J Intell Manuf (2018) 29:1545–1557
https://doi.org/10.1007/s10845-016-1198-x

Implementation and comparison of algorithms for multi-objective optimization based on genetic algorithms applied to the management of an automated warehouse

Gianluca Nastasi[1] · **Valentina Colla**[1] · **Silvia Cateni**[1] · **Simone Campigli**[2]

Received: 11 December 2014 / Accepted: 16 January 2016 / Published online: 27 January 2016

Abstract The paper presents strategies optimization for an existing automated warehouse located in a steelmaking industry. Genetic algorithms are applied to this purpose and three different popular algorithms capable to deal with multi-objective optimization are compared. The three algorithms, namely the *Niched Pareto Genetic Algorithm*, the *Non-dominated Sorting Genetic Algorithm 2* and the *Strength Pareto Genetic Algorithm 2*, are described in details and the achieved results are widely discussed; moreover several statistical tests have been applied in order to evaluate the statistical significance of the obtained results.

Keywords Multi-objective optimization · Evolutionary algorithms · Niched Pareto Genetic Algorithm · Non-dominated Sorting Genetic Algorithm 2 · Strength Pareto Evolutionary Algorithm 2 · Warehouse

Introduction

Automated warehouses for stocking several products are widely spread in many industrial fields (Berg and Zijm 1999). The main objective in the realization of these automated stocking systems lies in their reliability and in the possibility to reduce human intervention by decreasing costs and improving workers' safety. Warehouses can be included in the wider class of logistic systems, whose efficiency is related to several factors, such as the research of the best position for distribution facility prospecting the optimal strategy which is satisfactory to the costumer demand at minimum cost and maximum profit (Ming-bao et al. 2007; Li and Wu 2007). The growing diffusion of these system and their importance in determining the efficiency of the whole manufacturing chain is attested by the increasing literature on this subject. Warehouse operation planning problems can be basically classified according to some macro-categories, i.e. material reception, storage and retrieval, order picking and shipping (Gu et al. 2007). A recent study analyses the requirements for database and interface system for automated warehouses (Guo and Jin 2014). However a key role in order to optimize the warehouse efficiency and reduce the inbound logistic costs is played by storage and retrieval strategies (Roodbergen and Vis 2009), which usually aim at satisfying criteria related to space exploitation, energy efficiency, storage safety and stability and easy product dispatch (especially in case of products subjected to deterioration and ageing). These criteria might conflict against each other, therefore a suitable trade-off need to be found among them. An example is provided in (Bortolini et al. 2014), where a linear programming model is presented which optimises the load assignment within an Automated Storage and Retrieval System through a multi-objective perspective, by considering time efficiency for the operations, energy consumption and risk of collapse during dangerous situations (e.g. seismic events).

Genetic Algorithms (GAs) (Goldberg 1989) can be successfully exploited to solve complex optimization problems with one or multiple objectives (Coello et al. 2002) and many examples can be found in literature of their application in

✉ Gianluca Nastasi
g.nastasi@sssup.it

Valentina Colla
v.collai@sssup.it

Silvia Cateni
s.cateni@sssup.it

Simone Campigli
simone.campigli@roboris.it

[1] Scuola Superiore Sant'Anna, Pisa, Italy

[2] Roboris s.r.l., Pisa, Italy

logistics-related problems (Han et al. 2015; Takeyasu and Kainosho 2014; Hiremath et al. 2013; Liu and Xu 2011) and also specifically for automated warehouses management and control (Colla et al. 2010b; Figueiredo et al. 2012; Popović et al. 2014).

This paper presents and compares three different GA-based approaches for facing a Multi-Objective Optimization Problem (MOOP) related the allocation strategies in a real automated warehouse for the primary steelmaking industry.

The paper is organized as follows: second section provides a theoretical background on the MOOP; third section briefly introduces the GAs, while fourth section introduces and compared three GA-based algorithms for facing MOOPs, i.e. the Niched Pareto Algorithm (NPGA), the Non-dominated Sorting Genetic Algorithm 2 (NSGA2) (Hiremath et al. 2013; Bandyopadhyay and Bhattacharya 2013) and the Strength Pareto Evolutionary Algorithm 2 (SPEA2). A brief description of the considered automated warehouse is presented in fifth section, while in sixth section some numerical results are presented and widely discussed by also exploiting some statistical tests to evaluate the statistical significance of the achieved results. Finally some concluding remarks are given in last section.

Multiobjective optimization problem

Real world optimization problems are often characterized by the presence of many different objectives that should be combined to reach a final solution. The process of finding one (or more) optimal trade-off among several possibly conflicting objective functions is called Multi-Objective Optimization Problem (MOOP) and can be defined as the problem of "finding a vector of decision variables which satisfies constraints and optimizes a vector function whose elements represent the objective functions. These functions form a mathematical description of performance criteria which are usually in conflict with each other. Hence, the term optimize means finding such a solution which would give the values of all the objective functions acceptable to the designer" (Osyczka 1985).

A generic MOOP can be described as follows:

$$\begin{cases} \max \mathbf{F}(\mathbf{x}) = [f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_k(\mathbf{x})]^T \\ \text{subject to } \mathbf{G}(\mathbf{x}) = [g_1(\mathbf{x}), g_2(\mathbf{x}), \ldots, g_m(\mathbf{x})] \leq 0 \\ \mathbf{x} \in D \end{cases} \quad (1)$$

where $\mathbf{x} = (x_1, x_2, \ldots, x_n)^T$ is the vector of the decision variables, $f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_k(\mathbf{x})$ are the k objective functions, $g_1(\mathbf{x}), g_2(\mathbf{x}), \ldots, g_m(\mathbf{x})$ are the m constraints, and $D \in R^n$ is the solution space. The vector function that should be optimized forms a mathematical description of the performance criteria which can also conflict against each other. In this sense, the term *optimize* means reaching a solution which provide the decision maker with a satisfactory trade-off among the values of all objective functions (Amuso and Enslin 2007). The utopian solution is defined as in (2)

$$\mathbf{F^*} = (f_1^*, f_2^*, \ldots, f_k^*)^T \quad (2)$$

where $f_1^*, f_2^*, \ldots, f_k^*$ denote the individual minima of each respective objective function. The ideal solution (feasible only for trivial problems) can be reached if all $f_i(\mathbf{x}$ have their minimum in the same point. This ideal result can be obtained by determining $\mathbf{x^*} = [x_1^*, x_2^*, \ldots, x_k^*]$ such that it satisfies equation (1) by providing the optimum values of all objective functions. Finding an ideal solution is very rare, especially as the dimensionality of the problem increases; this is the reason why the optimal term must be defined as a good compromise between the optimality of different objective functions.

The idea of a trade-off optimality criteria was introduced by Edgeworth in 1881 and subsequently generalized by Pareto in 1896 (Coello et al. 2002). A solution is included to the so-called *Pareto set* if another solution does not exist that can improve at least one of the objectives without degrading any other objective. The Pareto optimum usually provides not a single result but rather a set of solutions, the so–called *non dominated* solutions. In other words, a solution $\mathbf{x^*}$ is said to be Pareto optimal if and only if there are no other solutions dominating it. The set of all Pareto solutions is called the *Pareto optimal set* and the corresponding objective vectors are said to be on the *Pareto front*. Once the Pareto set is found, the solution which optimizes the objective function considered mostly important or prioritary for the problem under consideration is selected. To solve this type of problems, very often evolutionary algorithms rather than a purely mathematical resolution are chosen. Evolutionary algorithms can be applied to complex domain problems as they ensure a convergence to feasible solutions for a given problem, although the convergence to the optimum is not guaranteed.

