5	47.5
18	116.6	90.0(SPC) 90.0(TPC) 84.0(UC)	87.0	87.0	93.8	66.0(SPC) 65.0(TPC) 66.0(UC)	65.0	64.0	82.4	60(SPC) 63.8(TPC) 61.5(UC)	60.8	59.3	71.0	49.0(SPC) 53.5(TPC) 61.0(UC)	67.5	50.0
20	120.8	93.0(SPC) 93.0(TPC) 99.0(UC)	108.0	99.0	97.1	89.0(SPC) 74.0(TPC) 84.0(UC)	84.0	76.0	85.2	75.8(SPC) 69.0(TPC) 66.0(UC)	74.3	59.8	73.4	66.0(SPC) 64.5(TPC) 67.5(UC)	70.0	66.5
22	124.8	105.0(SPC) 120.0(TPC) 105.0(UC)	105.0	109.5	100.2	87.0(SPC) 85.0(TPC) 81.0(UC)	87.0	89.0	87.9	78.0(SPC) 77.3(TPC) 76.5(UC)	81.0	75.8	75.6	68.5(SPC) 69.5(TPC) 66.5(UC)	71.0	64.5
24	127.9	115.5(SPC) 115.5(TPC) 115.5(UC)	115.5	111.0	102.6	95.0(SPC) 89.0(TPC) 90.0(UC)	90.0	90.0	90.0	80.3(SPC) 78.8(TPC) 73.5(UC)	87.8	74.3	77.3	64.0(SPC) 67.0(TPC) 65.5(UC)	66.5	63.0

After analysing results from table 2, we found that genetic algorithm provided 7 times the best result for the SPC implementation, 12 times for the TPC implementation and 9 times for the UC implementation. The basic TS approach provided 3 times the best result, and PTS provided 15 times the best result showing the better performance in general terms. The conventional algorithm implementation was significantly outperformed by all the tested methods.

The genetic approaches showed a similar behaviour for their three implementations corresponding to single point crossover (SPC), two point crossover (TPC), and uniform crossover (UC), although TPC implementation showed slightly better results for most of the cases. In general, GA approaches behaved better for few cars' cases, as well as for lower floors. When the number of cars increases and floors are higher, tabu approaches tend to behave better, especially in the probabilistic approach. The probabilistic tabu search was the method presenting a higher number of successes (15 times).

Next figure 5 shows the comparative results for GA in its best implementation (TPC), conventional TS algorithm, and PTS algorithm. The figure depicts how PTS provides better results, especially for complex cases with many cars in the elevator group.

