Postprint : Bautista, J. & Alfaro-Pozo, R. Prog Artif Intell (2017). doi:10.1007/s13748-017-0110-z

Free and regular mixed-model sequences by a linear program-assisted hybrid algorithm GRASP-LP

Joaquín Bautista [1], Rocío Alfaro-Pozo [2,1]

[1]Research Group OPE-PROTHIUS. ETSEIB Universitat Politècnica de Catalunya. Avda. Diagonal, 647, 7th floor, 08028 Barcelona, Spain.
[2]EAE Business School. Aragó, 55, 08015, Barcelona, Spain.
joaquin.bautista@upc.edu; ralfaro@eae.es

Abstract. A linear program-assisted hybrid algorithm (GRASP-LP) is presented to solve a mixed-model sequencing problem in an assembly line. The issue of the problem is to obtain manufacturing sequences of product models with the minimum work overload, allowing the free interruption of operations at workstations and preserving the production mix. The implemented GRASP-LP is compared with other procedures through a case study linked with the Nissan' Engine Plant from Barcelona.

Keywords: GRASP; Linear programming; Sequencing; Mixed-model assembly lines: Production mix preservation

1 Preliminaries

Flexibility is the paradigm of the vast majority of the current production systems. Today production systems must be able to manufacture different versions of a product without physical changes at modules or workstations and with negligible setup times between different-type consecutive units; furthermore, they must respond quickly to any variation in the production plan. For this reason flexibility is what makes it important the sequencing problem.

This flexibility is crucial at many manufacturing sectors, such as the Automotive, where production is carried on mixed-model assembly lines and the product mix changes frequently. This leads to the two main problems of this type of assembly lines: the balancing problem and the sequencing problem.

Balancing problem appears in first place and it consists of assigning efficiently the set of assembly tasks for a product into the set of workstations arranged in series. The resulting line's configuration must meet the coherent order of tasks, and the set of restrictions linked with the task-attributes, such as the processing time, the required space and the involved risk [1].

Once the line is configured and the demand plan is defined, the sequencing problem appears. This problem focuses on determining the manufacturing order of prod-

ucts according to different criteria, such as the production maximization given the available time to carry out the all demand plan [2].

As has already been evoked, the sequencing problem can respond to different productive concerns [3]; among the most common, we find the following:

o1. Maximizing the completed product units at the assembly line, reducing simultaneously, the useless time of operators, the unnecessary waiting and the production loses caused by the over workloads at the workstations [4].
o2. Minimizing the number of broken restrictions by the sequence –solution– given several determinants of technological and ergonomic nature that may affect some especial components of the assembling [5].
o3. Maintaining the manufacturing-product and component-consumption rates as constant as possible in order to minimize the maximum stock levels of components [6].

In view of the most common objectives for sequencing problem and taking as reference the work by Bautista, Alfaro-Pozo and Batalla (2016) [7], this paper lies in a specific sequencing problem, the MMSP-W (Mixed Model Sequencing Problem with Workload Minimization).

The problem aims to establish a bijective application between the elements of a set, named T, of manufacturing cycles (which are enumerated t $(t = 1, .., T)$), and the elements of a set, named Ψ, of products (with T products). The elements of Ψ can be grouped in exclusive classes that are denoted as ψ_i and they meet the following: $\Psi = \bigcup_{i \in I} \psi_i$ and $\psi_i \cap \psi_{i'} = \emptyset, \forall \{i, i'\} \in I$; where I is the set of product types (which are enumerated as i $(i = 1, .., n)$).

To complete each product type, $i \in I$, it is required a processing time, $p_{i,k}$ $(i \in I, k \in K)$, that is measured at normal activity or work pace (activity factor: $\alpha^N = 1$), at each workstation, k, from the set of workstations of the assembly line, K, (which is enumerated as k $(k = 1, .., m)$).

Obviously, differences between classes, ψ_i, (SUVs –Sport Utility Vehicle–, vans, trucks…) mean heterogeneous processing times, $p_{i,k}$. However, the time allowed for processors (operators and robots) to perform any operation corresponding to any product type and carried out at any workstation is always the same. This time is named cycle time, it is denoted as c and it is also measured at normal activity.

Discrepancies between the cycle time and the processing times lead to two possible situations for the processors of workstations:

s1. Standby status with useless time: downtime between the finalization instant of one operation and the start of the next operation because the product is not ready.
s2. Lock status by work overload: processors do not have enough time to complete the operation.

The last situation, s2, may be occasionally moderated at the k $(k = 1, .., m)$ workstation by granting a time greater than the cycle time, c, to each processor, i.e., by

allowing a time window, l_k ($l_k > c$), to complete the product unit. Obviously, the said concession will reduce the available time of the operator to work on the next product unit. And, accordingly, the available time to work at the next workstation $(k+1)$ on the retained product unit, after its release, will decrease.

Despite processors dispose of the time window, the time may not enough to complete the operation and therefore, the operation should be interrupted. This interruption on an unfinished product unit can be made in two ways:

i1. Forced interruption: it occurs when the operator reach the time window limit, l_k, at its workstation without completing the corresponding processing time.
i2. Free interruption: it occurs when the product unit is released even though the operation is not completed, before the operator reaches the limit of the time window. Obviously, if the time window limit is reached the operation is also interrupted.

In either case, forced or free interruption, the final purpose of the MMSP-W is to obtain a sequence of products that minimizes the total work overload of the assembly line or, alternatively and equivalently, maximizes the total completed work (see Theorem 1 in [3]).

In addition to the heterogeneous processing times of operations, the mix of product types or models also produces variations in the consumption of components. These variations are an undesirable aspect in production systems governed by the Just in Time –JIT– [6] ideology, as occurs in the automotive sector, where manufacturing sequences with regular consumption of components are desirable. This desirable regularity property in JIT environments is favored by the objective (o3) of sequencing problems, which focuses on reaching sequences with constant production and component-consumption rates.

Based in this premise, we address a variant of the MMSP-W that combines the objectives o1 and o3 in order to adapt the problem to real world environments. Accordingly, we study a mixed-model sequencing problem with the objective of maximizing productivity, reducing the useless time of operators and regularizing the production by means of preserving the production mix in the manufacturing sequence (*pmr*). Besides, unlike [7], in this paper the operations can be freely interrupted (i2). Specifically, the objective o.1 is represented by the objective functions (work overload, W, and useless time, U), the objective o.3 is incorporated into the problem by production mix restrictions (*pmr*) and, finally, the condition i2 is considered by the introduction of some inequalities. This variant of the sequencing problem is called MMSP-W/pmr/free.

