Progress in Artificial Intelligence (2021) 10:465–488
https://doi.org/10.1007/s13748-021-00249-z

REGULAR PAPER

Exact and heuristic procedures for the Heijunka-flow shop scheduling problem with minimum makespan and job replicas

Joaquín Bautista-Valhondo[1]

Received: 23 October 2020 / Accepted: 7 June 2021 / Published online: 3 July 2021

Abstract
In this paper, a new problem of job sequences in a workshop is presented, taking into account non-unit demands for the jobs and whose objective is to minimize the total completion time for all the jobs (C_{max}) satisfying a set of restrictions imposed on the problem to preserve the production mix. Two procedures are proposed to solve the new problem: Mixed Integer Linear Programming and a Metaheuristic based on Multistart and Local Search. The two proposed procedures are tested using instance set Nissan-9Eng.I, in both cases giving rise to highly satisfactory performance both in quality of solutions obtained and in the CPU times required. Through a case study of the Nissan engine manufacturing plant in Barcelona, our economic-productive analysis reveals that it is possible to save an average of € 1162.83 per day, manufacturing 270 engines, when we transform the current assembly line into a Heijunka-Flow Shop.

Keywords Flow shop scheduling problem · Overall demand · Heijunka · Mixed integer linear programming · Multistart · Local search · Metaheuristic

1 Preliminaries

The *Flow Shop Scheduling Problem* (FSP) is a sequencing problem that has received considerable attention from professionals and researchers in recent decades due in part to the wide range of production environments it can model [19].

A recent version of FSP is the $Fm/\beta/\gamma/d_i$ family of sequencing problems [3] and 2020), which is to establish an application between the elements of a set T of ordinals (T elements) corresponding to the positions in the production sequence: $\pi(T) = (\pi_1, .., \pi_T)$, and the elements of a set J of jobs or products (D elements, with $D = T$).

The jobs or products in group J are classified into exclusive types or classes, J_i, satisfying the following properties: $J = \bigcup_{i \in I} J_i$ and $J_i \cap J_{i'} = \emptyset$, $\forall \{i, i'\} \in I$, where I is the set of job types ($i = 1, .., n$).

In $Fm/\beta/\gamma/d_i$ problems, the β parameter can take the permutation (prmu) or blocking (block) values, while the γ parameter corresponds the efficiency metrics to optimize (C_{max}, C_{med}, etc.), vector $\vec{d} = (d_1, d_2, .., d_n)$ represents the demand plan for the considered job types, and d_i symbolizes the number of jobs of type $i \in I$ within J, that is to say $d_i = |J_i|\ \forall i \in I$, satisfying: $\sum_{\forall i} d_i = D = T$.

The units of J travel in order through a set K of m stations on an assembly line arranged in series, and the production of a job of type $i \in I$ requires a heterogeneous processing time $p_{i,k}$ in workstation $k \in K$ ($k = 1, .., m$).

The purpose of problems $Fm/\beta/\gamma/d_i$ is to obtain a sequence of replicated jobs or products (d_i), in a line with m machines, with the possibility of wblocking or not, according to the β parameter, and with the objective of optimizing the efficiency metric represented by the γ parameter ($C_{\max}$, C_{med}, etc.).

Therefore, using the notation proposed by Graham et al. [11], both the $Fm/prmu/\gamma$ problems [1, 10, 13, 20, 22, 23] as the $Fm/block/\gamma$ problems [4, 8, 16, 18, 21] are particular cases of the family $Fm/\beta/\gamma/d_i$, when $d_i = 1$ for all $i \in I$.

On the other hand, completing all jobs in the shortest time possible ($\min C_{\max}$)) is not the only desirable objective when establishing a product manufacturing sequence. In production environments that are governed by the Just-in-Time manufacturing ideals [17], the production sequences must have properties that are linked to the *Heijunka* concept

✉ Joaquín Bautista-Valhondo
joaquin.bautista@upc.edu

[1] IOC ETSEIB Universitat Politècnica de Catalunya, Av. Diagonal 647, 08028 Barcelona, Spain

[9, 12, 14], whose meaning is to achieve regularity of production.

