A Multi-Objective GA to Demand-side Management in an Automated Warehouse

J. J. Cárdenas, A. García, J. L. Romeral, J. C. Urresty
MCIA Group, Universitat Politècnica de Catalunya
C/ Colon 1 TR 2-225, 08222, Terrassa, Catalonya, Spain
{juan.jose.cardenas; antoni.garcia; luis.romeral; julio.urresty}@mcia.upc.edu

Abstract

The simultaneous operation of the automated storage and retrieval machines (ASRs) in an automated warehouse can increase the likelihood that high power demand peaks turn unstable the electric system. Furthermore, high power peaks mean the need for more electrical power contracted, which in turns leads to more fixed operation cost and inefficient use of the electrical installations. In this context, we present a multi-objective genetic algorithm approach (MOGA) to implement demand-side management (DSM) in an automated warehouse. It works minimizing the total energy demand, but without increasing substantially the time for the operation. Simulations show the performances of the new approach.

1. Introduction

Warehouse is a facility, which provides the services about material storage and management to a manufacturing firm or customer. Its efficiency is depended about many factors and it is important because costs incurred are reflected in the production or distribution accounts, and are ultimately on to the consumer [1]. Generally speaking, the efficiency of warehouse operations is influenced by many factors such as warehouse layout, storage policy, order picking policy, etc. [2].

On other hand, it is observed that the simultaneous operation of the automated storage and retrieval machines (ASRs) in an automated warehouse can increase the likelihood that high power demand peaks turn unstable the electric system. Apart from instability, these peaks cause the deterioration of the power quality, increasing the dips and swells events, even operation stops caused for power cuts. In general, the factories oversize their electrical installations such that the probability of this happens are very low. The over sizing is to use cable with size greater than necessary, has greater electrical cabinets and boxes, as well as to has generators and transformers of greater power capacity. Of course, this increases significantly the cost of building an automated warehouse and the problems about power quality remain. Furthermore, it is usual that the utilities charge in the electrical bill the contracted power, which is the power that the utilities foresee available for a specify costumer. If the costumer over pass this power it has to pay penalties, which increase the electrical bill.

Hence, making efficient use of electricity is as important as the optimization of any other resource or raw material. In addition, energy optimization issues are raising more and more due to the current worldwide energetic problem. Therefore, it is essential to look for methods or algorithms in order to improve the energy use efficiency, even when we have to decrease a little the performance of other parameters.

Nowadays, at scientific literature we can find dissertations about subjects related with automated warehouse, focusing in problems like optimization of the best route of the ASRs, minimizing the storage and retrieval times, and making policies to establishing what it is the best way to allocation of the goods in the warehouse. All these subjects are important in order to reduce the operating time and improve the efficiency of the system [3-6]. However, the simultaneous problem of power peaks in factory installations has not been taken into account directly in the scientific literature.

A simply solution in order to improve the energy used could be sorting the start of the ASRs so that the simultaneous start is avoided. It could be useful since the peak power of rotating machines generally happen at the beginning of the movement. However, this solution has several buts: the peaks not always appear at the beginning, and there are peaks at the end or in the middle of the movement. Also, the length of the peaks is variable and it depends on random variables as the weight of the load, the distance to the goal place where the load will be stored and the start point, the velocity of the ASRs in x-axis as y-axis, etc. This shows clearly that the problem is not trivial and it is not deterministic for its random nature and it requires of a dynamic algorithm. So stochastic methods are naturally selected for modelling and optimization solution of such kind of problem [7, 8]. This is the main reason because the use of Genetic

Algorithms (GA) as first approach to search for the optimal timing distribution of ASRs.

GA can run to optimize more than one goal. For instance, having the energy management as an scheduling problem, as already introduced, other objectives can be taken into consideration, i.e, GA tries to reduce the total energy necessary, but without increasing substantially the time for the total movements. This kind of GA is known as Multi-objective GA [9].