Genetic algorithms

Genetic Algorithms (GAs) are stochastic search algorithms based on the basic principles of biological evolution and natural selection. The idea of evolutionary computing was proposed in the 1960s by I. Rechenberg and in 1975 John Holland invented the Genetic Algorithms (Holland 1975; Goldberg 1989).

In GAs, each optimization decision variable is coded into a *gene*, that can be represented by a real number, a bit or a string of bits. The corresponding genes for all parameters generate a *chromosome* which represents each individual. A chromosome can be an array of real numbers or a binary string depending on the treated problem. An individual is a possible

solution while a set of individuals is called *population*. Solutions from one population are selected and mated through a *selection operator*, such as *tournament* or *roulette wheel selection*, where the probability to be selected is somehow proportional to the individual own *fitness* (i.e. a quantitative measure of its goodness with respect to the optimization problem). These set of selected individuals (the *mating pool*) is used to generate a new population through genetic operators called *crossover* and *mutation*. *Crossover* selects genes from parent chromosomes and forms a new offspring. The simplest method is to choose randomly two corresponding genes belonging to different parents in order to obtain a son. Specific crossover made for a certain problem can improve performance of the genetic algorithm. On the other hand *mutation* is performed in order to prevent that all the solutions of the population get trapped into a local optimum. *Mutation* changes randomly a gene belonging to the new offspring. In general, the new population is expected to hold better features with respect the previous one. This process is repeated until some conditions are satisfied, in terms, for instance, of goodness of the best solution, average fitness of the whole population or maximum number of developed populations (*generations*).

GAs have been applied to a large range of real world problems (Colla et al. 2010a; Cateni et al. 2010; Colla et al. 2008a; Borup and Parkinson 1992; Fonseca and Fleming 1998; Kroo et al. 2000; Krus et al. 1996), as they are very robust and can handle all type of fitness landscape and mixture of real and discrete parameters.

Genetic algorithms applied to multi-objective optimization

In this section three GA-based strategies for Multi-objective Optimization are presented, which have been applied to the proposed practical problem: the Niched Pareto Genetic Algorithm (NPGA), the Non-dominated sorting GA2 (NSGA2) and the Strength Pareto Evolutionary Algorithm 2 (SPEA2). In the following a subsection is devoted to each algorithm.

Niched Pareto Genetic Algorithm (NPGA)

The specific aspects of the NPGA (Horn et al. 1994; Lu et al. 2011) are concentrated in implementation of the GA. The most widely used selection method is the so-called *tournament*. In tournament selection a set of individuals is randomly selected from the considered population and the best one of this subset is assigned to the *mating pool*. It is possible control the selection pressure and hence the convergence speed by regulating the size of the tournament. Tournament selection presupposes that the user desires a single answer to the problem. After several generations the population will converge to a uniform one. NPGA, in order to avoid convergence and to conserve multiple Pareto optimal solutions, modifies tournament selection in two manners: introducing *Pareto domination tournament* and, in the case of non-dominant tournament, *sharing* is performed in order to select the winner.

Pareto domination tournaments

The binary relation of domination conducts to a binary tournament where two randomly extracted individuals are compared; in this way, when an individual dominates the other one, it is considered the winner although providing insufficient domination pressure. In order to improve domination pressure and control of the same pressure, a sampling scheme is performed. The two candidates for the reproduction are equally and randomly selected from the population. In addition, a set of solutions to be used for comparison called *comparison set* is increasingly defined by randomly selecting individuals from the population. Both of the previously selected candidates are compared to all elements of the *comparison set*; if a candidate is dominated by at least one element and the other does not, then the latter is considered the winner of the tournament and it will be part of those chosen for reproduction. In the event that both participants are dominated by the *comparison set*, the *fitness sharing* is exploited to determine the winner. The size of the *comparison set* allows controlling the selection pressure and is called *domination pressure*.

Sharing on the non-dominated frontier

Fitness sharing was initially proposed by Goldberg and Richardson (1987), analysed in detail by Deb (1989) and finally applied successfully to several complex and real contexts. The main objective of *fitness sharing* is to distribute the whole population over a number of segments in the search space, where each segment receives a part of the population in proportion to its size. In order to compute the *fitness sharing*, two parameters must be determined: *objective fitness* and niche count m_i. The objective fitness is calculated through the so called *Goldberg's ranking*. *Goldberg's ranking* is used in order to achieve equal reproductive potential for all individuals on the Pareto front. All individuals that are not dominated by others are initially identified in the population and rank 1 is assigned to them. They are then ignored and the same procedure is repeated in order to define set of individuals belonging to rank 2 and so on until all individuals have been classified. A simple example where two objective functions are minimized using the Goldberg's Ranking algorithm is shown in Fig. 1.

The niche count m_k represents an estimate of how the niche is crowded (the neighbourhood) to the k-th individual

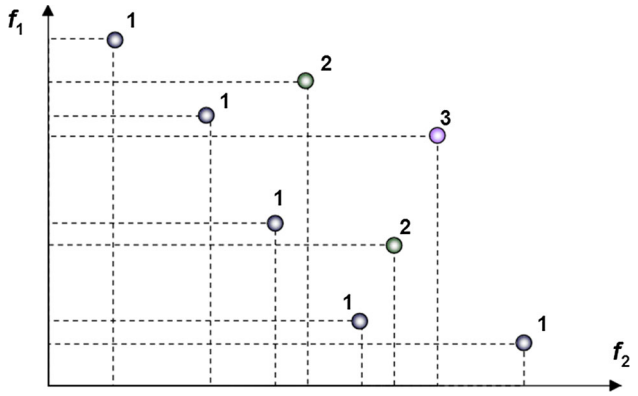

Fig. 1 An example concerning Goldbergs Ranking procedure

$\mathbf{x_k}$. This value is calculated over all elements belonging to the i-th population P_i as follows:

$$m_k = \sum_{\mathbf{x_j} \in P_i} Sh\left[d\left(\mathbf{x_k}, \mathbf{x_j}\right)\right] \tag{3}$$

where $d(\cdot, \cdot)$ is a distance operator, $Sh[d]$ is the value of the *sharing function*. $Sh[d]$ is a decreasing function of the distance value and $Sh[d \geq \sigma_{share}] = 0$.

Typically, the *triangular sharing function* is adopted, where $Sh[d] = 1 - \frac{d}{\sigma_{share}}$ and $Sh[d] = 0$ for $d > \sigma_{share}$. In this context the parameter σ_{share} is called *niche radius* and is fixed by the user in order to estimate the expected or desired minimal separation between solutions.

