Adding diversity to two multiobjective constructive metaheuristics for time and space assembly line balancing

Manuel Chica[1], Óscar Cordón[1], Sergio Damas[1], and Joaquín Bautista[2,3]

Abstract We present a new mechanism to introduce diversity into two multiobjective approaches based on ant colony optimisation and randomised greedy algorithms to solve a more realistic extension of a classical industrial problem: time and space assembly line balancing. Promising results are shown after applying the designed constructive metaheuristics to ten real-like problem instances.

Key words: Time and Space Assembly Line Balancing Problem, Constructive Metaheuristics, Multiobjective Optimisation, Automotive Industry.

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 to 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. Consequently, 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 –referred to as simple assembly line balancing problems (SALBP)– was proposed to model this situation [3, 15]. Taking this

European Centre for Soft Computing, Mieres (Asturias), Spain
{manuel.chica, oscar.cordon, sergio.damas}@softcomputing.es
· Universitat Politècnica de Catalunya, Barcelona, Spain
joaquin.bautista@upc.edu · Nissan Chair (http://www.nissanchair.com)

family as a base and adding spatial information to enrich it, Bautista and Pereira recently proposed a more realistic framework: the time and space assembly line balancing problem (TSALBP) [2]. This framework considers an additional space constraint to become a simplified version of real-world problems. The new space constraint emerged due to the study of the specific characteristics of the Nissan plant in Barcelona (Spain).

As many real-world problems, TSALBP formulations have a multicriteria nature [4] because they contain three conflicting objectives to be minimised: the cycle time of the assembly line, the number of the stations, and the area of this stations. In this paper we have selected the TSALBP-1/3 variant which tries to minimise the number of stations and their area for a given product cycle time. We have made this decision because it is quite realistic in the automotive industry. The final aim is to provide the plant manager with a well spread Pareto front of solutions with different trade-offs between the number of stations and the area of these stations. This will allow the plant manager to choose the most appropriate one for his/her industrial context.

TSALBP-1/3 has an important set of constraints like precedences or cycle time limits for each station. Thus, the use of constructive approaches like ant colony optimisation (ACO) [10] is more convenient than others like local or global search procedures. ACO is a constructive metaheuristic [13] inspired by the shortest path searching behaviour of various ant species. Many different ACO algorithms have successfully solved different combinatorial problems such as the travelling salesman problem, the quadratic assignment problem, the sequential ordering problem, production scheduling, timetabling, project scheduling, vehicle and telecommunication routing, and investment planning [10].

Due to the two aforementioned reasons, i.e., the multiobjective nature of the problem and the need to solve it through constructive algorithms, a sensible choice is to use a Pareto-based multiobjective ACO (MOACO) algorithm [12]. This family involves different variants of ACO algorithms which aim to find not only one solution, but a set of the best solutions according to several conflicting objective functions.

In [7], we successfully tackled the TSALBP-1/3 by means of a specific procedure based on the multiple ant colony system (MACS) algorithm [1]. However, we noticed that intensification could be too high in a specific region of the Pareto front because of the station-oriented approach that was accomplished. In particular, the approximations to the obtained Pareto fronts showed a significant lack of diversity and an excessive convergence to the left-most region of the objective space. That is an undesirable situation for the plant managers who should be provided with all the configurations of their contextual interest in the objective space.

In this paper we aim to introduce a new mechanism in two constructive metaheuristics in order to avoid that local convergence behaviour. On the one hand, we induce the generation of more diverse solutions by means of a multi-colony approach [14] according to different station filling rates in the MACS algorithm. On the other hand, the new filling threshold mechanism is also included on another method. In particular, we consider a multiobjective randomised greedy algorithm (MORGA), based on the first stage of the GRASP method [11] and proposed in [6]. It follows the same constructive scheme and Pareto-based approach used in the MACS algorithm. In this way, we have been able to compare the influence of incorporating the new diversity improvement in both approaches, the MOACO algorithm and the MORGA. These algorithms, with and without the new diversification component, will be tested on ten real-like TSALBP-1/3 instances.

The paper is structured as follows. In Section 2, the problem formulation and the MOACO algorithm and the MORGA are explained. Then, the proposed multi-colony approach to improve the proposals is described in Section 3. The experimentation setup as well as the analysis of results is presented in Section 4. Finally, some concluding remarks are discussed in Section 5.

2 Preliminaries

In this section the problem preliminaries are presented first. Then, the main features of the MACS algorithm and the MORGA to tackle the TSALBP-1/3 variant are described.

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 an operation time for its execution $t_j > 0$ that is determined as a function of the manufacturing technologies and the employed resources. A task j is assigned to a single station k. Each station k has thus assigned a subset of tasks S_k ($S_k \subseteq V$), called its workload.