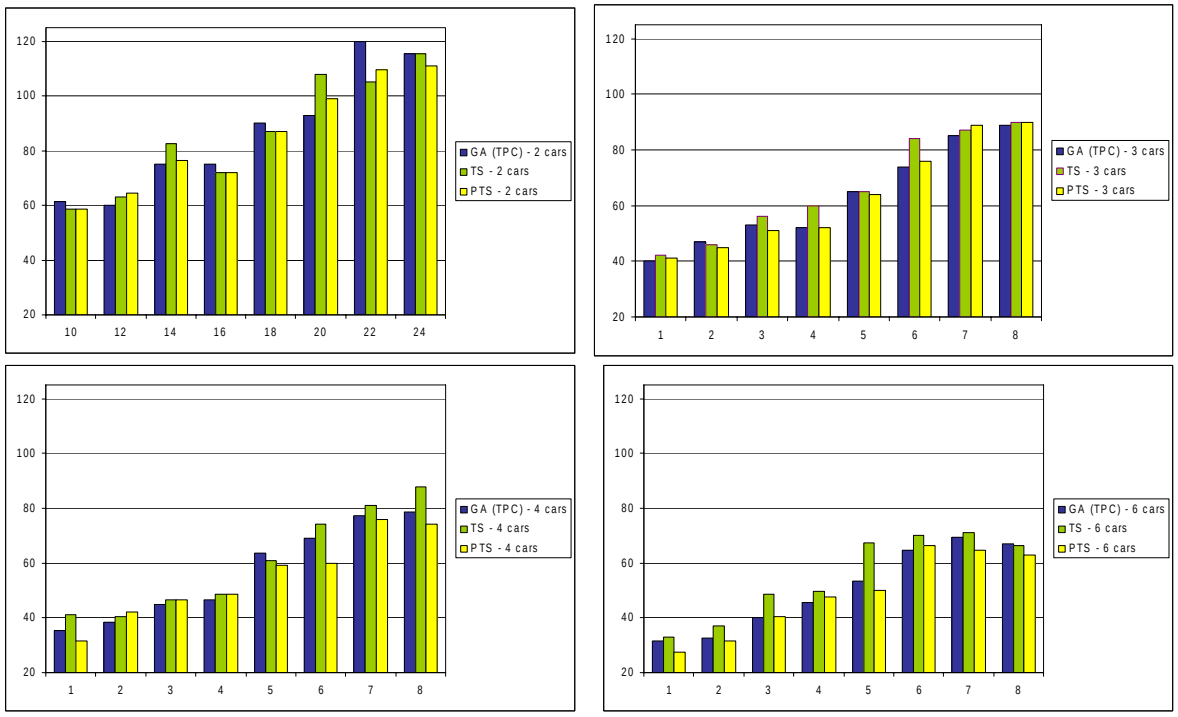

Figure 6. AJT results in buildings from 10 to 24 floors for each algorithm, and elevator group under consideration

Next figure 6 shows the evolution of AJT related to the number of cars in the elevator group and calculated for each specific floor for the best implementation regarding the quality of service (AJT) that is the PTS algorithm in our tests. The figure states how AJT increases for higher floors, and how AJT decreases when cars in the group raise, as it was expected.

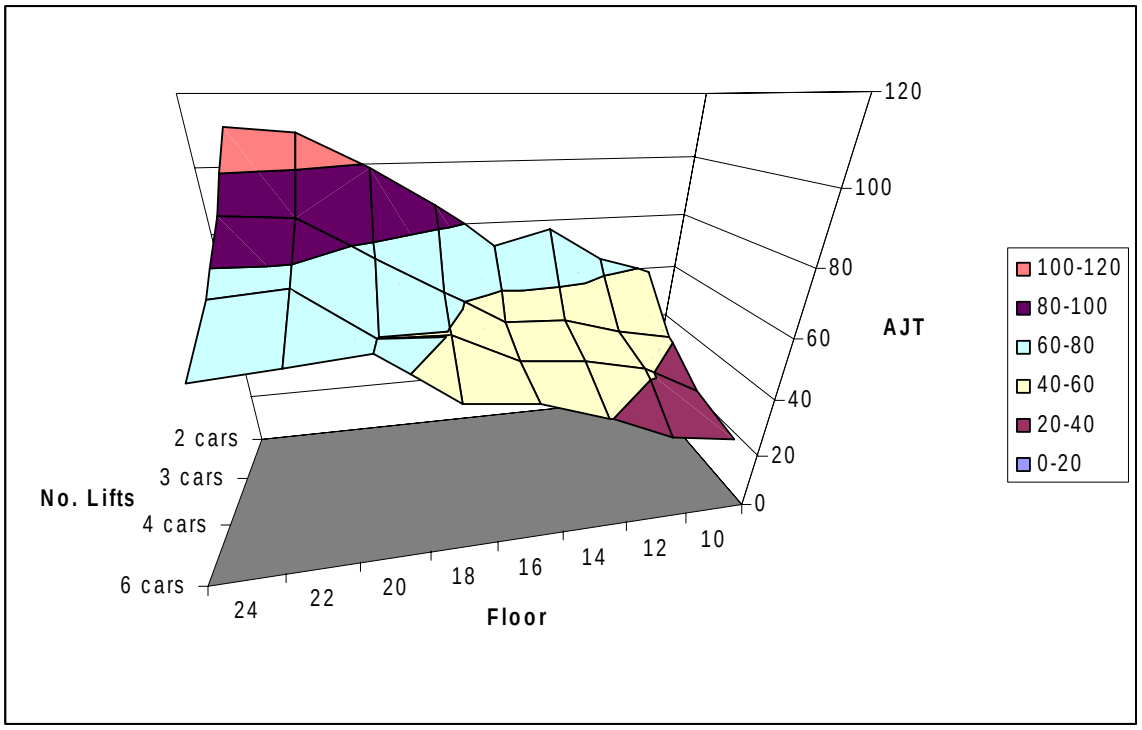

Figure 7. PTS algorithm: AJT results in buildings from 10 to 24 floors depending on the number of cars in the group

