Memetic Computing manuscript No.
(will be inserted by the editor)

A new diversity induction mechanism for a multi-objective ant colony algorithm to solve a real-world time and space assembly line balancing problem

Manuel Chica · Oscar Cordón · Sergio Damas · Joaquín Bautista

Received: date / Accepted: date

Abstract Time and space assembly line balancing considers realistic multi-objective versions of the classical assembly line balancing industrial problems. It involves the joint optimisation of conflicting criteria such as the cycle time, the number of stations, and/or the area of these stations. The different problems included in this area also inherit the precedence constraints and the cycle time limitations from assembly line balancing problems. The presence of these hard constraints and their multi-criteria nature make these problems very hard to solve. Multi-objective constructive metaheuristics (in particular, multi-objective ant colony optimisation) have demonstrated to be suitable approaches to solve time and space assembly line balancing problems.

The aim of this contribution is to present a new mechanism to induce diversity in an existing multi-objective ant colony optimisation algorithm for the 1/3 variant of the time and space assembly line balancing problem. This variant is quite realistic in the automative industri as it involves the joint minimisation of the number and the area of the stations given a fixed cycle time limit. The performance of our proposal is validated considering ten real-like problem instances. Moreover, the diversity induction mechanism is also tested on a real-world instance from the Nissan plant in Barcelona (Spain).

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.

Manuel Chica, Oscar Cordón and Sergio Damas
European Centre for Soft Computing
33600 Mieres, Spain
E-mail: {manuel.chica,oscar.cordon,sergio.damas}@softcomputing.es

Joaquín Bautista
Nissan Chair - Universitat Politècnica de Catalunya
Barcelona, Spain
E-mail: joaquin.bautista@nissanchair.com

Keywords Time and space assembly line balancing problem · ant colony optimisation · multi-objective 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] [14]. Taking this 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 these stations. In this paper we deal with 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 choice 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 hard 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 [12]. Due to the two aforementioned reasons, i.e., the multi-objective nature of the problem and the need to solve it through constructive algorithms, a sensible choice is to use a Pareto-based multi-objective ACO (MOACO) algorithm [11]. 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 [5–7], 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 [14] that was accomplished. In particular, the approximations to the Pareto fronts obtained showed a 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 induce a new mechanism to avoid that local convergence behaviour. The goal is to induce the generation of more diverse solutions by means of a multi-colony approach [13] based on the use of different station filling rates in the ants' construction procedure. Our MACS-TSALBP-1/3 algorithm with and without the new diversification component will be tested on ten real-like TSALBP-1/3 instances, showing the performance improvement obtained. Furthermore, the designed algorithms will also be applied to a real-world problem instance from the Nissan industry plant in Barcelona.

The paper is structured as follows. In Section 2, the problem formulation and the MACS principles are explained. Then, the proposed multi-colony approach to improve the basic MACS algorithm operation mode 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, an overview of the TSALBP is discussed. Then, the mathematical formulation of the TSALBP-1/3 is detailed. Finally, the main features of the MACS algorithm are briefly described.

2.1 The Time and Space Assembly Line Balancing Problem

The manufacturing of a production item is divided up 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 [14] 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 [15].

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 multi-objective 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:

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

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

subject to:

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

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

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

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

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

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

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 multi-objective 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:

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

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 $p(j)$:

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

Every time an ant crosses edge $< i, j >$, it performs the local pheromone update as follows:

$$\tau_{ij} = (1 - \rho) \cdot \tau_{ij} + \rho \cdot \tau_0 \tag{11}$$

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:

$$\tau_0 = \frac{1}{\hat{f}^0 \,.\, \hat{f}^1} \tag{12}$$

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 (12) 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 $< i, j >$:

$$\tau_{ij} = (1 - \rho) \cdot \tau_{ij} + \frac{\rho}{f^0(S) \cdot f^1(S)} \tag{13}$$

3 A new diversitiy induction mechanism in MACS for the TSALBP-1/3

This section presents our proposal of a new mechanism to induce diversity in a MACS algorithm for the TSALBP-1/3. Section 3.1 reviews our previous approach based on MACS to tackle the TSALBP-1/3. Then, Section 3.2 details our diversity induction mechanism.

3.1 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.

