Computers & Industrial Engineering 61 (2011) 103–117

Contents lists available at ScienceDirect

Computers & Industrial Engineering

journal homepage: www.elsevier.com/locate/caie

An advanced multiobjective genetic algorithm design for the time and space assembly line balancing problem

Manuel Chica [a], Óscar Cordón [a,b,*], Sergio Damas [a]

[a] European Centre for Soft Computing, 33600 Mieres, Spain
[b] Dept. Computer Science and Artificial Intelligence, E.T.S. Informática y Telecomunicaciones, 18071 Granada, Spain

ARTICLE INFO

Article history:
Received 30 August 2010
Received in revised form 21 December 2010
Accepted 4 March 2011
Available online 11 March 2011

Keywords:
Assembly lines
Time and space assembly line balancing
Evolutionary multiobjective optimization
NSGA-II

ABSTRACT

Time and space assembly line balancing considers realistic multiobjective versions of the classical assembly line balancing industrial problems involving the joint optimization of conflicting criteria such as the cycle time, the number of stations, and/or the area of these stations. In addition to their multi-criteria nature, the different problems included in this field inherit the precedence constraints and the cycle time limitations from assembly line balancing problems, which altogether make them very hard to solve. Therefore, time and space assembly line balancing problems have been mainly tackled using multiobjective constructive metaheuristics. Global search algorithms in general – and multiobjective genetic algorithms in particular – have shown to be ineffective to solve them up to now because the existing approaches lack of a proper design taking into account the specific characteristics of this family of problems. The aim of this contribution is to demonstrate the latter assumption by proposing an advanced multiobjective genetic algorithm design for the 1/3 variant of the time and space assembly line balancing problem which involves the joint minimization of the number and the area of the stations given a fixed cycle time limit. This novel design takes the well known NSGA-II algorithm as a base and considers the use of a new coding scheme and sophisticated problem specific operators to properly deal with the said problematic questions. A detailed experimental study considering 10 different problem instances (including a real-world instance from the Nissan plant in Barcelona, Spain) will show the good yield of the new proposal in comparison with the state-of-the-art methods.

1. Introduction

An assembly line is made up of a number of workstations, arranged either in series or in parallel. These stations are linked together by a transport system that aims to supply materials to the main flow and move the production items from one station to the next one. Since the manufacturing of a production item is divided into a set of tasks, a usual and difficult problem is to determine how these tasks can be assigned to the stations fulfilling certain restrictions. The aim is to get an optimal assignment of subsets of tasks to the stations of the plant. Moreover, each task requires an operation time for its execution which is determined as a function of the manufacturing technologies and the employed resources.

A family of academic problems – called simple assembly line balancing problem (SALBP) – was proposed to model this situation (Baybars, 1986; Scholl, 1999). Taking this family as a base and adding spatial information to enrich the problem, Bautista and Pereira recently proposed a more realistic framework: the time and space assembly line balancing problem (TSALBP) (Bautista & Pereira, 2007). It emerged due to the study of the specific characteristics of the Nissan automotive plant located in Barcelona, Spain. Hence, this framework considers an additional space constraint to become a simplified version of real-world problems. In addition, TSALBP formulations have a multi-criteria nature as many real-world problems. These formulations involve minimising three conflicting objectives: the cycle time of the assembly line, the number of stations, and their area. One of these formulations is the TSALBP-1/3 variant which tries to minimise the number and the area of the stations for a given product cycle time. This is a very usual situation in real-world factories as the said Nissan automotive plant where the annual production is usually set by market objectives.

One of the most important aspects in TSALBP-1/3 is the set of constraints, including the set of tasks precedences and the cycle time limitation for each station. Since constructive metaheuristics such as ant colony optimization (ACO) (Dorigo & Stützle, 2004) have a good capability to deal with constrained combinatorial optimization problems, they have demonstrated to be more appropriate than non constructive procedures (Glover & Kochenberger, 2003) to solve the TSALBP-1/3 up to now. Specifically, in Chica,

* Corresponding author. Tel.: +34 985 456545; fax: +34 985 456699.
E-mail addresses: manuel.chica@softcomputing.es (M. Chica), oscar.cordon@softcomputing.es (Ó. Cordón), sergio.damas@softcomputing.es (S. Damas).

0360-8352/$ - see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cie.2011.03.001

Cordón, Damas, Bautista, and Pereira (2008a, 2010) the authors proposed the use of a multiobjective ACO algorithm based on the multiple ant colony system (MACS) (Barán & Schaerer, 2003) for this problem. The MACS algorithm obtained the best results in comparison with a multiobjective random search, a multiobjective randomised greedy algorithm, and a multiobjective genetic algorithm (Chica et al., 2010). In particular, the latter method – a multiobjective extension of an existing genetic algorithm for SALBP (Sabuncuoglu, Erel, & Tayner, 2000) based on the use of the well-known NSGA-II (Deb, Pratap, Agarwal, & Meyarivan, 2002), the state-of-the-art evolutionary multiobjective optimization (EMO) algorithm – showed a very low performance.

Although single and multiobjective genetic algorithms have been successfully applied to many different industrial engineering problems as supply chain optimization, job shop scheduling, plant design, and packing and distribution (Altiparmak, Gen, Lin, & Paksoy, 2006; Dietz, Azzaro-Pantel, Pibouleau, & Domenech, 2008; Gao, Gen, Sun, & Zhao, 2007; Leung, Wong, & Mok, 2008) – and even to assembly and disassembly line balancing (Kim, Kim, & Kim, 1996; McGovern & Gupta, 2007; Simaria & Vilarinho, 2004) – the fact that genetic algorithms require careful designs in order to deal with constrained optimization problems is well known (Michalewicz, Dasgupta, Riche, & Schoenauer, 1996; Santana-Quintero, Hernández-Díaz, Molina, Coello, & Caballero, 2010). Hence, the weak performance of the latter multiobjective genetic algorithm when solving the TSALBP-1/3 was due to its inability to deal with the inherent problem characteristics and not to any drawback related to the EMO approach followed. In fact, EMO could be a powerful tool to accurately solve this very complex problem.

Therefore, in this contribution a new design of a multiobjective genetic algorithm is developed, also based on NSGA-II but incorporating specific components to appropriately deal with the TSALBP constraints. On the one hand, a new individual representation will be proposed which is more faithful to the solution phenotype and thus more appropriate for the problem solving. On the other hand, novel crossover, repair, and mutation operators will be designed to overcome the non constructive nature of genetic algorithms when dealing with the TSALBP constraints. Finally, a diversity induction mechanism will be incorporated to obtain well spread Pareto fronts.

Different variants of the proposed EMO algorithm design, based on the use of only some of the latter components, will be considered to ensure the actual need of the cooperative action of all of them in order to achieve the best performance. The resulting variants of the algorithm will be compared among them and the best performing ones will be benchmarked with the existing multiobjective genetic algorithm and the state-of-the-art algorithm to solve the problem, MACS-TSALBP-1/3. We will consider nine well-known problem instances from the literature for this experimental study. Furthermore, the algorithms will be applied to a real-world problem instance from the Nissan industry plant in Barcelona. In order to evaluate the performance of the different methods, a detailed analysis of results will be developed considering the usual multiobjective performance indicators (metrics).

This paper is structured as follows. In Section 2, the formulation of the TSALBP-1/3 and the existing methods to solve it, i.e. the MACS algorithm, a multiobjective randomised greedy algorithm, and the multiobjective extension of the genetic algorithm for SALBP, are reviewed. Then, our novel multiobjective genetic algorithm design for the problem is described in Section 3. The used performance indicators and problem instances, the developed experiments, and the analysis of the obtained results to test the performance of the different algorithms are presented in Section 4. Finally, in Section 5, some concluding remarks and proposals for future work are provided.

2. Preliminaries

This section is devoted to describe some required preliminaries to properly understand the work developed in this contribution. First, the formulation of the TSALBP-1/3 is introduced. Then, the composition of the different metaheuristic methods which have been proposed in the literature to tackle this complex industrial engineering problem is briefly reviewed.

2.1. The time and space assembly line balancing problem

The manufacturing of a production item is divided into a set V of n tasks. Each task j requires a positive operation time t_j for its execution. This time is determined as a function of the manufacturing technologies and the resources employed. Each task j can be only assigned to a single station k. A subset of tasks S_k ($S_k \subseteq V$) is thus assigned to each station k ($k = 1,2,\ldots,m$). They are referred as its workload.

Every task j has a set of "preceding tasks" P_j which must be accomplished before starting that task. These constraints are represented by an acyclic precedence graph, whose vertices correspond to the tasks and where a directed arc $\langle i,j\rangle$ indicates that task i must be finished before starting task j on the production line. Thus, task j cannot be assigned to a station that is before the one where task i was assigned.

Each station k presents a station workload time $t(S_k)$ that is equal to the sum of the tasks' duration assigned to it. In general, the SALBP (Baybars, 1986; Scholl, 1999) focuses on grouping these tasks into workstations by an efficient and coherent method. In short, the goal is to achieve a grouping of tasks that minimises the inefficiency of the line or its total downtime satisfying all the constraints imposed on the tasks and stations.

On the other hand, there is a real need of introducing space constraints in the assembly lines' design because of two main reasons: (a) the length of the workstation is limited in the majority of the situations, and (b) the required tools and components to be assembled should be distributed along the sides of the line. Based on these realistic features, a new real-like problem comes up.

In order to model it, Bautista and Pereira (2007) extended the SALBP into the TSALBP by means of the following formulation: the area constraint must be considered by associating a required area a_j to each task j. We can see in Fig. 1 the graph of the first eight tasks of the real-world instance of Nissan. Each task has a time and area information. The arcs denote the precedence relations between the different tasks. For instance, task 4 requires an area of 1 unit, an operation time of 60, and it cannot start before tasks 1 and 5 finish.

Apart from the area of the tasks, every station k will require a station area $a(S_k)$, equal to the sum of the areas of all the tasks assigned to that station. This needed area must not be larger than the available area A_k of the station k. For the sake of simplicity, A_k is assumed to be identical for all the stations and denoted by A, where $A = \max_{k=1,2,\ldots,m} A_k$.

Overall, the TSALBP may be stated as: given a set of n tasks with their temporal and spatial attributes, t_j and a_j, and a precedence graph, each task must be assigned to just one station such that:

1. all the precedence constraints are satisfied,
2. there is not any station with a workload time $t(S_k)$ greater than the cycle time c,
3. there is not any station with a required area $a(S_k)$ greater than the global available area A.