El The Heijunka (regularity) concept can be applied to any constituent element of Just in Time production, the most obvious criteria being the following:

C1. Regularize the consumption of the parts. The purpose of this criterion is to control the stock levels of the component parts of mixed products (e.g., in the manufacture of engines: block, cylinder head, cylinders and pistons, camshaft, gear change, etc.) throughout the manufacturing process on the assembly line.

C2. Regularize workloads at line stations. The purpose of this criterion is to avoid or smooth the work overloads that are generated when a manufacturing sequence consecutively contains a series of products rich in process time. This criterion is purely ergonomic and its objective is to avoid or reduce the risk of injury to line operators due to intermittent overloads.

C3. Regularize the manufacture of mixed products throughout the manufacturing sequence. This criterion tries to collect, in a simple way and to facilitate management, the benefits of criteria C1 and C2, since it encourages, without optimizing, both the regularity of the consumption of the component parts and the regularity of the workloads in the production line.

On the other hand, the incorporation of Heijunka in production sequence problems can be characterized by three methods:

M1. Constraints: For example, imposing minimum and maximum manufacturing levels on the job types $(i = 1, ..., n)$ in each manufacturing cycle $(t = 1, \dots, T)$ and/or imposing minimum and maximum consumption values on the component parts of mixed products in each manufacturing cycle.

M2. Objective function: Maximizing the constancy of the product manufacturing rates [15] and/or the component consumption rates [5] and/or the rates of the required processing times in the workstations.

M3. Mixed characterization: There is also the possibility of establishing a mixed characterization of Heijunka, which incorporates into the sequence models the two previous methods: (a) restrictions and (b) an objective function.

In this work, the third criterion (C3) and the first method (M1) have been added to the genuine $Fm/prmu/C_{\max}/d_i$ problem to achieve sequences with minimum makespan ($C_{\max}$: time that elapses from the start of work to the end) and with some properties that propitiate the regularity of product manufacturing through restrictions.

The main contributions of this work are: (i) description and formulation of a new problem that we call $Hejunka - Fm/prmu/C_{\max}/d_i$; (ii) design and implementation of a Metaheuristic based on Multistart and Local Search (MS-Q) to solve the new problem; (iii) a computational analysis of MS-Q and MILP (CPLEX solver) performance in CPU time and quality of solutions using real-dimension instances related to case study; and (iv) an economic-productive feasibility study to implement the solutions on a production line.

The remaining text has the following structure. Section 2 is dedicated to presenting the new problem under study which is illustrated with an example in Sect. 3. In Sect. 4, the designed MS-Q procedure is described. In Sect. 5, a case study with its data is shown, as well as the procedures used and their results. Finally, Sect. 6 offers some conclusions about this work.

2 Heijunka – $Fm/prmu/C_{\max}/d_i$ Problem

To incorporate Heijunka, we will indicate that the sequence $\pi(T) = (\pi_1, \dots, \pi_T)$, which is composed of T units of jobs, has the property of preservation of the production mix if the set of restrictions (1) is satisfied. We also call this property *Quota property*:

$$\lambda_i t \le X_{i,t} \le \lambda_i t \equiv |X_{i,t} - \lambda_i t| < 1 \ \ \forall i \in I, \ \ \forall t \in \mathrm{T}; X_{i,T} = d_i \ \forall i \in I \quad (1)$$

where:

- I : set of product types, $i = 1, .., |I|$.
- T: set of manufacturing cycles in every demand plan, $t = 1, .., |\mathrm{T}|; T \equiv |\mathrm{T}|$.
- d_i : demand for units of type $i \in I$ in an arbitrary demand plan.
- λ_i : proportion of units of type $i \in I$: $\lambda_i = d_i/T \ \forall i \in I$.
- $X_{i,t}$: number of units of type $i \in I$ in the partial sequence $\pi(t) \subseteq \pi(T)$: actual production associated with the partial sequence $\pi(t)$.

The *Quota property* (1) imposes that the actual production $X_{i,t}$, for every product ($i \in I$ and every manufacturing cycle $t \in \mathrm{T}$, must be an integer as close as possible to its ideal production $\lambda_i t$. The ideal production $(\lambda_i t)$ is defined as the quota of manufacturing time given to a product $(i \in I)$ until the end of each production cycle $(t = 1, .., |\mathrm{T}|)$.