Each task j has a set of direct predecessors, P_j, which must be accomplished before starting it. These constraints are normally represented by means of an acyclic precedence graph, whose vertices stand for the tasks and where a directed arc (i, j) indicates that task i must be finished before starting task j on the production line. Thus, if $i \in S_h$ and $j \in S_k$, then $h \leq k$ must be fulfilled. Each station k presents a station workload time $t(S_k)$ that is equal to the sum of the tasks' lengths assigned to the station k. SALBP [15] focuses on grouping tasks in workstations by an efficient and coherent way. There is a large variety of exact and heuristic problem-solving procedures for it [16].

The need of introducing space constraints in the assembly lines' design is based on 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. Hence, an area constraint may be considered by associating a required area a_j to each task j and an available area A_k to each station k that, for the sake of simplicity, we shall assume it to be identical for every station and equal to $A : A = max_{\forall k \in \{1..n\}} \{A_k\}$. Thus, each station k requires a station area $a(S_k)$ that is equal to the sum of areas required by the tasks assigned to station k.

This leads us to a new family of problems called TSALBP in [2]. It may be stated as: given a set of n tasks with their temporal t_j and spatial a_j attributes ($1 \leq j \leq n$) and a precedence graph, each task must be assigned to a single station such that: (i) every precedence constraint is satisfied, (ii) no station workload time ($t(S_k)$) is greater than the cycle time (c), and (iii) no area required by any station ($a(S_k)$) is greater than the available area per station (A).

TSALBP presents eight variants depending on three optimisation criteria: m (the number of stations), c (the cycle time) and A (the area of the stations). Within these variants there are four multiobjective problems and we will tackle one of them, 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 since the annual production of an industrial plant (and therefore, the cycle time c) is usually set by some market objectives. For more information we refer the interested reader to [5].

2.2 TSALBP-1/3 Formulation

According to the TSALBP formulation [2], the 1/3 variant deals with the minimisation of the number of stations, m, and the area ocuppied by those stations, A, in the assembly line configuration. We can mathematically formulate this TSALBP variant as follows:

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

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

subject to:

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

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

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

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

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

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

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 equation 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 Multiple ant colony system

MACS was proposed as a was proposed as a multiobjective extension of the ant colony system (ACS) [9]. MACS uses a single pheromone trail matrix τ and several heuristic information functions η^k (in our case, η^0 for the operation time t_j of each task j and η^1 for its area a_j). From now on, we restrict the description of the algorithm to the case of two objectives. In this way, an ant moves from node i to node j by applying the following transition rule:

$$j = \begin{cases} \arg\ \max_{j \in \Omega}(\tau_{ij} \cdot [\eta_{ij}^0]^{\lambda\beta} \cdot [\eta_{ij}^1]^{(1-\lambda)\beta}), & \text{if } q \leq q_0, \\ \hat{i}, & \text{otherwise.} \end{cases} \tag{9}$$

where Ω represents the current feasible neighbourhood of the ant, β weights the relative importance of the heuristic information with respect to the pheromone trail, and λ is computed from the ant index h as $\lambda = h/M$, with M being the number of ants in the colony, $q_0 \in [0,1]$ is an exploitation-exploration parameter, q is a random value in $[0,1]$, and $\hat{i}$ is a node selected according to the probability distribution $p(j)$:

$$p(j) = \begin{cases} \frac{\tau_{ij} \cdot [\eta^0_{ij}]^{\lambda\beta} \cdot [\eta^1_{ij}]^{(1-\lambda)\beta}}{\sum_{u \in \Omega} \tau_{iu} \cdot [\eta^0_{iu}]^{\lambda\beta} \cdot [\eta^1_{iu}]^{(1-\lambda)\beta}}, & \text{if } j \in \Omega, \\ 0, & \text{otherwise.} \end{cases} \quad (10)$$

Every time an ant crosses edge $< i, j >$, it performs the local pheromone update as follows: $\tau_{ij} = (1-\rho) \cdot \tau_{ij} + \rho \cdot \tau_0$

Initially, τ_0 is calculated by taking the average costs, $\hat{f}^0$ and $\hat{f}^1$, of each of the two objective functions, f^0 and f^1, from a set of heuristic solutions by applying the expression:

$$\tau_0 = \frac{1}{\hat{f}^0 \cdot \hat{f}^1} \quad (11)$$

However, the value of τ_0 is not fixed during the algorithm run, as usual in ACS, but it undergoes adaptation. At the end of each iteration, every complete solution built by the ants is compared to the Pareto archive P_A which was generated till that moment. This is done in order to check if a new solution is a non-dominated one. If so, it is included in the archive and all the dominated solutions are removed. Then, τ_0' is calculated by applying equation (11) with the average values of each objective function taken from the current solutions of the Pareto archive. If $\tau_0' > \tau_0$, being τ_0 the initial pheromone value, pheromone trails are reinitialised to the new value $\tau_0 = \tau_0'$. Otherwise, a global update is performed with each solution S of the Pareto set approximation contained in P_A applying the following rule on its composing edges $< i, j >$:

$$\tau_{ij} = (1-\rho) \cdot \tau_{ij} + \frac{\rho}{f^0(S) \cdot f^1(S)} \quad (12)$$

2.4 A MACS algorithm for the TSALBP-1/3

In this section we describe the customisation made on all the components of the general MACS algorithm scheme to build our solution methodology.