The TSALBP presents different formulations depending on which of the three considered parameters (c, the cycle time; m, the number of stations; and A, the area of the stations) are tackled

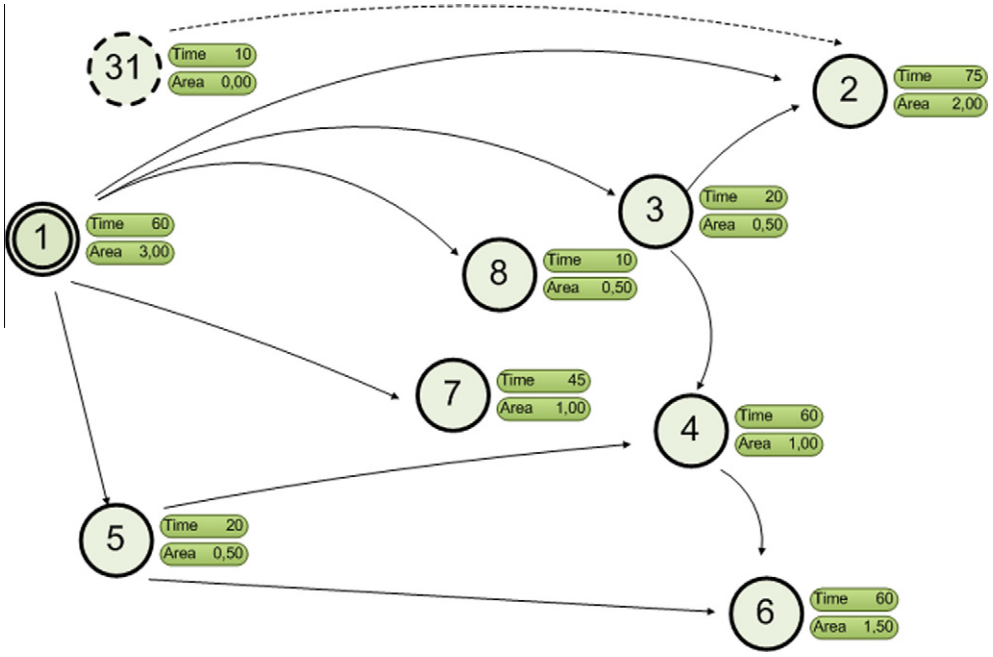

Fig. 1. A precedence graph which represents the first 8 tasks of the real-world instance of Nissan. Time and area information are shown next to each task. Task 31 is also shown because of its precedence relation with respect to task 2.

as objectives to be optimised and which others are provided as fixed variables. The eight possible combinations result in eight different TSALBP variants. Within them, there are four multiobjective variants depending on the given fixed variable: c, m, A, or none of them. While the former three cases involve a bi-objective problem, the latter defines a three-objective problem.

We will tackle one of these formulations, the TSALBP-1/3. It consists of minimising the number of stations m and the station area A, given a fixed value of the cycle time c. We chose this variant because it is quite realistic in the automotive industry, our field of interest, since the annual production of an industrial plant (and therefore, the cycle time c) is usually set by market objectives. Besides, the search for the best number of stations and area makes sense if the goal is reducing costs and make workers' day better by setting up less crowded stations. More information about the justification of the choice can be found in Chica et al. (2010).

2.2. Mathematical formulation of the TSALBP-1/3

According to the TSALBP formulation (Bautista & Pereira, 2007), the 1/3 variant deals with the minimization of the number of stations, m, and the area occupied by those stations, A, in the assembly line configuration. We can mathematically formulate this TSALBP variant as follows:

$$Minf^0(x) = m = \sum_{k=1}^{UB_m} \max_{j=1,2,\ldots,n} x_{jk}, \tag{1}$$

$$f^1(x) = A = \max_{k=1,2,\ldots,UB_m} \sum_{j=1}^{n} a_j x_{jk} \tag{2}$$

subject to:

$$\sum_{k=E_j}^{L_j} x_{jk} = 1, \quad j = 1,2,\ldots,n \tag{3}$$

$$\sum_{k=1}^{UB_m} \max_{j=1,2,\ldots,n} x_{jk} \leqslant m \tag{4}$$

$$\sum_{j=1}^{n} t_j x_{jk} \leqslant c, \quad k = 1,2,\ldots,UB_m \tag{5}$$

$$\sum_{j=1}^{n} a_j x_{jk} \leqslant A, \quad k = 1,2,\ldots,UB_m \tag{6}$$

$$\sum_{k=E_i}^{L_i} k x_{ik} \leqslant \sum_{k=E_j}^{L_j} k x_{jk}, \quad j = 1,2,\ldots,n; \quad \forall i \in P_j \tag{7}$$

$$x_{jk} \in \{0,1\}, \quad j = 1,2,\ldots,n; k = 1,2,\ldots,UB_m \tag{8}$$

where:

- n is the number of tasks,
- x_{jk} is a decision variable taking value 1 if task j is assigned to station k, and 0 otherwise,
- a_j is the area information for task j,
- UB_m is the upper bound for the number of stations m,
- E_j is the earliest station to which task j may be assigned,
- L_j is the latest station to which task j may be assigned,
- UB_m is the upper bound of the number of stations. In our case, it is equal to the number of tasks, and

Constraint in Eq. (3) restricts the assignment of every task to just one station, (4) limits decision variables to the total number of stations, (5) and (6) are concerned with time and area upper bounds, (7) denotes the precedence relationship among tasks, and (8) expresses the binary nature of variables x_{jk}.

2.3. Previous approaches for the TSALBP-1/3

The specialised literature includes a large variety of exact and heuristic problem-solving procedures as well as metaheuristics for solving the SALBP (Scholl & Voss, 1996, 2006). Among them, the use of genetic algorithms (Sabuncuoglu et al., 2000; Anderson & Ferris, 1994; Kim, Kim, & Kim, 2000, 2009), tabu search (Chiang, 1998), simulating annealing (Heinrici, 1994), and ant colony optimization (Bautista & Pereira, 2007; Blum, 2008) have been considered. Besides, multicriteria formulations of the SALBP have also been tackled using genetic algorithms (Leu, Matheson, & Rees, 1994), differential evolution (Nearchou, 2008), and ant colony optimization (McMullen & Tarasewich, 2006).

However, there are not many proposals for solving the multiobjective 1/3 variant of the TSALBP (Chica et al., 2010). Among them, the following can be found: (a) a MACS algorithm, (b) a multiobjective randomised greedy algorithm, and (c) a multiobjective extension of the SALBP genetic algorithm proposed in Sabuncuoglu et al. (2000). We briefly review these algorithms in the next three subsections, as two of them will be considered as baselines for our new proposal in the experimental study developed in Section 4.

2.3.1. The MACS algorithm for the TSALBP-1/3

MACS (Barán & Schaerer, 2003) was proposed as an extension of ant colony system (ACS) (Dorigo & Gambardella, 1997) to deal with multiobjective problems. The original version of MACS uses one pheromone trail matrix and several heuristic information functions. However, in the case of the TSALBP-1/3, the experimentation carried out in Chica et al. (2010) showed that the performance was better when MACS was only guided by the pheromone trail information. Therefore, the heuristic information functions were not used.

Since the number of stations is not fixed, the algorithm uses a constructive and station-oriented approach (Scholl, 1999) to face the precedence problem (as usually done for the SALBP, Scholl & Becker, 2006). Thus, the algorithm will open a station and select one task till a stopping criterion is reached. Then, a new station is opened to be filled and the procedure is iterated till all the existing tasks are allocated.

The pheromone information has to memorise which tasks are the most appropriate to be assigned to a station. Hence, a pheromone trail has to be associated to a pair $(station_k, task_j)$, $k = 1,\ldots,n$, $j = 1,\ldots,n$, with n being the number of tasks, so the pheromone trail matrix has a bi-dimensional nature. Since MACS is Pareto-based, the pheromone trails are updated using the current non-dominated set of solutions (Pareto archive). Two station-oriented single-objective greedy algorithms were used to obtain the initial pheromone value τ_0.

In addition, a novel mechanism was introduced in the construction procedure in order to achieve a better search diversification–intensification trade-off able to deal with the problem difficulties. This mechanism randomly decides when to close the current station taking as a base both a station closing probability distribution and an ant filling threshold α_i. The probability distribution is defined by the station filling rate (i.e., the overall processing time of the current set of tasks S_k assigned to that station) as follows:

$$p(closing\ k) = \frac{\sum_{i \in S_k} t_i}{c} \tag{9}$$

At each construction step, the current station filling rate is computed. In case it is lower than the ant's filling percentage threshold α_i (i.e., when it is lower than $\alpha_i \cdot c$), the station is kept opened. Otherwise, the station closing probability distribution is updated and a random number is uniformly generated in [0,1] to take the decision whether the station is closed or not. If the decision is to close the station, a new station is created to allocate the remaining tasks. Otherwise, the station will be kept opened. Once the latter decision has been taken, the next task is chosen among all the candidate tasks using the MACS transition rule to be assigned to the current station as usual. The procedure goes on till there is no more remaining task to be assigned.

Thus, the higher the ant's threshold, the higher the probability of a totally filled station, and *vice versa*. This is due to the fact that there are less possibilities to close it during the construction process. In this way, the ant population will show a highly diverse search behaviour, allowing the algorithm to properly explore the different parts of the optimal Pareto front by appropriately distributing the generated solutions.

The interested reader is referred to Chica et al. (2010) for a complete description of the MACS proposal for the TSALBP-1/3.

2.3.2. A multiobjective randomised greedy algorithm

A multiobjective randomised greedy algorithm for the TSALBP-1/3 was also proposed in Chica et al. (2010) based on a diversification generation mechanism which behaves similarly to a GRASP construction phase (Feo & Resende, 1995).

In Chica et al. (2010) randomness is introduced in two processes. On the one hand, allowing the selection of the next task to be assigned to the current station to be randomly taken among the best candidates. It starts by creating a candidate list of unassigned tasks. For each candidate task j, its heuristic value η_j is computed by measuring the preference of assigning it to the current opened station. η_j is proportional to the processing time and area ratio of that task (normalised with the upper bounds given by the time cycle, c, and the sum of all tasks' areas, respectively), as well as the ratio between the number of successors of task j and the maximum number of successors of any eligible task. Then, all the candidate tasks are sorted according to their heuristic values and a quality threshold is set for them, given by $q = max_{\eta_j} - \gamma \cdot (max_{\eta_j} - min_{\eta_j})$. All the candidate tasks with a heuristic value η_j greater or equal than q are selected to be in the restricted candidate list (RCL). In the former expression, γ is the diversification–intensification trade-off control parameter. When γ is equal to 1 a completely random choice is obtained, inducing the maximum possible diversification. In contrast, if $\gamma = 0$ the choice is close to a pure greedy decision, with a low diversification. Proceeding in this way, the RCL size is adaptive and variable, thus achieving a good diversification–intensification trade-off. In the last part of the construction step, a task is randomly selected among those of the RCL. The construction procedure finishes when all the tasks have been allocated in the needed stations.

On the other hand, randomness is also introduced in the decision of closing the current station. This is done according to a probability distribution given by the filling rate of the station (see Eq. (9)). The filling thresholds approach is also used to achieve a diverse enough Pareto front. A different threshold is selected in isolation at each iteration of the multiobjective randomised greedy algorithm, i.e., the construction procedure of each solution considers a different threshold. As a consequence, the algorithm uses the same constructive approach than the MACS algorithm, considering filling thresholds and closing probabilities at each construction step. The main difference is the probabilistic criterion to select the next task that will be included in the current station.

The algorithm is run a number of iterations to generate different solutions. The final output consists of a Pareto set approximation composed of the non-dominated solutions among them.

2.3.3. A multiobjective extension of a single-objective genetic algorithm for the SALBP

An extension of an existing single-objective genetic algorithm for the SALBP was proposed in Chica et al. (2010) to deal with the TSALBP-1/3. The authors chose the proposal introduced in Sabuncuoglu et al. (2000) and adapted it by means of the state-of-the-art multiobjective NSGA-II approach. In short, the features of this TSALBP-NSGA-II designed can be summarised as follows:

- Coding: the original order-based encoding scheme proposed in Sabuncuoglu et al. (2000) is considered. The length of the chromosome is equal to the number of tasks. The task-station assignment is implicitly encoded in the genotype and it is obtained by using a simple station-oriented constructive mechanism (Scholl, 1999) guided by fulfilling the available cycle time of each station. A station is opened and sequentially filled with the tasks listed in the chromosome order while the overall

processing time of the set of assigned tasks does not exceed the assembly line cycle time. Once there is not available time to place the next task in the current station, this station is closed and a new empty one is opened to assign the remaining tasks. The procedure stops when all the tasks are allocated.