Keeping in mind the application of the MMSP-W/pmr/free problem through a case study that is inspired in the BCN Nissan's Engine Plant, a linear programing-assisted hybrid algorithm is implemented. Particularly, a GRASP algorithm (Greedy Randomized Adaptive Search Procedure) to obtain sequences with minimum work overload and a linear program to include regularity and free interruption of operations are designed and implemented in this paper. Besides, in order to assess the performance of

the proposed hybrid procedure the results are compared with those obtained in previous researches that are the state of art of the problem under study: the Bounded Dynamic Programming with linear programing assistance (BDP-2) [10] and with Mixed Integer Linear Programming (MILP) [9].

Accordingly, the paper is structured as follows: the MMSP-W/pmr/free is formulated in section 2; section 3 describes the GRASP algorithm and the linear program that assists GRASP and improves the solution given by the first; the case study is presented in section 4, showing the results given by the three assessed procedures; finally, the conclusions are collected in section 5.

2 The MMSP-W with production mix preservation and free interruption of operations

Given:

a. The sets type of products ($I: i = 1, .., |I|$) and workstations ($K: k = 1, .., m$).
b. The cycle time, c, the temporal windows, l_k ($k \in K$), the number of processors assigned to each workstation, b_k ($k \in K$), and the processing times, $p_{i,k}$ ($i \in I \wedge k \in K$) of operations, at normal activity.
c. The demand plan $\vec{d} = (d_1, \dots, d_n)$, where d_i is the amount of units of type $i \in I$; and the production mix vector, $\vec{\lambda} = (\lambda_1, \dots, \lambda_n)$, where λ_i is the proportion of the model $i \in I$ in the plan, fulfilling: $\vec{\lambda} = \vec{d}/D$ y $T \equiv D = \sum_{\forall i} d_i$.

We formulate the basic MMSP-W/pmr/free as follows:

$$W\big(\pi(T)\big) = \sum_{t=1}^{T} \sum_{k=1}^{m} b_k w_{k,t}(\pi_t) \tag{1}$$

$$U\big(\pi(T)\big) = \sum_{t=1}^{T} \sum_{k=1}^{m} b_k u_{k,t}(\pi_t) \tag{2}$$

$$0 \le w_{k,t}(\pi_t) \le max\left(0, s_{k,t}(\pi_t) + p_{\pi_t,k} - e_{k,t}(\pi_t)\right) \quad \forall k \in K \; \forall t \in \mathrm{T} \tag{3}$$

$$u_{k,t}(\pi_t) = s_{k,t}(\pi_t) - e_{k,t-1}(\pi_{t-1}) \quad \forall k \in K \; \forall t \in \mathrm{T} \tag{4}$$

$$s_{k,t}(\pi_t) = max\big(e_{k,t-1}(\pi_{t-1}), e_{k-1,t}(\pi_t), (k + t - 2)c\big) \; \forall k \in K \; \forall t \in \mathrm{T} \tag{5}$$

$$e_{k,t}(\pi_t) = s_{k,t}(\pi_t) + p_{\pi_t,k} - w_{k,t}(\pi_t) \; \forall k \in K \; \forall t \in \mathrm{T} \tag{6}$$

$$e_{k,t}(\pi_t) \le (k + t - 2)c + l_k \; \forall k \in K \; \forall t \in \mathrm{T} \tag{7}$$

$$e_{k,0}(\pi_0) = e_{0,t}(\pi_t) = 0 \quad \forall k \in K \; \forall t \in \mathrm{T} \tag{8}$$

$$\lfloor \lambda_i t \rfloor \le X_{i,t} \le \lceil \lambda_i t \rceil, \; X_{i,T} = d_i \; \forall i \in I \; \forall t \in \mathrm{T} \tag{9}$$

The problem consists on obtaining a sequence of products, $\pi(T) = (\pi_1, \dots, \pi_T)$, with the following properties: (i) minimum work overload W, (ii) minimum useless time U, (iii) demand plan satisfaction, $\vec{d}$, (iv) production mix preservation constraints satisfaction, and (v) free interruption of operations.

In the formulation, definitions (1) and (2) determine, respectively, the work overload, W, and the useless time, U, generted by the sequence, $\pi(T)$. The inequalities (3) bounds the partial work overload at all workstation, k, and all cycle, t, allowing

the free interruption of any operation between its start instant and its completion instant or the instant that is fixed by the temporal window: $(k + t - 2)c + l_k$. The equalities (4) define the partial useless time at each workstation and cycle regarding the sequence, $\pi(T)$. Equations (5) determine the minimum start instants, $s_{k,t}$, while 6), (7) and (8) determine the minimum finish instants, $e_{k,t}$, for the $m \times D$ operations. Finally, conditions (9) force to preserve the production mix in all cycle and to meet the demand plan, $\vec{d}$.

In order to formulate the production mix preservation, we use the variables $X_{i,t}$ that symbolize the amount of units of type $i \in I$ contained in the partial sequences: $\pi(t) \equiv (\pi_1, \dots, \pi_t) \subseteq \pi(T)$ $(\forall\, t = 1, .., T)$.

3 Hybrid algorithm GRASP-LP

GRASP is a multi-start meta-heuristic algorithm [11] whose procedure is based on the construction of an initial solution and the improvement of this solution through the iterative application of an embedded local search, whose objective is to reach a local optimum in a specific neighborhood.

On the other hand, the linear programming is a classic optimization technique that allows modeling and solving problem with linear objective functions and constraints, by means of exact algorithms [12].

The nature of the MMSP-W/pmr/free problem leads to the application of both resolution techniques: GRASP is centered on the combinatory aspect of the problem obtaining the best sequence, $\pi(T) = (\pi_1, \dots, \pi_T)$, with forced interruptions; and the LP is focused on the optimization of the continuous variables, minimizing the functions (1) and (2). Specifically, the procedure GRASP-LP, proposed by us to solve the MMSP-W/pmr/free problem consists of two different stages: the first corresponds to the GRASP and provides the best sequence with forced interruptions after a pre-fixed number of iterations (construction and improvement phases); and the second corresponds to a linear program that minimizes the overall work overload and useless time given by the sequence resulting from the first stage but considering free interruptions.

3.1 Phase 1: Sequence construction

A sequence $\pi(T) = (\pi_1, \dots, \pi_T)$ is progressively built by assigning, at each t $(t = 1, \dots, T)$ stage a product from the list of candidates to occupy the t position of the sequence –this list is named $CL(t)$–. Therefore, when the t stage is reached, a product $i \in CL(t)$ is incoporated into the $\pi(t-1) = (\pi_1, \dots, \pi_{t-1})$ secuence already consolidated (see scheme in table 1).

