Incorporating Preferences to a Multi-objective Ant Colony Algorithm for Time and Space Assembly Line Balancing*

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

[1] European Centre for Soft Computing, Mieres (Asturias), Spain
{manuel.chica,oscar.cordon,sergio.damas}@softcomputing.es
[2] Universitat Politècnica de Catalunya, Barcelona, Spain
{joaquin.bautista,jorge.pereira}@upc.edu

Abstract. We present an extension of a multi-objective algorithm based on Ant Colony Optimisation to solve a more realistic variant of a classical industrial problem: Time and Space Assembly Line Balancing. We study the influence of incorporating some domain knowledge by guiding the search process of the algorithm with preferences-based dominance. Our approach is compared with other techniques, and every algorithm tackles a real-world instance from a Nissan plant. We prove that the embedded expert knowledge is even more justified in a real-world problem.

1 Introduction

The Time and Space Assembly Line Balancing Problem (TSALBP) [1] belongs to a family of academic problems which came up with the name of Simple Assembly Line Balancing Problem (SALBP) [2]. It considers an additional space constraint to become a simplified but closer version to real-world problems. In this paper we tackle the 1/3 variant of the TSALBP, which tries to minimise the number of stations and their area for a given product cycle time, a very complex and realistic multi-criteria problem in the automotive industry.

In our previous work [3], we successfully solved the TSALBP-1/3 by means of multi-objective and constructive approaches. We selected the Multiple Ant Colony System (MACS) algorithm [4] because of its good performance to develop a multi-objective ant colony optimisation (MOACO) proposal [5] for it. In the current contribution we aim to extend this proposal by incorporating problem-specific information provided by the plant experts in the form of preferences in the decision variable space which will allow us to guide the search. We will use an *a priori* approach to incorporate these expertise although *a posteriori* and *interactive* procedures have been mostly applied in the Evolutionary Multi-objective Optimisation and Operations Research communities [6].

* UPC Nissan Chair as well as the Spanish Government partially funded this work by means of PROTHIUS-II project: DPI2007-63026 including EDRF fundings.

M. Dorigo et al. (Eds.): ANTS 2008, LNCS 5217, pp. 331–338, 2008.

This improved model aims to reduce the size of the Pareto set and, collateraly, increase the quality of the Pareto front by increasing the MACS convergence capability and focusing on actually interesting and useful solutions for the decision-maker using problem-specific knowledge.

Our MACS algorithm is applied both to academic real-like problem instances and to a real-world instance which has more peculiarities than the others. It corresponds to the assembly process of the Nissan Pathfinder engine, developed at the Nissan industry plant of Barcelona (Spain).

The paper is structured as follows. In Section 2, the problem formulation, and our previous MOACO proposal are explained. The preferences based on domain knowledge are detailed in Section 3. Experiments and analysis of results are shown in Section 4. In Section 5, some concluding remarks are discussed.

2 Preliminaries

In this section the problem preliminaries and our previous MOACO proposal are presented. First, an overview of the assembly line balancing problem is discussed. Then, 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. Each station k is assigned to a subset of tasks S_k ($S_k \subseteq V$), called its workload. A task j must be assigned to one station k.

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 [2] focuses on grouping tasks in workstations by an efficient and coherent way. There is a large variety of exact and heuristic problem-solving procedures [7], even an hybrid ant colony algorithm-beam search approach has been recently developed in [8].

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 [1]. 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 optimization 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. Besides, the search for the best number of stations and areas makes sense if we want to reduce costs and make workers' day better by setting up less crowded stations. For more information we refer to [3].

2.2 A MACS Algorithm to Solve TSALBP-1/3

In this section, a brief summary of our previous multi-objective proposal based on the MACS algorithm is presented. The complete MACS description can be found in [4], and our proposal is detailed in [3].

Since the number of stations is not fixed, we use a constructive and station-oriented approach (as usually done for the SALBP [7]) to face the precedence problem. Thus, our algorithm will open a station and select one task till a stopping criterion is reached. Then, a new station is again opened to be filled.