2.4.1 Heuristic information

MACS works with two different heuristic information values, η^0_j and η^1_j, each of them associated to one criterion. In our case, η^0_j is related with the required operation time for each task and η^1_j with the required area:

$$\eta^0_j = \frac{t_j}{c} \cdot \frac{|F_j|}{\max_{i \in \Omega} |F_i|} \qquad \eta^1_j = \frac{a_j}{UB_A} \cdot \frac{|F_j|}{\max_{i \in \Omega} |F_i|}$$

where UB_A is the upper bound for the area (the sum of all tasks' areas) and F_j is the set of tasks that come after task j. The second term in both formulae represents a ratio between the number of successors of the task j (the cardinality of the successors set F_j) and the maximum number of successors of any eligible task belonging to the ant's feasible neighbourhood Ω. Both sources of heuristic information range in $[0,1]$, with 1 being the most preferable.

As usual in the SALBP, tasks having a large value of time (a large duration) and area (occupying a lot of space) are preferred to be firstly allocated in the stations. Apart from area and time information, we have added another information related to the number of successors of the task which was already used in [2]. Tasks with a larger number of successors are preferred to be allocated first.

Heuristic information is one-dimensional since it is only assigned to tasks. In addition, it can be noticed that heuristic information has static and dynamic components. Tasks' time t_j and area a_j are always fixed while the successors rate is changing through the constructive procedure. This is because it is calculated by means of the candidate list of feasible and non-assigned tasks at that moment.

2.4.2 Pheromone trail and τ_0 calculation

The pheromone trail information has to memorise which tasks are the most appropriate to be assigned to a station. Hence, pheromone has to be associated to a pair $(station_k, task_j)$, being $k = 1,...,n$ and $j = 1,...,n$. In this way, contrary to heuristic information, our pheromone trail matrix has a bi-dimensional nature since it links tasks with stations.

In every ACO algorithm, an initial value for the pheromone trails has to be set up. This value is called τ_0 and it is normally obtained from an heuristic algorithm. We have used two station-oriented single-objective greedy algorithms, one per heuristic, to compute it. These algorithms open the first station and select the best possible task according to their heuristic information (related either with the duration time and successors rate η_j^0, or the area and successors rate η_j^1). This process is repeated till there is not any task that can be included because of the cycle time limit. Then, a new station must be opened. When no more tasks are to be assigned, the greedy algorithm finishes. τ_0 is then computed from the costs of the two solutions obtained by the greedy algorithm using the following MACS equation: $\tau_0 = \frac{1}{f^0(S_{time}) \cdot f^1(S_{area})}$

2.4.3 Randomised station closing scheme and transition rule

Our approach follows a *station-oriented* procedure, which starts opening a station and selecting the most suitable task to be assigned. When the current station is loaded maximally, it is closed and the next one is opened and ready to be filled. At the beginning, we decided to close the station when it was full in relation to the fixed cycle time c, as usual in SALBP and TSALBP applications. We found that this scheme did not succeed because the obtained Pareto fronts did not have enough diversity. Thus, we introduced a new mechanism in the construction algorithm to close the station according to a probability, given by the filling rate of the station:

$$p\,(closing\,S_k) = \frac{\sum_{i \in S_k} t_i}{c} \tag{13}$$

This probability distribution is updated at each construction step. A random number is uniformly generated in $[0,1]$ after each update to decide whether the station is closed or not. If the decision is not to close the station, we choose the next task among all the candidate tasks using the MACS transition rule and the procedure goes on.

Because of the one-dimensional nature of the heuristic information, the original transition rule (Equation 9) that chooses among all the candidate tasks at each step, has been modified:

$$j = \begin{cases} \arg\ \max_{j \in \Omega}(\tau_{kj} \cdot [\eta_j^0]^{\lambda\beta} \cdot [\eta_j^1]^{(1-\lambda)\beta}), & \text{if } q \le q_0, \\ \hat{i}, & \text{otherwise,} \end{cases} \tag{14}$$

where $\hat{i}$ is a node selected by means of the following probability distribution:

$$p(j) = \begin{cases} \frac{\tau_{kj} \cdot [\eta_j^0]^{\lambda\beta} \cdot [\eta_j^1]^{(1-\lambda)\beta}}{\sum_{u \in \Omega} \tau_{ku} \cdot [\eta_u^0]^{\lambda\beta} \cdot [\eta_u^1]^{(1-\lambda)\beta}}, & \text{if } j \in \Omega, \\ 0, & \text{otherwise.} \end{cases} \tag{15}$$

2.5 MORGA

Our diversification generation mechanism behaves similarly to a GRASP construction phase [11]. The most important element in this kind of construction is that the selection of the task at each step must be guided by a stochastic greedy function that is adapted with the pseudo-random selections made in the previous steps.