First in the paper, we present the modeling of ASRs that is indispensable for the application. This is presented in the Section II. In the Section III we explain how deal with the problem using a MOGA and introduce the main concepts about this evolutionary algorithm. Section IV presents the got simulation results and finally Section V draws some conclusions and further work.

2. Load Profile Modeling of ASRs

In a warehouse, the automated storage and retrieval machines (ASRs) pick up and unload the pallets loaded of goods, from a single starting point to some places on the shelves. Each storage place has associated x and y coordinates on the shelves. Normally the path followed for the ASRs has triangle shame, as it is showed in the Fig. 1. We have called cycle to this kind of movement.

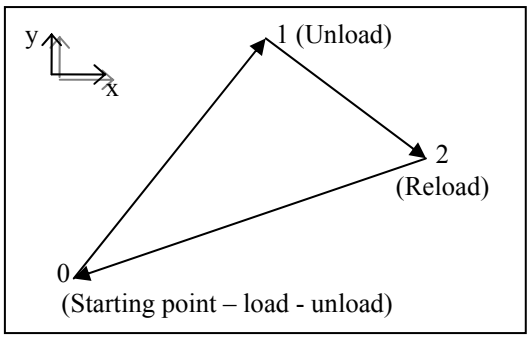

Fig. 1. The typical path followed by the ASRs (a cycle)

These movements are continually made (hence the name "cycle") until the planning of storage and retrieval is finished. Considering this, we have made measurements of different movements to random coordinates on the shelves with and without load on the ASRs. We have extracted from the real data a model that is easy to use and it requires a few resources to its management. Therefore, for a particular point (x_1, y_1), we have obtained a model for each movement decomposing it in x and y sub movements. We also have assumed that the starting point has the coordinates (0, 0). For this first approach, we have made the analysis and management for only the movement with load that goes from point *0* to the point *1* (Fig. 1) and return to *0* point without load for two (2) ASRs. The aim of this is to simplify the development but demonstrating the feasibility and suitability of using such algorithms in this type of problem. Moreover, to expand the GA to the full cycle, and later do it to higher number of ASRs and cycles will be an easy task. It is due that we have used generic vector indexation for all variables, as explained in the next section.

It should be noted that the load profile data could also be extracted from a prognosis module of load profiles. Therefore, we have not directly used the real data because we suppose that in a full application we will have a prognosis module of load profiles. This hypothetical prognosis module could has as main inputs the allocation coordinates [X, Y] from optimization allocation module. It would be expected that such load profile prognosis is not so accuracy and detailed for a load profile with minutes' resolution. Despite this, the most important is that the general algorithm for DSM does not ask for an accuracy prognosis of a load profile. However, it is required that the prognosis shows the form of the load profile and especially indicates with any degree of accuracy and reliability of the occurrence of power peak demand.

The measurements of currents were made with and without load on the ASRs. The selected load profiles (Fig. 2, Fig. 3 and Fig. 4) show clearly the main characteristics of each one of the movements. It was detected a strong dependency among them and the input variables X, Y (Cartesian coordinates of the positions in the warehouse) and the load weight. However, the form of each profile in general is mainly dependent of the kind of movement to execute (translation on the x-axis in forward and reverse movement, elevation or descent on the y-axis). As expected, the load weight and X, Y position influence mainly the maximum peaks and the duration of the profiles.

Fig. 2 is a clear example of the simplicity of the used models. We can see in red, the real Load Profile (LP) of some tested ASRs and in black, the proposed model. This LP is for a particular movement, in this case the translation on x axis (as it was explained former).