Experiments showed that the performance is better if MACS is only guided by the pheromone trail information. Such 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 $(station_k, task_j)$, $k = 1...n$, $j = 1...n$, so our pheromone trail matrix has a bi-dimensional nature. We used two station-oriented single-objective greedy algorithms to obtain the initial pheromone value τ_0.

In addition, we introduced a new mechanism in the construction algorithm to close a station according to a probability distribution, given by the filling rate of the station: $p(closing) = \left(\sum_{\forall i \in S_k} t_i\right)/c$. It helps the algorithm to reach more diverse solutions from closing stations by a deterministic process. The probability is computed at each construction step so its value is progressively increased. Then, a random number is generated to decide if the station is closed.

Besides, there is a need to achieve a better intensification-diversification trade-off. That was achieved by introducing different filling thresholds associated to the ants. These thresholds make the different ants have a different search behaviour. The higher the ant's threshold, the more filled the station will be (there will be less possibilities to close the station during its creation 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 appropiately spreading the generated solutions.

3 Adding Preferences Based on Domain Knowledge

In this work, we have included some expert information into the dominance definition considered by the algorithm (using an *a priori* approach). This addition of domain knowledge allows us to derive a Pareto set composed of a smaller number of more likely solutions for the final user as well as to induce a better convergence to the actual Pareto front as a collateral effect [6]. In this section we describe how we have modified the original dominance definition including specific preferences given by the expert for the current problem.

We have applied these preferences based on domain knowledge when there are some solutions with the same objective values [1], i.e. the same value for area and number of stations (A and m) for a fixed cycle time (c). Decision between two solutions with different c, A and m values is made by using the traditional dominance relationship.

Since the number of solutions of this kind can be quite large, thus difficulting the final choice of the best one for each combination of objective values to the user, it is important to establish a criterion based on the expert's preferences to choose, among those solutions, the one having the best quality according to the industrial context. Thus, we can discriminate between two solutions with same c, A and m values considering that:

(a) The workload of the plant must be well-balanced in every station. For m stations, all the station workload times $t(S_k)$ for $k = 1..m$ are alike. Due to this criterion, and considering the same number of employees per station, a well-balanced plant provides less human resources' conflicts. Likewise, it eliminates the need of shifts among the workers of the different stations.
(b) The required space for worker's instruments must be as similar as possible. This preference aims to offer solutions in which every worker has the same working conditions. If we reduce the extra effort in movements and the crowding sensation, it will eliminate industrial disputes.

The latter criteria are formulated through the following preference measures:

$$P_t(\sigma) = \sum_{k=1}^{m}(c - t(S_k))^2 \qquad P_a(\sigma) = \sum_{k=1}^{m}(A - a(S_k))^2$$

where σ represents a solution with known c, A and m values. $S_k, \forall k = 1...m$ is the assignment of the different tasks to the k-th station in σ.

Bearing in mind these measures, the following preferences-based dominance relations can be considered:

Definition 1. *A solution σ_1 is said to partially dominate another solution σ_2 with respect to time, both with identical c, A and m values, if $P_t(\sigma_1) < P_t(\sigma_2)$.*

[1] Notice that preferences are usually applied to guide the search to the specific Pareto front zones that are interesting for the user (i.e. in the objective space) while, in our case, it is applied on the decision variable space to reduce the number of solutions considering different task assignments presenting the same objective values.