We introduce randomness in two processes. On the one hand, allowing each decision to be randomly taken among the best candidates. On the other hand, closing the station according to a probability distribution.

We use the same constructive approach, with closing probabilities at each constructive step, than in the MACS algorithm. The probabilistic criterion to select the next task that will be included in the current station is changed to be only based on heuristic information. Therefore, to make a decision among all the current feasible candidate tasks we use a single heuristic value given by:

$$\eta_j = \frac{t_j}{c} \cdot \frac{a_j}{UB_A} \cdot \frac{|\ F_j\ |}{\max_{i \in \Omega} |\ F_i\ |} \tag{16}$$

The decision is made randomly among the selected tasks in the restricted candidate list (RCL) by means of the following procedure: we calculate the heuristic value of every feasible candidate task to be assigned to the current open station. Then, we sort them according to their heuristic values and, finally, we set a quality threshold for the heuristic given by $q = \max_{\eta_j} - \gamma \cdot (\max_{\eta_j} - \min_{\eta_j})$.

All the tasks with a heuristic value η_j greater or equal than q are selected to be in the RCL. γ is the diversification-intensification trade-off control parameter.

When $\gamma = 1$ there is a completely random choice inducing the maximum possible diversification. In contrast, if $\gamma = 0$ the choice is close to a pure greedy decision, with a low diversification.

As MACS, the MORGA construction algorithm also incorporates a mechanism which allows us to close a station according to a probability distribution, given by the filling rate of the station (see equation (13)).

3 Using a multi-colony approach on the MACS-TSALBP-1/3 and MORGA algorithms

The MACS-based TSALBP-1/3 algorithm proposed in [7] carries the problem of not providing enough intensification in some Pareto front areas, since there is a low probability of filling stations completely. Hence, there is a need to find a better intensification-diversification trade-off. This objective can be achieved by introducing different filling thresholds associated to the ants that build the solution. These thresholds make the different ants in the colony have a different search behaviour. Thus, the ACO algorithm becomes multi-colony [14]. In our case, thresholds are set between 0.2 and 0.9 and they are considered as a preliminary step before deciding to close a station.

Therefore, the solution construction procedure is modified. We compute the station closing probability distribution as usual based on the station current filling rate (equation (13)). However, only when the ant's filling threshold has been overcome, the random decision of either closing a station or not according to that probability distribution is considered. Otherwise, the station will be kept opened. Thus, the higher the ant's threshold is, the more complete the station will be likely to be. 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 fronts by appropriately spreading the generated solutions.

We have also used the same filling thresholds technique for the MORGA. In the MACS algorithm, these filling thresholds are applied in parallel following the multi-colony approach. Unlike the MACS algorithm, different thresholds are only used in isolation at each iteration in the case of the MORGA.

4 Experimentation

4.1 Problem instances and parameter values

Ten problem instances with different features have been selected for the experimentation: `arc111` with cycle time limits of $c = 5755$ and $c = 7520$ (P1 and P2), `barthol2` (P3), `barthold` (P4), `heskia` (P5), `lutz2` (P6), `lutz3` (P7), `mukherje` (P8), `scholl` (P9), and `weemag` (P10). Originally, these instances were SALBP-1 instances only having time information. However, we have created their area information by reverting the task graph to make them bi-objective (as done in [2]) [1].

We run each algorithm 10 times with different random seeds, setting the time as stopping criteria (900 seconds). All the algorithms were launched in the same computer: Intel Pentium^{TM} D with two CPUs at 2.80GHz, and CentOS Linux 4.0. On the one hand, the values of the parameters used in all the MACS algorithms with and without the new diversification component are as follows. We consider ten different ants, $\beta = 2$, and $\rho = 0.2$. Different values of the transition rule parameter q_0 are also studied. In particular: $q_0 = 0.2, 0.5, 0.8$. On the other hand, the MORGA was launched with different diversification-intensification parameter values, $\gamma = \{0.1, 0.2, 0.3\}$

With respect to the parameters of our proposal on using different filling thresholds, there are two ants for each of the five ants' thresholds considered: $\{0.2, 0.4, 0.6, 0.7, 0.9\}$ in the MACS algorithm. The same threshold values were used for the MORGA.

4.2 Metrics of performance

We will consider two different multiobjective metrics [8, 17] to evaluate the performance of the two variants of the MACS-based TSALBP-1/3 algorithm and the MORGA.