- Initial population: it is randomly generated by assuring the feasibility of the precedence relations.
- Crossover: a kind of order preserving crossover (Goldberg, 1989; Bäck, Fogel, & Michalewicz, 1997) is considered to ensure that feasible offsprings are obtained satisfying the precedence restrictions. This family of order-based crossover operators emphasises the relative order of the genes from both parents. In our case, two different offspring are generated from the two parents to be mated, proceeding as follows. Two cutting points are randomly selected for them. The first offspring takes the genes outside the cutting points in the same sequence order as in the first parent. That is, from the beginning to the first cutting point and from the second cutting point to the end. The remaining genes, those located between the two cutting-points, are filled in by preserving the relative order they have in the second parent. The second offspring is generated the other way around, i.e. taking the second parent to fill in the two external parts of the offspring and the first one to build the central part. Notice that, preserving the order of the genes of the other parent in the central part will guarantee the feasibility of the obtained offspring solution in terms of precedence relations. The central genes also satisfy the precedence constraints with respect to those that are in the two external parts.
- Mutation: the same mutation operator considered in the original single-objective genetic algorithm (Sabuncuoglu et al., 2000), a scramble mutation, is used. A random cut-point is selected and the genes after the cut-point are randomly replaced (scrambled), assuring feasibility.
- Diversity: the similarity-based mating scheme for EMO proposed in Ishibuchi, Narukawa, Tsukamoto, and Nojima (2008) to recombine extreme and similar parents was used in this algorithm to try to improve the diversity and spread of the Pareto set approximations.

This NSGA-II design for the TSALBP-1/3 showed poor results in comparison with MACS (Chica et al., 2010). The Pareto front approximations generated showed a very low cardinality and converged to a narrow region located in the left-most zone of the objective space (i.e. solutions with small values of the number of stations, m). The latter fact is justified by the TSALBP-1/3 nature as a strongly constrained combinatorial optimization problem, which was not properly tackled by the global search algorithm considered (a multiobjective genetic algorithm) and by the basic order encoding used.

Nevertheless, in the next section we will propose an advanced EMO design able to overcome the problems of the latter basic multiobjective genetic algorithm and to successfully solve the TSALBP-1/3.

3. An advanced NSGA-II-based approach for the TSALBP-1/3

As said, the weak performance of the previous EMO algorithm (Section 2.3.3) when solving the TSALBP-1/3 cannot be explained because of the chosen multiobjective genetic algorithm. It is well known that NSGA-II has shown a large success when solving many different multiobjective numerical and combinatorial optimization problems (see Chapter 7 in Coello, Lamont, & Van Veldhuizen (2007) for a detailed review classified in different application areas). On the contrary, that weak behaviour was due to the inherent characteristics of the combinatorial optimization problem being solved. In principle, the use of global search procedures as genetic algorithms could be less appropriate than constructive metaheuristics to deal with the TSALBP-1/3 because of the hard constraints (precedence relations and stations' cycle time limitation). In addition, the representation used does not seem to be adequate because it is not a natural coding for the problem.

Hence, authors propose a novel design, based on the original NSGA-II search scheme (Deb et al., 2002) as well. However, a more appropriate representation and more effective operators are used to solve the TSALBP-1/3. From now on, the new algorithm will be referred as *advanced TSALBP-NSGA-II* because of its problem-specific design and potential application to other TSALBP variants. The previous method will be referred to as *basic TSALBP-NSGA-II* in order to stress the difference between both approaches. The main features and operators of the *advanced TSALBP-NSGA-II* are described in the next subsections.

3.1. Representation scheme

The most important problem of the *basic TSALBP-NSGA-II* method was the representation scheme, based on that usually considered by the existing genetic algorithm approaches for the SALBP. We should note that the SALBP is a single-objective problem and thus it is not strictly necessary to represent a solution as an assignment of tasks to stations to solve it. Instead, an order encoding is used to define a specific task ordering in a chromosome and the latter assignment is determined in a constructive fashion, as seen in Section 2.3.3.

However, the latter representation is not a good choice for the TSALBP-1/3. It carries the problem of biasing the search to a narrow area of the Pareto front (as demonstrated by the experimental results in Chica et al. (2010) and in the current contribution). Here is where our new proposal, the *advanced TSALBP-NSGA-II*, takes the biggest step ahead with respect to the existing basic algorithm. The new coding scheme introduced will explicitly represent task-station assignments regardless the cycle time of the assembly line, thus ensuring a proper search space exploration for the joint optimization of the number and the area of the stations. Furthermore, the representation will also follow an order encoding to facilitate the construction of feasible solutions with respect to the precedence relations constraints.

The allocation of tasks among stations is made by employing separators.[1] Separators are thus dummy genes which do not represent any specific task and they are inserted into the list of genes representing tasks. In this way, they define groups of tasks being assigned to a specific station. The maximum possible number of separators is $n-1$ (with n being the number of tasks), as it would correspond to an assembly line configuration with n stations, each one composed of a single task. Tasks are encoded using numbers in $\{1,\ldots,n\}$, as in the previous representation, while separators take values in $\{n+1,\ldots,2\cdot n-1\}$. Hence, the genotype is again an order-based representation. Fig. 2 shows an example of the new coding scheme.

The number of separators included in the genotype is variable and it depends on the number of existing stations in the current solution. Therefore, the algorithm works with a variable-length coding scheme, although its order-based representation nature avoids the need of any additional mechanism to deal with this issue. The maximum size of the chromosome is $2\cdot n-1$ to allow the presence of separators for the maximum number of possible stations. On the other hand, the representation scheme ensures the encoded solutions are feasible with respect to the precedence

[1] We should notice that, although this representation is not very extended, the use of separators in an order encoding was previously considered in a document clustering application (Robertson & Willett, 1994).

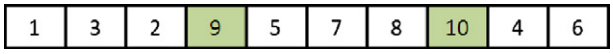

Fig. 2. Coding scheme example: for the first 8 tasks of the real-world instance of Nissan, a genotype representing three stations is represented, having 3, 3, and 2 tasks, respectively. Separators are those genes coloured.

relations constraints. However, the cycle time limitation could be violated and it will be a task of the genetic operators to ensure feasibility with respect to that constraint.

In summary, the proposed representation shows two advantages. On the one hand, it is clear and natural and thus it fulfils the rule of thumb that the genetic coding of a problem should be a natural expression of it. On the other hand, the genotype keeps on being a permutation, thus allowing us to consider the extensively used genetic operators for this representation.

3.2. The crossover operator

The main difficulty arising when using non-standard representations is the design of a suitable crossover operator able to combine relevant characteristics of the parent solutions into a valid offspring solution. Nevertheless, as our representation is order-based, the crossover operator can be designed from a classical order-based one. Crossover operators of the latter kind which have been suggested in the literature include partially mapped crossover (PMX), order crossover, order crossover # 2, position based crossover, and cycle crossover, among others (Poon & Carter, 1995). We have selected one of the most extended ones, PMX, which has been already used in other genetic algorithm implementations for the SALBP (for example in Sabuncuoglu et al. (2000)).

PMX generates two offspring from two parents by means of the following procedure: (a) two random cut points are selected, (b) for the first offspring, the genes outside the random points are copied directly from the first parent, and (c) the genes inside the two cut points are copied but in the order they appear in the second parent. The same mechanism is followed up with the second offspring but with the opposite parents. See Fig. 3 where an example of the operator is shown.

Thanks to our advanced coding scheme and to the use of a permutation-based crossover, the feasibility of the offspring with respect to precedence relations is assured. However, since information about the tasks-stations assignment is encoded inside the chromosome, it is compulsory to assure that: (a) there is not any station exceeding the fixed cycle time limit, and (b) there is not any empty station in the configuration of the assembly line.

Therefore, a repair operator must be applied for each offspring after crossover. The use of these kinds of operators is very extended in evolutionary computation when dealing with combinatorial optimization problems with hard restrictions (Chootinan & Chen, 2006). They should be carefully developed as a poor design of the repair operator can bias the convergence of the genetic algorithm or can make the crossover operator lose useful information from the parents. The goals and methods of our repair operator are the following:

- Redistribute spare tasks among available stations: changing the order of the genes in the parents to generate the offspring can cause the appearance of stations with an excessive cycle time. The repair operator must reallocate the spare tasks in other stations. First, the critical stations (those exceeding the cycle time) and their tasks are localised. Then, the feasible stations available to reallocate each task of the critical station, fulfilling precedence and cycle time restrictions, are calculated. If one spare task can be reallocated in more than one different station, the algorithm will choose one of them randomly for the reallocation. This process is repeated till either the critical station satisfies the cycle time restriction or there is no feasible move to be done. In the latter case, the critical station will be randomly divided in two or more feasible stations by adding the needed separators to balance the load.
- Removing empty stations: no empty stations are allowed. For the genotype of the individual, this means that two or more genes representing separators cannot be placed together. Thus, the repair operator will find and remove them to only keep the necessary separators.[2]

3.3. Mutation operators

Two mutation operators have been specifically designed and applied uniformly to the selected individuals of the population. The first one is based on reordering a part of the sequence of tasks and reassigning them to stations. The second one is introduced to induce more diversity in order to achieve a well distributed Pareto front approximation. The need of using the second operator will be demonstrated in the experimentation carried out in Section 4.3.1. We respectively call scramble and divider to the two mutation operators and they are described as follows:

- Scramble mutation: after choosing two points randomly, the tasks between those points are scrambled forming a new sequence of tasks in such a way the mutated solution keeps on being feasible with respect to the precedence relations. The existing separators among the two mutation points are ignored and a new reallocation of those tasks is considered by randomly generating new separator locations within the task sequence. An illustrative example is in Fig. 4. To do so, a similar mechanism to the filling thresholds of the MACS algorithm have been followed (see Section 2.3.1). The task sequence is analysed from left to right and each position has a random choice for the insertion of a separator. The probability distribution associated to the separator insertion depends on the current station filling rate according to the cycle time (see Eq. (9)). Besides, it is biased by a given α threshold defined in [0,1], which represents the minimum percentage of cycle time filling allowed for the new defined stations. Only positions making the station filling rate be higher or equal to *alpha* are likely to insert a separator and the random choice is only made in those specific cases. Hence, a low value of α will promote stations with fewer tasks, thus favouring the exploration of the left-most region of the Pareto front (assembly line configurations with a large number of stations and small area sizes, see Figs. 10 and 14). On the contrary, high values of the parameter will create stations having more tasks and being close to the cycle time limit, favouring the exploration of the right-most region of the Pareto front (configurations with a small number of stations and large area sizes). In this way, the scramble mutation becomes a parameterised operator with a parameter α defining its search behaviour. The joint use of different variants of the scramble mutation operator with different α values will properly explore the different parts of the search space in order to converge to the optimal Pareto front. The experimentation developed in the current contribution shows how better results are achieved when using two different scramble operators with α equal to 0 and 0.8.
- Divider mutation: this operator was introduced to obtain better distributed Pareto front approximations generated by the algorithm by looking for those solutions having a larger number of

[2] Notice that, the application of the current operator is not actually needed and it is more related to aesthetic reasons. The coding scheme, the designed genetic operators and the multiobjective fitness function would actually allow the algorithm to work with chromosomes encoding empty stations by directly ignoring them.