Under such conditions, we can present a model for the $Fm/prmu/C_{\max}/d_i$ that accounts for two types of aspects:

Efficiency: objective function to minimize the maskespan $C_{\max}$.

Technical-productive: Quota property to enforce preservation of the production mix in the Heijunka manufacturing sequence $\pi(T)$.

Effectively, assuming the following data is known:

- The set of job types $(I\,:\,i=1,..,|I|)$ and the set of stations $(K\,:\,k=1,..,|K|)$.
- The processing times $p_{i,k}$ $(i\in I\wedge k\in K)$ of the operations.
- The demand vectors $\vec{d}=\left(d_1,\ldots,d_{|I|}\right)$ and production mix $\vec{\lambda}=\left(\lambda_1,\ldots,\lambda_{|I|}\right)$.

The problem is finding a *Quota sequence* of T jobs $\pi(T)=\left(\pi_1,\ldots,\pi_T\right)$ with minimum makespan $C_{\max}$ that satisfies the demand plan represented by the vector $\vec{d}$. The formulation of the model is as follows.

2.1 Model Q-FSP

$$\min\ \mathcal{F}(\pi(T))=C_{\max}\equiv C_{m,T} \tag{2}$$

$$C_{k,t}\left(\pi_t\right)=S_{k,t}\left(\pi_t\right)+p_{\pi_t,k}\quad\forall k\in K\;\forall t=1,..,T \tag{3}$$

$$S_{k,t}\left(\pi_t\right)=\max\left(C_{k,t-1}\left(\pi_{t-1}\right),C_{k-1,t}\left(\pi_t\right)\right)\;\forall k\in K\;\forall t=1,..,T \tag{4}$$

$$X_{i,\,t}=\left|\left\{\pi_\tau\in\pi(t)\subseteq\pi(T)\,:\;\pi_\tau=i\in I\right\}\right|\quad\forall i\in I\;\forall t=1,..,T \tag{5}$$

$$\lambda_i t\le X_{i,\,t}\le\lambda_i t\quad\forall i\in I\;\forall t=1,..,T \tag{6}$$

$$X_{i,\,T}=d_i\quad\forall i\in I \tag{7}$$

$$C_{k,0}=0\;\forall k\in K \tag{8}$$

$$C_{0,t}=0\;\forall t=1,..,T \tag{9}$$

In the model Q-FSP, the identity (2) expresses the minimization of the objective function $\mathcal{F}(\pi(T))$ that attends to the time of completion of the last job or product π_T of the production sequence $\pi(T)$ in the last machine ($k=m$); that is: $C_{\max}\equiv C_{m,T}$. The equality (3) determines the minimum time of completion of the t-th job π_t in production sequence $\pi(T)$ in machine $k\in K\,:\;C_{k,t}\left(\pi_t\right)$. Meanwhile, the equality (4) determines the minimum start time $S_{k,t}$ of the t-th job π_t in $\pi(T)$ in machine $k\in K$. Formula (5) serves to count the number of jobs of type $i\in I$ in the partial sequence $\pi(t)\subseteq\pi(T)$. The conditions (6) impose the *Quota* property on the manufacturing sequence $\pi(T)$. The equalities (7) impose the satisfaction of the demand plan ($d_i\;\forall i\in I$). Finally, conditions (8) and (9) set the start of completion times.

3 An illustrative example

In order to illustrate the problem under study, the following example is presented: There are 6 jobs or products ($T=6$), of which 3 are type A, 1 is type B, and 2 are type C. The units of product are processed in 3 workstations ($|K|=3$) with different processing times. The processing time of each unit of type of product (A, B, C) in each workstation $\left(m_1,m_2,m_3\right)$ is that set out in Table 1.

The optimal manufacturing sequence for the proposed example, in order to minimize the completion time of all the jobs on the production line $\left(C_{\max}\right)$, for the problem $Fm/prmu/C_{\max}/d_i$ is $\pi_1(6)=(C,C,A,A,A,B)$. Figure 1 shows the Gantt chart for this sequence.