On the one hand, we selected the hypervolume ratio (HVR) from the first group. It can be calculated as follows:

$$HVR = \frac{HV(P)}{HV(P^*)}, \tag{17}$$

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

---

[1] Problem instances and more information available at `http://www.nissanchair.com/TSALBP`

We should notice that the true Pareto fronts are not known in our real-world problem instances. Thus, we will consider a pseudo-optimal Pareto set, i.e. an approximation of the true Pareto set, obtained by merging all the (approximate) Pareto sets $\overline{P_i^j}$ generated for each problem instance by all the existing algorithms for the problem in the different runs [5]. Thanks to this pseudo-optimal Pareto set, we can compute HVR and consider it in our analysis of results.

On the other hand, we have also considered the binary set coverage metric C to compare the obtained Pareto sets two by two based on the following expression:

$$C(P,Q) = \frac{|\{q \in Q; \exists p \in P : p \prec q\}|}{|Q|}, \tag{18}$$

where $p \prec q$ indicates that the solution p, belonging to the approximate Pareto set P, dominates the solution q of the approximate Pareto set Q in a minimisation 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. Note that both $C(P,Q)$ and $C(Q,P)$ have to be considered, since $C(P,Q)$ is not necessarily equal to $1-C(Q,P)$.

We have used boxplots based on the C metric that calculates the dominance degree of the approximate Pareto sets of every pair of algorithms (see Figure 1 and 2). Each rectangle contains ten boxplots representing the distribution of the C values for a certain ordered pair of algorithms in the ten problem instances (P1 to P10). Each box refers to algorithm A in the corresponding row and algorithm B in the corresponding column and gives the fraction of B covered by A ($C(A,B)$).

4.3 Analysis of results

The experimental results obtained by the two MACS variants with and without the diversity mechanism can be seen in the C metric boxplots of Figure 1 and in the HVRvalues in Table 1. Some conclusions can be reached from the analysis of the C metric values:

- Comparing both versions of MACS, the original one with a specific value of q_0 and its counterpart multi-colony extension, we can see that significantly "better"[2] results are provided by the latter MACS with thresholds. It happens regardless of the value of q_0, and it is common in all the problem instances but P5 (heskia). This is because of the nature of that problem instance, whose pseudo-optimal Pareto front is not wide enough. Every solution of this problem instance is found in the central part of the objective space, so the diversity introduced by the filling thresholds is not useful.

[2] When we refer to the best or better performance comparing the C metric values of two algorithms we mean that the Pareto set derived from one algorithm significantly dominates that one achieved by the other. Likewise, the latter algorithm does not dominate the former one to a high degree.

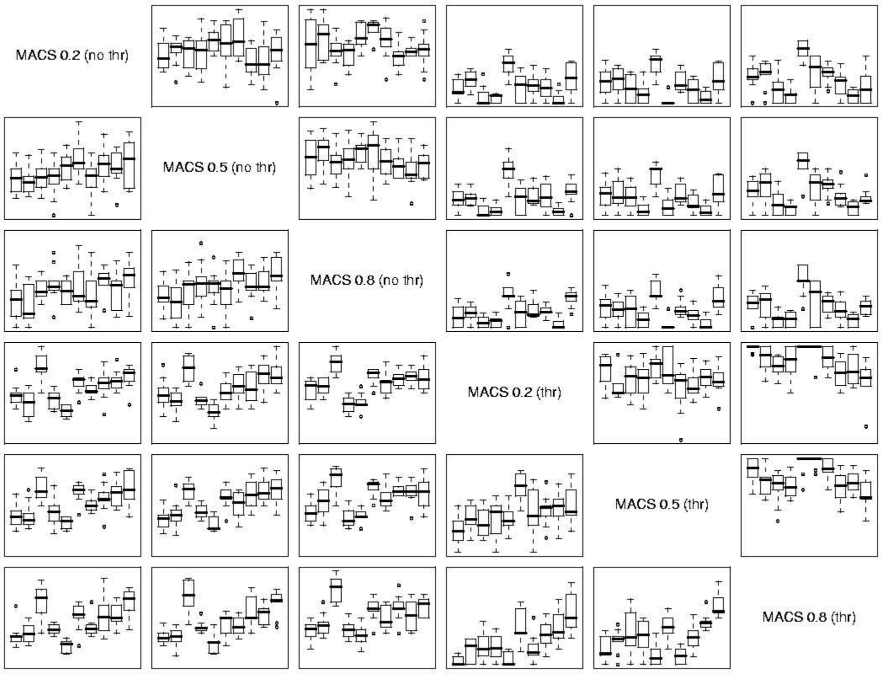

Fig. 1 *C* metric values represented by means of boxplots comparing MACS with and without multi-colony scheme (i.e. variable filling thresholds).