The Non-dominated Sorting Genetic Algorithm 2 (NSGA2)

In Srinivas and Deb (1995) proposed an algorithm, called Non-dominated Sorting Genetic Algorithm (NSGA) which differs from simple genetic algorithm only in the selection operator. Before performing the selection, the population is ranked considering the individual non domination. The non dominated individuals are identified and then assumed to consistute the first non dominated front in the population, to which a large dummy fitness value is assigned. These individuals are then shared with their dummy fitness value in order to conserve dissimilarity in the population. After sharing these non dominated individuals, they are temporarily set aside in order to process the remaining ones and also to identify individuals for the second non dominated front. To this new set of individuals is then assigned a new dummy fitness value which will be smaller than the minimum shared dummy fitness of the previous front. This procedure continues until the whole population is classified into different fronts. This method is used to search for non-dominated regions or Pareto fronts and its efficiency lies in the way multiple objectives can be solved; moreover both minimization and maximization problems can be handled. NSGA over the years has been widely criticized. The main criticisms are the high computational complexity of non dominated sorting, the lack of elitism (Zitzler et al. 2000; Rudolph 1999) and finally the need for specifying the sharing parameter (Fonseca and Fleming 1998). In Deb et al. (2002) proposed an improved version of NSGA, called NSGA2 which introduced a fast non dominated sorting algorithm. The main objective of NSGA2 is the introduction of an elitism-preserving approach and a parameterless niching operator for diversity preservation (the so-called *crowding distance* comparison operator) providing an improved computational complexity. In order to understand the NSGA2 procedure, the following three operations are introduced:

- Non dominated sorting on population P_i. The solutions in P_i, which do not dominate each other but dominate all other solutions belonging to P_i are saved in the first front named B_1. The same operation is performed in order to find solutions which are kept in the second front B_2 and so on until no solutions are left. To each front the corresponding rank is assigned, i.e. rank i is assigned to solutions in B_i.
- Solutions are then sorted in accord to their crowding distances in order to select solutions from the same non-dominated front. The crowding distance computation requires sorting the population on the basis of each objective function value in ascending order of magnitude. For each objective function, an infinite distance value is assigned to the boundary solutions. To the intermediate solutions a distance value is assigned which is equal to the absolute normalized difference in the function values of two consecutive solutions. The overall crowding distance is computed as the sum of individual distance values relative to each objective. Naturally each objective function is firstly normalized. The crowded-comparison operator selects the solution that has bigger crowding distance.
- An elitist preserving approach. An elitist preserving approach is used to avoid missing Pareto-optimal solution from a parent population P_i. In order to generate the next generation P_{i+1} the following steps are performed:
  1. Solutions which are considered infeasible are cast aside.
  2. P_{i+1} is filled with parent and offspring population on the basis of the non-dominated sorting rank until the number of solutions n is greater than the population scale N.
  3. In order to keep the size of P_{i+1} equal to N, $n - N$ solutions from the last included non-dominated front are removed from P_{i+1}.

The Strength Pareto Evolutionary Algorithm 2

The Strength Pareto Evolutionary Algorithm (SPEA) was introduced in Zitzler et al. (1999). SPEA exploits a typical population structure plus an external set that archives non-dominated solutions found during the generations.

The procedure starts with an initial population P_0 and an external empty set $\overline{P}_0$. The iterative algorithm at each generation i stores the non-dominated solutions achieved throughout the generations in the external population set $\overline{P}_i$, whose dimension grows through time. SPEA exploits the notion of Pareto dominance in order to fix scalar fitness values to individuals. Although SPEA produced good results with respect to other evolutionary algorithms (Zitzler and Thiele 1999). Zitzler et al. (2001) identified several weaknesses and proposed an improved strength Pareto Evolutionary Algorithm called SPEA2. The main limits of SPEA algorithm are the following: individuals dominated by the same archive individuals that are stored in $\overline{P}_i$ have the same fitness value. This can be a problem when $\overline{P}_i$ includes a single individual, as all the population individuals have an identical rank regardless of the fact that they might dominate each other, by thus identifying SPEA as a random search algorithm. Moreover if many individuals of P_i do not dominate each other, very poor information can be acquired considering the partial order defined by the dominance relation. SPEA2 was introduced in order to overcome these problems. SPEA2 uses an improved fitness assignment scheme that assesses how many individuals are dominated and by how many individuals they are dominated. Moreover the use of archive truncation method assures the preservation of boundary solutions. SPEA2 algorithm can be described by the following six steps (Amuso and Enslin 2007; Tami and Abido 2011):

1. An initial population P_0 of size N and an empty external (archive) Pareto optimal set $\overline{P}_0$ of maximum size M are created. This initialisation step is performed only once at the beginning of the procedure.
2. External set updating: at the generic i-th generation a fitness value is assigned to each individual. The fitness assignment includes density information into its evaluation, differently from SPEA. Other main differences lie in the fact that in SPEA2 the archive set size is fixed and SPEA2 does not use a clustering method to reduce the archive size, but a truncation method that preserves boundary points. A strength value $S(\mathbf{x_k})$ is assigned to the k-th element $\mathbf{x_k}$ of the population P_i, which is computed as:

$$S(\mathbf{x_k}) = \left|\left\{\mathbf{x_j} | \mathbf{x_j} \in \left(P_i + \overline{P}_i\right) \wedge \left(\mathbf{x_k} \succ \mathbf{x_j}\right)\right\}\right| \tag{4}$$

where $|.|$ indicates the cardinality of the set while the symbol $\succ$ corresponds to the Pareto dominance relation. The raw fitness $R(\mathbf{x_k})$ is then computed as:

$$R(\mathbf{x_k}) = \sum_{\substack{\mathbf{x_j} \in P_i + \overline{P_i} \\ \mathbf{x_k} \succ \mathbf{x_j}}} S(\mathbf{x_j}) \tag{5}$$

As it is possible that different individuals of the same population have identical raw fitness values, a density estimate is also performed in order to provide additional information. For each individual $\mathbf{x_k}$ the distances from all the other solutions present in archive and in the population are computed and stored in a list. The list is sorted in increasing order and the distance value ranked at the m-th position in such list can be indicated as $\sigma(\mathbf{x_k})^m \geq 0$. The density for $\mathbf{x_k}$ is thus calculated as:

$$D(\mathbf{x_k}) = \frac{1}{\sigma(\mathbf{x_k})^m + 2} \tag{6}$$

where m is typically equal to the square root of the sample size and obviously $D(\mathbf{x_k}) < 1$.
The individual fitness value of $\mathbf{x_k}$ is finally computed as follows:

$$F(\mathbf{x_k}) = R(\mathbf{x_k}) + D(\mathbf{x_k}) \tag{7}$$

3. All non dominated individuals in the i-th population P_i are stored in $\overline{P}_{i+1}$. The size of $\overline{P}_{i+1}$ is kept constant by means of a truncation operator, namely the non dominated individuals are stored into the archive set until the maximum dimension in eventually reached.
4. Two individuals are randomly selected from the update archive set and their fitness is compared: the one with highest fitness is copied to the mating pool.
5. Crossover and mutation operations are performed according to their probabilities to create the new population.
6. The termination criteria are checked: if they are satisfied, the procedure stops, otherwise it iterates starting from step No. 2.

Automated warehouse description

The considered automated warehouse is hosted by a company producing welded steel tubes, whose plant is composed by three main areas: a production area, a storage area and a shipping area. In the production area the tubes are manufactured, packaged and subsequently carried to the storage area, where they are stacked on the basis of their shape and length. Each tube is characterized by a specific type depending on its length, section shape, section size and steel quality: more than 1000 tube typologies are produced. When a pack is ready at the end of the production line, it is carried to the storage area through an automated material handling system which is composed by 43 automated cranes (trolleys). Trolleys move