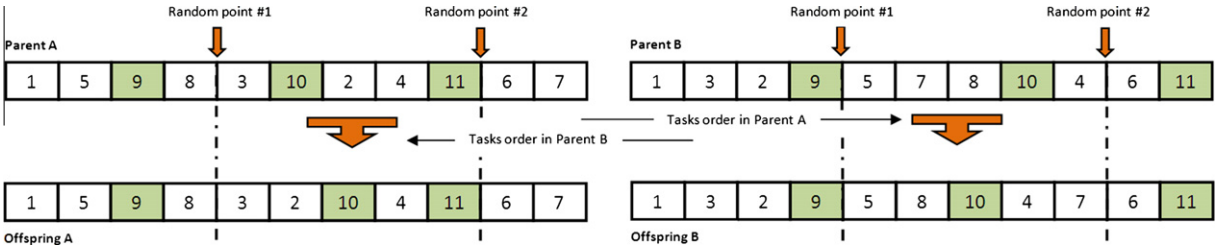

Fig. 3. An application example of the crossover operator. The tasks between the two random points are copied following the order of the other parent.

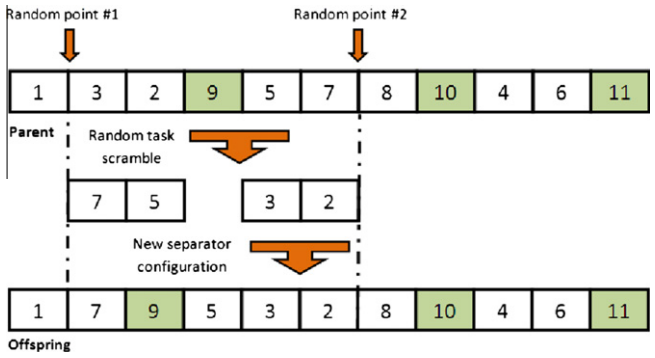

Fig. 4. The scramble mutation is applied to the first 8 tasks of the Nissan instance. The tasks between the two cut points are scrambled in the offspring.

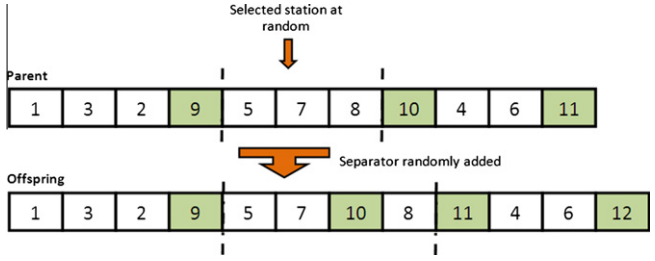

Fig. 5. The divider mutation is applied to the first 8 tasks of the Nissan instance. A new separator is chosen at random to split up the second station of the solution in two new stations.

stations with a low area (i.e., the right-most region of the Pareto front). The operator works as follows (Fig. 5): (a) it randomly selects one station with more than one task assigned, (b) it places a separator in the genotype, at a random position, to split up the current station into two stations.

3.4. Diversity induction mechanism

Finally, in order to additionally increase the diversity of the search to obtain better distributed Pareto front approximations, a set of techniques to inject diversity to the algorithm search were studied. As a result of that study, one successful and very recent NSGA-II diversity induction mechanism was adopted: Ishibuchi et al.'s similarity-based mating (Ishibuchi et al., 2008). In this way, the new design inherits the Ishibuchi et al.'s similarity-based mating from the existing *basic TSALBP-NSGA-II*, as this component helps the algorithm to get a better convergence (see the experimentation developed in Section 4.3.1).

This diversity induction mechanism is based on selecting two sets of candidates to become the couple of parents to be mated, with a pre-specified dimension γ and δ,[3] respectively. The chromosomes of each set are randomly drawn from the population by a binary tournament selection. Then, the average objective vector of the first set is computed. The most distant chromosome to the average objective vector among the γ candidates in this first set is chosen as the first parent. For the second parent, the most similar chromosome to the first parent in the objective space is selected among the δ candidates of the second set.

4. Experiments

This section is devoted to describe the experimental study developed to test our proposal. We first specify the problem instances, parameter values, and multiobjective performance indicators used for the computational tests. Then, we justify the need of using all the *advanced TSALBP-NSGA-II* components in the algorithm design to achieve the best performance. Finally, we benchmark our novel technique with respect to the existing *basic TSALBP-NSGA-II* and the state-of-the-art algorithm for the TSALBP-1/3, MACS.

4.1. Problem instances and parameters

Ten problem instances with different features have been selected for the experimentation: `arclll` with cycle time limits of $c = 5755$ and $c = 7520$ (P1 and P2), `barthol2` (P3), `barthold` (P4), `lutz2` (P5), `lutz3` (P6), `mukherje` (P7), `scholl` (P8), `wee-mag` (P9), and `Nissan` (P10). The 10 TSALBP-1/3 instances considered are publicly available at: http://www.nissanchair.com/TSALBP. Originally, these instances but `Nissan` were SALBP-1 instances[4] only having time information. However, their area information has been created by reverting the task graph to make them bi-objective (as done in Bautista & Pereira (2007)).

The real-world problem instance (P10) corresponds to the assembly process of the Nissan Pathfinder engine, assembled at the Nissan industrial plant in Barcelona (Spain) (Bautista & Pereira, 2007). As this real-world instance has special characteristics because it shows a lot of tasks having an area of 0, the repair operator for the crossover of the advanced TSALBP-NSGA-II was implemented by also redistributing the tasks with the highest-area station in the developed experiments.

We executed each algorithm 10 times with different random seeds, setting a fixed run time as stopping criterion (900 s). All the algorithms were launched in the same computer: Intel Pentium™ D with two CPUs at 2.80 GHz, and CentOS Linux 4.0 as operating system. Furthermore, the parameters of the developed algorithms and their operators are shown in Table 1.

[3] These parameters were originally noted as α and β in the original contribution (Ishibuchi et al., 2008). However, the notation for γ and δ have been changed to avoid misleading the reader with other parameters used in the current paper.

[4] Available at http://www.assembly-line-balancing.de.

Table 1
Used parameter values.

Parameter	Value	Parameter	Value
Basic TSALBP-NSGA-II			
Population size	100	Ishibuchi's γ, δ values	10
Crossover probability	0.8	Mutation probability	0.1
MACS			
Number of ants	10	β	2
ρ	0.2	q_0	0.2
Ants' thresholds (2 ants per each)	{0.2, 0.4, 0.6, 0.7, 0.9}		
Advanced TSALBP-NSGA-II			
Population size	100	Ishibuchi's γ, δ values	10
Crossover probability	0.8	Mutation probability	0.1
α values for scramble mutation	{0, 0.8}		

4.2. Multiobjective performance indicators

We will consider the two usual kinds of multiobjective performance indicators existing in the specialised literature (Zitzler, Deb, & Thiele, 2000, 2003; Deb, 2001; Knowles & Corne, 2002; Coello et al., 2007):

- Unary performance indicators: those which measure the quality of a non-dominated solution set returned by an algorithm.
- Binary performance indicators: those which compare the performance of two different multiobjective algorithms.

The first two subsections review the indicators from each group which are to be considered in the current contribution. We also present in the third subsection the use of attainment surface plots to ease the posterior analysis of results.

4.2.1. Unary performance indicator considered

The hypervolume ratio (*HVR*) (Coello et al., 2007) has become a very useful unary performance indicator. Its use is very extended as it can jointly measure the distribution and convergence of a Pareto set approximation. The *HVR* can be calculated as follows:

$$HVR = \frac{HV(P)}{HV(P^*)}, \tag{10}$$

where $HV(P)$ and $HV(P^*)$ are the volume (S indicator value) of the approximate Pareto set and the true Pareto set, respectively. When *HVR* equals 1, then the Pareto front approximation and the true Pareto front are equal. Thus, *HVR* values lower than 1 indicate a generated Pareto front that is not as good as the true Pareto front.

Since we are working with real problems, some obstacles which make difficult the computation of this performance indicator have to be kept in mind. First, it should be noticed that the true Pareto fronts are not known. In our case, a pseudo-optimal Pareto set will be considered, i.e. an approximation of the true Pareto set, obtained by merging all the Pareto set approximations P_i^j generated for each problem instance by any algorithm in any run. Thanks to this pseudo-optimal Pareto set, the *HVR* performance indicator values can be computed, considering them in our analysis of results.

Besides, there is an additional problem with respect to the *HVR* performance indicator. In minimization problems, as ours, there is a need to define a reference point to calculate the volume of a given Pareto front. If this anti-ideal solution is not correctly chosen, the *HVR* values can be unexpected (Knowles & Corne, 2002). Thus, the anti-ideal solution for each instance is defined as "logical" maximum values for the two objectives in each case. These reference points are specific for each problem instance.

4.2.2. Binary performance indicators considered

The previous performance indicator allows us to determine the absolute and individual quality of a Pareto front, but cannot be used for comparison purposes (Zitzler, Thiele, Laumanns, Fonseca, & Grunert da Fonseca, 2003). However, binary indicators aim to compare the performance of two different multiobjective algorithms by comparing the Pareto set approximations generated by each of them. In this contribution, we will consider two of them: the ϵ indicator I_ϵ and the set coverage indicator C.

The I_ϵ indicator (Zitzler et al., 2003) is a quality assessment method for multiobjective optimization that avoids particular difficulties of unary and classical methods (Knowles, 2006). Two different definitions are possible: the standard (multiplicative) I_ϵ and the additive indicator $I_{\epsilon+}$. We have opted by the multiplicative indicator. Given two Pareto front approximations, P and Q, the value $I_\epsilon(P,Q)$ is calculated as follows:

$$I_\epsilon(P,Q) = inf_{\epsilon\in\Re}\{\forall z^2 \in Q, \exists z^1 \in P : z^1 \preceq_\epsilon z^2\} \tag{11}$$

where $z^1 \preceq_\epsilon z^2$ iff $z_i^1 \leqslant \epsilon \cdot z_i^2, \forall i \in \{1,\ldots,o\}$, with o being the number of objectives, assuming minimization. $I_\epsilon(P,Q) < I_\epsilon(Q,P)$ indicates, in a weak sense, that the P set is better than the Q set because the minimum ϵ value needed so that approximation set P ϵ-dominates Q is smaller than the ϵ value needed for Q to ϵ-dominate P.

On the other hand, the classical set coverage indicator C (Zitzler et al., 2000) is computed as follows:

$$C(P,Q) = \frac{|\{q \in Q; \exists p \in P : p \preceq q\}|}{|Q|}, \tag{12}$$

where $p \preceq q$ indicates that the solution p, belonging to the approximate Pareto set P, weakly dominates the solution q of the approximate Pareto set Q in a minimization problem.

Hence, the value $C(P,Q) = 1$ means that all the solutions in Q are dominated by or equal to solutions in P. The opposite, $C(P,Q) = 0$, represents the situation where none of the solutions in Q are covered by the set P. Notice that, both $C(P,Q)$ and $C(Q,P)$ have to be considered, since $C(P,Q)$ is not necessarily equal to $1 - C(Q,P)$.