3.1.1 Heuristic information

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

$$\eta_j^0 = \frac{t_j}{c} \cdot \frac{\mid F_j \mid}{\max_{i \in \Omega} \mid F_i \mid} \tag{14}$$

$$\eta_j^1 = \frac{a_j}{UB_A} \cdot \frac{\mid F_j \mid}{\max_{i \in \Omega} \mid F_i \mid} \tag{15}$$

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.

3.1.2 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:

$$\tau_0 = \frac{1}{f^0(S_{time}) \cdot f^1(S_{area})} \tag{16}$$

3.1.3 Randomised station closing scheme and transition rule

Our approach follows a *station-oriented* procedure [14], 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. In order to diversify the search, allowing to build solutions composed of stations with small, medium, and large loads, 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:

$$p\left(closing\ k\right) = \frac{\sum_{i \in S_k} t_i}{c} \tag{17}$$

where S_k represents the subset of tasks assigned to the station k, t_i indicates the operation time required by task i for its execution, and c is the cycle time.

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, the ant chooses 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 MACS transition rule (Equation 9) that chooses among all the candidate tasks at each step, has been modified:

$$j = \begin{cases} \arg\ \max_{j \in \Omega}(\tau_{kj} \cdot [\eta_j^0]^{\lambda\beta} \cdot [\eta_j^1]^{(1-\lambda)\beta}), & \text{if } q \leq q_0, \\ \hat{i}, & \text{otherwise,} \end{cases} \tag{18}$$

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

$$p(j) = \begin{cases} \frac{\tau_{kj} \cdot [\eta_j^0]^{\lambda\beta} \cdot [\eta_j^1]^{(1-\lambda)\beta}}{\sum_{u \in \Omega} \tau_{ku} \cdot [\eta_u^0]^{\lambda\beta} \cdot [\eta_u^1]^{(1-\lambda)\beta}}, & \text{if } j \in \Omega, \\ 0, & \text{otherwise.} \end{cases} \tag{19}$$

3.2 A diversity induction mechanism based on a multi-colony approach for the MACS-TSALBP-1/3 algorithm

The randomised station closing scheme proposed in [5–7] and described in Section 3.1.3 aimed to avoid that all the solutions generated lie in the Pareto front regions with the minimum number of stations. Indeed, our preliminary experiments following a station-oriented procedure without such a scheme provided solutions in a narrow part of the Pareto front (see [6]). That was due to the fact that we initially decided to close every station only if it was full according to the fixed cycle time c, as usual in SALBP and TSALBP applications. The Pareto fronts likewise achieved did not have enough diversity.

However, the randomised station closing scheme proposed to avoid such behaviour entailed an important drawback. Since the closing probability was updated at each construction step and it was proportional to the sum of the overall processing time of the station (i.e., the sum of the processing times of all the tasks assigned to i, see equation 17), there was a low probability of filling stations completely. Notice that, when the randomised station closing scheme was not considered, the solutions were concentrated in the left-most region of the objective space. Thus, they corresponded to the lowest number of stations completely filled and requiring a very high station area. Nevertheless, that region of the objective space could not be achieved when the proposed station closing scheme was introduced. The MACS-based TSALBP-1/3 algorithm proposed did not provide enough diversification in those Pareto front regions.

Therefore, 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. A new diversity induction mechanism randomly deciding when to close the current station taking as a base both the station closing probability distribution and an ant filling threshold α_i can thus be proposed.

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 directly kept opened. Otherwise, the station closing probability distribution is updated using equation 17 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 are no more remaining tasks to be assigned.

Thus, the higher the ants 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, by using different filling thresholds, 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. Hence, these thresholds make the different ants in the colony have a different search behaviour. Thus, the ACO algorithm becomes multi-colony [13].

4 Experimentation

In this section, we first explain the instances and the used parameters for the different algorithms. Then, the performance indicators used to compare the algorithms are commented. Finally, the analysis of the achieved results as well as the application of the algorithms to the real instance of Nissan are described.