Table 1. Used parameter values

Parameter	Value	Parameter	Value
GENERAL		**MACS/ACS**	
Number of runs	10 (except for ACS)	Number of ants	10
Max. run time	900 seconds	β	2
PC Specs.	Intel Pentium^{TM} D	ρ	0.2
	2 CPUs at 2.80GHz	q_0	0.2
OS	CentOS Linux 4.0	Ants' thresholds	$\{0.2, 0.4, 0.6, 0.7, 0.9\}$
	GCC 3.4.6		(2 ants per threshold)
MORGA		**ACS**	
α_{MORGA}	0.3	Number of runs	11
			(one for each α_{ACS} value)
Diversity thresholds	$\{0.2, 0.4, 0.6, 0.7, 0.9\}$	α_{ACS} for the objective aggregation	$\{0, 0.1, ..., 0.9, 1\}$

Definition 2. *A solution σ_1 is said to partially dominate another solution σ_2 with respect to space, both with identical c, A and m values, if $P_a(\sigma_1) < P_a(\sigma_2)$.*

Definition 3. *A solution σ_1 is said to completely dominate another solution σ_2 with respect to time and space, both with identical c, A and m values, if:*

$$([P_t(\sigma_1) \leq P_t(\sigma_2)] \wedge [P_a(\sigma_1) < P_a(\sigma_2)]) \vee ([P_t(\sigma_1) < P_t(\sigma_2)] \wedge [P_a(\sigma_1) \leq P_a(\sigma_2)])$$

4 Experiments

We present two different algorithms, MORGA and ACS, to compare our MACS algorithm with them. Then, the considered experimental setup is showed. We finally lay out the developed experimentation and discuss the obtained results considering multi-objective metrics and some graphics.

4.1 Other Approaches to TSALBP-1/3

Apart from the MACS algorithm, we have also designed a modified ACS algorithm and a multi-objective randomised greedy algorithm (MORGA), already proposed in [3]. In both cases, we use the same constructive approach, non-dominated solution archive, and all the remaining multi-objective mechanisms than in our MACS algorithm. The ACS algorithm has been developed according to the original version [9] but including a weighted aggregation of the two objectives. An approximate Pareto set is built by fusing all the obtained solutions of the different runs of the algorithm. On the other hand, MORGA has a diversification generation mechanism that behaves similarly to a GRASP construction phase. We use an α_{MORGA} parameter which stands for the diversification-intensification trade-off control parameter in the decision step of the algorithm. The most suitable value for α_{MORGA} is 0.3 (see [3]), the one used in this work.

4.2 Problem Instances and Parameter Values

Five real-like problem instances have been selected for the experimentation (see Table 2). Originally, these instances were SALBP-1 instances [2] only having time

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

Table 2. Mean and standard deviation values ($\bar{x}(\sigma)$) of the unary metrics for barthol2, barthold, lutz2, nissan, scholl and weemag instances