The I_ϵ and C performance indicator values of the approximation sets of every pair of algorithms have been represented by boxplots (see Figs. 7, 9a, 11 and 13a for I_ϵ, and Figs. 8, 9b, 12 and 13b for C). In the figures, each rectangle represents one of the 10 problem instances (ranging from P1 to P10). Inside each rectangle, boxplots representing the distribution of the I_ϵ and C values for a certain pair of algorithms are drawn. Given Fig. 7 as an example, the top-left rectangle shows the boxplots comparing three pairs of algorithms: TN vs. V1, TN vs.V2, and TN vs. V3 (see Section 4.3 for the notations of these algorithms). As I_ϵ and C are binary indicators, two boxplots have been drawn for each algorithm comparison. The white boxplots represent the distributions $I_\epsilon(TN,Vx)$ generated in the 10 runs, while the coloured boxplots do so for the $I_\epsilon(Vx,TN)$ values. In each boxplot, the minimum and maximum values are the lowest and highest lines, the upper and lower ends of the box are the upper and lower quartiles, a thick line within the box shows the median, and the isolated points are the outliers of the distribution.

4.2.3. Attainment surface plots

An attainment surface is the surface uniquely determined by a set of non-dominated points that divides the objective space into the region dominated by the set and the region that is not dominated by it (Fonseca & Fleming, 1996). Given r runs of an algorithm, it would be nice to summarise the r attainment surfaces obtained, using only one summary surface. Such summary attainment surfaces can be defined by imagining a diagonal line in the direction of increasing objective values cutting through the r attainment surfaces generated (see the plot in Fig. 6). The intersection on this line that weakly dominates at least $r - p + 1$ of the surfaces and is

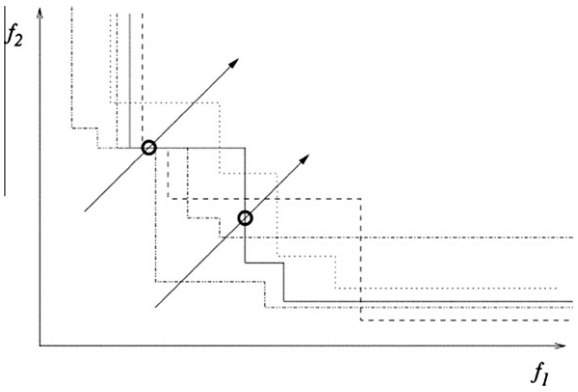

Fig. 6. Five attainment surfaces are shown representing the output of five runs of an algorithm. The two diagonal lines intersect the five surfaces at various points. In both cases, the circle indicates the intersection that weakly dominates at least $5-3+1=3$ surfaces and is also weakly dominated by at least three surfaces. Therefore, these two points both lie on the third summary attainment surface (reprinted from Knowles (2006)).

weakly dominated by at least p of them, defines one point on the "pth summary attainment surface". In our case, this surface is the union of all the goals that have been attained in the $r = 10$ independent runs of the algorithm.

Hence, the corresponding attainment surfaces will be represented in order to allow an easy visual comparison of the performance of the different benchmarked algorithms. These graphics offer a visual and quantitative information (Fonseca & Fleming, 1996), sometimes more useful than numeric values, mainly in complex problems as ours.

4.3. Experimentation and analysis of results

In this section, we analyse the performance of the *advanced TSALBP-NSGA-II*. First, a comparison of three limited variants of the new proposal is done to ensure the need of using all its features. As comparing all the possible algorithm components combinations is excessive, the most significant have been selected. Three algorithms (V1, V2, and V3) have been selected as variants of the *advanced TSALBP-NSGA-II* by removing Ishibuchi's diversity operator, the new divider mutation operator, and the scramble

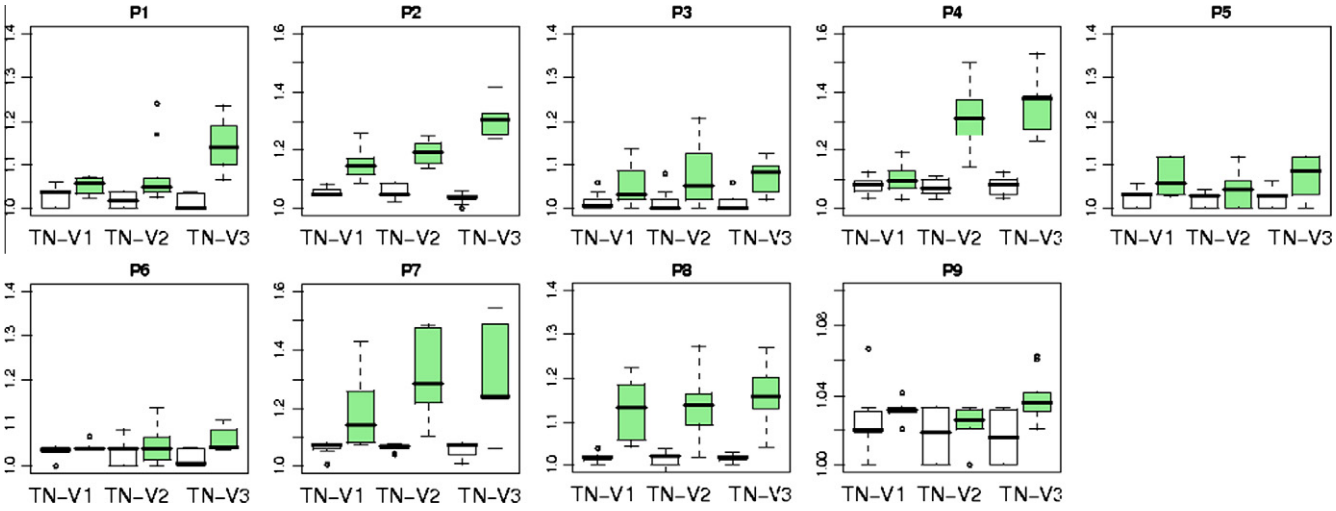

Fig. 7. Boxplots representing the binary I_ϵ indicator values for comparisons between the *advanced TSALBP-NSGA-II* (TN) and its limited variants (Vx) for instances P1–P9. White boxplots correspond to $I_\epsilon(TN,Vx)$ distribution, coloured boxplots to $I_\epsilon(Vx,TN)$. Lower values indicate better performance.

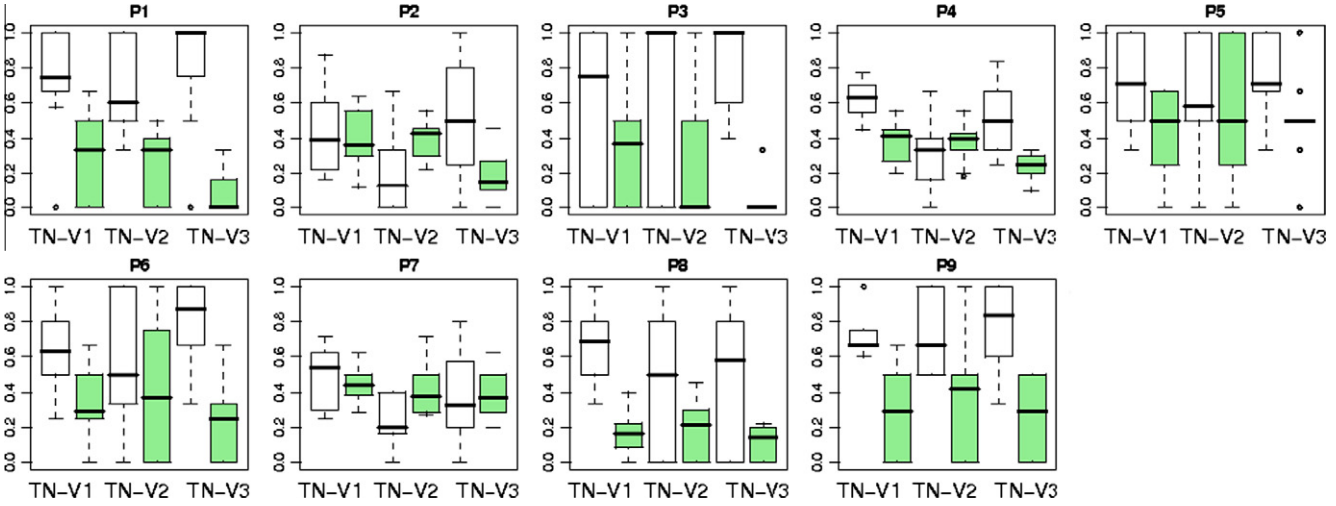

Fig. 8. Boxplots representing the binary C indicator values for comparisons between the *advanced TSALBP-NSGA-II* (TN) and its limited variants (Vx) for instances P1–P9. White boxplots correspond to $C(TN,Vx)$, coloured boxplots to $C(Vx,TN)$. Larger values indicate better performance.

mutation's parameters, respectively. Finally, a comparison between the complete version of the *advanced TSALBP-NSGA-II* and the state-of-the-art algorithms for the TSALBP-1/3 is done. **The source codes of all the algorithms considered in the current experimental study are publicly available at** http://www.nissanchair.com/TSALBP.

4.3.1. Comparison of the advanced TSALBP-NSGA-II variants

We will analyse the performance of the full design of the *advanced TSALBP-NSGA-II* algorithm described in Section 3 in comparison with the following three limited variants of it:

- V1. The difference with respect to the complete version is the lack of use of the Ishibuchi's operator. As said, this operator is able to induce more diversity into the search mechanism of the EMO algorithm in order to generate well distributed Pareto front approximations.
- V2. It only differs from the complete variant in the absence of the new divider mutation operator that was explained in Section 3.3.
- V3. The components that were suppressed in the V1 and V2 variants, that is Ishibuchi's diversity induction operator and the divider mutation operator, are now discarded in conjunction. In addition, the scramble mutation operator is used without considering the α parameter that controls the filling of the stations (which is the same that setting $\alpha = 0$).

We will consider two independent analyses in the current section. First, the performance of the *advanced TSALBP-NSGA-II* algorithm and its three limited variants (V1, V2, and V3) will be analysed in the first nine problem instances (P1–P9). Later, the same study is performed in the real-world Nissan instance (P10).

Figs. 7 and 8 show the binary performance indicators comparisons for the first nine instances. In the first figure, the I_ϵ indicator values are clearly lower in the case of the former (white boxplots) than in the latter ones (coloured boxplots) in almost every case. This means that the performance of the *advanced TSALBP-NSGA-II* is significantly better according to this indicator.

With respect to the *C* indicator (Fig. 8), a similar conclusion is achieved. The *advanced TSALBP-NSGA-II* gets better coverage values than the limited variants in almost all the problem instances: better results than V1 in the 9 problem instances, better than V2 in 6 of the 9 instances, and better than V3 in 8 of the 9 instances (all but P7). V2 gets a better yield than the complete version of the *advanced TSALBP-NSGA-II* in P2, P4, and P7.

The quality assessment of the unary performance indicator *HVR* for the *advanced TSALBP-NSGA-II* and its limited variants is shown in Table 2. Here, the values of the indicator show even clearer results. The full version of the algorithm gets the best values in all the problem instances. Therefore, the convergence and distribution of the Pareto front approximations generated by the *advanced TSALBP-NSGA-II* are the highest ones according to this indicator.

The I_ϵ and *C* performance indicators of the Nissan problem instance are shown in Fig. 9 and the *HVR* values in Table 3. We can obtain the same conclusions than with the problem instances P1–P9. There is just a different behaviour in the I_ϵ indicator, where a limited variant, V2, obtains the same performance than the complete version of the algorithm (TN).