For its part, Fig. 2 shows the Gantt chart corresponding to an optimal sequence for the problem Hejunka $-\,Fm/prmu/C_{\max}/d_i$, in which the satisfaction of the Quota property of all types of product is imposed in all manufacturing cycles. The sequence $\pi_2(6)=(C,A,A,C,A,B)$ has a value of the objective function $C_{\max}\left(\pi_2\right)=34$.

Considering the sequences $\pi_1(6)$ y $\pi_2(6)$, it can be stated:

(i) The solution $\pi_1(6)$ presents a value of $C_{\max}$ less by one unit of time than that corresponding to the solution $\pi_2(6)$ (i.e. : $C_{\max}\left(\pi_2\right)-C_{\max}\left(\pi_1\right)=34-33=1$). This means that $\pi_1(6)$ is more efficient than $\pi_2(6)$ in terms of completion time for all jobs.

(ii) The solution $\pi_2(6)=(C,A,A,C,A,B)$ satisfies the Quota property at all positions in the sequence.

(iii) The solution $\pi_1(6)=(C,C,A,A,A,B)$ violates the Quota property at 3 positions in the sequence, as detailed in Table 2.

In view of Table 2, we can state that the sequence $\pi_1(6)=(C,C,A,A,A,B)$ does not satisfy the Quota property for product types A and C in the cycle $t=2$ nor for product type C in cycle $t=3$, therefore, the sequence $\pi_1(6)$ violates the Quota property in 8.33% of the constraints.

In the subsection dedicated to the implementation of solutions in a production line, the advantages offered by planning sequences satisfying the Quota property within the Heijunka ideology are described.

Table 1 Processing times ($p_{i,k}$) required by the units of product, according to type, in each workstation

	$A\ \left(d_A=3\right)$	$B\ \left(d_B=1\right)$	$C\ \left(d_c=2\right)$	$\sum_{\forall i} d_i\times p_{i,k}$
m_1	5	4	3	25
m_2	5	4	4	27
m_3	4	3	5	25
$\sum_{\forall k} d_i\times p_{i,k}$	42 (3×14)	11 (1×11)	24 (2×12)	$p_{\text{tot}}=77$

The total processing time required to the production line is $p_{\text{tot}}=77$

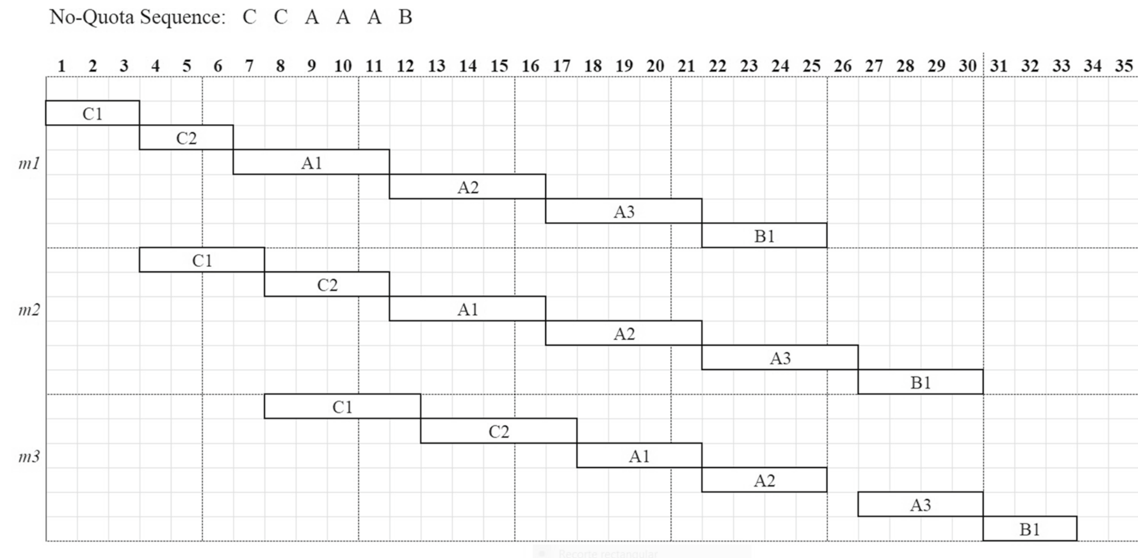

Fig. 1 Gantt chart for the sequence $\pi_1(6) = (C, C, A, A, A, B)$. The sequence $\pi_1(6)$ is optimal for the problem $Fm/prmu/C_{\max}/d_i$, and its value is $C_{\max}(\pi_1) = 33$