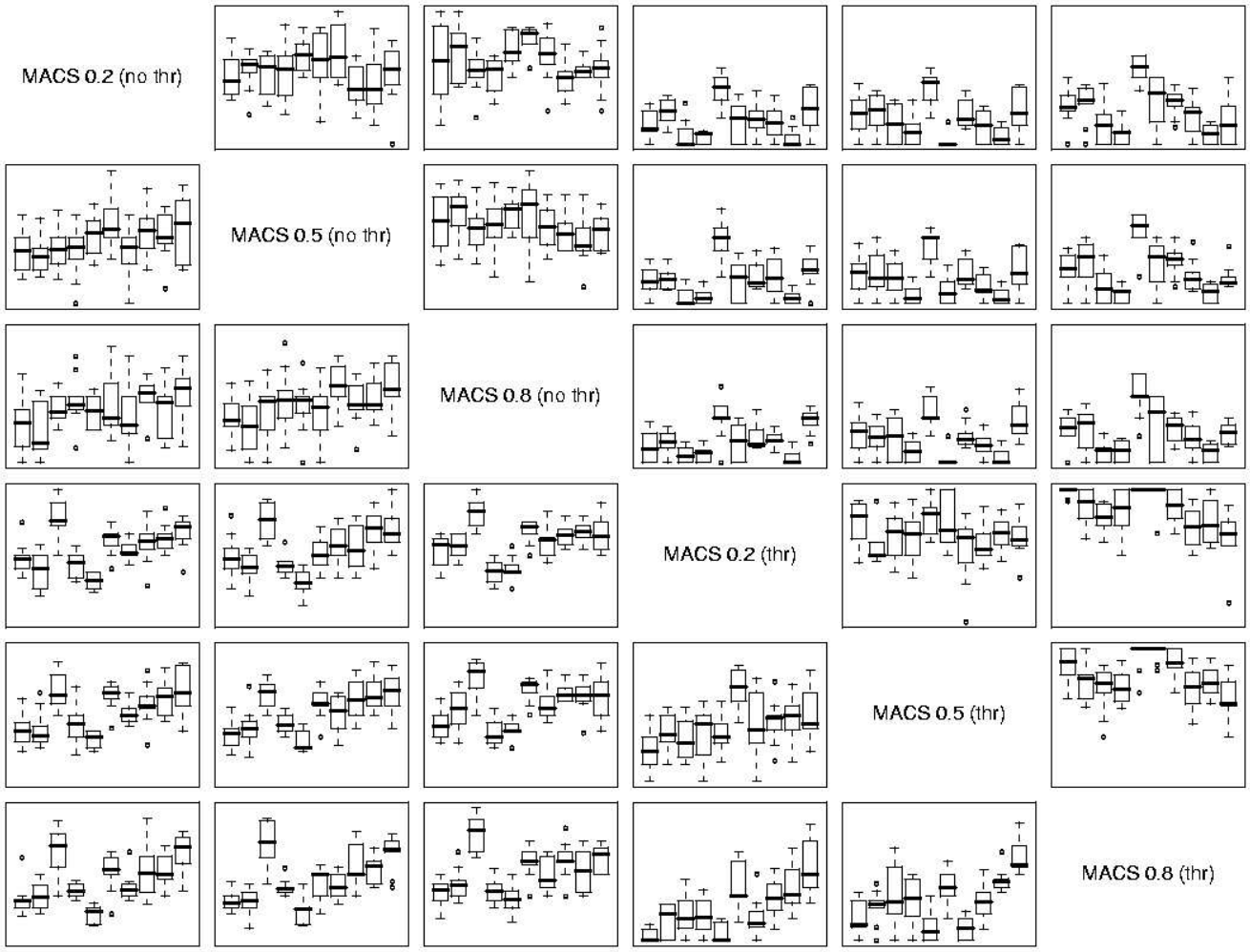

Fig. 1 C performance indicator values represented by means of boxplots comparing MACS with and without multi-colony scheme (i.e. variable filling thresholds).

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]). In addition, we have considered a real-world problem instance corresponding to the assembly process of the Nissan Pathfinder engine, assembled at the Nissan industrial plant in Barcelona (Spain) [2]. All the TSALBP-1/3 instances considered are publicly available at *http://www.nissanchair.com/TSALBP*.

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 parameters concerning our proposal on using different filling thresholds are as follows. There are two ants for each of the five ants' thresholds considered: $\{0.2, 0.4, 0.6, 0.7, 0.9\}$.

4.2 Multi-objective performance indicators

We will consider two different multi-objective performance indicators (wrongly called metrics in the past) [8, 16] to evaluate the performance of the two variants of the MACS-based TSALBP-1/3 algorithm.