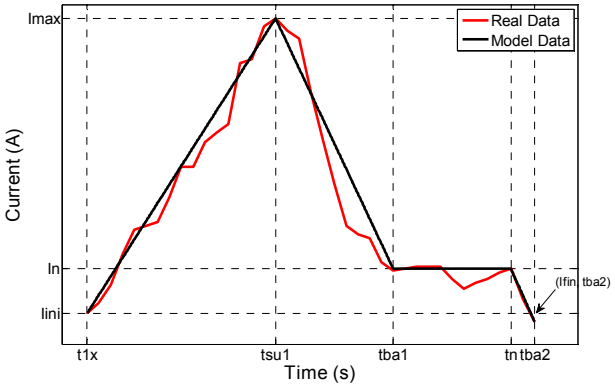

Fig. 2. The load profile of an ASR without load in forward movement on the x-axis

Only five points characterize the model proposed for this movement, and only three of them are

representatives. The others are the initial and the final points. These points are:

1. *Init point: (I_{ini}, t_{1x});*
2. *Maximun peak: (I_{max}, t_{su1})*
3. *Nominal current: (I_n, t_{ba1});*
4. *Down point: (I_n, t_n);*
5. *Final point: (I_{fin}, t_{ba2})*

For the others movements (up and down) the modeling are similar, the only different is the amount of parameters because as we said before the LP is depend about the kind of movement.

In the Fig. 3, again, we have in red the real data and in black the proposed model for the up movement. In this case, the parameters of the model are:

1. *Init point: (I_{ini}, t_{1y});*
2. *Nominal current: (I_n, t_{su1});*
3. *Down point: (I_n, t_n);*
4. *Final point: (I_{fin}, t_{ba1})*

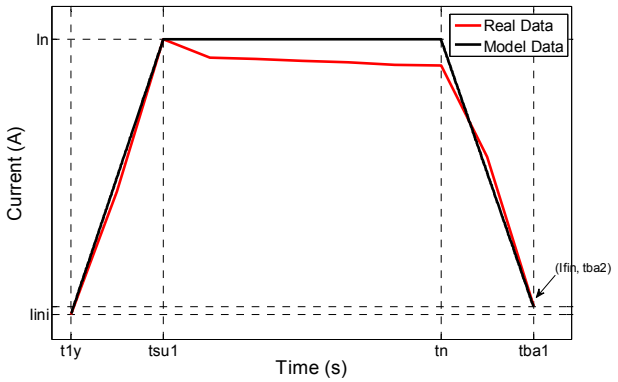

Fig. 3. The load profile of an ASR without load in up movement.

Finally, in the Fig. 4 we have the model and real data for the down movement. The main parameters are:

1. *Init point: (I_{ini}, t_{2y});*
2. *Current first peak: (I_{peak1}, t_{su1})*
3. *Nominal current: (I_n, t_{ba1}) to (I_n, t_n);*
4. *Current second peak: (I_{peak2}, t_{su2});*
5. *Final point: (I_{fin}, t_{ba2})*

Each of these movements is related to its corresponding motor on the ASR. It has a motor for movement in each axis, hence the LP in Fig. 2 corresponds to the motor for movement in x-axis (horizontal translation) and the LP in Fig. 3 and Fig. 4 correspond to the motor of the movement in y-axis (up and down). Therefore, to get the total LP of a storage process (round trip) we have superimposed the corresponding LPs depending on their temporal location. In the Fig. 5 is showed the LP for a full movement of storage below to an ASR. The LP is divided in two parts: at the top is the LP of the axis-x movement and the bottom is the LP of the axis-y movement.

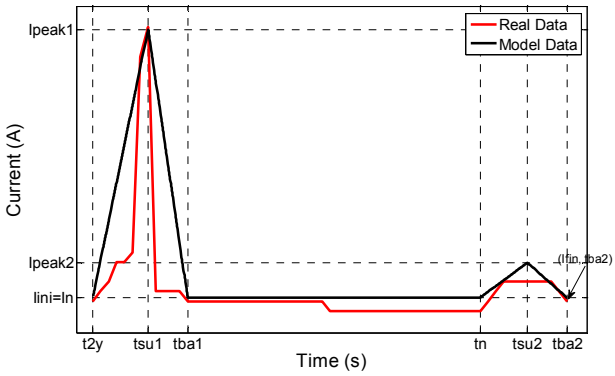

Fig. 4. The load profile of an ASR without load in down movement.