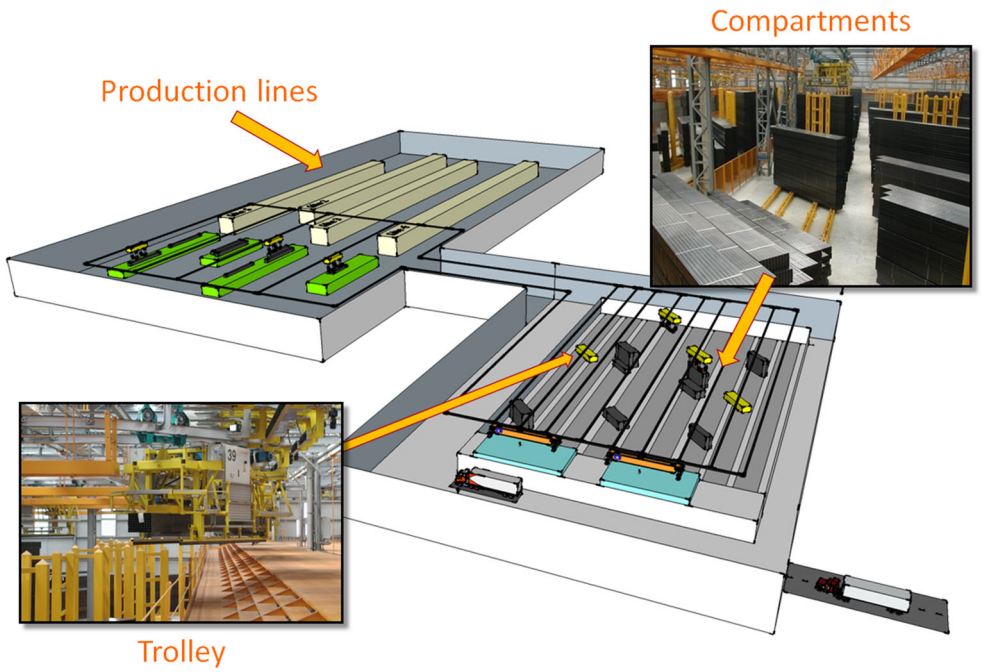

Fig. 2 The plant (Colla et al. 2010b)

through a suspended railway system and they are able to self-governing movements thanks to an on-board Programmable Logic Controller (PLC). The task of the main PLC is to give three types of missions to trolley: pack retrieval at the end of a production line, transport to the storage area and then stacking or pack retrieval in the storage area for dispatching or, finally, packs recording in the storage area. The automated warehouse system is linked to the information system of the company in order to receive production and dispatching order. The storage area is divided into nine aisles and it has a dimension of about 50×90 m. There are no predefined slots where tubes are stored, in facts aisles are dynamically divided into compartments and each compartment is in turn composed by stacks of packs of the same typology of tubes, whose sizes are assigned by the allocation algorithm. Adjacent compartments are separated by a space of 0.5 m. Tube packs stay in the warehouse until they are bought by costumers through a Business-to-Business (B2B) web portal, which is directly linked to the automated warehouse management system. Finally tubes pack are moved to the dispatching area where are loaded on trucks which deliver them to customers. The warehouse is completely automated and its Warehouse Management System (WMS) can be resumed in the following four parts which are ruled by specific algorithms (Fig. 2):

1. Trolleys management system;
2. Allocation system;
3. Reordering system;
4. Dispatching system.

The allocation system is used to optimize the space organization and to encourage the distribution of packs along different aisles. The algorithm selects a suitable collocation finding an empty space where the compartments fit with a residual space smaller than a fixed threshold. If this space does not exist the algorithm searches for an empty space whose residual overcomes another threshold. The reordering subsystem tries to compact compartments in order to free space for other productions. Usually three reordering strategies are performed: FIFO (First In- First Out) reordering, stacks reordering and space reordering. FIFO reordering objective consists in collocating older packs that are included in almost empty compartments over new ones in order to prevent tube oxidation that can be a cause of customer rejections. Two or more compartments are merged together so that older packs stay on the top of the stacks through a specific algorithm that uses a threshold on packs age. The objective of the stacks reordering strategy is to compact compartments which have too few packs distributed in many stacks. Finally the space reordering algorithm de-fragments empty spaces in order to reach wider spaces reducing the small ones. The compartments chosen for this objective are those that contain at least one adjacent empty space which is greater than a fixed threshold. The selected compartments are carried to a new position where they fit better in order to free a large area of aisle. As mentioned above the algorithms that currently perform allocation and reordering strategies of tubes packs are just based on heuristic and simple rules. The main limit of these strategies regards the fact that they do not develop a complete search on all available free spaces or compartments.

Also allocation and reordering strategies are not optimized in the sense that they are not driven by objective functions. In fact the application of these procedures could conduct the warehouse to a false saturation status and a manual storage operation in another not automatized warehouse should be performed losing time, efficiency and safety. For this aim three Key Performance Indicators (KPIs) are introduced in order to obtain objective functions for new strategies.

1. Total stored Weight (TSW). This index depends on the number and dimension of the items which are stored in the warehouse and its equation is the following:

$$TSW = \sum_{k=1}^{N_c} \sum_{j=1}^{N_k} w_{kj} \tag{8}$$

where N_c is the number of associated compartments, $N_i(k)$ represents the number of items of the k_{th} compartment and w_{kj} is the weight of the j-th item. TSK should be maximised.

2. Empty Space (ES). This index identifies the overall amount of empty space in the aisles, is usually expressed in centimeters and is computed as follows:

$$ES = \sum_{k=1}^{N_s} l_k \tag{9}$$

where N_s is the number of empty spaces in the warehouse while l_k is the individual length. Also this index should be maximised.

3. Fragmentation (F). This index provides a measure of how much the empty space is fragmented into small slots. $F \simeq 1$ means that the warehouse contains a lot of small empty spaces, vice-versa when $F \simeq 0$ there are a few big empty slots. F can be expressed as follows:

$$F(t) = 1 - \frac{Ns(0)}{Ns(t)} \tag{10}$$

where $Ns(0)$ indicates the number of empty spaces when no compartment is allocated. $Ns(t)$ identifies the number of empty spaces counted at the time of the measurement. This index should be minimised.

The aim of the present work is to find a suitable trade-off among all the above listed KPIs by defining a vector $\mathbf{z}$. Therefore the MOOP can be expressed as follows:

$$\begin{cases} \max \mathbf{F}(\mathbf{x}) = [TSW(\mathbf{x}), ES(\mathbf{x}), -F(\mathbf{x})]^T \\ \mathbf{x} \in D \\ \mathbf{G}(\mathbf{x}) \leq 0 \end{cases} \tag{11}$$

where $\mathbf{x}$ represents the position and stacks configuration that a compartment can assume in the warehouse, D is the domain of $\mathbf{x}$. The achievement of a solution through traditional optimization techniques is very difficult and not always possible, also due to the difficulty of translating and evaluating both the objective vector $\mathbf{F}(\mathbf{x})$ and the constraints vector $\mathbf{G}(\mathbf{x})$ in terms of mathematical equations, as they depend on the layout, the type of material handling system, the shape and properties of packages as well as the storage and reordering algorithms (Colla et al. 2008b). For instance, the evaluation of the fitness function requires to simulate how the WMS will store the packs into the warehouse that, in turn, requires to consider all the constraint on their placement as explained before (e.g. compartments must be composed by the same typology of tubes; compartments heights depend on their configuration, packs older then a certain threshold must stay on top; etc.). These can't be described in terms of equations, since they are based on a complex algorithm. Thus, the overall WMS has been reproduced and simulated by means of a software simulator, according to the information provided by the plant managers and the routines available at the plant, in a way that only feasible configurations are handled and, for each of them, the KPIs can be computed but this does not allow to solve the above-mentioned MOOP through traditional optimization techniques. This is the reason why an approach based on evolutionary computation has been selected.

Numerical results

NPGA, NSGA2 and SPEA2 have been implemented and tested for the problem of warehouse optimization, by exploiting the simulation system described in (Colla et al. 2008b), which is directly fed with the same real data that are processed by the WMS and produced an overview of the distribution of the tube packs within the aisles and compartments which is identical to the one available for the operator through the interface of the WMS. Moreover, within this system also the traditional allocation and reordering procedures have been implemented, Therefore through this system, a direct comparison is possible among the different allocation strategies.