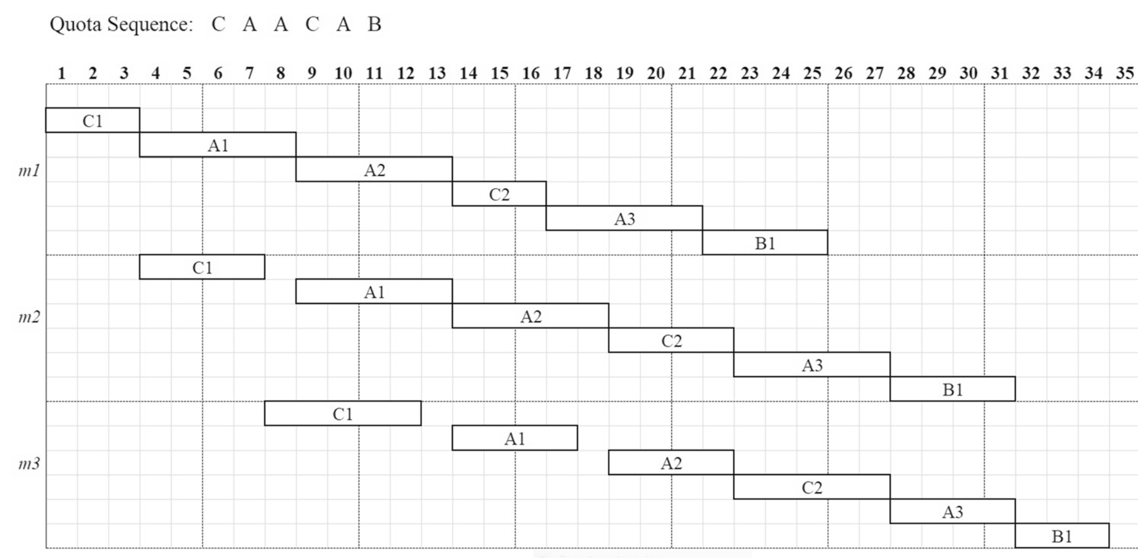

Fig. 2 Gantt chart for the sequence $\pi_2(6) = (C, A, A, C, A, B)$. The sequence $\pi_2(6)$ is optimal for the $Hejunka - Fm/prmu/C_{\max}/d_i$ problem, and its value is $C_{\max}(\pi_2) = 34$

Table 2 Solution $\pi_1(6) = (C, C, A, A, A, B)$: the values of the accumulated productions $X_{i,t}$ and the intervals $[a, b]$ are shown

i	$t = 1$		$t = 2$		$t = 3$		$t = 4$		$t = 5$		$t = 6$	
	$X_{i,t}$	$[a, b]$	$X_{i,t}$	$[a, b]$	$X_{i,t}$	$X_{i,t}$	$X_{i,t}$	$[a, b]$	$X_{i,t}$	$[a, b]$	$X_{i,t}$	$[a, b]$
A	0	[0,1]	**0**	[1,1]	1	[1,2]	2	[2,2]	3	[2,3]	3	[3,3]
B	0	[0,1]	0	[0,1]	0	[0,1]	0	[0,1]	0	[0,1]	1	[0,1]
C	1	[0,1]	**2**	[0,1]	**2**	[1,1]	2	[1,2]	2	[1,2]	2	[2,2]

The values $a = \lambda_i t$ and $b = \lambda_i t$ are respectively lower and upper limits that are imposed on the variables $X_{i,t}$ $(\forall i \forall t)$ to achieve a Quota sequence

4 Metaheuristic procedure for *Heijunka* − *Fm*/*prmu*/C_{max}/d_i

The proposed metaheuristic is based on a Multistart procedure with Local Search similar to Bautista and Alfaro [2]. Indeed, the proposed procedure, MS-Q, consists of a first phase (constructive phase) which provides an initial solution through a randomized greedy procedure, and a second phase (improvement phase) which uses local search procedures to reach the local optima in one or more specific neighborhoods.