Regarding computational time, tabu search approaches showed worse behaviour than genetic algorithm implementations (Table 3). GA implementations provide implementable computational time in real controllers for most of the cases, even in largest configurations. PTS computational times are suitable for cases with lower group configurations. The 4 cars case is in the limit and the 6 cars case would not be directly implementable. Anyhow, all the soft computing approaches reveals very robust with respect to the number of floors in

the building, being the computational time key factor the number of cars in the group. Also tests show how conventional TS results are out of limits regarding computational times.

Table 3. Computational time results in buildings from 10 to 24 floors for each algorithm, and elevator group under consideration (results are provided in seconds)

Building floors	2 cars			3 cars			4 cars			6 cars		
	GA*	TS	PTS	GA*	TS	PTS	GA*	TS	PTS	GA*	TS	PTS
10	1.6 3.0 1.6	3.2	0.5	2.6 2.5 2.4	4.4	0.8	2.9 2.2 1.8	9.3	6.2	1.9 1.9 2.0	43.6	27.3
12	2.0 1.9 2.2	6.5	0.4	2.3 2.5 2.5	4.8	0.8	1.7 1.7 1.7	22.0	2.9	2.0 2.5 2.0	57.2	22.3
14	2.2 2.0 2.0	5.0	0.5	2.2 2.2 2.6	9.2	0.8	1.8 1.7 1.5	30.7	4.7	2.0 2.0 2.0	81.6	29.6
16	2.1 2.3 2.1	6.8	0.5	2.2 2.4 2.2	9.9	0.7	1.8 1.7 1.7	28.9	5.1	1.8 1.9 1.7	104.1	16.9
18	2.3 2.3 2.0	4.8	0.6	2.4 2.3 2.2	13.5	1.5	1.7 1.7 1.8	67.5	8.4	1.9 1.9 2.1	183.5	46.2
20	2.0 2.3 2.1	4.6	0.6	4.8 2.4 2.4	13.0	1.9	1.7 1.9 1.9	57.0	6.3	2.1 2.1 1.9	167.8	57.5
22	2.2 2.2 2.0	5.4	0.6	2.3 2.5 2.4	9.4	2.0	1.9 1.8 1.9	50.4	14.0	1.8 1.9 2.1	173.0	60.8
24	2.3 2.1 2.0	6.1	0.7	2.7 2.8 2.5	16.3	2.1	1.8 1.8 1.8	89.4	12.8	1.8 1.9 2.5	134.4	47.8

(*) Computational results are shown first for SPC, then for TPC and last for UC

However, it has to be taken into account that results are provided for a Pentium 4 CPU 3.00 GHz workstation. Real industry controller microchips are faster, and faster results would be probably expected from real implementations.

5. CONCLUSIONS

This paper presents two soft computing approaches (genetic algorithm and tabu search) for optimizing the average journey time in an elevator group control system. Even though

genetic algorithm has been previously considered in vertical transportation problems, the use of tabu search approaches is a novelty in vertical transportation and has not been considered previously.

Tests were carried out for high-rise buildings considering diverse sizes in the elevator group (2 to 6 cars). Results indicate that the waiting time and journey time of passengers were significantly improved when dealing with soft computing approaches. Even more, the probabilistic tabu search approach showed the best results when it was compared to the rest of implementations, although several genetic implementations also provided very good results, especially attending to the computational time.