The product $i \in CL(t)$ must meet two conditions to access the list:

(c.1) The amount of units, $X_{i,t-1}$, of type $i \in I$, in the sequence, $\pi(t-1)$, must be lower than its demand in the production plan: $X_{i,t-1} < d_i$.

(c.2) The production of the i product until the period t ($X_{i,t} = X_{i,t-1} + 1$) of the sequence must satisfy the production mix restrictions: $\lfloor \lambda_i t \rfloor \leq X_{i,t} \leq \lceil \lambda_i t \rceil$. That is, the n^{th} unit of type $i \in I$ (n_i) must be manufactured at t_{n_i} cycle of the interval $[t_{min}(n_i), t_{max}(n_i)]$, fulfilling: $t_{min}(n_i) \leq t_{n_i} \leq t_{max}(n_i)$, $(\forall n_i = 1, .., d_i)$.

If the $CL(t)$ list becomes empty by imposing the conditions, (c.1) and (c.2), the condition (c.1) will be maintain while (c.2) will be relaxed.

Subsequently, the candidate products, $i \in CL(t)$, at the stage t are sorted. This sorting responds to two hierarchical priority indices.

The first one related to the work overload generated by the sequence, $\pi_i(t) \equiv \pi(t-1) \cup \{i\}$, which results from adding the product $i \in CL(t)$ at the sequence $\pi(t-1)$. That is:

$$f_i^{(t)} = W(\pi_i(t)) = W(\pi(t-1)) + \sum_{k=1}^{m} b_k w_{k,t}(i) \quad (\forall i \in CL(t) \wedge \forall t = 1, .., T) \tag{10}$$

where $w_{k,t}(i)$ is the partial work overload burden borne by the processor of the workstation $k \in K$ when the t^{th} product unit is type i. This work overload, with forced interruptions, is determined according to equation (11):

$$w_{k,t}(i) = max\big(0, s_{k,t}(i) + p_{i,k} - (k + t - 2)c - l_k\big) \tag{11}$$

In (11), $s_{k,t}(i)$ is the start instant of the operation at the k workstation when a product type i occupies the t^{th} position of the sequence. This instant depends on both the start of the t^{th} manufacturing cycle at the k workstation and the finish instant of operations in progress at the k and $k-1$ stations. Considering the rule for the forced interruption and the initial condition $s_{1,1}(i) = 0 \; \forall i \in I$, the start and finish instants of operations are determined as follows:

$$s_{k,t}(i) = max\big(e_{k,t-1}(\pi_{t-1}), e_{k-1,t}(i), (k + t - 2)c\big) \tag{12}$$

$$e_{k,t-1}(\pi_{t-1}) = s_{k,t-1}(\pi_{t-1}) + p_{\pi_{t-1},k} - w_{k,t-1}(\pi_{t-1}) \tag{13}$$

$$e_{k-1,t}(i) = s_{k-1,t}(i) + p_{i,k-1} - w_{k-1,t}(i) \tag{14}$$

The second index (dependent to the first) attends to obtain sequences that minimize the useless time at workstations. That is:

$$g_i^{(t)} = U\big(\pi_i(t)\big) = U\big(\pi(t-1)\big) + \sum_{k=1}^{m} b_k u_{k,t}(i) \quad (\forall i \in CL(t) \wedge \forall t = 1, .., T) \tag{15}$$

where $u_{k,t}(i)$ is the useless time available for the processor of the k station between the instants $e_{k,t-1}(\pi_{t-1})$ and $s_{k,t}(i)$. Consequently:

$$u_{k,t}(i) = s_{k,t}(i) - e_{k,t-1}(\pi_{t-1}) \tag{16}$$

The indices $f_i^{(t)}$ and $g_i^{(t)}$ allow sorting in ascending order the elements of the $CL(t)$ list resulting in the $\overline{CL}(t)$ list. It should be noted that the useless time, $\big(g_i^{(t)}\big)$,

only affects this sorting of elements whether there are tie in the work overload values, $\left(f_i^{(t)}\right)$, because the indices are hierarchically applied.

After the sorting, the $\overline{CL}(t)$ list is reduced by the admission factor Λ (percentage of products that will be sorted among the best candidates). This reduction gives rise the restricted list, $\overline{RCL}(t,\Lambda)$, that is equal than the $\overline{CL}(t)$ list when $\Lambda = 100\%$.

The table 1 shows the scheme for the implemented GRASP constructive phase.

Table 1. GRASP constructive phase for a sequence with forced interruption of operations, with minimum work overload and useless time and with production mix preservation.

0. *Initialization:*
   Input: $\Lambda, I, K, D, c, \left(d_i, p_{i,k}, l_k\right)\ \forall i \in I\ \forall k \in K$
   Initialize: $T = D, t = 0, \pi(t) = \{\emptyset\}, (n_i = 0, \lambda_i = d_i/D)\ \forall i \in I$
1. *Creation of the set of candidate product types:*
   $t \leftarrow t + 1$
   Let $CL(t) = \{i \in I: (n_i < d_i) \wedge (t_{min}(n_i + 1) \leq t \leq t_{max}(n_i + 1))\}$
   - If $CL(t) = \{\emptyset\} \Rightarrow CL(t) = \{i \in I: n_i < d_i\}$
2. *Assessment of candidate product types:*
   $\forall i \in CL(t)$, by: (10) - (16), determine:
   $f_i^{(t)} = W\left(\pi_i(t)\right) = W\left(\pi(t-1)\right) + \sum_{k=1}^{|K|} b_k w_{k,t}(i)$
   $g_i^{(t)} = U\left(\pi_i(t)\right) = U\left(\pi(t-1)\right) + \sum_{k=1}^{|K|} b_k u_{k,t}(i)$
3. *Sorting of candidate product types:*
   Let $\overline{CL}(t) = \left(i_1, \dots, i_{|RCL(t)|}\right)$ be the sorted list of candidate products,
   - It will be met: $pos\left(i, \overline{CL}(t)\right) < pos\left(i', \overline{CL}(t)\right)\ \forall \{i, i'\} \subseteq \overline{CL}(t)$, if the condition is satisfied:
   $\left[\left(f_i^{(t)} < f_{i'}^{(t)}\right)\right] \vee \left[\left(f_i^{(t)} = f_{i'}^{(t)}\right) \wedge \left(g_i^{(t)} < g_{i'}^{(t)}\right)\right]$
4. *Selection of the product type from the restricted list* $\overline{RCL}(t,\Lambda) \subseteq \overline{CL}(t)$:
   - Let $pos^* = -int(-\Lambda \cdot |\overline{CL}(t)| \cdot RND)$ be the selected position, then, it is selected the product type i^* that is in the said position:
   $i^* = i_{pos^*} \in \overline{RCL}(t,\Lambda) = \left(i_1, \dots, i_{|\overline{RCL}(t,\Lambda)|}\right)$ con $\overline{RCL}(t,\Lambda) \subseteq \overline{CL}(t)$
5. *Update:*
   $n_{i^*} \leftarrow n_{i^*} + 1\ ;\ \pi(t) \equiv \pi(t-1) \cup \{i^*\}$
6. *Finalization test:*
   if $t < T$ go to step 1
   else, END