It is important to underline that at the end of the GA-based optimization procedure, a Pareto front is produced which contains a set of optimal solutions for the desired problem. Among these solutions, the one is selected which maximizes TSW, as, for the problem under consideration, it represents the most significant index. For each algorithm 10 different simulations have been performed in order to obtain reliable results. Each simulation include the processing of the real data referring to 1 full year of production corresponding to 352 days.

Figure 3 shows the trend of the TSW in the warehouse (in tons) at the end of each day, which is measured at the

Fig. 3 Total stored weigth

completion of the placement, shipping and reordering. For all 3 tested algorithms a negative peak is present during the month of August due to the closure of the plant.

Figure 4 shows the daily trend of the index ES obtained by applying the 3 previously listed GA-based strategies, while Fig. 5 depicts the daily trend concerning the fragmentation index F. In this last case, all the simulations start with from a null value of F, which corresponds to the perfect compaction of the compartments.

Tables 1, 2 and 3 summarise the results of the simulation in terms of average values of the considered performance indexs (in the following indicated as $\overline{TSW}$, $\overline{ES}$ and $\overline{F}$) as well as the maximum value of TSW (indicated as TSW_{max}. In order to prove the effectiveness of the proposed methods, the last 3 columns of each table show the percentage difference with respect to the results obtained through the usual basic heuristic strategy (HS), through NPGA as well as through a Fuzzy GA-based strategy (FuzGA) (which is based on a fuzzy aggregation of the objective functions) which has already been described in some previous works (Colla et al. 2008b, 2009, 2010b).

NPGA provides worse results than those obtained with NSGA2 and SPEA2 but still better than HS. The strategies based on the algorithms NSGA2 and SPEA2 provide the best results regarding TSW, TSW_{max} and the flow of incoming tube packs in the warehouse, which are, in conclusion, the most important KPIs to optimize. The other two KPIs (ES and F) are slightly sacrificed in order to maximize TSW which is considered more relevant and drives the final selection of the solution among those belonging to the Pareto front. In order to better understand these results, let us consider a situation in which a newly produced tubes batch is arriving. The optimization algorithm is then run and outputs a set of Pareto optimal solutions (i.e. compartments where to place new packs) that represents different trade-offs between the KPIs. Some of these solutions could, for instance, maximize ES but not the number of inserted packs. This condition does not maximize TSW and should thus be avoided, since this is the main objective. In order to select the most suitable solution to apply among the Pareto optimal solutions returned by the optimization algorithm, the following procedure is followed: firstly, the solutions that maximize TSW are selected and then, among these latter ones, the solutions that optimize ES and F are selected. If more than one solution satisfies these criteria, a random selection is performed. This explains why HS seems to have better performances in terms of $\overline{ES}$, but far worse values for TSW and $\overline{F}$ indicate that the available empty space is actually more fragmented and, thus, less usable for the allocation of batches that are produced thereafter.

Regarding the SPEA2 algorithm, noticeably an increase of the TSW of further 337 ton/day (+3.75 %) is achieved with respect to the SPGA strategy. NSGA2 provides an even higher increase of TSW exceeding the 450 ton/day (+5 %).

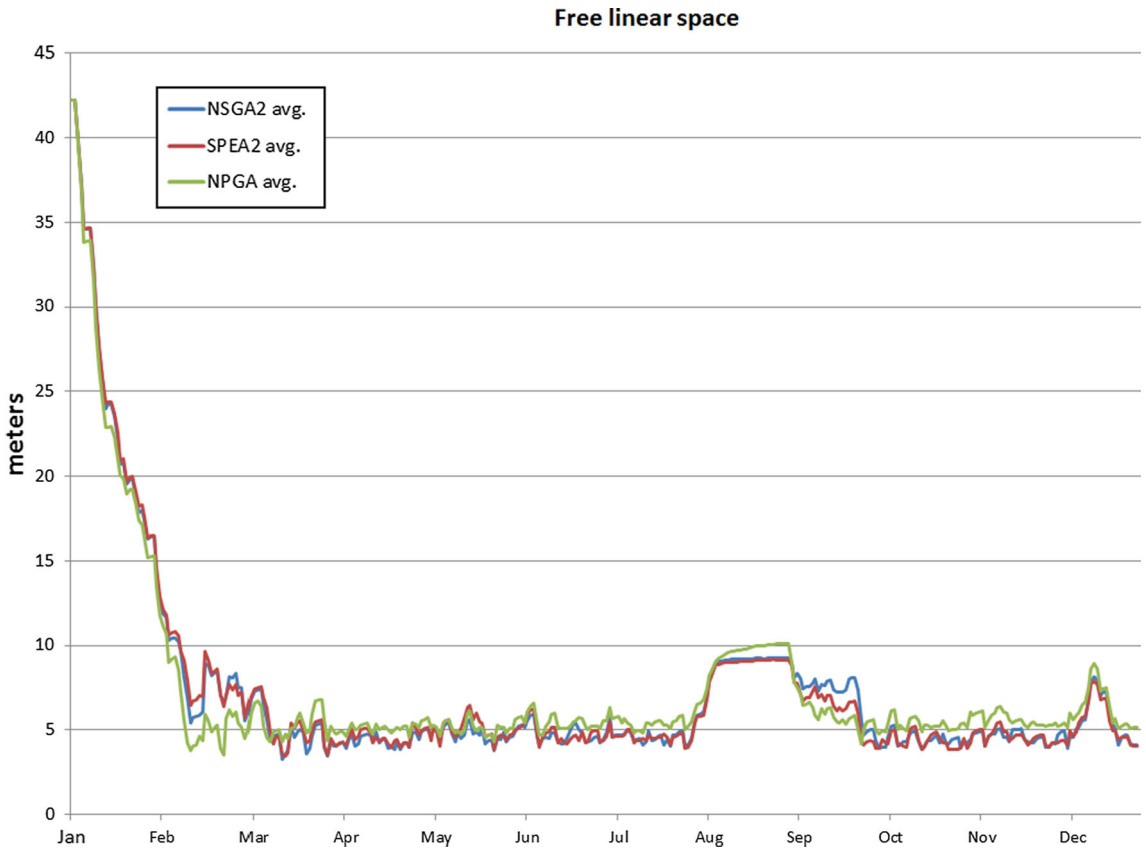

Fig. 4 Free linear space

Another surprising result is reached using the strategy based on the algorithm NSGA2 considering the maximum value of TSW that gives an increase of 1077 ton/day (+11 %) with respect to the highest recorded value. Therefore NSGA2 appears to be the most effective algorithm in order to favour the full occupation of the available space in the warehouse. This conclusion, however, needs to be further validated by assessing whether the results provided by the NSGA2 are actually significantly different with respect to those provide by NPGA and SPEA2 and to this aim some statistical test have been performed, which are described in the following section statistical tests.

Figure 6 compares the trends of TSW of a single run of each examined algorithm along the considered year of production.

Statistical tests

In order to evaluate the statistical significance of the results that have been applied through the three considered GA-based multi-objective optimization strategies, appropriate statistical tests have been developed, which are introduced in Garcia et al. (2009). In particular, the statistical analysis has been focused on the TSW, as such KPI is considered the most relevant one in the long term by the personnel of the company where the automated warehouse is located. The purpose of the applied tests is to assess how much statistically significant the difference between the results provided by NSGA2 and the results provided by the other strategies is. In fact, despite the results shown in Fig. 6 may seem to privilege SPEA2 and NSGA2, due to the intrinsic non-deterministic nature of the evolutionary procedure, a final verification stage is needed to prove that the better performance depicted by the previous tables and figures are not obtained "by chance" or through a specific and not repeatable series of trials. All statistical tests have been performed through the R software (R Development Core Team 2008) assuming a critical significance value $\alpha = 0.05$. As the Sheskin's conditions (Sheskin 2003) for the adoption of parametric statistical tests are not verified, non-parametric tests must be exploited.