	barthol2	**barthold**	**lutz2**	**nissan**	**scholl**	**weemag**
	No. of non-dominated solutions					
MORGA	12.1 (1.5)	2437.2 (578.3)	1227.3 (652.7)	809.9 (103.6)	595.4 (67.5)	10.8 (2.3)
ACS	5 (0)	2 (0)	5 (0)	3 (0)	3 (0)	5 (0)
MACS	13.5 (2.8)	12 (1.4)	268.9 (22.4)	571.9 (81.1)	50.6 (85.7)	15.6 (4.4)
MACS prefs.	10.8 (1.5)	12 (1.2)	7.6 (1.0)	7.2 (0.8)	13 (2.1)	7.9 (1.2)
	No. of different Pareto front solutions					
MORGA	7.1 (0.5)	9.3 (1.6)	7.4 (0.8)	7.6 (0.7)	3.9 (0.7)	6.7 (0.9)
ACS	5 (0)	2 (0)	5 (0)	3 (0)	3 (0)	5 (0)
MACS	12.8 (2.8)	11 (0.9)	6.7 (0.6)	7.6 (1.0)	14.6 (2.0)	8.2 (1.5)
MACS prefs.	10.8 (1.5)	12 (1.2)	7 (0.8)	7.2 (0.8)	13 (2.1)	7.8 (1.2)
	Metric S					
MORGA	393060.69 (28.2)	711632.81 (58.3)	26115 (178.0)	8815.17 (7.9)	16337440 (752.6)	63679.2 (21.5)
ACS	390161 (0)	709528 (0)	26470 (0)	8713 (0)	16458204 (0)	65062 (0)
MACS	391719.09 (1204.8)	725348.19 (2127.4)	26027.3 (3.5)	8889.75 (0.7)	16550586 (6318.0)	65148.1 (5.7)
MACS prefs.	391410.59 (166.4)	726088 (2202.9)	26071.1 (135.7)	8864.45 (31.9)	16552952 (7248.2)	65151.6 (17.5)
	Metric M2*					
MORGA	5.82 (0.7)	7.29 (1.0)	6.89 (0.7)	6.73 (0.3)	3.85 (0.6)	6.35 (0.9)
ACS	4 (0)	2 (0)	5 (0)	3 (0)	2 (0)	4.5 (0)
MACS	10.86 (2.1)	9.49 (0.6)	6.21 (0.5)	6.88 (0.8)	11.51 (1.4)	7.46 (1.3)
MACS prefs.	9.38 (1.2)	10.19 (1.0)	6.62 (0.7)	6.54 (0.7)	10.09 (1.7)	7.15 (1.1)
	Metric M3*					
MORGA	80.6 (0.2)	684.09 (0)	22.45 (3.7)	21.11 (1.6)	2163.08 (0)	20.56 (1.7)
ACS	58.6 (0)	712.08 (0)	21.21 (0)	8.25 (0)	3563.24 (0)	24.19 (0)
MACS	61.99 (12.9)	407.91 (21.0)	19.73 (0.6)	21.12 (1.3)	1645.49 (38.7)	24.61 (1.0)
MACS prefs.	64.82 (6.6)	403.31 (23.3)	20.27 (0.9)	19.62 (2.6)	1658.79 (40.8)	24.39 (1.6)
	No. of applications of preferences-based dominance					
MACS prefs.	8.3 (3.0)	5.6 (2.9)	478.1 (105.4)	935.4 (231.4)	240.2 (145.1)	39.5 (18.2)

information. However, we have created their area information by reverting the task graph to make them bi-objective (as done in [1]). Apart from these real-like instances, we have considered a real-world problem corresponding to the assembly process of the Nissan Pathfinder engine, developed at the Nissan industry plant of Barcelona (Spain). See [1] for additional information on this problem instance.

The MACS algorithm and MORGA have been run ten times with ten different seeds for each of the five real-like and the Nissan instances. The ACS algorithm has been run eleven times with different values for the α_{ACS} parameter in order to spread all the extent of the Pareto front. Every considered parameter value is shown in Table 1.

4.3 Results Analysis

We have applied some multi-objective unary metrics to measure the performance of the different approaches: the number of total and different (in the objective