The sequence of tasks, $\pi(T)$, resulting from the GRASP constructive phase, can break the preservation condition of the production mix when the $CL(t)$ list is empty at the step 1 of any execution stage of the algorithm. When this occurs, all products with pending demand are considered. Indeed, an Exchange procedure is activated in order to solve a maximum constraint satisfaction problem $\left[t_{min}(n_i) \leq t_{n_i} \leq t_{max}(n_i),\ \forall\ n_i = 1,., d_i: i \in I\right]$ that transforms the original sequence, $\pi(T)$, in other sequence, $\hat{\pi}(T)$, that satisfies the preservation constraints.

3.2 Phase 2: Solution improvement by local search

Similar to [7], the local improvement phase begins with the $\hat{\pi}(T)$ sequence, which satisfies the conditions (9). This phase consists of executing four descent algorithms on four neighborhoods consecutively and repetitively until none of them improves the best-obtained solution while the iteration.

The descent algorithms are based on exchange and insertion of products (see [7]) and they are address to exploring sequence cycles in both increasing and decreasing direction. Such procedures are: (i) forward exchange (figure 1), (ii) backward exchange (figure 2), (iii) forward insertion (figure 3), and (iv) backward insertion (figure 4).

Figure 1. Forward exchange. Given the t position of the sequence $\hat{\pi}(T)$ is searched the next position with the same product type, $t'(t' > t)$ or, otherwise, $t' = T + 1$; defined the range $[t + 1, t' - 1]$ the contained product types are exchanged from one position to another in the upwards direction in order to improve the solution. The example shows how product type 9 can be exchanged with product type 3 or 4.

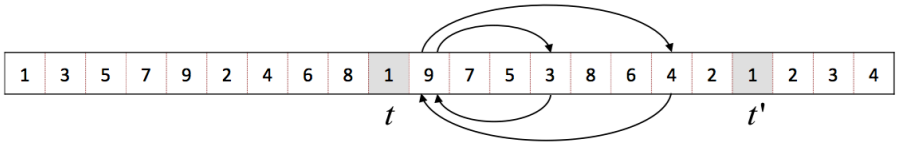


Figure 2. Backward exchange. This procedure is like the forward exchange but in downwards direction. In this case the range of exchange elements is defined by $[t' + 1, t - 1]$, where t' may be $t' = 0$ whether there is no previous product type. In the example, we can see how product type 2 is exchanged with product type 5 that is in a previous position in the sequence, and similarly, product type 6 with the 3 one.

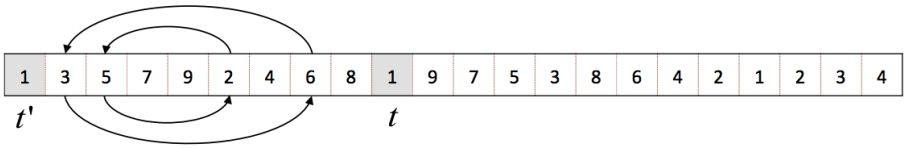


Figure 3. Forward insertion. Similar to the forward exchange, once the range $[t+1, t'-1]$ with $t'(t' > t)$ is defined the intermediate elements are inserted to improve the solution in other later positions of the sequence but within the range. If solution improves, the insertion in consolidated. In the scheme, product type 9 located in position $t+1$ is inserted in position $t'-4$, causing the movement of product type 7 from position $t+2$ to $t+1$.

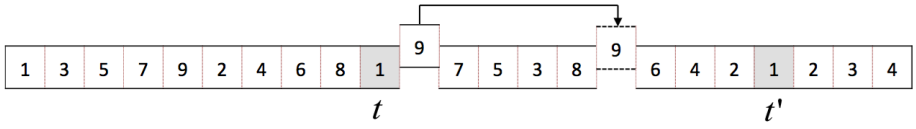
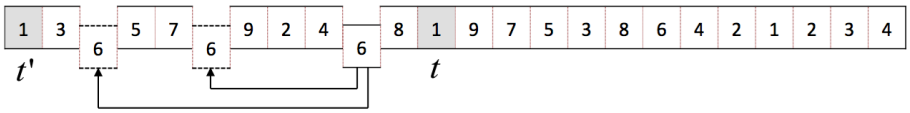


Figure 4. Backward insertion. Given the range $[t'+1, t-1]$ the intermediate elements are inserted in previous positions from the partial sequence in order to improve the solution. In the example, it is show how product type 6, initially located in $t-2$ is moved to positions $t'+2$ or $t'+5$ to asses if there is improvement.



To select the best solution it should be noted that in case of two sequences with production mix preservation, the sequence with less total work overload, $W(\pi(T))$, will be preferred, and, in the event of a tie in the work overload, the sequence with less useless time, $U(\pi(T))$, will be the selected.

Finally, the results of this second phase will be the $\pi^*(T) = (\pi_1^*, \dots, \pi_T^*)$ sequence.

3.3 Phase 3: Overall Work Overload Minimization by Linear Programming

After the GRASP improvement phase, the $\pi^*(T) = (\pi_1^*, \dots, \pi_T^*)$ sequence with the least amount of work overload is used as input for a linear program, LP-W, whose objective is to minimize the overall work overload by means of allowing the free interruption of operations.

Let LP-W be: $\min W = \sum_{t=1}^{T} \sum_{k=1}^{|K|} b_k w_{k,t}$ (17)

Subject to:

$v_{k,t} + w_{k,t} = p_{\pi_t^*,k}$ $\forall k = 1,.,m;\ \forall t = 1,.,T$ (18)

$s_{k,t} \geq (k+t-2)c$ $\forall k = 1,.,m;\ \forall t = 1,.,T$ (19)

$s_{k,t} \geq s_{k,t-1} + v_{k,t-1}$ $\forall k = 1,.,m;\ \forall t = 2,.,T$ (20)

$s_{k,t} \geq s_{k-1,t} + v_{k-1,t}$ $\forall k = 2,.,m;\ \forall t = 1,.,T$ (21)

$s_{k,t} + v_{k,t} \leq (k+t-2)c + l_k$ $\forall k = 1,.,m;\ \forall t = 1,.,T$ (22)

$v_{k,t}, w_{k,t} \geq 0$ $\forall k = 1,.,m;\ \forall t = 1,.,T$ (23)

where $s_{k,t}$, $v_{k,t}$ and $w_{k,t}$ are real variables that represent the start instant, the completed work and the work overload of the t^{th} operation at the k workstation, respectively.