Fig. 5 shows how the timeline is segmented for each of the steps performed by the ASR in a storage movement. t_{1x} and t_{1y} are the start times. So Δt_{1x} and Δt_{1y} are the used delays to avoid the simultaneous start of the motors among those belong to the same ASR and among the motors of the different ASRs in the warehouse. The translation movement with load is made between T_x and t_{1x} times; the up movement with load, between T_y and t_{1y} times; so far as the going movements are executed, then the going time is determinate for the longest between T_x and T_y. When [X, Y] position are gotten the fork movement is executed. This is to leave the load on its storage position and it takes the time interval between t_f and the final going time (T_x or T_y as appropriate). Then, the ASR has to return to the starting position and pick up the next load. The return without load movement has the following sequence: again there are delay times to avoid simultaneous start and these are Δt_{2x} and Δt_{2y}; the return horizontal and the down movements are executed between T_{xtotal} and t_{2x}, T_{ytotal} and t_{2y} times, respectively. The total time of the whole movement will be the maxim between T_{xtotal} and T_{ytotal}. It has been show the modeling of the load profiles, and it is an important highlights their simplicity and practicality, which make it suitable to be implemented for a prognosis algorithm, for example, mean artificial neuronal networks.

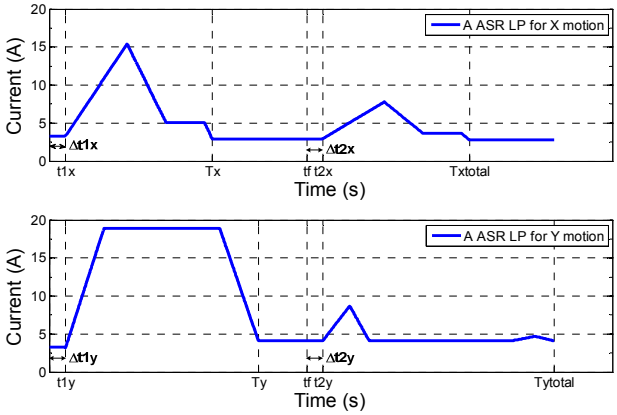

Fig. 5. Decomposed Full LP for an ASR.

3. Genetic Algorithm Solution

The proposed solution based on a Multi-Objective Genetic Algorithm (MOGA) is presented in the Fig. 6 as a general block diagram. Its main objective is to get the optimal delay times $[\Delta t_{1x}, \Delta t_{1y}, \Delta t_{2x}, \Delta t_{2y}]_{Mx4}$ (M is the number of ASRs) for each ASR in the warehouse but without increasing substantially the time for the operation. The first block has as inputs the $[X, Y]_{Mx2}$ coordinates, which determine the position where the goods have to be stored or retrieved. This block is used to get the LP prognosis, which has been replaced on this occasion for data derived from actual measurements (see section 2). The next block is the MOGA optimization, where the MOGA is executed and it has as outputs the delays that optimize the use of the available electric power. The final block is only used to visualize the results and do the comparison between the init and final data.

For this first implementation we have only had into account two ASRs ($M = 2$), which are named A and B. In addition, the route used by the ASRs is only a movement of going and back. The aim of this is to simplify the development but demonstrating the feasibility and suitability of using such algorithms in this type of problem. However, there are no constraints to increase the number of ASRs relative to the MOGA implementation. The having more ASRs or machines to management mainly affect the convergence time of the MOGA. This was tested in another application where fictitious load profiles were used to simulate the management of multiples machines (for example, more than 10 machines). Although the convergence time was increased, the generation's number remains about constant and such increase it was not so much in comparison with simulations with a low number of LPs (for example, less than 10 machines) The extension from two ASRs to M ASRs does not have enough problem since a generic vector and matrix indexation has been used.