- We find less performance differences with a lower value of q_0. It makes sense since MACS with higher q_0 values gives more importance to a higher intensification in the selection procedure and thus, the Pareto fronts are more similar. Hence, the algorithm does not take advantage of the diversity induced by the thresholds approach.
- If we compare every MACS variant with and without thresholds, regardless of the value of q_0, the conclusion is that MACS 0.2 with thresholds is the best approach. It gets better results than MACS 0.5 and 0.8 with thresholds in every problem instance. It is only dominated by MACS 0.2 and 0.5 without thresholds in P5. Even in a non-common problem instance like P5, results are good enough. Hence, the diversity of the task selection procedure (a low value of q_0 parameter) and the use of variable station filling thresholds are both important to solve the problem appropriately. Nevertheless, if we select MACS 0.8 with thresholds and MACS without thresholds with lower values of q_0 (0.2 and 0.5) to be compared, we can notice that the former algorithm outperforms the latter two in five and six problem instances respectively. On the contrary, the latter two are better in four of them. All of these algorithms have thus quite similar results. Consequently, the variable filling thresholds in isolation are not enough to get a good yield. There is also a demand for diversity in the randomised task selection procedure of the algorithm which requires a good diversification-intensification trade-off.

Table 1 Mean and standard deviation values (in brackets) of the HVR metric for the MACS algorithm. In each problem instance, the best mean value is in bold

A1: MACS 0.2 (without thr.), **A2**: MACS 0.5 (without thr.), **A3**: MACS 0.8 (without thr.)
A4: MACS 0.2 (with thr.), **A5**: MACS 0.5 (with thr.), **A6**: MACS 0.8 (with thr.)

	P1	P2	P3	P4	P5
A1	0.5532 (0.023)	0.6655 (0.009)	0.6418 (0.026)	0.4297 (0.043)	**0.9686** (0.006)
A2	0.5549 (0.019)	0.6600 (0.017)	0.6331 (0.012)	0.4475 (0.034)	0.9660 (0.006)
A3	0.5331 (0.008)	0.6418 (0.014)	0.6172 (0.016)	0.4629 (0.061)	0.9608 (0.007)
A4	**0.9051** (0.01)	**0.8962** (0.013)	**0.8852** (0.020)	0.8176 (0.027)	0.8695 (0.022)
A5	0.8770 (0.009)	0.8839 (0.016)	0.8617 (0.016)	0.7969 (0.024)	0.8471 (0.013)
A6	0.8353 (0.008)	0.8522 (0.010)	0.8285 (0.022)	**0.8191** (0.018)	0.8114 (0.018)
	P6	P7	P8	P9	P10
A1	0.6729 (0.022)	0.8222 (0.315)	0.5522 (0.019)	0.6014 (0.017)	0.7830 (0.019)
A2	0.6833 (0.036)	0.7101 (0.246)	0.5480 (0.013)	0.5968 (0.015)	**0.7819** (0.035)
A3	0.6486 (0.036)	0.6523 (0.239)	0.5365 (0.019)	0.6070 (0.019)	0.7789 (0.014)
A4	**0.8430** (0.022)	**0.972**3 (0.066)	0.8979 (0.011)	**0.8941** (0.011)	0.7674 (0.028)
A5	0.8368 (0.016)	0.8812 (0.058)	**0.8988** (0.013)	0.8829 (0.012)	0.7535 (0.037)
A6	0.7284 (0.054)	0.7330 (0.066)	0.8656 (0.011)	0.8506 (0.013)	0.7067 (0.052)

Table 2 Mean and standard deviation values (in brackets) of the HVR metric for the MORGA. In each problem instance, the best mean value is in bold

A1: MORGA 0.1 (without thr.), **A2**: MORGA 0.2 (without thr.), **A3**: MORGA 0.3 (without thr.)
A4: MORGA 0.1 (with thr.), **A5**: MORGA 0.2 (with thr.), **A6**: MORGA 0.3 (with thr.)

	P1	P2	P3	P4	P5
A1	0.5792 (0.012)	0.6602 (0.018)	0.6017 (0.023)	0.4278 (0.04)	0.9137 (0.007)
A2	0.5779 (0.012)	0.6550 (0.008)	0.6294 (0.042)	0.3957 (0.035)	0.9294 (0.010)
A3	0.5624 (0.026)	0.6789 (0.017)	0.6028 (0.019)	0.4129 (0.017)	**0.9302** (0.009)
A4	0.9258 (0.005)	0.9093 (0.005)	**0.7560** (0.005)	0.8457 (0.020)	0.8642 (0.007)
A5	0.9333 (0.007)	0.9121 (0.005)	0.6528 (0.008)	0.9262 (0.019)	0.8953 (0.038)
A6	**0.9542** (0.007)	**0.9385** (0.007)	0.6488 (0.009)	**0.9366** (0.016)	0.9149 (0.052)
	P6	P7	P8	P9	P10
A1	0.5784 (0.020)	0.6914 (0.223)	0.5176 (0.015)	0.5861 (0.012)	0.7911 (0.026)
A2	0.5909 (0.029)	0.5447 (0.09)	0.5316 (0.022)	0.5807 (0.016)	0.7939 (0.027)
A3	0.6451 (0.043)	0.6730 (0.237)	0.5301 (0.026)	0.5873 (0.017)	0.7994 (0.031)
A4	0.7611 (0.029)	0.7034 (0.260)	0.8769 (0.009)	0.8606 (0.004)	0.8568 (0.018)
A5	0.8361 (0.033)	0.7498 (0.039)	0.8797 (0.008)	**0.8663** (0.004)	0.8726 (0.017)
A6	**0.8847** (0.038)	**0.7466** (0.067)	**0.9011** (0.006)	0.8610 (0.007)	**0.8837** (0.022)