After setting a prefixed number of iterations (construction plus improvement), MS-Q metaheuristic obtains in phase-1 manufacturing sequences, $\pi(T) = (\pi_1, \dots, \pi_T)$, that satisfy the Quota property, and then, in phase-2, those sequences are subjected to local optimization in order to minimize the completion time of the last job in the last workstation, that is: $C_{\max}$.

4.1 Phase 1: construction of a Quota sequence

The problem of the construction of a Quota sequence, which we will call *Quota-Product Rate Variation Problem* (Q-PRV), can be formulated as a Binary Linear Programming (BLP) representing maximum constraints satisfaction problem, as follows.

4.2 Model maxsat Q-PRV

$$\min\ \mathcal{Z}_{\text{sum}}(\pi(T)) = \sum_{t=1}^{T}\sum_{i=1}^{n} z_{i,t} \Leftrightarrow \max\ \mathcal{Z}'_{\text{sum}}(\pi(T)) = \sum_{t=1}^{T}\sum_{i=1}^{n}(1 - z_{i,t}) \tag{10}$$

$$\sum_{i=1}^{n} x_{i,t} = 1 \quad \forall t = 1,..,T \tag{11}$$

$$\sum_{t=1}^{T} x_{i,t} = d_i \quad \forall i = 1,..,n \tag{12}$$

$$X_{i,t} = \sum_{\tau=1}^{t} x_{i,\tau} \qquad \forall i = 1,..,n; \forall t = 1,..,T \tag{13}$$

$$\left|X_{i,t} - \lambda_i t\right| < 1 + z_{i,t} \qquad \forall i = 1,..,n; \forall t = 1,..,T \tag{14}$$

$$x_{i,t} \in \{0,1\} \qquad \forall i = 1,..,n; \forall t = 1,..,T \tag{15}$$

$$z_{i,t} \in \{0,1\} \qquad \forall i = 1,..,n; \forall t = 1,..,T \tag{16}$$

$$X_{i,t} \in \mathbb{Z}^+ \cup \{0\} \qquad \forall i = 1,..,n; \forall t = 1,..,T \tag{17}$$

where $x_{i,t}$ $(\forall i \forall t)$ is a binary variable that equals 1 if and only if a unit of type of product $i \in I$ occupies position t of the manufacturing sequence $\pi(T)$, while binary variable $z_{i,t}$ $(\forall i \forall t)$ takes the value 0 when the type of product $i \in I$ satisfies the property Quota in the production cycle t and is equal to 1 otherwise.

In the Maxsat Q-PRV model, the objective function (10) corresponds to the minimization of the number of Quota constraints violated ($\mathcal{Z}_{\text{sum}}$). Equalities (11) impose that each position in the sequence has a job assigned, while equalities (12) force compliance with the demand plan $\vec{d} = (d_1, \dots, d_n)$. The equalities (13) are used to determine the accumulated productions $X_{i,t}$ $(\forall i \forall t)$ of all types of jobs and up to each manufacturing cycle. The inequalities (14) force the satisfaction of the Quota property by all types of jobs $(\forall i \in I)$ in all positions of the sequence $(\forall t \in \mathrm{T})$. Finally, conditions (15) and (16) impose that the variables $x_{i,t}$ and $z_{i,t}$ are binary, while conditions (17) force the accumulated production $(X_{i,t})$ are integers and not negative.

To generate Quota sequences in accordance with the Maxsat Q-PRV model, an enumerative deterministic procedure can be designed based on the branching and cutting of partial solutions; however, in this work we have chosen to use random to promote the diversity of the initial solutions generated in Phase 1, thus allowing them to belong to different regions of the feasible solutions space.