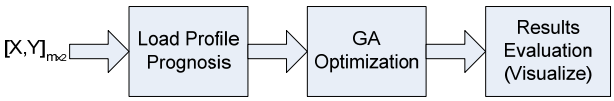

Fig. 6. The general block diagram of the proposed solution.

3.1. LP Prognosis

An important premise for the good performance of this block is that it must be possible to have a schedule of the allocation of goods for a certain period. Therefore, taking advantage of the modelling here presented, it is possible to implement an algorithm to do the prognosis of the load profile in function of the [X, Y] coordinates. It could be an algorithm with artificial neural networks or even an easier algorithm like a lookup table. This part of the whole application is pending of developing. Now, we have used the directly obtained models from the real data.

In order to implement this module, we divided it in two parts, as showed in the Fig. 7: The "Get Model Parameters" and the "Make Load Profile" sub models. The first has the task of getting the main parameters of each LP, as showed in "Load Profile modelling of ASRs" section. Now it is made of a semi-automated way. The second sub model uses the obtained parameter to concatenate the vector that represents each LP. These vectors are *A.lp* and *B.lp* and others parameters that are incasuled in *A* and *B* structures.

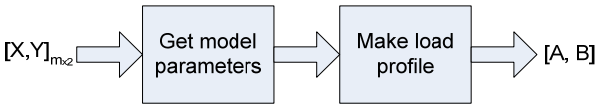

Fig. 7. Block diagram of the prognosis module.

3.2. MOGA and GA solution

Multi objective problem looks for the optimization of two or more utility functions since it is normal that in optimization problem we want to minimize or maximize more than one variable. Like this case, where we seek to decrease the power peaks but without affect the performance of the whole storage system, particularly the times of operation. In order to get these objectives, here we propose a multi objective genetic algorithm.

In the multi objective optimization, is normal that whereas one variable is being optimized the other one(s) is (are) being affected negatively. Then the search of an optimal solution has to be traded off in some way. So the concept of *nondominated* or *noninferior* variables and *Pareto optimality* appear [10].

GA consists on searching algorithms based on the mechanics of natural selection and natural genetics. They combine survival of the fittest among string structures (chromosomes) with a structures yet randomized information exchange to form a search algorithm with some of the innovative flair of human search. In every generation, a new set of artificial chromosomes (strings) are created using pieces of the fittest of the old; an occasional new part is tried for good measure; these new chromosomes are gotten means functions or operators that mainly emulate the evolutionary processes of selection, mating and mutation. While randomized, genetic algorithms are no simple random walk, they efficiently exploit historical information to speculate on new search points with expected improved performance. These algorithms are computationally simple yet powerful in their search for improvement. Furthermore, they are not fundamentally limited by restrictive assumptions about the search space (assumptions concerning continuity, existence of derivatives, unimodality, and other matters) Genetic Algorithms have been developed by John Holland, his collogues, and his students at the University of Michigan [10].

The Fig. 8 shows the flowchart of a standard GA. Firstly, the initial population is obtained by means of random initialization of the each chromosome of the population. The number of individuals or chromosomes is a very important parameter of the GA. So the diversity of the population depends on population size and that guarantees the no getting into local minimum. Nevertheless, a very large population could make so slow the execution of the GA. For this application, we chose a population initial of 60 individuals (it is a good start point to try with a population of $15 \times n$, where n is the number of fitness functions to be optimized).

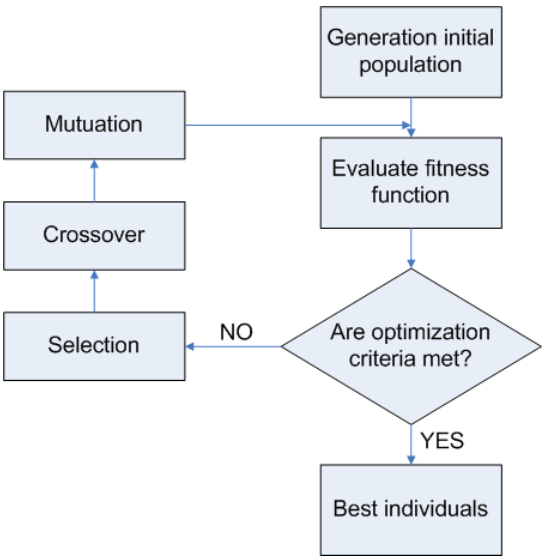

Fig. 8. Flowchart of a GA.