The latter global yields can be also observed in the attainment surfaces of the different problem instances. As an example, we show those for P3 and P7 in Fig. 10 (two graphics of this kind are shown in this section due to the lack of space, although similar results are obtained in every instance). These attainment surfaces can also help us to find out why the removed components of the limited variants are needed, as it will be analysed in the following items:

Table 2
Mean and standard deviation $\bar{x}(\sigma)$ of the *HVR* performance indicator values for the *advanced TSALBP-NSGA-II* (TN) and its limited variants (Vx) for instances P1–P9. Higher values indicate better performance.

	HVR				
	P1	P2	P3	P4	P5
TN	**0.989 (0)**	**0.958 (0.02)**	**0.906 (0.05)**	**0.955 (0.01)**	**0.892 (0.06)**
V1	0.972 (0.02)	0.914 (0.01)	0.869 (0.03)	0.927 (0.03)	0.835 (0.03)
V2	0.945 (0.04)	0.905 (0.02)	0.855 (0.04)	0.812 (0.06)	0.855 (0.09)
V3	0.915 (0.04)	0.843 (0.03)	0.858 (0.05)	0.778 (0.04)	0.822 (0.02)
	P6	P7	P8	P9	
TN	**0.913 (0.06)**	**0.916 (0.02)**	**0.946 (0.04)**	**0.943 (0.02)**	
V1	0.885 (0.06)	0.862 (0.04)	0.857 (0.03)	0.915 (0.02)	
V2	0.887 (0.06)	0.801 (0.04)	0.856 (0.05)	0.914 (0.03)	
V3	0.831 (0.08)	0.801 (0.03)	0.836 (0.04)	0.907 (0.03)	

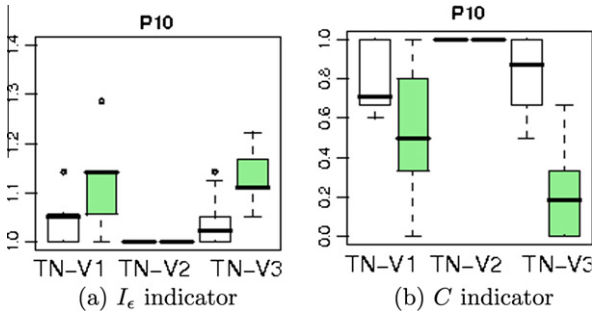

Fig. 9. The corresponding boxplots representing the binary indicators values for comparisons between the *advanced TSALBP-NSGA-II* (TN) and its limited variants (Vx) for the Nissan problem instance (P10).

Table 3
Mean and standard deviation $\bar{x}(\sigma)$ of the *HVR* performance indicator values for the *advanced TSALBP-NSGA-II* (TN) and its limited variants (Vx) for the Nissan problem instance. Higher values indicate better performance.

	HVR P10 (Nissan)
TN	**0.884 (0.07)**
V1	0.796 (0.06)
V2	**0.884 (0.06)**
V3	0.815 (0.07)

- First, the Ishibuchi's diversity induction operator will help the algorithm to get a better spread Pareto front approximation. We can draw that conclusion comparing the dashed green line (corresponding to V1) and the solid blue (that of the *advanced TSALBP-NSGA-II* algorithm) line in the attainment surfaces of Fig. 10.
- On the other hand, the use of a divider mutation operator (suppressed in the V2 variant) and the incorporation of different values for the α parameter of the scramble mutation operator are both very important. Consequently, the attainment surfaces of the V2 and V3 variants are much less spread than the complete version of the *advanced TSALBP-NSGA-II*.
- The difference of performance is more important between the *advanced TSALBP-NSGA-II* and the V3 variant. In this case, the V3 variant cannot even achieve the level of convergence of the complete algorithm as can be seen in the attainment surfaces and the *HVR* performance indicator.

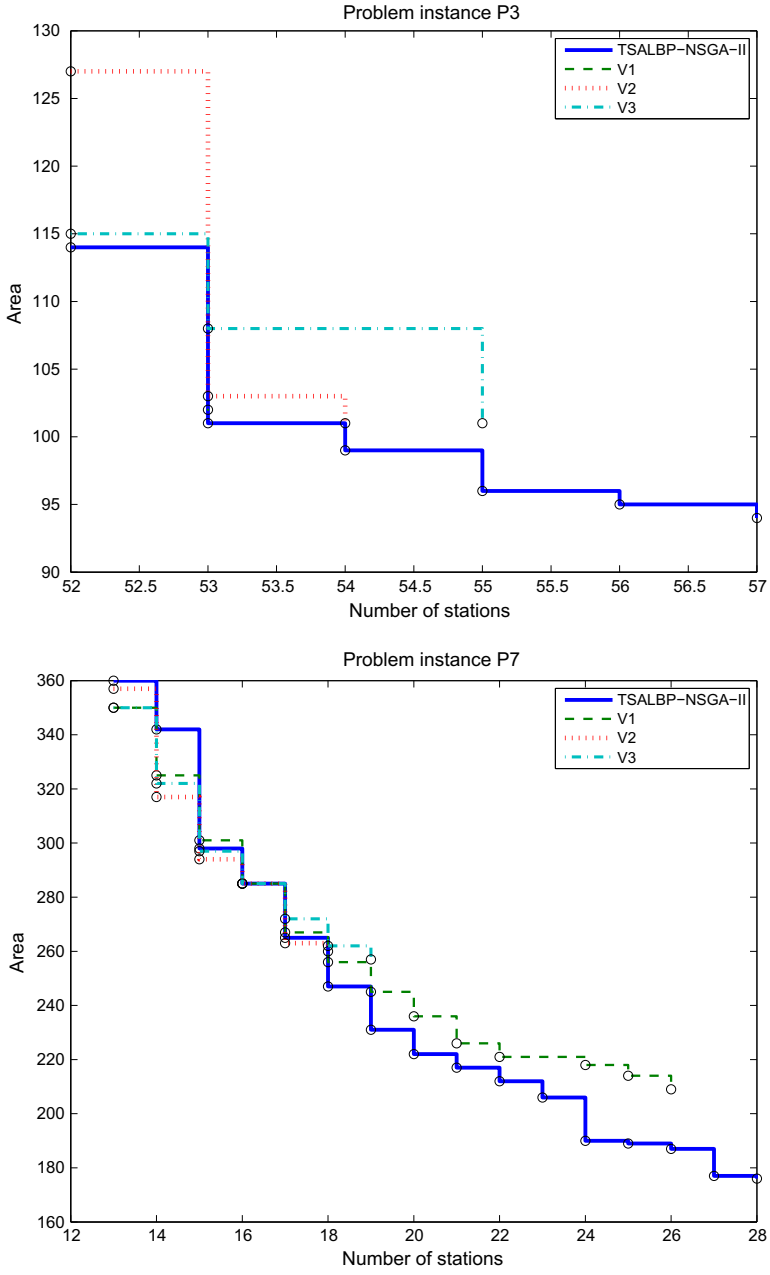

Fig. 10. Attainment surface plots for the P3 and P7 problem instances.

Consequently and in view of this experimental study, it can be concluded that every included component in the *advanced TSALBP-NSGA-II* helps to increase the performance of the algorithm, and the absence of any of them induces a considerable fall both in the convergence and diversity of the Pareto set approximations generated. It is thus clear that all the designed components are required to achieve the best diversification–intensification trade-off in the multiobjective search space.

4.3.2. Comparison of our proposal with the state-of-the-art algorithms

The MACS algorithm, reviewed in Section 2.3.1, achieved the best results for the solving of the TSALBP-1/3 in comparison with the multiobjective randomised greedy algorithm and the *basic TSALBP-NSGA-II* (Chica et al., 2010). Although the latter one reached better solutions in a specific small region of the Pareto front than the MACS algorithm, its behaviour was worse in the rest of the Pareto front, as already explained. The latter fact motivated us to design an EMO algorithm able to outperform the MACS algorithm in all the Pareto front, the goal of the present work.

In this section, these two algorithms are compared, the state-of-the-art MACS and the *basic TSALBP-NSGA-II*, with our complete proposal, the *advanced TSALBP-NSGA-II*. We use the same multiobjective performance indicators considered in the previous section and proceed in the same way performing two independent analysis (P1–P9 and P10).

The results corresponding to the two binary indicator values on the first nine instances are represented by means of boxplots in Figs. 11 and 12. The respective *HVR* values are included in Table 4. Besides, attainment surfaces for some instances are plotted in Fig. 14.

In view of the results corresponding to the I_ϵ and the C indicators in Figs. 11 and 12, a clear conclusion can be drawn: the *advanced TSALBP-NSGA-II* outperforms both MACS and the *basic TSALBP-NSGA-II* without any doubt.

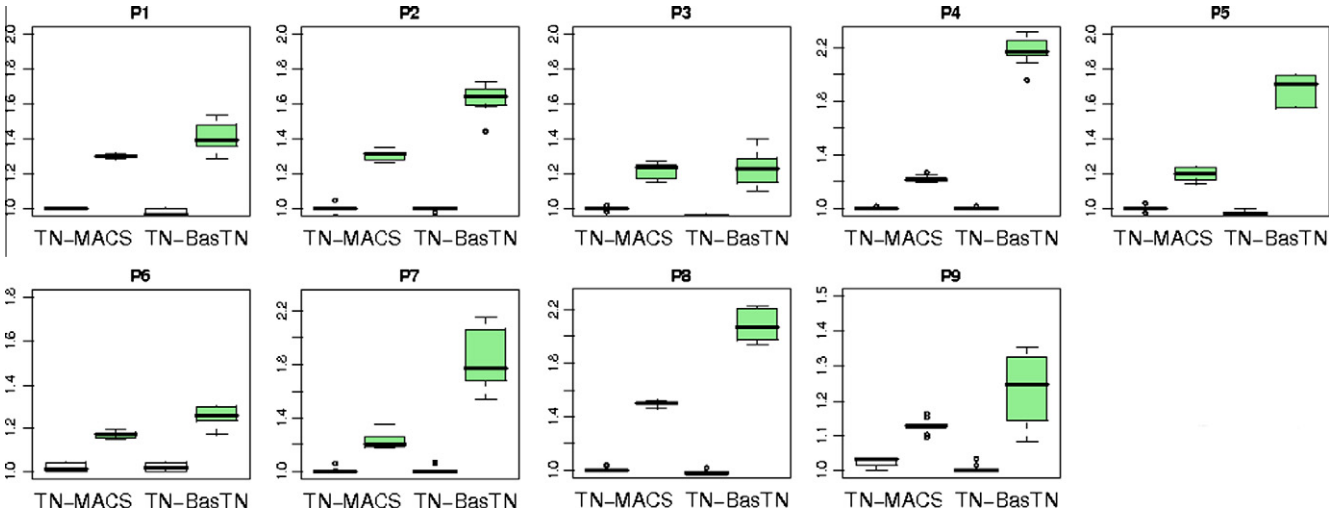

Fig. 11. Boxplots representing the binary I_ϵ indicator values for comparisons between the *advanced TSALBP-NSGA-II* (TN) and the state-of-the-art algorithms (MACS and BasTN) for instances P1 to P9. White boxplots correspond to $I_\epsilon(TN, MACS/BasTN)$, coloured boxplots to $I_\epsilon(MACS/BasTN, TN)$. Lower values indicate better performance.