In particular in the present work the method proposed in Friedman (1937) has been applied, which is one of the most employed non-parametric tests and is widely adopted in literature. This procedure requires the creation of a double entry table with a number of rows equal to the number of the performed tests (here $N_r = 352$) and a number of columns N_c equal to the number of algorithms to compare (here $N_c = 3$). For each row (i.e. for each repetition of the test) the perfor-

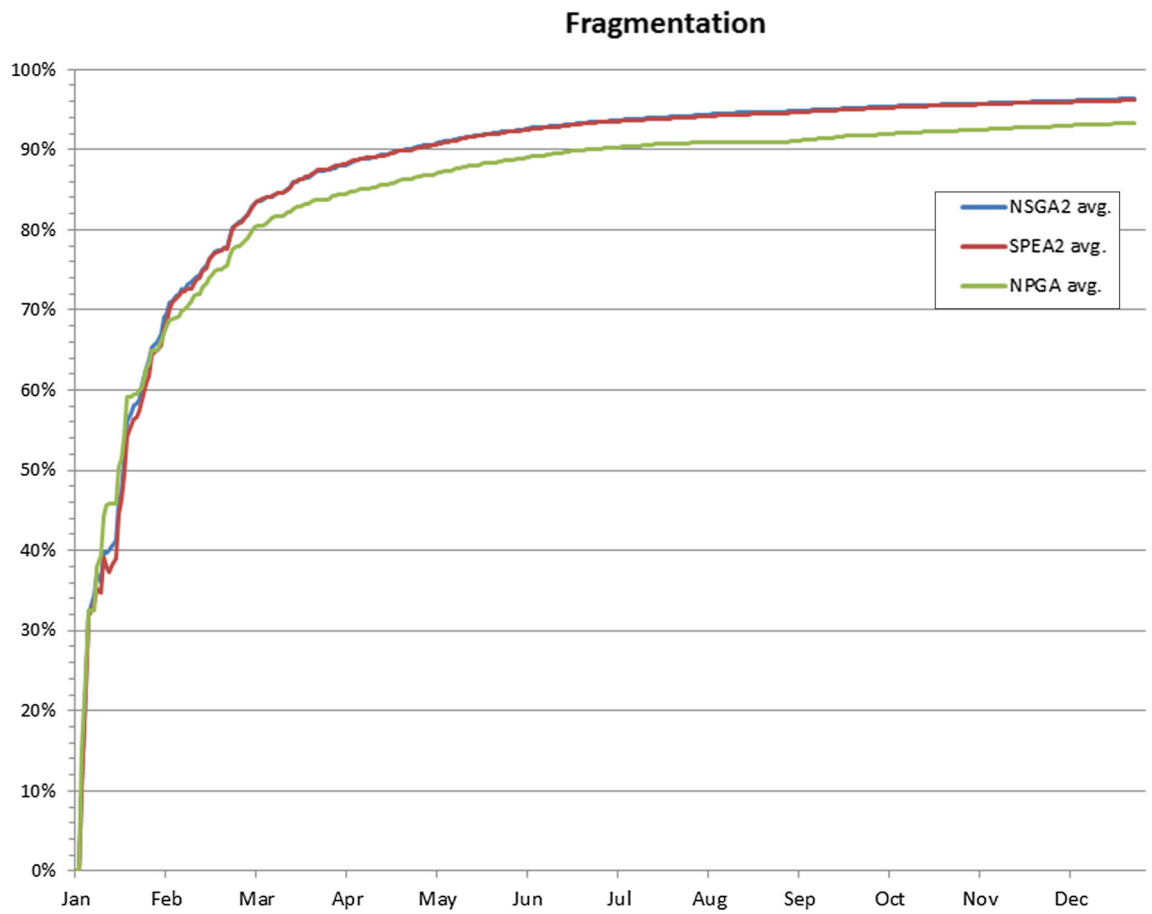

Fig. 5 Fragmentation

Table 1 Average KPI values obtained through the NPGA strategy

KPI	Value	Unit	Versus HS %	Versus FuzGA %	Versus SPGA %
$\overline{TSW}$	8150.75	Ton/day	+24.18	+11.29	−9.18
TSW_{max}	9715.58	Ton	+11.88	+6.48	−5.12
$\overline{F}$	84.70	%	−5.36	−4.28	−5.80
$\overline{ES}$	74.76	m	−22.91	+7.38	+6.61

Table 2 Average KPI values obtained through the SPEA2 strategy

KPI	Value	Unit	Versus HS %	Versus FuzGA %	Versus SPGA %
$\overline{TSW}$	9312.73	Ton/day	+41.88	+27.15	+3.75
TSW_{max}	11,031.0	Ton	+27.03	+20.90	+7.72
$\overline{F}$	87.39	%	−2.35	−1.24	−2.81
$\overline{ES}$	72.83	m	−24.09	+4.61	+4.08

mances of the procedures are ranked by assigning them an integer score $1, \ldots, N_c$, where the unitary value is assigned to algorithm that provided the worst result in the associated test. Finally the sum of the ranks T_j $(1 \leq j \leq N_c)$ is computed for each column.

If the three algorithms provide similar results, the values of the sum of the ranks T_j should be similar $\forall j$ end equal to an expected value $E\{T\}$ depending on the number of tests, which can be calculated as follows:

Table 3 Average KPI values obtained through the NSGA2 strategy

KPI	Value	Unit	Versus HS %	Versus FuzGA %	Versus SPGA %
$\overline{TSW}$	9426.94	Ton/day	+43.62	+28.71	+5.03
TSW_{max}	11,317.6	Ton	+30.31	+24.04	+10.52
$\overline{F}$	87.63	%	−2.09	−0.97	−2.54
$\overline{ES}$	72.82	m	−24.91	+4.59	+4.17