On the other hand, we show the results of the MORGA with and without the diversity mechanism. In Figure 2, the boxplots of the C metric are shown. Similar conclusions can be obtained:

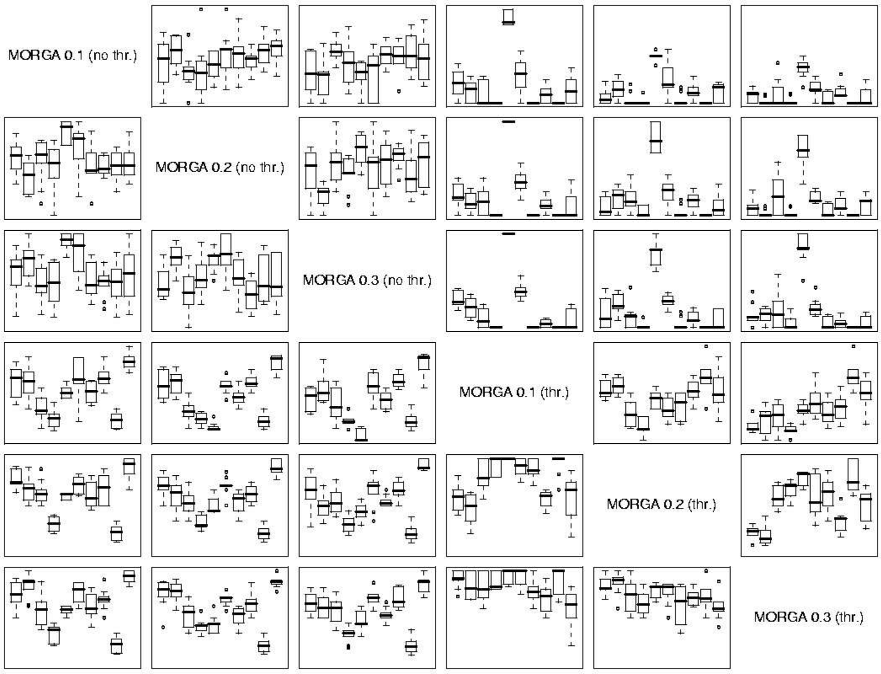

Fig. 2 *C* metric values represented by means of boxplots comparing the MORGA with and without using the variable filling thresholds.

- The MORGA variants with the diversity mechanism almost always achieve better performance than those without it.
- Only in the P5 instance, there are solutions of the MORGA variants with the diversity mechanism which are dominated by the algorithms without the new approach.
- It is clear how the MORGA with $\gamma = 0.3$ is the best of the MORGA variants, and its version with the diversity mechanism the best algorithm.

In general terms, we can draw similar conclusions analysing the HVR metric values (see Tables 1 and 2). They are always higher in variants with thresholds as they better converge towards the true (i.e., pseudo-optimal) Pareto fronts. For example, that is shown in the Pareto fronts of Figure 3 that graphically shows the aggregated Pareto fronts corresponding to P3 and P10 instances for the MACS algorithm and MORGA.

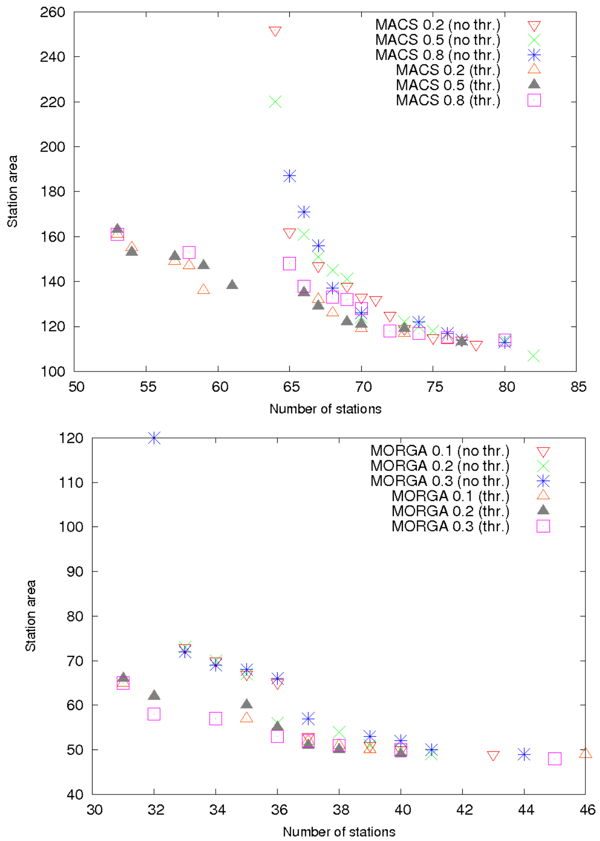

Fig. 3 Pareto fronts of the MACS algorithm and MORGA for the P3 and P10 problem instances respectively.