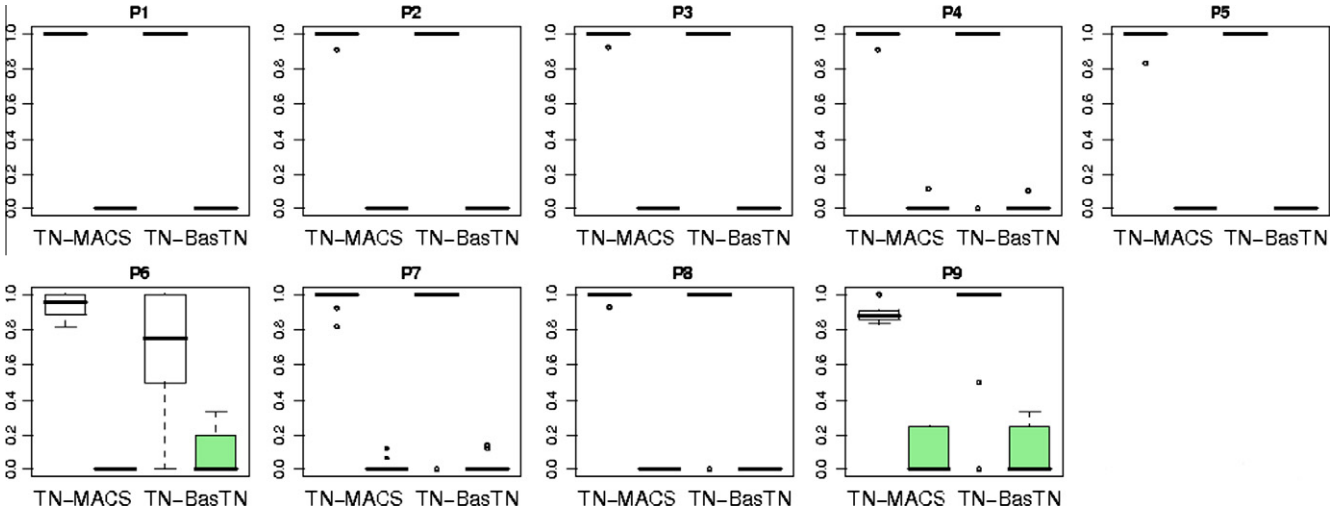

Fig. 12. Boxplots representing the binary C indicator values for comparisons between the *advanced TSALBP-NSGA-II* (TN) and the state-of-the-art algorithms (MACS and BasTN) for instances P1–P9. White boxplots correspond to $C(TN, MACS/BasTN)$, coloured boxplots to $C(MACS/BasTN, TN)$. Larger values indicate better performance.

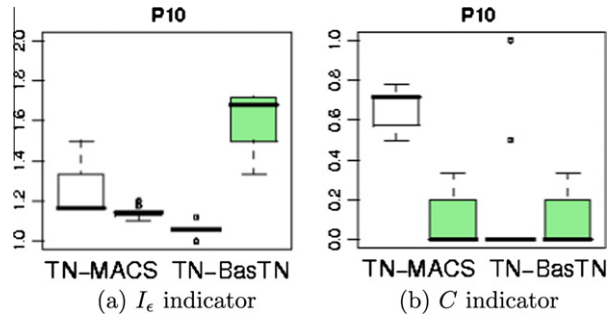

Fig. 13. The corresponding boxplots representing the binary indicators values for comparisons between the *advanced TSALBP-NSGA-II* (TN) and the state-of-the-art algorithms (MACS and BasTN) for the Nissan problem instance.

Table 4
Mean and standard deviation $\bar{x}(\sigma)$ of the *HVR* performance indicator values for the *advanced TSALBP-NSGA-II* (TN), MACS (S1) and the *basic TSALBP-NSGA-II* (S2) for instances P1–P9. Higher values indicate better performance.

	HVR				
	P1	P2	P3	P4	P5
TN	**0.989 (0)**	**0.958 (0.02)**	**0.906 (0.05)**	**0.955 (0.01)**	**0.892 (0.06)**
S1	0.763 (0)	0.766 (0.01)	0.722 (0.01)	0.723 (0.02)	0.599 (0.02)
S2	0.762 (0.03)	0.700 (0.03)	0.639 (0.07)	0.134 (0.06)	0.008 (0.01)
	P6	P7	P8	P9	
TN	**0.913 (0.06)**	**0.916 (0.02)**	**0.946 (0.04)**	**0.943 (0.02)**	
S1	0.585 (0.02)	0.740 (0.01)	0.514 (0.01)	0.820 (0.01)	
S2	0.546 (0.03)	0.434 (0.05)	0.157 (0)	0.432 (0.2)	

The same fact is observed analysing the unary indicator *HVR* results. The convergence and diversity of the *advanced TSALBP-NSGA-II* is higher than those of the state-of-the-art algorithms in all the instances. The overall good performance of the *advanced TSALBP-NSGA-II* can be clearly observed in the attainment surfaces of Fig. 14. There is a high distance between the attainment surfaces obtained by the *advanced TSALBP-NSGA-II* and those corresponding to the remaining algorithms in the P2, P3, and P8 instances. Notice that in the plot of the P3 instance the attainment surfaces of the limited V1, V2, and V3 variants of the *advanced TSALBP-NSGA-II* are also included. It can be observed that not only the complete

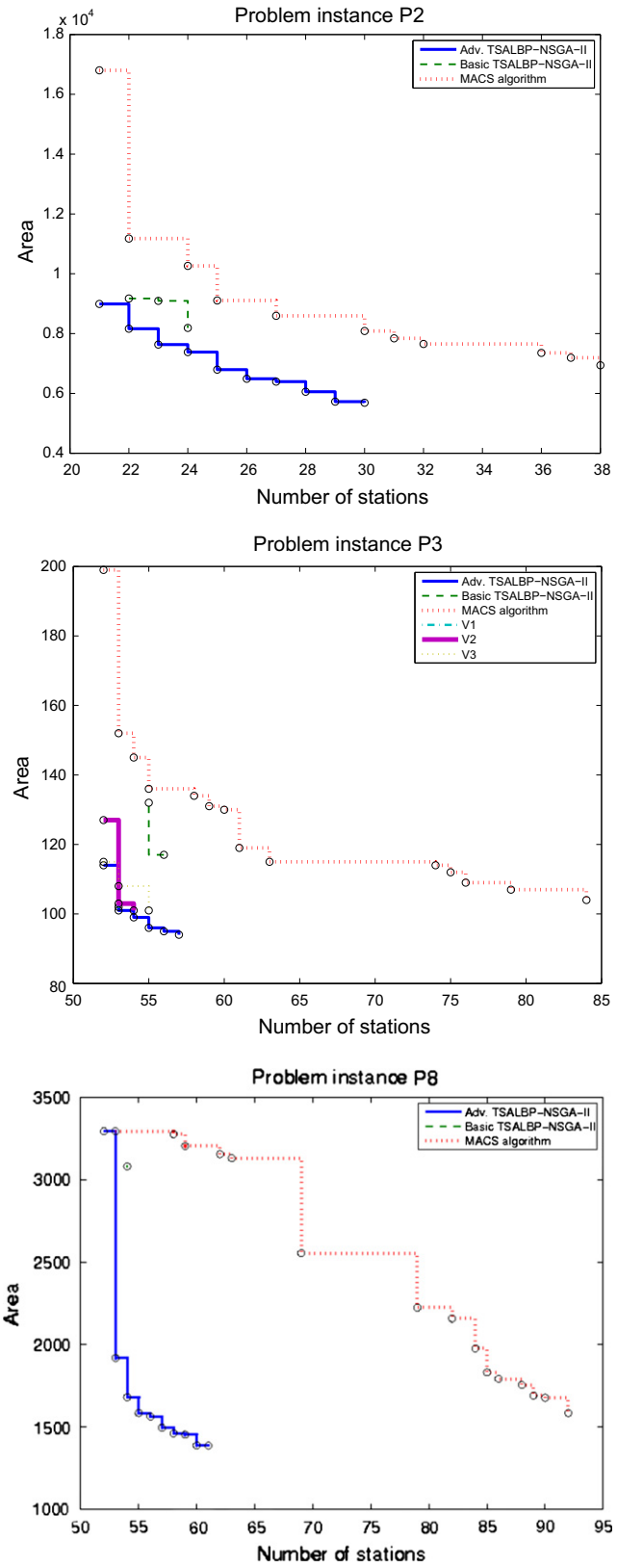

Fig. 14. Attainment surface plots for the P2, P3, and P8 problem instances.

version of the TSALBP-NSGA-II achieves better results than MACS and the *basic TSALBP-NSGA-II*. Its limited variants are also better optimisers for this TSALBP-1/3 instance.

Table 5
Mean and standard deviation $\bar{x}(\sigma)$ of the *HVR* performance indicator values for the *advanced TSALBP-NSGA-II* (TN) and the state-of-the-art algorithms, MACS (S1) and the *basic TSALBP-NSGA-II* (S2) for the Nissan problem instance. Higher values indicate better performance.

	HVR P10 (Nissan)
TN	**0.884 (0.07)**
S1	0.849 (0.01)
S2	0.316 (0.03)

The case of the real-world instance of Nissan is analysed in view of the performance indicators of Fig. 13 and the *HVR* values of Table 5. The behaviour of the algorithms is similar to that reported for the latter instances. The only exception is the I_ϵ indicator, where the MACS algorithm gets slightly better results than the *advanced TSALBP-NSGA-II*. Regarding the attainment surface of the Nissan instance (Fig. 15), although the convergence of the *advanced TSALBP-NSGA-II* is clearly higher than the rest of the algorithms, the MACS algorithm achieves the two most extreme solutions of the pseudo-optimal Pareto front which are not found by the *advanced TSALBP-NSGA-II*. This is the reason why the value of the I_ϵ indicator associated to the MACS algorithm was slightly better than the one obtained by the *advanced TSALBP-NSGA-II*, although the latter method is showing the best overall convergence to the pseudo-optimal Pareto front.

According to the previous analysis of the performance indicators and attainment surfaces, we can assert that the *advanced TSALBP-NSGA-II* outperforms the state-of-the-art algorithms in all the considered problem instances, Nissan included.

5. Concluding remarks

A novel multiobjective genetic algorithm design has been proposed to tackle the TSALBP-1/3 resulting in a new approach called the *advanced TSALBP-NSGA-II*. The need of all the main components of the proposal has been justified in a experimental study. The performance of this new technique has been compared with the state-of-the-art algorithms, the MACS multiobjective ACO approach and a previous multiobjective extension of an existing genetic algorithm for the SALBP, called *basic TSALBP-NSGA-II*. The comparisons were carried out using up-to-date multiobjective performance indicators. The *advanced TSALBP-NSGA-II* clearly outperformed the latter two methods when solving nine of the 10 TSALBP-1/3 instances considered as well as it also showed an advantage in the real-world Nissan problem instance.

It has been demonstrated that the existing *basic TSALBP-NSGA-II* showed a poor performance due to the use of non-appropriate representation and genetic operators to solve the problem. Since the TSALBP-1/3 is a very complex combinatorial optimization problem with strong constraints, a deep study of the best design options for the specific context was mandatory to get a high performance problem solving technique. Therefore, it has been demonstrated that multiobjective genetic algorithms are suitable to solve these kind of multiobjective assembly line balancing problems if a good design is used.