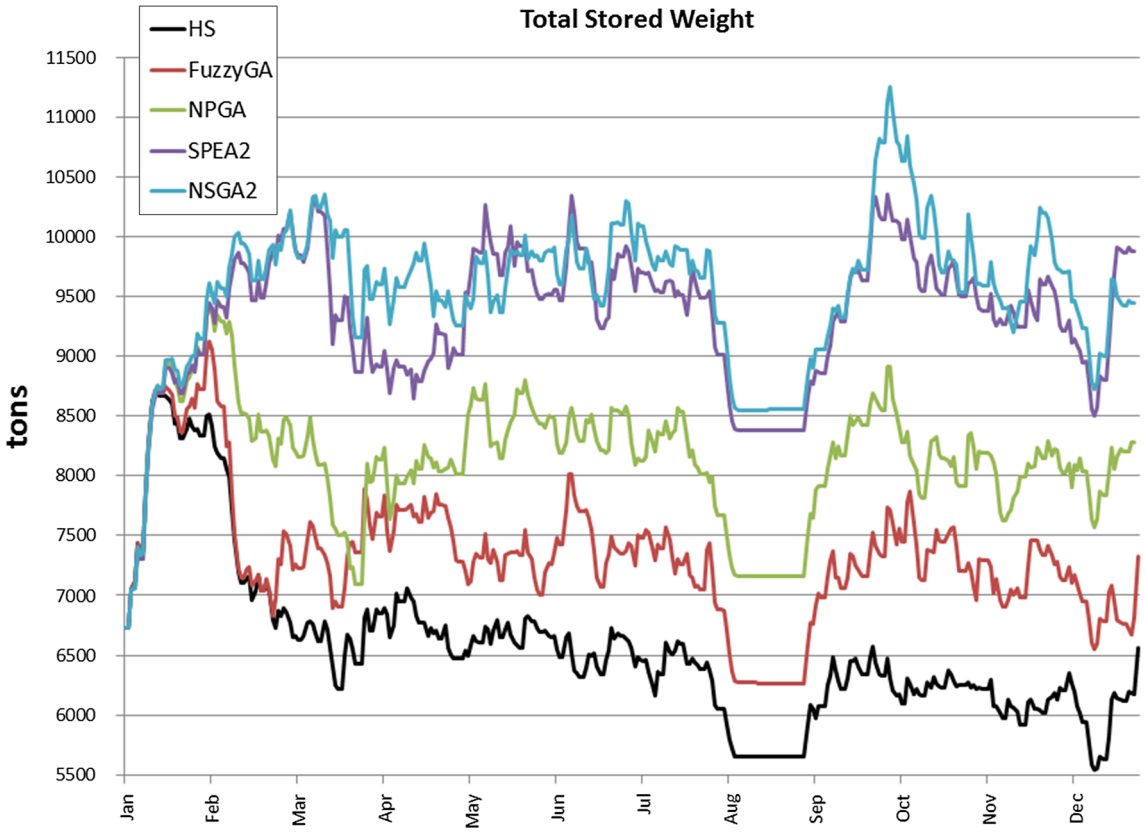

Fig. 6 Total stored material weigth considering the three strategies

$$E\{T\} = \frac{N_r(N_c+1)}{2} \tag{12}$$

where N is the number of rows. In the present case $E\{T\} = 704$.

The fact that the actual values of results are T_j $(1 \leq j \leq N_c)$ are very different from $E\{T\}$ means that in at least one column lower or higher ranks are concentrated, i.e. the corresponding algorithm performs significantly worse or better than the other ones. Table 4 shows the T_j values obtained in the developed test by the three considered GA-based multi-objective optimization algorithms: the highest value corresponds to the NSGA2 algorithm while the lowest value is associated to the NPGA. The results clearly show that NSGA2 outperforms the other ones.

In order to establish if the calculated values of T_j $(1 \leq j \leq N_c)$ are actually significantly different from $E\{T\}$, the index χ^2_{FR} is computed, which is distributed approximately as the index χ^2 with $N_c - 1$ degrees of freedom (see Eq. (13)):

Table 4 Results obtained performing Friedman test

Algorithm	Ranks sum	Average rank
NPGA	358	1.0170
SPEA2	746	2.1193
NSGA2	1008	2.8636

$$\chi^2_{FR} = \frac{12}{N_r N_c(N_c+1)} \sum_{j=1}^{N_c} \left(T_j - \frac{N_r(N_c+1)}{2}\right)^2 \tag{13}$$

The evaluated index tends to zero if the observed values of T_j are similar to the expected value. In order to state that the

Table 5 Results obtained performing Wilcoxon test

Algorithm	Wilcoxon value	p value
NPGA	117,821.5	$2.2e-16$
SPEA2	71,028.5	0.0004495

observed values of T_j are significantly different from $E\{T\}$, the computed value of χ^2_{FR} is compared to tabulated threshold values, which depend on the desired level of significance α and the number of degrees of freedom. In the present case $\chi^2_{FR} = 624.11$ which is higher than the tabulated threshold value for $\alpha = 0.05$ and 2 degrees of freedom, thus the conclusion is reached that the results obtained with the three algorithms are significantly different from each other.

However, in order to state that the best results are those obtained with the NSGA2, the Friedman algorithm is not sufficient and also another test, called Wilcoxon test, has been performed. This test is used to directly compare the performance of NSGA2 with respect to another one according to their ranking. Wilcoxon test provides the so called *p-value*, i.e. the probability that the difference in the performance achieved by the two algorithms is obtained by chance and is not statistically significant. Such value must be compared to the selected significance level α: if it is lower than α, it means that the performance difference is statistically meaningful, i.e. the selected algorithms actually outperforms the other one it is compared to. Table 5 shows the results obtained by comparing NSGA2 to the other two algorithms through the Wilcoxon test.

As shown in the table, both the p are below the significance threshold $\alpha = 0.05$, which confirms the goodness of NSGA2.

Finally, in order to further prove this statement, we calculated the Family Wise Error Rate (FWER) which is defined as the probability of obtaining one or more false results during the tests. The FWER is calculated as follows:

$$FWER = 1 - \prod_{i=1}^{N_A} (1 - p_i) \tag{14}$$

where N_A identifies the number of algorithms that are compared with NSGA2 (here $N_A = 2$) and p_i is the p_{value} obtained by the Wilcoxon test. In the present case $FWER = 4.4950e^{-4}$, that is a value much lower than the significance threshold $\alpha = 0.05$.

Thus the final conclusion can be reached that, in the MOOP related to the considered warehouse allocation strategies, the NSGA2 algorithm outperforms the other tested strategies (NPGA and SPEA2).

Conclusions

In this paper three different solutions based on GAs are applied to the multi-objective optimization of storage strategies for an existing warehouse within a steelmaking industry. The results have been presented and widely discussed in order to select the best algorithm. All the three proposed evolutionary approaches significantly outperform the heuristic procedure which was originally applied in the warehouse as well as a previously developed algorithm which merged the different objectives into a single one in order to apply a Single-Objective Optimization procedure. In particular, the NSGA2 allows to improve the Total Stored Weight of more than 43 % in average with respect to the heuristic method and of more than 28 % with respect to the Fuzzy approach. Also the other objectives are highly improved.

Such results have also been assessed by means of statistical tests which provides proof of the significance of the difference among the performance achieved by the different optimization strategies. In fact, the obtained results prove that NSGA2 provides the best results compared to the other implemented algorithms.

Ongoing and future work includes the application to the same case study and the comparison of other evolutionary algorithms which are not based on GAs, such as, for instance, Particle Swarm Optimization (PSO).

Acknowledgements The authors wish to gratefully acknowledge ILVA Racconigi Works for providing data and relevant information and for the fruitful discussions, which significantly contributed to the present analysis.

References

Amuso, V. J., & Enslin, J. (2007). The strength pareto evolutionary algorithm 2 (SPEA2) applied to simultaneous multimission waveform design. *Waveform Diversity and Design*, *1*, 407–417.

Bandyopadhyay, S., & Bhattacharya, R. (2013). Applying modified nsga-ii for bi-objective supply chain problem. *Jounal of Intelligent Manufacturing*, *24*(4), 707–716.

Bortolini, M., Botti, L., Cascini, A., Gamberi, M., & Mora, C. (2014). Multi-objective assignment strategy for warehouses served by automatic storage and retrieval system. In *12th International conference on industrial logistics*, ICIL (pp. 127–134).

Borup, L., & Parkinson, A. (1992). Comparison of four non-derivative optimization methods on two problems contaning heuristic and analythic knowledge. In D. A. Hoeltzel (Ed.), *Advances in design automation*. New York: The American Society of Mechanical Engineers.

Cateni, S., Colla, V., & Vannucci, M. (2010). Variable selection through genetic algorithms for classification purposes. In *10th IASTED international conference on artificial intelligence and applications*, AIA2010, vol 1, (pp. 6–11).

Coello, C. A., Veldihuizen, D. A. V., & Lamont, G. B. (2002). *Evolutionary algorithms for solving multi-objective problems*. New york: Kluwer Academic Publishers.