Table 1 Mean and standard deviation values (in brackets) of the HVR performance indicator for all the problem instances.

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	0.91 (0.005)	0.91 (0.006)	0.86 (0.008)	0.93 (0.005)	**0.96** (0.002)
A2	0.91 (0.003)	0.91 (0.008)	0.86 (0.006)	0.93 (0.005)	**0.96** (0.002)
A3	0.91 (0.005)	0.91 (0.005)	0.85 (0.005)	0.93 (0.005)	**0.96** (0.001)
A4	**0.99** (0.001)	**0.99** (0.002)	**0.98** (0.001)	**0.97** (0.001)	**0.96** (0.006)
A5	0.98 (0.002)	**0.99** (0.002)	**0.98** (0.001)	**0.97** (0.001)	0.95 (0.003)
A6	0.98 (0.001)	0.98 (0.001)	**0.98** (0.001)	0.97 (0.002)	0.94 (0.005)
	P6	P7	P8	P9	P10
A1	0.89 (0.01)	0.91 (0.008)	0.91 (0.005)	0.87 (0.003)	0.93 (0.007)
A2	0.89 (0.008)	0.91 (0.01)	0.92 (0.008)	0.87 (0.004)	0.93 (0.009)
A3	0.89 (0.008)	0.91 (0.011)	0.92 (0.005)	0.87 (0.004)	0.92 (0.011)
A4	**0.99** (0.001)	**0.99** (0.001)	**0.99** (0.001)	**0.99** (0.001)	**0.99** (0.001)
A5	**0.99**(0)	**0.99** (0.001)	**0.99** (0.001)	**0.99** (0.001)	**0.99** (0.001)
A6	**0.99** (0.005)	**0.99** (0.001)	**0.99** (0.002)	**0.99** (0.001)	**0.99** (0.001)

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

$$HVR = \frac{HV(P)}{HV(P^*)}, \tag{20}$$

where $HV(P)$ and $HV(P^*)$ are the volume (S performance indicator) 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.

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 performance indicator C to compare the obtained Pareto sets two by two based on the following expression:

$$C(P,Q) = \frac{|\{q \in Q\,;\, \exists p \in P : p \prec q\}|}{|Q|}, \tag{21}$$

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 performance indicator that calculates the dominance degree of the approximate Pareto sets of every pair of algorithms (see Figure 1 and Figure 3). 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 performance indicator boxplots of Figure 1 and in the HVR values in Table 1. Some conclusions can be reached from the analysis of the C performance indicator 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"[1] 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

[1] When we refer to the best or better performance comparing the C performance indicator 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.

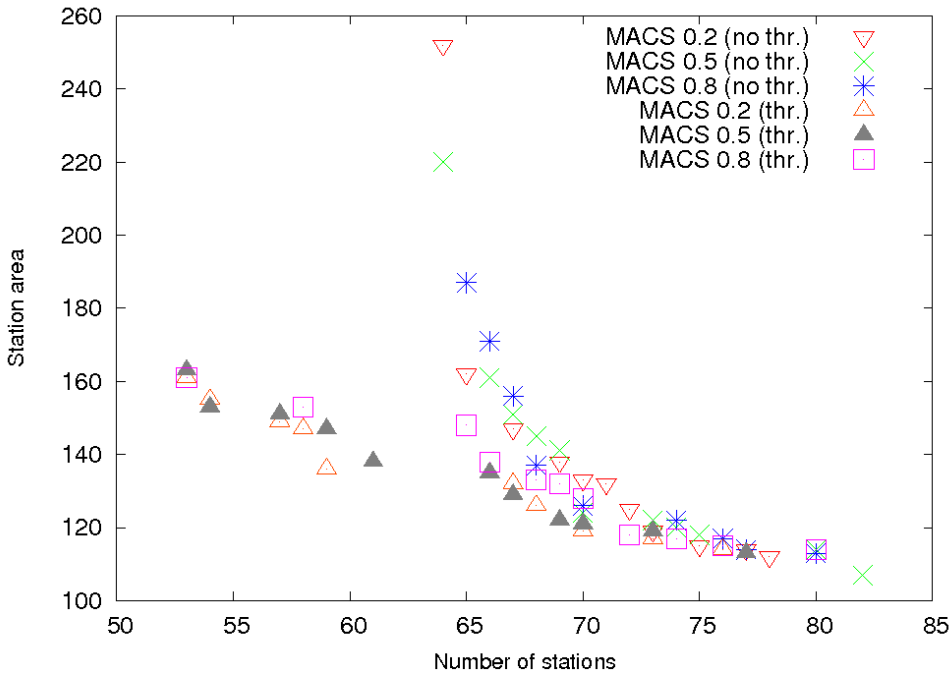

Fig. 2 Pareto fronts for the P3 problem instance.

solution of this problem instance is found in the central part of the objective space, so the diversity introduced by means of the filling thresholds is not useful.