Future work will be devoted to: (a) apply a local search procedure to increase the performance of the algorithms, (b) add interactive preferences into the *advanced TSALBP-NSGA-II* to guide the search to the Pareto front regions preferred by the expert (Chica, Cordón, Damas, Bautista, & Pereira, 2008b, 2009, 2011), and (c) perform some further improvements in the *advanced TSALBP-NSGA-II* to slightly increase the spread of the Pareto front it generates in order to get even better results in the Nissan instance.

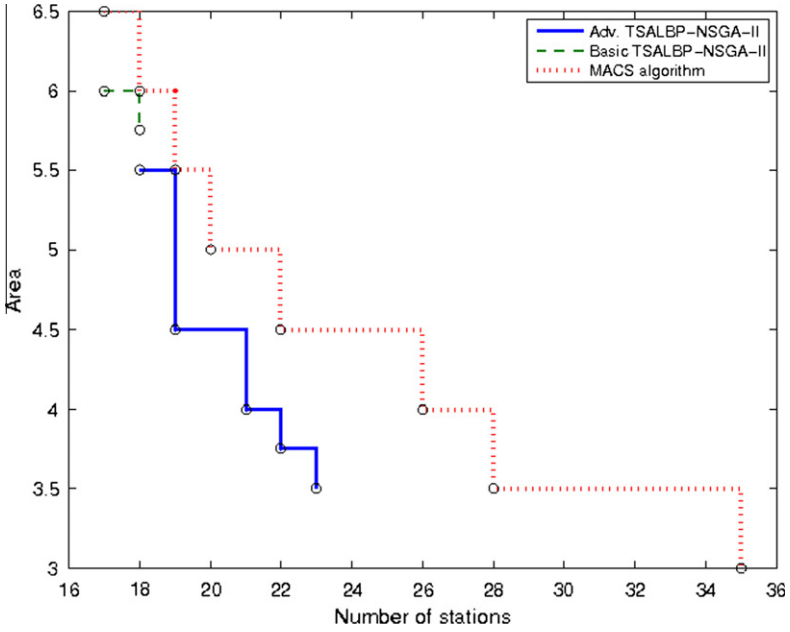

Fig. 15. Attainment surface plot for the real-world instance of Nissan (P10).

Acknowledgements

This work has been supported by the Spanish Ministerio de Ciencia e Innovación (MICINN) under Project TIN2009-07727, including EDRF fundings. We would like to express our most sincere gratitude to our collaborator, Dr. Bautista, Director of the Nissan Endowed Chair at the Technical University of Catalonia, for supporting this research with his experience on the TSALBP.

References

Altiparmak, F., Gen, M., Lin, L., & Paksoy, T. (2006). A genetic algorithm approach for multi-objective optimization of supply chain networks. *Computers and Industrial Engineering, 51*(1), 196–215.

Anderson, E. J., & Ferris, M. C. (1994). Genetic algorithms for combinatorial optimization: The assembly line balancing problem. *ORSA Journal on Computing, 6*, 161–173.

Bäck, T., Fogel, D., & Michalewicz, Z. (Eds.). (1997). *Handbook of evolutionary computation*. IOP Publishing and Oxford University Press.

Barán, B., & Schaerer, M. (2003). A multiobjective ant colony system for vehicle routing problem with time windows. In *21st IASTED international conference* (pp. 97–102), Innsbruck, Germany.

Bautista, J., & Pereira, J. (2007). Ant algorithms for a time and space constrained assembly line balancing problem. *European Journal of Operational Research, 177*, 2016–2032.

Baybars, I. (1986). A survey of exact algorithms for the simple assembly line balancing problem. *Management Science, 32*(8), 909–932.

Blum, C. (2008). *Beam-ACO for Simple Assembly Line Balancing, 20*(4), 618–627.

Chiang, W. C. (1998). The application of a tabu search metaheuristic to the assembly line balancing problem. *Annals of Operations Research, 77*, 209–227.

Chica, M., Cordón, O., Damas, S., & Bautista, J. (2010). Multiobjective, constructive heuristics for the 1/3 variant of the time and space assembly line balancing problem: ACO and random greedy search. *Information Sciences, 180*, 3465–3487.

Chica, M., Cordón, O., Damas, S., & Bautista, J. (2011). Including different kinds of preferences in a multi-objective ant algorithm for time and space assembly line balancing on different Nissan scenarios. *Expert Systems with Applications, 38*, 709–720.

Chica, M., Cordón, O., Damas, S., Bautista, J., & Pereira, J. (2008b). Incorporating preferences to a multi-objective ant algorithm for time and space assembly line balancing. In *Ant Colony Optimization and Swarm Intelligence. Lectures Notes in Computer Sciences* (Vol. 5217, pp. 331–338). Springer.

Chica, M., Cordón, O., Damas, S., Bautista, J., & Pereira, J. (2008a). A multiobjective ant colony optimization algorithm for the 1/3 variant of the time and space assembly line balancing problem. In *12th international conference on processing and management of uncertainty in knowledge-based systems (IPMU)* (pp. 1454–1461), Málaga, Spain.

Chica, M., Cordón, O., Damas, S., & Bautista, J. (2009). Integration of an EMO-based preference elicitation scheme into a multi-objective ACO algorithm for time and space assembly line balancing. In *2009 IEEE international conference on multicriteria decision making (IEEE MCDM 2009)* (pp. 157–162), Nashville, USA.

Chootinan, P., & Chen, A. (2006). Constraint handling in genetic algorithms using a gradient-based repair method. *Computers and Operations Research, 33*(8), 2263–2281.

Coello, C. A., Lamont, G. B., & Van Veldhuizen, D. A. (2007). *Evolutionary algorithms for solving multi-objective problems* (2nd ed.). Springer.

Deb, K. (2001). *Multi-objective optimization using evolutionary algorithms*. Wiley.

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation, 6*(2), 182–197.

Dietz, A., Azzaro-Pantel, C., Pibouleau, L., & Domenech, S. (2008). Strategies for multiobjective genetic algorithm development: Application to optimal batch plant design in process systems engineering. *Computers and Industrial Engineering, 54*(3), 539–569.

Dorigo, M., & Gambardella, L. (1997). Ant colony system: A cooperative learning approach to the traveling salesman problem. *IEEE Transactions on Evolutionary Computation, 1*(1), 53–66.

Dorigo, M., & Stützle, T. (2004). *Ant colony optimization*. Cambridge: MIT Press.

Feo, T. A., & Resende, M. G. C. (1995). Greedy randomized adaptive search procedures. *Journal of Global Optimization, 6*, 109–133.

Fonseca, C. M., & Fleming, P. J. (1996). On the performance assessment and comparison of stochastic multiobjective optimizers. In *Proceedings of the 4th international conference on parallel problem solving from nature (PPSN), Lecture notes in computer science* (Vol. 1141, pp. 584–593), Berlin, Germany.

Gao, J., Gen, M., Sun, L., & Zhao, X. (2007). A hybrid of genetic algorithm and bottleneck shifting for multiobjective flexible job shop scheduling problems. *Computers and Industrial Engineering, 53*(1), 149–162.

Glover, F., & Kochenberger, G. A. (Eds.). (2003). *Handbook of metaheuristics*. Kluwer Academic.

Goldberg, D. E. (1989). *Genetic algorithms in search, optimization and machine learning*. Addison-Wesley Publishing.

Heinrici, A. (1994). A comparison between simulated annealing and tabu search with an example from the production planning. In H. Dyckhoff, U. Derigs, M. Salomon (Eds.), *Operations research proceedings 1993* (pp. 498–503), Berlin, Germany.

Ishibuchi, H., Narukawa, K., Tsukamoto, N., & Nojima, Y. (2008). An empirical study on similarity-based mating for evolutionary multiobjective combinatorial optimization. *European Journal of Operational Research, 188*(1), 57–75.

Kim, Y. K., Kim, Y. J., & Kim, Y. (1996). Genetic algorithms for assembly line balancing with various objectives. *Computers and Industrial Engineering, 30*(3), 397–409.

Kim, Y. K., Kim, Y., & Kim, Y. J. (2000). Two-sided assembly line balancing: A genetic algorithm approach. *Production Planning and Control, 11*, 44–53.

Kim, Y. K., Song, W. S., & Kim, J. H. (2009). A mathematical model and a genetic algorithm for two-sided assembly line balancing. *Computers and Operations Research, 36*(3), 853–865.

Knowles, J. (2006). ParEGO: A hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems. *IEEE Transactions on Evolutionary Computation, 10*(1), 50–66.

Knowles, J., & Corne, D. (2002). Instance generators and test suites for the multiobjective quadratic assignment problem. In *Proceedings of the 2002*

congress on evolutionary computation (CEC) (Vol. 1, pp. 711–716), Honolulu, HI, USA.

Leu, Y. Y., Matheson, L. A., & Rees, L. P. (1994). Assembly line balancing using genetic algorithms with heuristic-generated initial populations and multiple evaluation criteria. *Decision Sciences, 25*, 581–606.

Leung, S., Wong, W., & Mok, P. (2008). Multiple-objective genetic optimization of the spatial design for packing and distribution carton boxes. *Computers and Industrial Engineering, 54*(4), 889–902.

McGovern, S. M., & Gupta, S. M. (2007). A balancing method and genetic algorithm for disassembly line balancing. *European Journal of Operational Research, 179*(3), 692–708.

McMullen, P. R., & Tarasewich, P. (2006). *Multi-objective assembly line balancing via a modified ant colony optimization technique, 44*, 27–42.

Michalewicz, Z., Dasgupta, D., Riche, R. G. L., & Schoenauer, M. (1996). Evolutionary algorithms for constrained engineering problems. *Computers and Industrial Engineering, 30*(4), 851–870.

Nearchou, A. C. (2008). *Multi-objective balancing of assembly lines by population heuristics, 46*(8), 2275–2297.

Poon, P. W., & Carter, J. N. (1995). Genetic algorithm crossover operators. *Computers and Operations Research, 22*(1), 135–147.

Robertson, A. M., & Willett, P. (1994). Generation of equifrequent groups of words using a genetic algorithm. *Journal of Documentation, 50*(3), 213–232.

Sabuncuoglu, I., Erel, E., & Tayner, M. (2000). Assembly line balancing using genetic algorithms. *Journal of Intelligent Manufacturing, 11*, 295–310.

Santana-Quintero, L. V., Hernández-Díaz, A. G., Molina, J., Coello, C. A., & Caballero, R. (2010). A hybrid multi-objective optimization algorithm using differential evolution and rough set theory for constrained problems. *Computers and Operations Research, 37*(3), 470–480.

Scholl, A. (1999). *Balancing and sequencing of assembly lines* (2nd ed.). Heidelberg: Physica-Verlag.

Scholl, A., & Becker, C. (2006). State-of-the-art exact and heuristic solution procedures for simple assembly line balancing. *European Journal of Operational Research, 168*(3), 666–693.

Scholl, A., & Voss, S. (1996). Simple assembly line balancing – Heuristic approaches. *Journal of Heuristics, 2*, 217–244.

Simaria, A. S., & Vilarinho, P. M. (2004). A genetic algorithm based approach to the mixed-model assembly line balancing problem of type II. *Computers and Industrial Engineering, 47*(4), 391–407.

Zitzler, E., Deb, K., & Thiele, L. (2000). Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary Computation, 8*(2), 173–195.

Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C. M., & Grunert da Fonseca, V. (2003). Performance assessment of multiobjective optimizers: An analysis and review. *IEEE Transactions on Evolutionary Computation, 7*(2), 117–132.