By using the LP-W, the GRASP-LP hybrid procedure is place on a equal footing to compete on the problem resolution with other procedures from the literature: MILP [9] y BDP-2 [10].

4 Computational experiment. Case study

The computational experience is focused on analyzing the performance of GRASP-LP against other procedures in regard with the quality of the solutions and CPU times. Specifically, we compare the results obtained by:

- BDP-2: BDP algorithm with production mix preservation. We take into account the two versions of this algorithm according to the pseudo-dominances of vertices, the 2/1 and the 2/2, (see [10]).
- MILP: $4 \cup 3\_pmr$ model (see [9])
- GRASP-LP: procedure presented in this paper.

Like [7], this comparison between the performances of all procedures is made through a case study linked with the Nissan's Engine Plant in Barcelona.

The case study consists of an assembly line where different types of engines are assembled and where 42 operators work with a cycle time of 175 seconds. The line assembles nine types of engines, which are grouped into three families (SUVs –Sport Utility Vehicle–, Vans and Trucks).

Specifically, the main characteristics of the case study are the following:

- Number of workstations: $|K| \equiv m = 21$.
- Number of product types: $|I| = 9$ $(i = 1,..,9)$.
- Cycle time: $c = 175\ s.$, and temporal window: $l_k = 195\ s.\ (\forall k = 1,..,21)$.
- Number of homogeneous processors (considering each processor as a team of two operators with the same skills): $b_k = 1$ $(\forall k = 1,..,21)$.
- Processing times $p_{i,k}$ $(\forall i \in I, \forall k \in K)$ by product and workstation. The set of processing times are compressed between $89\ s.$ and $185\ s.$ At normal activity (see [3]: Table 5).
- Number of demand plans: $|\mathrm{E}| = 23$ ($\varepsilon = 1,..,23$). All plans have the same daily demand (see [3]: Table 6, Block I, NISSAN-9ENG).
- Daily demand: $T \equiv D_\varepsilon = 270$ units $(\forall \varepsilon = 1,..,23)$.

Once the codes of procedures were compiled, they have been run on an iMac (Intel Core i7 2.93 GHz, 8 GB de RAM). Mainly, the three procedures have the following characteristics:

- BDP-2: (i) maximum number of transitions from each vertex equal than the number of products types $|I| = 9$; (ii) there are different window widths, $H = (1, 36, 81, 126)$, for all 23 demand plans (which implies running 184 the algo-

rithm, taking into account the two versions); (iii) initial solution, Z_0, for H_n equal to best solution obtained with H_{n-1}, except for $H_1 = 1$, and where $Z_0 \to \infty$; (iv) the average CPU time by demand plan is 5026.6 *s*.; and (v) the production mix preservation and the free interruption of operations have been introduced by the linear programming assistance (Gurobi solver).

- MILP: (i) mathematical model compiled and run on the Gurobi solver v4.5.0; (ii) maximum CPU time available to run each demand plan equal to 7200 *s*. (23 executions), and average time used by demand plan equal to 6605.1 *s*.; and (iii) production mix restrictions and free interruption of operations.
- GRASP-LP: (i) maximum number of iterations by demand plan equal to 10; (ii) three possible values for the admission factor $\Lambda = (25\%, 50\%, 100\%)$ (690 solutions in 69 executions); (iii) average CPU time per demand plan consumed by the two GRASP phases equal to 425.3 *s*.; (iv) production mix restrictions and free interruption of operations are introduced through the use of linear programming after the GRASP execution; (v) linear programming run on an iMac (Intel Core 2 Duo 2.33 GHz, 3 GB de RAM) with only one processor and on the CPLEX solver v11.0.

Table 2 provides the best results given by BDP-2 (see table 3 in [10]), MILP (see column 4 ∪ 3_pmr in Table 3 from [9]) and GRASP-LP (this paper), in regards with the work overload, W, of the 23 demand plans, $\varepsilon \in E$. The table also shows the algorithm that wins at each demand plan and the unity gains of GRASP-LP against BDP-2 (ΔGvB), GRASP-LP against MILP (ΔGvM) and BDP-2 against MILP (ΔBvM). These unity gains are determined as follows:

$$\Delta \mathcal{P} v \mathcal{P}'(\varepsilon) = \frac{W(\varepsilon)_{\mathcal{P}'} - W(\varepsilon)_{\mathcal{P}}}{min(W(\varepsilon)_{\mathcal{P}'}, W(\varepsilon)_{\mathcal{P}})}$$
$$\forall \varepsilon \in \mathrm{E},\ \forall \mathcal{P} \in \{G, B\}, \forall \mathcal{P}' \in \{B, M\} \tag{24}$$

From the analysis of table 2, we can state:

- Regarding the best solutions, the winning procedure is BDP-2 with 12 best solutions out of 23 demand plans; the second best procedure is GRASP-LP, which obtains best solution in 7 occasions (demand plans: 1, 7, 8, 10, 12, 17 and 23), while MILP is in the last position with 5 best solutions (3, 10 , 19, 21 and 22). MILP and GRASP-LP obtain the same solution in plan 10, and MILP demonstrates the optimum solutions for the demand plans 10 and 19.
- GRASP-LP wins BDP-2 on 10 occasions out of 23. The average unity gain of BDP-2 against GRASP-LP is 15%, when BDP-2 is the winner. On the other hand, when GRASP-LP wins, the unity gain of GRASP-LP against BDP-2 is 11%. On global average, the unity gain of BDP-2 against GRASP-LP is only by 4%.
- GRASP-LP wins MILP in 12 plans and ties in plan 10. The overall average unity gain of GRASP-LP against MILP is around 6%. At length, GRASP-LP wins MILP with a partial average unity gain of 22%, and MILP wins partially GRASP-LP with a gain of 12% approximately.
- BDP-2 wins MILP on 16 times out of de 23. The partial average unity gain when BDP-2 wins MILP and vice versa, is equal to 13% and 1%, respectively. On overall, BDP-2 wins MILP with a gain of 9%.

– BDP-2, MILP and GRASP-LP required 5026.6 *s*, 6605.1 *s* and 426.6 *s*, on average, respectively, to confirm their best solution at all demand plans.

Table 2. For each plan $\varepsilon \in$ E, work overload, W, given by each procedure $(W(\varepsilon)_B, W(\varepsilon)_M, W(\varepsilon)_G)$, unity gain between pair of procedures $(\Delta GvB, \Delta GvM, \Delta BvM)$, best solution, $W(\varepsilon)^*$, in terms of work overload and winning algorithm.