Next it is executed the calculation of the fitness functions. This is the function objective, which has to be minimized. In this case, we want to minimize the power peaks of the total load profile. Therefore, the fitness functions are determinate by means of the next equations:

t: time.

k: number of the actual iteration.

M: number of ASRs.

$fv_{1,k}$: value of the fitness function 1 for the iteration k (Power peak value)

$P_{k,max}$: max power peak for the iteration k.

$lp(t)_{k,i}$: the load profile of the ASR i, in the iteration k.

x: vector or matrix $[\Delta t_{1x}, \Delta t_{1y}, \Delta t_{2x}, \Delta t_{2y}]_{Mx4}$ of time delays that determinate the sequence of start of the ASRs.

$fv_{2,k}$: value of the fitness function 2 for the iteration k (total delay time value)

$T_{total,k}$: total time of the total load profile for the iteration k.

$T_{total,0}$: initial total time of the prognosis of the total load profile.

$$fv_{1,k} = P_{k,max} = max\left(\sum_{i=1}^{M} lp(t)_{k,i}\right) \quad (1)$$

$$fv_{2,k} = T_{total,k} - T_{total,0} \quad (2)$$

$$F = [fv_1\ fv_2] \quad (3)$$

$$\min_x F \quad (4)$$

Then the equation (4) is the target of the MOGA and the x vector is formed for the time delays that determinate the start of the ASRs as it is showed in the Fig. 5. Thus, the immediate coding of the chromosome of the GA is the showed in the Fig. 9. The first 4 gens belong to the A ASR and the next 4 to the B ASR.

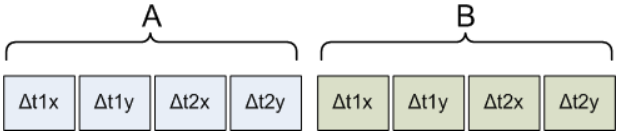

Fig. 9. String coding for the GA of the proposed solution.

The next step of the GA is the evaluations of the several optimization criteria are met. If so, the GA is stopped, or else, the evolutionary operators are used in order to get the next generation.

The used operators in this application were the *selection*, *crossover* and *mutation* ones. As its name suggests the selection consists in to select the best individuals from actual population. There are many ways to execute this operator [10]. Here we pass them to the next generation directly. The number of selected individuals is another parameter of the GA and determinates the degree of opportunity reproduction of the best individuals. This is named the selective pressure. The crossover or mating is the process of crossing over the genetic material from the parents to create the genetic material of the children. The crossover can be done of different ways. In this case, we have used the scattered function, which takes the genetic material from two parents and crosses over them following a generated random binary vector. Where the binary vector is 1 the gens are taken from parent 1 and in another case from parent 2. For example, if *p1* and *p2* are the parents

p1 = [a b c d e f g h]

p2 = [1 2 3 4 5 6 7 8]

And the binary vector is *[1 1 0 0 1 0 0 0]*, the function returns the following child:

child1 = [a b 3 4 e 6 7 8]

Mutation operation applies random changes to individual parents to form children. Mutation provides genetic diversity and enables the genetic algorithm to search a broader space. We have used Gaussian mutation, which adds a random number taken from a Gaussian distribution with mean 0 to each entry of the parent vector.

In summary, the final settings of the GA used were:

- Initial population: 60 random generated individuals
- Next generation: 2 selection + 16 crossover + 2 mutation
- Selection: stochastic uniform.
- Crossover: scattered
- Mutation: Gaussian

Next, we continue with the simulation results that were got by means of this configuration of the GA first approach solution.