Another indirect way of constructing sequences that satisfy all or a large part of the Quota constraints (14) is to determine integer values for the real production variables $X_{i,t}$ as close as possible to their ideal values $\lambda_i t$ and that, in addition, these values are consistent with the rest of the restrictions of the Maxsat Q-PRV model. To do this, it is enough to change the objective function (10) for a function that measures the discrepancies between the real and ideal accumulated productions. Some examples of discrepancy functions that we refer to are the following:

$$\min\ \Delta_1(\pi(T)) = \sum_{t=1}^{T}\sum_{i=1}^{n}\left(X_{i,t} - \lambda_i t\right)^2 \tag{18}$$

$$min\ \Delta_2(\pi(T)) = \sum_{t=1}^{T}\sum_{i=1}^{n}\left|X_{i,t} - \lambda_i t\right| \tag{19}$$

$$\min \Delta_3(\pi(T)) = \max_{1\le t\le T}\max_{1\le i\le n}\left(X_{i,t} - \lambda_i t\right)^2 \tag{20}$$

$$\min \Delta_4(\pi(T)) = \max_{1\le t\le T}\max_{1\le i\le n}\left|X_{i,t} - \lambda_i t\right| \tag{21}$$

In this work, the function (18), sum of quadratic discrepancies:$\Delta_1(\pi(T))$, is fundamental to construct a random generator of Quota sequences.

First, in Phase 1 a sequence of jobs $\pi(T) = \left(\pi_1, \dots, \pi_T\right)$ is constructed satisfying the Upper Quota property ((i.e. $X_{i,\,t} \le \lambda_i t, \quad \forall i \forall t$)), progressively and assigning at each stage t $(t = 1, \dots, T)$ a job from the CL(t) list of candidates that can be drawn to occupy the position t of the manufacturing sequence. Consequently, when stage t is reached, it is added to the sequence consolidated in the previous stage, $\pi\left(t-1\right) = (\pi_1, ., \pi_{t-1})$, a job $i \in \mathrm{CL}(t)$. List $CL(t)$ is constructed like this:

$$\mathrm{CL}(t) \; = \left\{ i \in I : (n_i < d_i) \wedge \left(n_i + 1 \le \lambda_i t\right) \right\} \tag{22}$$

where n_i is the number of jobs of type $i \in I$ that contains the production sequence $\pi\left(t-1\right) = (\pi_1, ., \pi_{t-1})$.

Therefore, for a job type $i \in I$ to enter the list $CL(t)$ of stage t, it must meet the following two conditions:

1. The job type does not have its demand fulfilled: $n_i = X_{i,\,t-1} < d_i$.
2. The difference between the upper Quota value $\lambda_i t$, corresponding to the ideal production of stage t, and the consolidate production up to the previous stage must be greater than or equal to one unit: $\lambda_i t - n_i \ge 1$.

Note that the candidate list, CL(t), only contains jobs or products that satisfy the upper Quota property; this is done like this because if the strict satisfaction of the Quota property is imposed: $\lambda_i t \le n_i + 1 \le \lambda_i t \equiv \left| n_i + 1 - \lambda_i t \right| < 1$, then there is a risk, and this is often the case, that CL(t) remains empty.

Second, the sum of quadratic discrepancies associated with each candidate job that is contained in the list CL(t) is evaluated, using the indices $g_i^{(t)}$:

$$g_i^{(t)} = \sum_{k=1}^{n} \left(n_k + \delta_{i,k} - \lambda_k t\right)^2 \quad \forall i \in \mathrm{CL}(t) \tag{23}$$

where n_k is the number of jobs of type $k \in I$ that contains the sequence consolidated in the previous stage, $\pi(t-1)$, and $\delta_{i,k}$ is the Kronecker delta: $\delta_{i,i} = 1 \wedge \delta_{i,k} = 0 \;\text{ if }\; i \ne k$.

Third, the jobs in the list CL(t) are ordered according to the increasing order of the priority indices $g_i^{(t)}$, giving rise to the ordered list $\overline{\mathrm{CL}}(t)$.

Alternatively, the sorting of the list CL(t) to construct the list $\overline{\mathrm{CL}}(t)$ can be made more efficient by using the priority indices $f_i^{(t)}$ which are defined as in (24).

$$f_i^{(t)} = \lambda_i t - n_i \quad \forall i \in \mathrm{CL}(t) \tag{24}$$

The equivalence between the orderings of the jobs according to the indices $g_i^{(t)}$ and $-f_i^{(t)}$ is demonstrated below.

Theorem 1 Given a partial sequence of jobs $\pi\left(t-1\right) = (\pi_1, ., \pi_{t-1})$ and a list of jobs $CL(t)$ constructed according to (22), then, the ordering of jobs of CL(t) according to the indices $g_i^{(t)}$ (see (23)) is opposite to the ordering according to the indices $f_i^{(t)}$ (see (24)).