$\varepsilon \in$ E	$W(\varepsilon)_B$	$W(\varepsilon)_M$	$W(\varepsilon)_G$	ΔGvB	ΔGvM	ΔBvM	$W(\varepsilon)^*$	$Winner$
1	166	186	98	0.69	0.90	0.12	98	GRASP-LP
2	318	383	342	-0.08	0.12	0.20	318	BDP-2
3	444	423	430	0.03	-0.02	-0.05	423	MILP
4	305	307	419	-0.37	-0.36	0.01	305	BDP-2
5	633	661	662	-0.05	-0.00	0.04	633	BDP-2
6	428	478	525	-0.23	-0.10	0.12	428	BDP-2
7	740	731	728	0.02	0.00	-0.01	728	GRASP-LP
8	112	160	92	0.22	0.74	0.43	92	GRASP-LP
9	739	751	911	-0.23	-0.21	0.02	739	BDP-2
10	1209	1208	1208	0.00	0.00	-0.00	1208	GR/MILP
11	92	122	96	-0.04	0.27	0.33	92	BDP-2
12	293	287	268	0.09	0.07	-0.02	268	GRASP-LP
13	277	336	294	-0.06	0.14	0.21	277	BDP-2
14	381	423	397	-0.04	0.07	0.11	381	BDP-2
15	422	442	429	-0.02	0.03	0.05	422	BDP-2
16	216	251	227	-0.05	0.11	0.16	216	BDP-2
17	466	488	464	0.00	0.05	0.05	464	GRASP-LP
18	610	619	698	-0.14	-0.13	0.01	610	BDP-2
19	949	945	948	0.00	-0.00	-0.00	945	MILP
20	129	150	169	-0.31	-0.13	0.16	129	BDP-2
21	565	561	725	-0.28	-0.29	-0.01	561	MILP
22	991	984	987	0.00	-0.00	-0.01	984	MILP
23	111	121	107	0.04	0.13	0.09	107	GRASP-LP
Average	-	-	-	-0.04	0.06	0.09	-	-

It should be noted that all sequences given by the three procedures (BDP-2, MILP and GRAS-LP) satisfy the production mix preservation property (*pmr*), which has been established through the restrictions (9) from the MMSP-W/pmr/free model. Accordingly, all sequences fulfill: $\lfloor \lambda_{i,\varepsilon} t \rfloor \leq X_{i,t,\varepsilon} \leq \lceil \lambda_{i,\varepsilon} t \rceil,\ X_{i,T,\varepsilon} = d_{i,\varepsilon}\ \forall i \in I, \forall t \in T, \forall \varepsilon \in E$, where:

– $d_{i,\varepsilon}$: is the demand of units type $i \in I$ in the plan $\varepsilon \in$ E
– $\lambda_{i,\varepsilon}$: the proportion of units type $i \in I$ in the plan $\varepsilon \in$ E; that is: $\lambda_{i,\varepsilon} = d_{i,\varepsilon}/T$ $\forall i \in I, \forall \varepsilon \in$ E
– $X_{i,t,\varepsilon}$: Real production associated with the partial sequence $\pi_\varepsilon(t)$. That is: the units of type $i \in I$ that contains the partial sequence $\pi_\varepsilon(t) = (\pi_{1,\varepsilon}, \ldots, \pi_{t,\varepsilon}) \subseteq \pi_\varepsilon(T)$ of the plan $\varepsilon \in$ E.

Therefore, all sequences have the same quality in regard with the production mix preservation criterion defined by the *pmr* restrictions (9). For that reason, we resort to the non-regularity functions from typical problems of the manufacturing ideology JIT [6, 8], such as the Product Rate Variation Problem (PRVP) or the Output Rate Variation Problem (ORVP). This allows us to obtain a metric that enables to discriminate solutions regarding the production regularity.

Specifically, to measure the production non-regularity of a sequence $\pi_\varepsilon(T)$, we use the sum of the quadratic discrepancies between the ideal production of each product at each manufacturing cycle and at each demand plan (i.e.: $\lambda_{i,\varepsilon}t\ \forall i \in I, \forall t \in T, \forall \varepsilon \in E$) and the real production of the partial manufacturing sequence (i.e.: $X_{i,t,\varepsilon}$). Accordingly:

$$\Delta_Q(X,\varepsilon) = \sum_{t=1}^{T}\sum_{i=1}^{|I|}\left(X_{i,t,\varepsilon} - \lambda_{i,\varepsilon}t\right)^2 \quad \forall \varepsilon \in E \tag{25}$$

Table 3. For all demand plan $\varepsilon \in$ E, value of $\Delta_Q(X,\varepsilon)$ function by procedure $(\Delta_Q(X,\varepsilon)_B, \Delta_Q(X,\varepsilon)_M, \Delta_Q(X,\varepsilon)_G)$, unity gain between pair of procedures $(\Delta GvB, \Delta GvM, \Delta BvM)$ and the best solution value $\Delta_Q(X,\varepsilon)_{BMG}$.

$\varepsilon \in$ E	$\Delta_Q(X,\varepsilon)_B$	$\Delta_Q(X,\varepsilon)_M$	$\Delta_Q(X,\varepsilon)_G$	ΔGvB	ΔGvM	ΔBvM	$\Delta_Q(X,\varepsilon)^*$	*Winner*
1	400.0	400.0	400.0	0.00	0.00	0.00	400.0	All
2	327.9	423.5	397.0	-0.21	0.07	0.29	327.9	BDP-2
3	340.7	408.5	380.6	-0.12	0.07	0.20	340.7	BDP-2
4	333.6	421.3	396.4	-0.19	0.06	0.26	333.6	BDP-2
5	352.1	394.7	429.0	-0.22	-0.09	0.12	352.1	BDP-2
6	388.5	420.0	395.7	-0.02	0.06	0.08	388.5	BDP-2
7	423.6	396.0	403.4	0.05	-0.02	-0.07	396.0	MILP
8	347.6	448.1	414.0	-0.19	0.08	0.29	347.6	BDP-2
9	360.7	411.2	394.8	-0.09	0.04	0.14	360.7	BDP-2
10	330.8	381.1	415.5	-0.26	-0.09	0.15	330.8	BDP-2
11	384.1	447.3	429.0	-0.12	0.04	0.16	384.1	BDP-2
12	385.5	410.2	416.2	-0.08	-0.01	0.06	385.5	BDP-2
13	334.5	436.4	419.8	-0.25	0.04	0.30	334.5	BDP-2
14	353.9	414.9	408.9	-0.16	0.01	0.17	353.9	BDP-2
15	378.1	445.2	401.1	-0.06	0.11	0.18	378.1	BDP-2
16	340.0	404.9	388.1	-0.14	0.04	0.19	340.0	BDP-2
17	370.2	415.3	391.6	-0.06	0.06	0.12	370.2	BDP-2
18	336.3	419.6	402.6	-0.20	0.04	0.25	336.3	BDP-2
19	412.2	412.3	373.5	0.10	0.10	0.00	373.5	GRASP-LP
20	342.6	393.6	386.5	-0.13	0.02	0.15	342.6	BDP-2
21	384.8	404.2	409.8	-0.07	-0.01	0.05	384.8	BDP-2
22	317.7	395.8	382.7	-0.20	0.03	0.25	317.7	BDP-2
23	309.2	385.6	377.1	-0.22	0.02	0.25	309.2	BDP-2
Average	-	-	-	-0.12	0.03	0.16	-	-