4. Simulation Results

In the Fig. 10 is presented the simulation results. The initial LPs are at the top plot, the GA LP result are at the middle, and MOGA LP results are at the bottom plot. In all, the load profile of *A, B* and the *total* is in blue, red and black respectively. With a simple inspection, we can notice the differences in terms of duration, appearance and peak profiles. In the initial load profiles, we have a maxim peak of about 56 A that occurs around the 4 seconds.

This peak is generated by the maxim peak of *A* and the rising load profile of *B*. Remember that each load profile is built by the addition of the load profile of each motor in the same ASR and the *total* load profile by the addition of the two ASRs under test, *A* and *B*. These initial profiles are the supposed prognostic load profiles. At the middle are the final load profiles after the GA execution and reordered the start times of each motor in both ASRs. There we can see as they have changed. Now the maxim peak on the total load profile is 46 A at the 5.5 s time. It has been reduced in 21 %, which is an appreciable result. However, the total time has been increased. The initial total time was about 29 s and after the rearrangement, this was about 34 s, 5 s more than initial time. At the bottom of the plot, the final load profiles after the MOGA execution. Now, note carefully that the maxim power peak has decreased with respect to the initial one, but the total time remains constant. This shows the effectiveness of the MOGA versus the GA.

However, the only disadvantage of the MOGA is that it took more iterations than GA. The table 1 we can see the summary of the results of both implementations.

The Fig. 11 shows the Pareto front obtained by the MOGA. This lets us see the behaviour no lineal of the functions to optimize. In addition, we can see as the performance of the best individual decreases in terms of maximum delay, whereas performance increases in terms of reducing peak.

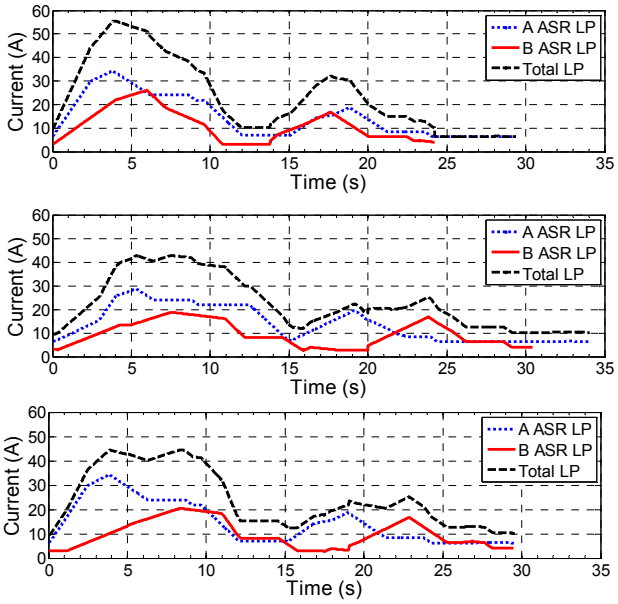

Fig. 10. Initial (at the top) LPs, GA (at the middle) and MOGA (at the bottom) LP results.

Description	GA	MOGA	Unit	Observation
ASR number	2	2	uni	A and B
Motors for control	2	2	uni	x and y axis
Total searched delays	8	8	uni	4 by ASR
Allowed maximum delay	5	5	s	
Iterations or generations	52	123		
Maximum got delay	4,9	5	s	
Initial total time	29,4	29.4	s	
End total time	34	29.5	s	With MOGA the total delay is less
Initial maximum total load peak	56	56	A	
End maximum total load peak	44	44,49	A	
Load peak reduction	21%	21%	%	

Table 1. Summary results for the Implemented GA and MOGA.

The CPU time used by the MOGA is considerably longer than the used by the GA. The first takes about 90 seconds and the second about 3 seconds. Take into account that due to the nature stochastic of both algorithms the CPU time changes in each execution, then the given values are the average times for several simulations. The CPU was an Intel® Core™2 Quad CPU Q6600 @ 2.400 GHz.