Computational times indicate that genetic algorithms could be directly implemented in industry controllers due to its very tight consumption times. On the other hand, tabu search algorithms can consume very much computational time when a large neighbourhood and a great number of iterations are used (cases of elevator groups with 6 cars). In such cases, the parameters of the tabu search algorithm must be carried out attending not so much to the algorithm accuracy but to the available time of trip of the elevator between different events (calls). However, the tabu search algorithm we are presenting here showed a very short time for computing its novel fitness estimation. Its major computational time consumption appears when dealing with the neighbourhood in the short term memory. Bounding this fact, the real implementation of the tabu search algorithm in the industry appears to be possible. For example, in real cases an alternative can be stopping the algorithm previously to reach the next event, which would occur after a known time interval. Of course all these decisions are very dependant on the computation speed of the electronic microchips installed by the company in the controller. Due to all these facts together with the great

potential of tabu search implementations regarding to AJT, an ongoing research on the capabilities of such soft computing techniques is necessary in our opinion. So, applications of TS approaches on vertical systems should gain interest and support in a near future, and new implementations should be expected.

Acknowledgements

The Spanish author acknowledges the financial support given by the Consejería de Innovation, Ciencia y Empresa of Andalusia, through its Excellence projects programme (project ref. P07-TEP-02832).

REFERENCES

[1] Closs, G.D. (1970) The computer control of passenger traffic in large lift systems, Ph.D. Thesis, *UMIST,* Manchester, UK.

[2] Prowse, R.W., Thomson, T., Howells, D. (1992) Design and control of lift systems using expert systems and traffic sensing, Elevator Technology 4, *IAEE Publ*., London.

[3] Ho, M., Robertson, B. (1994) Elevator group supervisory control using fuzzy logic, *Canadian Conference on Elevator and Computer Engineering*, 2, 11.4.4.

[4] A.T.P. So, W.L .Chan (1996) Dynamic zoning for intelligent supervisory control, *Int. J. of Elevator Engineering*, 1, pp. 47-59.

[5] Z. Qun, S. Ding, C. Yu, L. Xiaofeng (2001) Elevator group control system modeling based on object oriented petri net, *Elevator World* 49(8), pp. 99-105.

[6] C. E. Imrak, G.C. Barney (2001) The application of neural networks to lift traffic control, *Elevator World* 49(5), pp.82.

[7]A. Miravete (1999) Genetics and intense vertical traffic, *Elevator World*, July 11.4.4, 47(7), pp.118-120.

[8] M.L. Siikonen, T. Sus, H. Hakonen (2001) Passenger traffic flow simulation in tall buildings, *Elevator Word*, August 11.4.4, pp117-123.

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

[10] P. Cortés, J. Muñuzuri, L. Onieva (2006) Design and analysis of a tool for planning and simulating dynamic vertical transport, *Simulation: Transactions of the Society for Modeling and Simulation International* 82 (4), pp. 255-274.

[11] D.E. Goldberg (1989) *Genetic algorithms in search, optimization and machine learning*. NewYork: Addison-Wesley.

[12] B. Bolat, P. Cortés, E. Yalçin, M. Alişverişçi (2010) Optimal car dispatching for elevator groups using genetic algorithms, International Journal of Intelligent Automation and Soft Computing (AutoSoft) 16 (1), pp. 89-99.

[13] Barney G.C. (1987) Elevator abstracts including escalators, *Ellis Horwood Lim.*, Chichester.

[14] Siikonen, M-L., Korhonen, T. (1993) Defining the traffic mode of an elevator, based on traffic statistical data and traffic type definitions, *Kone Elevator*, U.S. Patent No.5 229 559.

[15] Buckles, B.P. Petry, F.E (1992) Genetic algorithms technology series, *IEEE Computer Society Press*, Los Alamitos,CA.

[16] Glover, F. (1989) Tabu Search, Part I, ORSA Journal on Computing 1 (3), 190-206.

[17] Glover, F. (1990) Tabu Search, Part II, ORSA Journal on Computing 2 (1), 4-32.

[18] Cibse Guide D,"*Transportation systems in buildings*", Cibse Pub. London, 2005.