Table 3 summarizes the values of the irregularity function $\Delta_Q(X,\varepsilon)$ for the best solutions from the set of Nissan-9Ing's instances, in regard with the work overload W:

- $\Delta_Q(X,\varepsilon)_B$: Production non-regularity of best solutions in work overload, W, given by BDP-2 for the ε demand plan.
- $\Delta_Q(X,\varepsilon)_M$: Production non-regularity of best solutions in work overload, W, given by MILP for the ε demand plan.
- $\Delta_Q(X,\varepsilon)_G$: Production non-regularity of best solutions in work overload, W, given by GRASP-LP for the ε demand plan.

Table 3 also indicates the winning algorithm for each demand plan and the unity gains obtained by GRASP-LP versus BDP-2 (ΔGvB), GRASP-LP versus MILP (ΔGvM) and BDP-2 versus MILP (ΔBvM). These gains are calculated in accordance with:

$$\Delta\mathcal{P}v\mathcal{P}'(\varepsilon) = \frac{\Delta_Q(X,\varepsilon)_{\mathcal{P}'} - \Delta_Q(X,\varepsilon)_{\mathcal{P}}}{min\left(\Delta_Q(X,\varepsilon)_{\mathcal{P}'}, \Delta_Q(X,\varepsilon)_{\mathcal{P}}\right)}$$

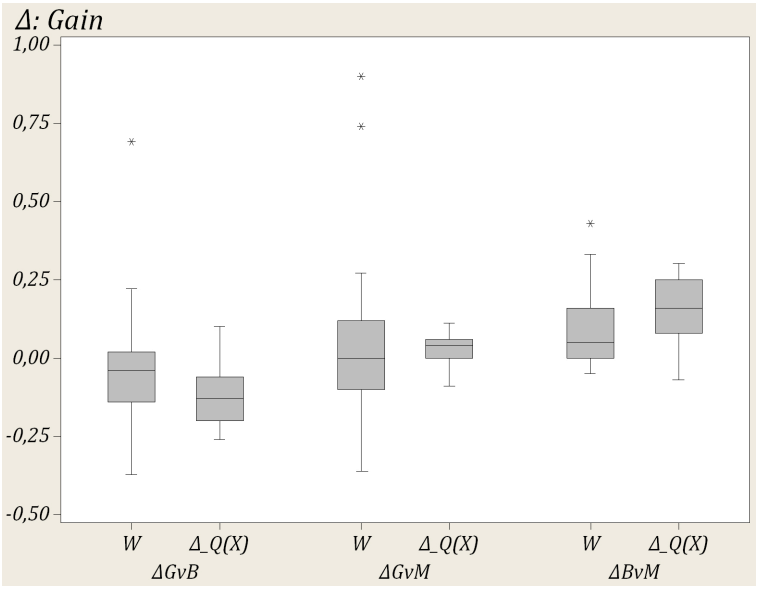
$$\forall\varepsilon \in \mathrm{E},\ \forall\mathcal{P} \in \{G,B\}, \forall\mathcal{P}' \in \{B,M\} \tag{26}$$

From table 3, we can denote:

- BDP-2 is the winning procedure with 20 best solutions out of the 23 demand plans.
- GRASP-LP reaches one best solution (plan 19).
- MILP obtains one best solution (plan 7).
- BDP-2, MILP and GRASP-LP give the same solution in plan 1.
- The average unity gain from BDP-2 against GRASP-LP is by 15%, when BDP-2 is the winner. However, when GRASP-LP wins, the gain of GRASP-LP versus BDP-2 is by 8%. Considering the overall average the unity gain is 12 % in favor of BDP-2 and against GRASP-LP.
- The overall average unity gain of GRASP-LP against MILP is only 3%. Specifically, GRASP-LP wins MILP with a partial average reduction of the non-regularity by 5%; and MILP wins GRASP-LP with a partial gain around 4%.
- In terms of partial averages, when BDP-2 wins MILP, and vice versa, the unity gains are 17% and 7% respectively. On the other hand, BDP-2 wins MILP with a gain by 16%, in overall terms.
- Finally, in accordance with the hierarchy criteria and subordinating the minimum non-regularity to the minimum work overload, the tie between MILP and GRASP-LP in the demand plan 10 is broken in favor of MILP; in particular $\Delta_Q(X,10)_M = 381.1$ and $\Delta_Q(X,10)_G = 415.5$.

To summarise the above results (tables 2 and 3) we analyse statistically by a box plot (Figure 5) the variation range of the unity gain functions, ΔGvB, ΔGvM and ΔBvM, both for the work overload function, $W(\varepsilon)$ (equation (24)), and for the function of non-regularity of production, $\Delta_Q(X,\varepsilon)$ (equation (26)).

Figure 5. Box-plot for unity gains between two different procedures ($\Delta GvB, \Delta GvM, \Delta BvM$) regarding the work overload (W) and the non-regularity of production ($\Delta_Q(X)$) functions. The abbreviations G, B and M indicate the GRASP-LP, BDP-2 and MILP procedures respectively.