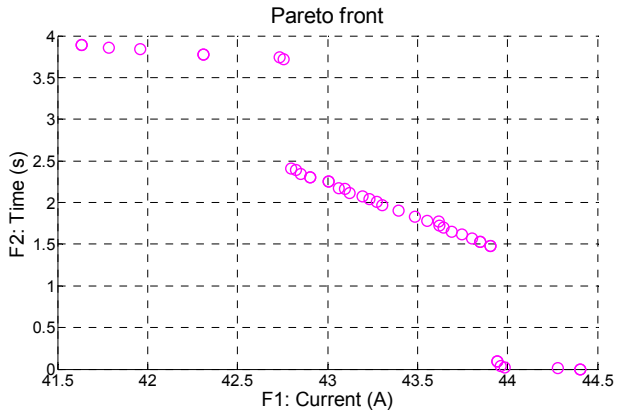

Fig. 11. Pareto front.

Despite these results, the MOGA and GA algorithms are considered feasible of being used since they will be used in off line application, together the algorithms for optimizing the pallets location. These are typically executed before the storage and retrieval process begins.

5. Conclusions

It is important to notice that energy optimization issues are raising more and more due to the current worldwide energetic problem.

A MOGA and GA approach for solving the sequencing problem to get optimal use of available power in an automated warehouse was implemented with appreciable results about reducing the instantaneous maxim power demand without increasing substantially the time for the operation.

This shows the feasibility and the potential of this kind of algorithm to solve problems of DSM in automated warehouse.

To develop a full DSM application, the next step is to implement a prognosis algorithm to get the load profile prognosis. This algorithm could be a neural network in order to take advantage of the method of modelling proposed here. By this way, the whole system can be tested in a real application.

References

[1] J. Drury and P. Falconer, *Building and planning for industrial storage and distribution*, 2 ed.: Architectural Press, 2003.

[2] S. Chwen-Tzeng, "Intelligent control mechanism of part picking operations of automated warehouse," in *Industrial Automation and Control: Emerging Technologies, 1995., International IEEE/IAS Conference on*, 1995, pp. 256-261.

[3] S.-n. Liu, "Optimization problem for AGV in automated warehouse system," in *Service Operations and Logistics, and Informatics, 2008. IEEE/SOLI 2008. IEEE International Conference on*, 2008, pp. 1640-1642.

[4] C. Yueting and H. Fang, "Research on particle swarm optimization in location assignment optimization," in *Intelligent Control and Automation, 2008. WCICA 2008. 7th World Congress on*, 2008, pp. 111-116.

[5] L. Meijuan, C. Xuebo, and Z. Meifeng, "Optimal Scheduling Approach of Storage/Retrieval Equipments Based on Genetic Algorithm," in *Intelligent Control and Automation, 2006. WCICA 2006. The Sixth World Congress on*, 2006, pp. 3345-3348.

[6] L. Meijuan and C. Xuebo, "Research on optimization problem of the automated warehouse using an improved ant colony algorithm," in *Intelligent Control and Automation, 2008. WCICA 2008. 7th World Congress on*, 2008, pp. 8716-8721.

[7] L. Y. Seng and P. Taylor, "Innovative Application of Demand Side Management to Power Systems," in *Industrial and Information Systems, First International Conference on*, 2006, pp. 185-189.

[8] Y. K. Penya, "Optimal Allocation and Scheduling of Demand in Deregulated Energy Markets," in *Tesis Doctoral - Vienna University of Technology*, 2006, p. 161.

[9] Y. K. Penya., "Last-generation Applied Artificial Intelligence for Energy Management in Building Automation," in *In the Proceedings of the 5th IFAC International Conference on Fieldbus Systems and their Applications, FeT 2003* Aveiro Portugal, 2003.

[10] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*: Addison-Wesley Longman, Inc., 1989.