Colla, V., Bioli, G., & Vannucci, M. (2008a). Model parameter optimisation for an industrial application: A comparison between traditional approaches and genetic algorithms. In *Proceedings—EMS 2008, European modelling symposium, 2nd UKSim Euro-*

pean Symposium on Computer Modelling and simulation art no 4625243, vol 1, (pp.34–39).

Colla, V., Nastasi G., & Matarese, N. (2010a). Gadf - genetic algorithms for distribution fitting. In *10th International conference on intelligent systems design and applications*, ISDA'10 art.no. 5687298, vol 1, (pp. 6–11).

Colla, V., Nastasi, G., Matarese, N., & Reyneri, L. (2009). Ga-based solutions comparison for storage strategies optimization for an automated warehouse. In *9th International conference on intelligent systems design and applications*, ISDA'09 (art.no.5364423), vol 1, (pp. 976–981).

Colla, V., Nastasi, G., Matarese, N., & Ucci, A. (2008b). Simulation of an automated warehouse for steel tubes. In *10th International conference on computer modelling and simulation, EUROSIM/UKSim2008*, vol 1, (pp. 150–155).

Colla, V., Nastasi, G., Matarese, N., & Reyneri, L. (2010b). Ga-based solutions comparison for warehouse storage optimization. *International Journal of Hybrid Intelligent Systems*, *7*, 283–297.

Deb, K. (1989). Genetic algorithms in multimodal function optimization. Master's thesis, University of Alabama, mS Thesis TCGA, report N 89002.

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA2. *IEEE Transactions on Evolutionary Computation*, *6*, 182–197.

Figueiredo, J., Oliveira, J., Dias, L., & Pereira, G. (2012). A genetic algorithm for the job shop on an ASRS warehouse. In *Lecture notes in computer science LNCS 7335 Part, 3*, (pp. 133–146).

Fonseca, C., & Fleming, P. J. (1998). Multiobjective optimization and multiple constraint handling with evolutionary algorithms - Part II: application example. *IEEE Transactions on Systems, Man and Cybernetics Part A: Systems and Humans*, *28*, 38–47.

Friedman, M. (1937). The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of American Statistical Association*, *32*, 675–701.

Garcia, S., Molina, D., Lozano, M., & Herrera, F. (2009). A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour. CEC'2005 special session on real parameter optimization. *Journal of Heuristics, 15*(6), 617–644.

Goldberg, D., & Richardson, J. (1987). Genetic algorithms with sharing for multi-modal function optimization. In *2nd International conference on genetic algorithm*, vol 1, (pp. 41–49).

Goldberg, D. (1989). *Genetic Algorithms in search, optimization and Machine Learning*. Boston, MA: Addison Wesley.

Gu, J., Goetschalckx, M., & McGinnis, L. (2007). Research on warehouse operation: A comprehensive review. *European Journal of Operational Research*, *177*(1), 1–21.

Guo, W., & Jin, H. (2014). Analysis and research of the automated warehouse management and control system. *Applied Mechanics and Materials*, *511–512*, 1095–1098.

Han, B., Zhang, W., Lu, X., & Lin, Y. (2015). On-line supply chain scheduling for single-machine and parallel-machine configurations with a single customer: Minimizing the makespan and delivery cost. *European Journal of Operational Research*, *244*(3), 704–714.

Hiremath, N., Sahu, S., & Tiwari, M. (2013). Multi objective outbound logistics network design for a manufacturing supply chain. *Journal of Intelligent Manufacturing*, *24*(6), 1071–1084.

Holland, H. J. (1975). *Adaptation in natural and artificial systems, an introductory analysis with application to biology, control and artificial intelligence*. Ann Arbor, Michigan: University of Michigan Press.

Horn, J., Nafpliotis, N., & Goldberg, D. E. (1994). A niched Pareto genetic algorithm for multiobjective optimization. In *1st IEEE conference on computational intelligence*, vol 1, (pp. 82–87).

Kroo, I., McMasters, J., & Smith, S. (2000). Highly nonplanar lifting systems. In *Presented at transportation beyond 2000: Technologies needed for engineering design*, NASA Langley Research Center.

Krus, P., Jansson, A., Berry, P., Hannson, E., & Ovrebo, K. (1996). A generic model concept for optimization in preliminary design. In Presented at world aviation congress and exposition, Los Angeles, California, USA.

Li, T., Wu, J. (2007). An improved genetic algorithm on logistics delivery in e-business. In *Natural computation, 2007. ICNC 2007. Third international conference on*, vol 3, (pp. 765–769).

Liu, Q., & Xu, J. (2011). A study on facility location-allocation problem in mixed environment of randomness and fuzziness. *Journal of Intelligent Manufacturing*, *22*(3), 389–398.

Lu, J., Yang, F., & Wang, L. (2011). Multi-objective rule discovery using the improved niched pareto genetic algorithm. In *3rd IEEE Conference on measuring technology and mechatronics automation*, vol 2, (pp. 657–661).

Ming-bao, P., Guo-guang, H., & Ling, X. (2007). Optimal number and sites of regional logistics centers by genetic algorithm and fuzzy c-mean clustering. In *Service systems and service management, 2007 international conference on*, vol 1, (pp. 1–5).

Osyczka, A. (1985). Multicriteria optimization for engineering design. *Design Optimization*, *1*, 193–217.

Popović, D., Vidović, M., & Bjelić, N. (2014). Application of genetic algorithms for sequencing of as/rs with a triple-shuttle module in class-based storage. *Flexible Services and Manufacturing Journal*, *26*(3), 432–453.

R Development Core Team (2008). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria, http://www.R-project.org, ISBN 3-900051-07-0.

Roodbergen, K. J., & Vis, I. F. (2009). A survey of literature on automated storage and retrieval systems. *European Journal of Operational Research*, *194*(2), 343–362.

Rudolph, G. (1999). Evolutionary search under partially ordered sets. Tech. rep., Dept. Comput. Sci/LS11, University of Dortmund, Germany, dept. Comput. Sci/LS11, University of Dortmund, Germany. Tech. rep. CI-67/99.

Sheskin, D. (2003). *Handbook of parametric and nonparametric statistical procedures*. New York, USA: Chapman and Hall/CRC.

Srinivas, N., & Deb, K. (1995). Multiobjective function optimization using nondominated sorting genetic algorithms. *IEEE Transactions on Evolutionary Computation*, *2*, 221–248.

Takeyasu, K., & Kainosho, M. (2014). Optimization technique by genetic algorithms for international logistics. *Journal of Intelligent Manufacturing*, *25*(5), 1043–1049.

Tami, M., & Abido, M. A. (2011). Multiobjective optimal power flow using improved strength pareto evolutionary algorithm (spea2). In *11th International conference on intelligent systems design and applications (ISDA)*, vol 1, (pp. 1097–1103).

van den Berg, J., & Zijm, W. (1999). Models for warehouse management: Classification and examples. *International Journal of Production Economic*, *59*, 519–528.

Zitzler, E., Laumanns, M., & Thiele, L. (2001). SPEA2: Improving the Strength Pareto Evolutionary Algorithm. TIK-Report103 1:1–20.

Zitzler, E., Deb, K., & Thiele, L. (2000). Comparison of multiobjective evolutionary algorithms: Empirical results. *IEEE Transactions on Evolutionary Computation*, *8*, 173–195.

Zitzler, E., Laumanns, M., & Thiele, L. (1999). Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach. *IEEE Transactions on Evolutionary Computation*, *3*, 257–271.

Zitzler, E., & Thiele, L. (1999). Multi-objective evolutionary algorithms: A comparative case study and strength pareto approach. *Transactions on Evolutionary Computation*, *3*, 257–271.