From the Box-plot we can state the following:

- Concerning the non-regularity of production $\Delta_Q(X)$, all unity gain values ($\Delta GvB, \Delta GvM, \Delta BvM$) are between the normal boundaries established by 1.5 times the interquartile range or the minimum and maximum values of each dataset. Therefore the unity gains (columns ΔGvB, ΔGvM *y* ΔBvM from Table 3) do not present outliers for no comparison between BDP-2, MILP and GRASP-LP.
- According the $\Delta_Q(X)$ metric and assessing GRASP-LP against MILP (column ΔGvM in Table 3), we can see a very narrow the interquartile range and a clearly biased value distribution.
- The $\Delta_Q(X)$ metric when the BDP-2 procedure is assessed against MILP and GRASP-LP has made evident the superiority of BDP-2 against the other two procedures. In this case the value distributions are centred and the interquartile ranges are similar (columns ΔGvB and ΔBvM in Table 3).
- Regarding work overload, W (columns ΔGvB, ΔGvM *and* ΔBvM in Table 2) the Box-plot shows outliers at all the procedure comparisons. Indeed, comparing GRASP-LP versus BDP-2 we have the outlier 0.69 that corresponds to demand plan #1; comparing GRASP-LP versus MILP we have two outliers, 0.9 and 0.74, for the demand plans #1 and #8, respectively; and comparing BDP-2 versus MILP we have the outlier 0.43 that corresponds to the plan #8.
- Finally, the comparison between BDP-2 against the other two procedures presents similar ranges and opposed biased distributions in accordance with the work over-

load values. However, the gain of GRASP-LP against MILP offers a centred distribution of values but with a greater range that the previous two.

5 Conclusions

This paper presents the mixed-model sequence with work overload minimization considering that processors can interrupt operations at any time before the end of the time available to work on the product unit (temporal window). Together with the work overload minimization, the problem proposed in this paper seeks to obtain regular sequences in terms of production by addressing the principles from the Just-in-Time ideology.

As resolution procedure for the problem, we present a hybrid procedure that combines a GRASP metaheuristic with a linear program, LP. This procedure together with two other procedures, MILP and BDP-2, are assessed by means of a case study linked with a real manufacturing environment.

After the computational experience, we have able to highlight the strengths and weaknesses of all procedures used (GRASP, BDP and MILP). To that end we focus on five qualities: (I) Guarantee of achieving the optimum, (ii) memory requirement, (iii) ease of implementation, (iv) quality of solutions and (v) speed.

As for the guarantee of optimal solution, MILP has an advantage over GRASP, because it is an exact procedure that explores the whole set of solutions. The weakness of MILP is the time required to complete the exploration of highly combinatory problems with industrial dimensions, such as our case study. For its part, BDP, in its version as an exact procedure, is also an exploration algorithm (in stages), therefore it presents the same advantages and disadvantages that MILP. Finally, GRASP is a heuristic procedure, whose purpose is not to guarantee optimum, but to obtain quality solutions in acceptable time.

In accordance with the memory requirements GRASP is undoubtedly the most appropriate procedure: GRASP allows to treat instances of industrial dimensions without great difficulty. Something similar happens with BDP when we limit the number of vertices to be explored in each stage, but in this case BDP becomes a heuristic and loses its strength. By its iterative performance, MILP needs in memory all the information of the mathematical model and it may be out of memory when the dimension of set of vertices to explore is very large. Accordingly, MILP is the worst procedure in regard with the memory characteristic, and it should be noted that GRASP is the simplest method to implement, and that both GRASP and BDP are more versatile than MILP.

Given the quality of solutions, the winning procedure, and therefore, state of art of the problem, is BDP with assistance of linear programming, in terms of work overload values. The second position is for GRASP, also with linear programming assistance, with only a difference by 4% in regard with the overall unity gain of BDP. However, GRASP-LP wins MILP by a work overload improvement of 6%. Considering the production regularity criteria and subordinating this to the work overload,

the winning procedure is also BDP. GRASP continues in second position worsening the regularity of sequences by 12% on average, with respect to BDP. In this case, the difference between quality of results given by MILP and GRASP is only by 3% being worst MILP.

Regarding CPU times, GRASP is the fastest, being 11.8 and 15.54 times faster than BDP and MILP, respectively; this is important in our case study because one minute of line stop means a cost of 137.14€.

Finally, as a general conclusion, the great strength of GRASP procedure studied in this paper has been to achieve high-quality solutions using the twelfth of the time that BDP used.

As future works, it is our goal to extend the proposed method to Beam Search and Ant Colony Algorithms. Similarly we want to compare results in case of prioritizing the regularity criteria against the work overload.

Acknowledgements. This has been funded by the Ministerio de Economía y Competitividad (Spanish Government) through the FHI-SELM2 (TIN2014-57497-P) project.

6 References

1. Bautista, J., Batalla-García, C., Alfaro-Pozo, R.: Models for assembly line balancing by temporal, spatial and ergonomic risk attributes. European Journal of Operational Research 251(3), pp. 814-829, (2016).
2. Boysen, N., Fliedner, M., Scholl, A.: Sequencing mixed-model assembly lines: Survey, classification and model critique. European Journal of Operational Research 192(2), pp. 349–373 (2009).
3. Bautista, J., Cano, A.: Solving mixed model sequencing problem in assembly lines with serial workstations with work overload minimisation and interruption rules. European Journal of Operational Research 210(3), pp. 495-513 (2011).
4. Cano-Belmán, J., Ríos-Mercado, R.Z., Bautista, J.: A scatter search based hyper-heuristic for sequencing a mixed-model assembly line. Journal of Heuristics 16(6), pp 749-770 (2010)
5. Bautista, J., Pereira, J., Adenso-Díaz, B.: A GRASP approach for the extended car sequencing problem. Journal of Scheduling 11, pp. 3-16 (2008).
6. Monden, Y.: Toyota Production System · An Integrated Approach to Just-In-Time. Springer US (1994).
7. Bautista, J., Alfaro-Pozo, R., Batalla-García, C.: GRASP for sequencing mixed models in an assembly line with work overload, useless time and production regularity. Progress in Artificial Intelligence 5(1), pp. 27-33 (2016).
8. Bautista, J., Cano, A., Alfaro, R., Batalla, C.: Impact of the Production Mix Preservation on the ORV Problem. Advances in Artificial Intelligence, Volume 8109 of the series Lecture Notes in Computer Science, pp. 250-259. Springer Berlin Heidelberg (2013).
9. Bautista, J., Cano, A., Alfaro, R.: Modeling and solving a variant of the mixed-model sequencing problem with work overload minimisation and regularity constraints. An application in Nissan's Barcelona Plant. Expert Systems with Applications 39(12), pp. 11001–11010 (2012).

10. Bautista, J., Cano, A., Alfaro, R.: A hybrid dynamic programming for solving a mixed-model sequencing problem with production mix restriction and free interruptions. Progress in Artificial Intelligence, First Online. DOI:10.1007/s13748-016-0101-5 (2016)
11. Resende, M.G.C., Ribeiro, C.C.: Greedy Randomized Adaptive Search Procedures: Advances, Hybridizations, and Applications: In: Gendreau, M. and Potvin, J.Y. (Eds.) Handbook of Metaheuristics, pp. 283-319. Springer US (2010).
12. Vanderbei, R.J.: Linear Programming · Foundations and Extensions. International Series in Operations Research & Management Science, Volume 114. Springer US (2008).