– 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 the latter, results are good enough.

Hence, the diversity of the task selection procedure (a low value of q_0 parameter) as well as 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.

In general terms, we can draw similar conclusions analysing the HVR values (see Table 1). 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 Figure 2 that graphically shows the aggregated Pareto fronts corresponding to P3.

4.4 Application to the real Nissan problem instance

In this section we consider the application of the algorithms to a real-world problem corresponding to the assembly process of the Nissan Pathfinder engine at the plant of Barcelona (Spain). The assembly of these engines is divided into 378 operation tasks, although we have grouped these operations into 140 different tasks. The Nissan instance data is also available at *http://www.nissanchair.com/TSALBP*. For more details about the Nissan instance the reader is referred to [2], where all the tasks and the time and area information are set.

The C performance indicator values are collected in Figure 3 by means of boxplots. Those values for the HVR performance indicator are shown in Table 2. Besides, Figure 4 shows the Pareto fronts of the algorithms for the Nissan instance.

We can observe that, as happened with the ten real-like instances, the variants of the new proposal of MACS with

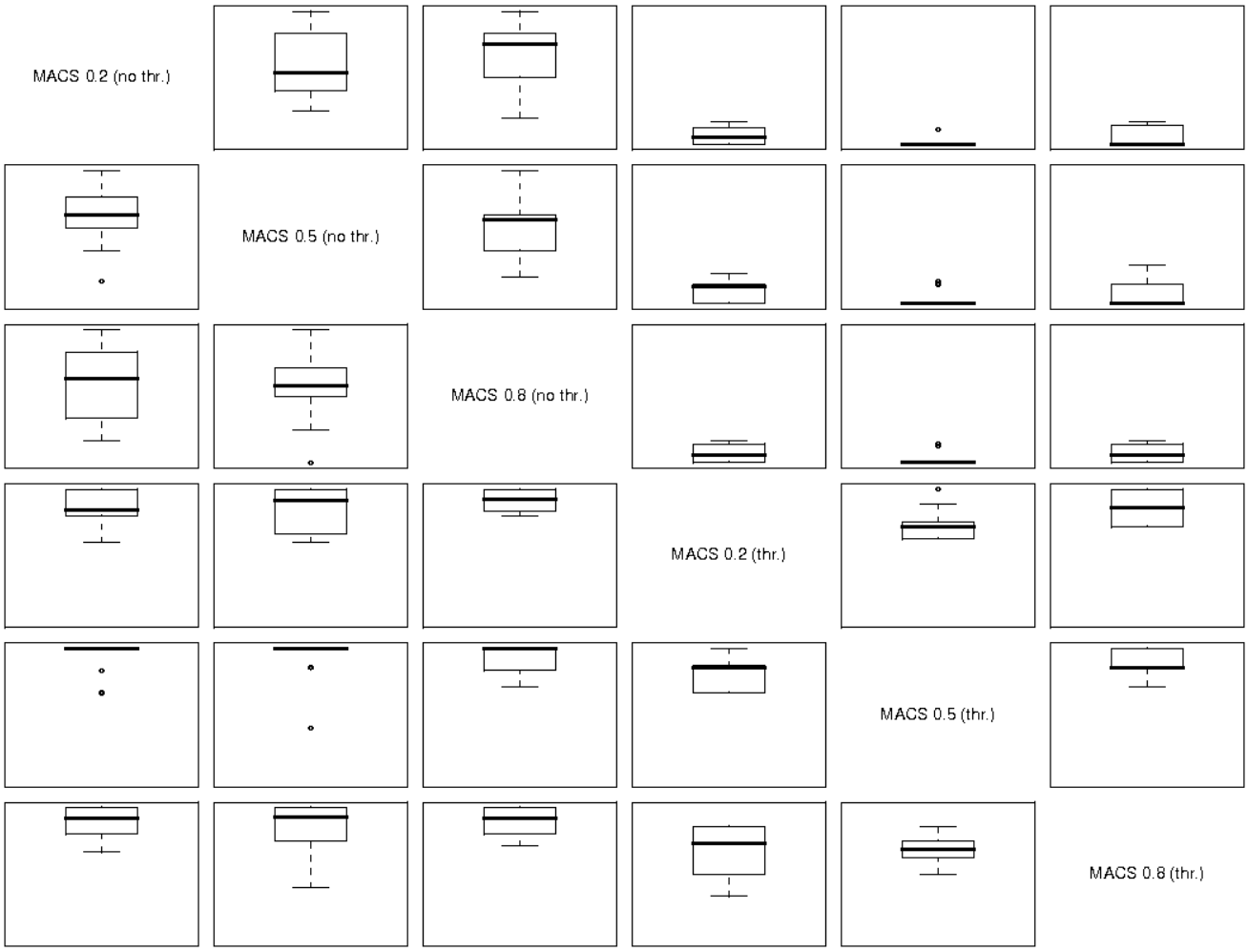

Fig. 3 C performance indicator values represented by means of boxplots comparing MACS with and without multi-colony scheme (i.e. variable filling thresholds) for the Nissan problem instance.