5 Concluding remarks

In a previous contribution [7] we demonstrated that the use of multiobjective constructive metaheuristics to tackle the TSALBP-1/3, particularly a MACS algorithm, was a good choice. And the consideration of a stochastic procedure to decide when to close a station performed better choice than a pure station-based approach. Nevertheless, that solution still leads to situations where intensification was too high in a specific region of the Pareto front. That is an undesirable situation for the plant managers who should be provided with all the configurations of their contextual interest in the objective space.

To solve this problem, in this contribution we showed a better intensification-diversification trade-off. It could be achieved in a MOACO algorithm by introducing different filling thresholds associated to the ants that build the solution in order to provide a different search behaviour to the different ants in the colony. We also applied a modified version of this new diversity mechanism to a multiobjective randomised greedy algorithm (MORGA).

Ten well-known problem instances of the literature were selected to test our proposal. From the obtained results we have found out that the best yield to globally solve the problem belongs to the new MACS-TSALBP-1/3 algorithm using the multi-colony scheme with $q_0 = 0.2$. Likewise, the MORGA with additional diversity clearly outperforms the results of the basic one.

In the future we aim to consider other multiobjective constructive metaheuristics and apply a local search to increase the current performance.

Acknowledgement

This work is supported by the UPC Nissan Chair and the Spanish Ministerio de Educación y Ciencia under project DPI2007-63026 and by the Spanish Ministerio de Ciencia e Innovación under project TIN2009-07727, both including EDRF fundings.

References

1. Barán, B., Schaerer, M.: A multiobjective ant colony system for vehicle routing problem with time windows. In: 21st IASTED Conference, pp. 97–102. Innsbruck (Germany) (2003)
2. Bautista, J., Pereira, J.: Ant algorithms for a time and space constrained assembly line balancing problem. European Journal of Operational Research **177**, 2016–2032 (2007)
3. Baybars, I.: A survey of exact algorithms for the simple assembly line balancing problem. Management Science **32**(8), 909–932 (1986)
4. Chankong, V., Haimes, Y.Y.: Multiobjective Decision Making Theory and Methodology. North-Holland (1983)
5. Chica, M., Cordón, O., Damas, S., Bautista, J.: Multi-objective, constructive heuristics for the 1/3 variant of the time and space assembly line balancing problem: ACO and randomised greedy. Tech. Rep. AFE-09-01, European Centre for Soft Computing, Asturias (Spain) (2009). (submitted to Information Sciences)
6. Chica, M., Cordón, O., Damas, S., Bautista, J.: A multiobjective GRASP for the 1/3 variant of the time and space assembly line balancing problem. In: 23th International Conference on Industrial, Engineering & Other Applications of Applied Intelligent Systems (IEA-AIE 2010). Cordoba (Spain) (2010). To appear
7. Chica, M., Cordón, O., Damas, S., Bautista, J., Pereira, J.: 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) (2008)
8. Coello, C.A., Lamont, G.B., Van Veldhuizen, D.A.: Evolutionary Algorithms for Solving Multi-objective Problems (2nd edition). Springer (2007)
9. Dorigo, M., Gambardella, L.: Ant colony system: a cooperative learning approach to the traveling salesman problem. IEEE Transactions on Evolutionary Computation **1**(1), 53–66 (1997)
10. Dorigo, M., Stützle, T.: Ant Colony Optimization. MIT Press, Cambridge (2004)
11. Feo, T.A., Resende, M.G.C.: Greedy randomized adaptive search procedures. Journal of Global Optimization **6**, 109–133 (1995)

12. García Martínez, C., Cordón, O., Herrera, F.: A taxonomy and an empirical analysis of multiple objective ant colony optimization algorithms for the bi-criteria TSP. European Journal of Operational Research **180**, 116–148 (2007)
13. Glover, F., Kochenberger, G.A. (eds.): Handbook of Metaheuristics. Kluwer Academic (2003)
14. Middendorf, M., Reischle, F., Schmeck, H.: Multi colony ant algorithms. Journal of Heuristics **8**(3), 305–320 (2002)
15. Scholl, A.: Balancing and Sequencing of Assembly Lines (2nd. Edition). Physica-Verlag, Heidelberg (1999)
16. Scholl, A., Becker, C.: State-of-the-art exact and heuristic solution procedures for simple assembly line balancing. European Journal of Operational Research **168**(3), 666–693 (2006)
17. Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C.M., Grunert da Fonseca, V.: Performance assessment of multiobjective optimizers: an analysis and review. IEEE Transactions on Evolutionary Computation **7**(2), 117–132 (2003)