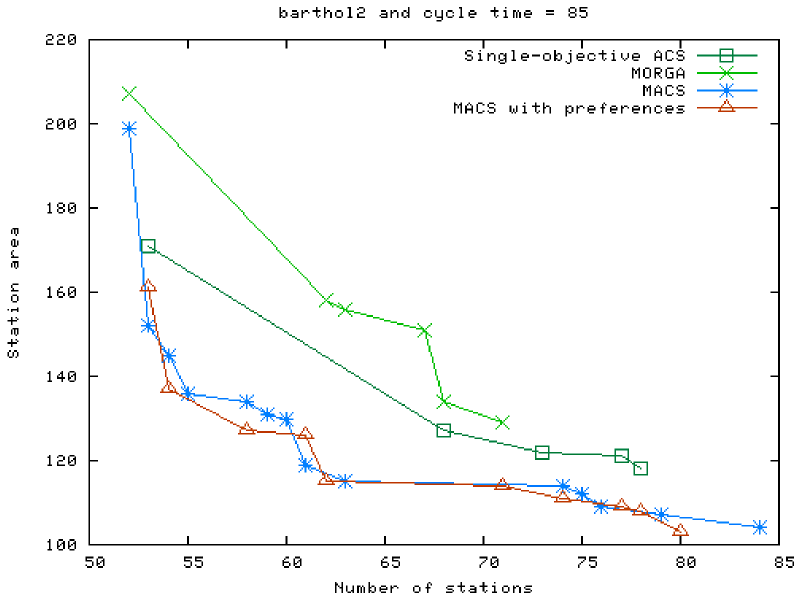

Fig. 1. Pareto fronts for the `barthol2` problem instance

vectors) Pareto solutions returned by each algorithm, and the S, $M2^*$ and $M3^*$ metrics [6]. S, the size of the space covered, measures the volume enclosed by the Pareto front, $M2^*$ evaluates the distribution of the solutions and $M3^*$ evaluates the extent of the obtained Pareto fronts [3]. In addition, the number of applications of the preferences-based dominance criterion for each problem is shown in Table 2. We can also observe the obtained values for the different unary metrics. The different variants of the MACS algorithm achieve better and vaster Pareto fronts than MORGA and ACS in almost every case. Mainly, they obtain more diverse Pareto fronts with a better convergence. The single-objective ACS algorithm is the worst choice because it only achieves few solutions in the Pareto set. However, the values of $M3^*$ for ACS are higher than MACS. This behaviour can be explained since ACS convergence to the actual Pareto front is not good enough, above all, in its most left part.

The preferences-based MACS variant shows the best convergence and reduces the number of non-dominated solutions with the same objective values as expected while keeping a similar value of different solutions. In some cases, this reduction is quite important (see `nissan` instancee, from an average of 571.9 solutions to 7.2). We should highlight that the real-world instance of Nissan is the most appropriate to use preferences based on domain knowledge. Indeed, the number of applications of the preferences-based dominance is the highest.

The graphical representation of the returned aggregated Pareto fronts for the `barthol2` instance is shown in Figure 1. We can arrive to the same previous

[3] $M1^*$ has not been applied because we do not know the optimal Pareto fronts.

conclusions by observing it. MACS variants are the best approaches, and even MACS with preferences achieves a better convergence. MORGA and ACS are not able to reach all the Pareto front surface, getting only few solutions. We have only included the Pareto front of this problem instances for the lack of space but pretty similar behaviours are obtained in the remainder. For the same reason, the binary metric C [6] is not included here, but their values and boxplots can be found in an external appendix at `http://www.nissanchair.com/TSALBP/`

5 Concluding Remarks

An existing MOACO proposal has been extended by introducing preferences based on domain knowledge to tackle the TSALBP-1/3. Other competitive alternatives to baseline the performance of our MOACO proposal, MORGA and a modified ACS, have been used. From the obtained results we have found out that the MACS algorithm gets better results than MORGA and ACS in every considered metric. It has a better convergence, distribution and number of solutions. The enrichment of MACS with domain knowledge provides excellent results, reducing the number of solutions in the Pareto set which have the same objective values and getting a better convergence. Our improved proposal is especially suitable for real-world problems like Nissan's one.

References

1. 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)
2. Scholl, A.: Balancing and Sequencing of Assembly Lines, 2nd edn. Physica-Verlag, Heidelberg (1999)
3. Chica, M., Cordón, O., Damas, S., Bautista, J., Pereira, J.: 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-08-01, European Centre for Soft Computing, Asturias (Spain) (2008)
4. Barán, B., Schaerer, M.: A multiobjective ant colony system for vehicle routing problem with time windows. In: 21st IASTED Conf., Germany, pp. 97–102 (2003)
5. 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)
6. Coello, C.A., Lamont, G.B., Van Veldhuizen, D.A.: Evolutionary Algorithms for Solving Multi-objective Problems, 2nd edn. Springer, Heidelberg (2007)
7. Scholl, A., Becker, C.: State-of-the-art exact and heuristic solution procedures for simple assembly line balancing. European Journal of Operational Research 168, 666–693 (2006)
8. Blum, C., Bautista, J., Pereira, J.: Beam-ACO applied to assembly line balancing. In: Dorigo, M., Gambardella, L.M., Birattari, M., Martinoli, A., Poli, R., Stützle, T. (eds.) ANTS 2006. LNCS, vol. 4150, pp. 96–107. Springer, Heidelberg (2006)
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)