Table 2 Mean and standard deviation values (in brackets) of the HVR performance indicator for the real Nissan problem instance.

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.)

A1	A2	A3	A4	A5	A6
0.55 (0.023)	0.54 (0.020)	0.54 (0.012)	0.94 (0.015)	**0.95** (0.013)	0.93 (0.017)

the diversity mechanism are the best algorithms. The difference is clear observing all the performance indicators. The convergence of MACS with the multi-colony extension is higher than without taking into account this diversity mechanism (see values of the HVR performance indicator in Table 2, showing a great difference (from 0.55 to 0.94)). The boxplots of C also reinforce the existence of this difference (Figure 3).

However, the behaviour of the MACS variants with the multi-colony approach is different with respect to the case of the ten real-like instances. In this case, MACS 0.5 achieves the best performance, even better than the MACS 0.2, which was the best variant for the other instances.

To sum up, as it happened with the other problem instances, the MACS with multi-colony mechanism outperforms the previous version of MACS considering globally all the performance indicators as well as the graphical Pareto front representation. The MACS 0.5 is more suitable for the Nissan problem instance than the 0.2 variant. This is because the real instance needs more intensification in the search since it has particular characteristics.

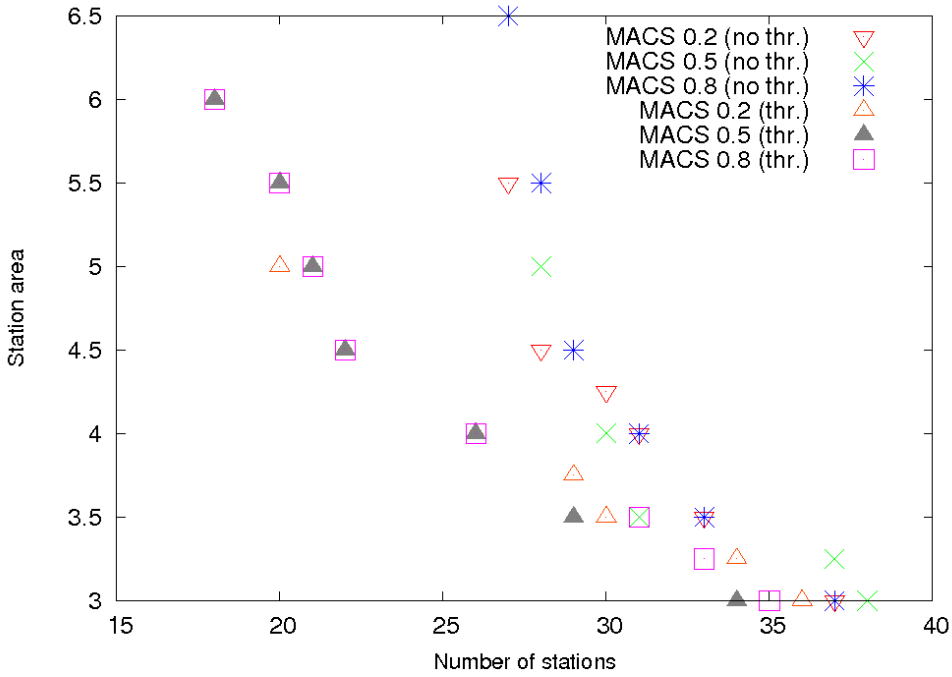

Fig. 4 Pareto fronts for the Nissan problem instance.

5 Concluding remarks

In previous works [5–7] we demonstrated that the use of a MACS algorithm to tackle the TSALBP-1/3 by considering a stochastic procedure to decide when to close a station was a better choice than a pure station-based approach. Nevertheless, that solution still led to situations where intensification was too high in a specific Pareto front region. 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 could be achieved by introducing different filling thresholds associated to the ants that build the solution. This new mechanism aimed to provide a different search behaviour to the different ants in the colony (multi-colony ACO).

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 $q_0 = 0.2$. In addition, the algorithms were applied to a real instance of Nissan, confirming the good performance of the MACS with the multi-colony mechanism. In this case, the best variant was the MACS multi-colony variant with $q_0 = 0.5$, followed by that with the $q_0 = 0.2$.

As future work, we aim to apply a local search to increase the performance of the algorithm. We also aim to consider other MOACO algorithms to solve the TSALBP-1/3.

References

1. B. Barán and M. Schaerer. A multiobjective ant colony system for vehicle routing problem with time windows. In *21st IASTED International Conference*, pages 97–102, Innsbruck (Germany), February 2003.
2. J. Bautista and J. Pereira. Ant algorithms for a time and space constrained assembly line balancing problem. *European Journal of Operational Research*, 177:2016–2032, 2007.
3. I. Baybars. A survey of exact algorithms for the simple assembly line balancing problem. *Management Science*, 32(8):909–932, 1986.
4. V. Chankong and Y. Y. Haimes. *Multiobjective Decision Making Theory and Methodology*. North-Holland, 1983.
5. M. Chica, O. Cordón, S. Damas, and J. Bautista. Multi-objective, constructive heuristics for the 1/3 variant of the time and space assembly line balancing problem: ACO and randomised greedy. Technical Report AFE-09-01, European Centre for Soft Computing, Asturias (Spain), 2009. (submitted to Information Sciences).
6. M. Chica, O. Cordón, S. Damas, J. Bautista, and J. Pereira. 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)*, pages 1454–1461, Málaga (Spain), June 2008.
7. M. Chica, O. Cordón, S. Damas, J. Bautista, and J. Pereira. On using heuristic information in a multi-objective ACO algorithm for the time and space assembly line balancing problem. In M. Middendorf and A. P. Engelbrecht, editors, *Applied Swarm Intelligence*. Series in Studies in Computational Intelligence, Springer, 2009. in press.

8. C. A. Coello, G. B. Lamont, and D. A. Van Veldhuizen. *Evolutionary Algorithms for Solving Multi-objective Problems (2nd edition)*. Springer, 2007.
9. M. Dorigo and L. Gambardella. Ant colony system: a cooperative learning approach to the traveling salesman problem. *IEEE Transactions on Evolutionary Computation*, 1(1):53–66, 1997.
10. M. Dorigo and T. Stützle. *Ant Colony Optimization*. MIT Press, Cambridge, 2004.
11. C. García Martínez, O. Cordón, and F. Herrera. 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.
12. F. Glover and G. A. Kochenberger, editors. *Handbook of Metaheuristics*. Kluwer Academic, 2003.
13. M. Middendorf, F. Reischle, and H. Schmeck. Multi colony ant algorithms. *Journal of Heuristics*, 8(3):305–320, 2002.
14. A. Scholl. *Balancing and Sequencing of Assembly Lines (2nd. Edition)*. Physica-Verlag, Heidelberg, 1999.
15. A. Scholl and C. Becker. State-of-the-art exact and heuristic solution procedures for simple assembly line balancing. *European Journal of Operational Research*, 168(3):666–693, 2006.
16. E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. Grunert da Fonseca. Performance assessment of multiobjective optimizers: an analysis and review. *IEEE Transactions on Evolutionary Computation*, 7(2):117–132, 2003.