Proof Indeed, let $H_{k,t} = n_k - \lambda_k t$ $(\forall k \in I, \forall t \in \mathrm{T})$, then, it can be stated:

$$g_i^{(t)} \le g_j^{(t)} \Leftrightarrow \sum_{k=1}^{n} \left(n_k + \delta_{i,k} - \lambda_k t\right)^2 \le \sum_{k=1}^{n} \left(n_k + \delta_{j,k} - \lambda_k t\right)^2 \Leftrightarrow$$

$$\sum_{k=1}^{n} \delta_{i,k}^2 + \sum_{k=1}^{n} H_{k,t}^2 + 2\sum_{k=1}^{n} \delta_{i,k} H_{k,t} \le \sum_{k=1}^{n} \delta_{j,k}^2 + \sum_{k=1}^{n} H_{k,t}^2 + 2\sum_{k=1}^{n} \delta_{j,k} H_{k,t} \Leftrightarrow$$

$$\sum_{k=1}^{n} \delta_{i,k} H_{k,t} \le \sum_{k=1}^{n} \delta_{j,k} H_{k,t} \Leftrightarrow H_{i,t} \le H_{j,t} \Leftrightarrow \lambda_i t - n_i \ge \lambda_j t - n_j \Leftrightarrow f_i^{(t)} \ge f_j^{(t)}$$

After this ordering, the list $\overline{\mathrm{CL}}(t)$ is reduced through a mechanism that is a function of the admission factor α (percentage of candidate jobs), with this operation, the restricted list $\overline{\mathrm{RCL}}(t, \alpha)$ is obtained, which coincides with $\overline{\mathrm{CL}}(t)$ when $\alpha = 100\% = 1$, while if $\alpha = 1/|I|$, the best candidate job from such lists is selected at each stage t.

Taking into account all the above, Algorithm A1 is formalized.

A1: Algorithm A1 for the constructive phase of a Upper Quota sequence of jobs: $\pi(T)$

// *Initialization*
input $\alpha, I, D, d_i \; \forall i \in I$
initialize $T = D, t = 0, \pi(t) = \{\emptyset\}, (n_i = 0, \lambda_i = d_i/D) \; \forall i \in I$
// *Create the candidate set* $CL(t)$ - see formula (22) -
while $(t \leq T)$ **do**
  set $t = t + 1$
  set $CL(t) = \{i \in I: (n_i < d_i) \wedge (n_i + 1 \leq \lceil \lambda_i t \rceil)\}$
// *Evaluate alternative according to* $f_i^{(t)}$ *priority indexes* – Theorem 1 and (24) -
  for all $(i \in CL(t))$ **do**
    set $f_i^{(t)} \equiv \lambda_i t - n_i$
  end for
// *Sort alternatives according to the decreasing order of* $f_i^{(t)}$ – Theorem 1 and (24) -
  sort $CL(t)$: **set** $\overline{CL}(t)$ as the ordered list from $CL(t)$ according the $f_i^{(t)}$ values.
// *Select alternative according to the admission factor* α
  set $pos_\alpha = -int(-\alpha \cdot |\overline{CL}(t)| \cdot RND) \equiv -int(-|\overline{RCL}(t, \alpha)| \cdot RND)$
  set $i_\alpha = i \in \overline{CL}(t): pos_i = pos_\alpha$
// *Update*
  set $n_{i_\alpha} \leftarrow n_{i_\alpha} + 1$
  set $\pi(t) = \pi(t-1) \cup \{i_\alpha\}$
end while
// *End Algorithm A*1

Note that Algorithm A1 is a general method of generating Upper Quota sequences, $\pi(T)$, independently of any other goal. Sometimes, the Algorithm A1 obtains solutions that also satisfy the Lower Quota property $\left(\lambda_i t \leq X_{i,t} \;\; \forall i \in I \; \forall t \in \mathrm{T}\right)$, when this purpose is not achieved then Algorithm A2 is run.

A2: Algorithm A2 for the constructive phase of the Quota sequence of jobs: $\hat{\pi}(T)$

1: // *Initialization*
2: **input** $I, D, d_i\ \forall i \in I, \pi(T) = (\pi_1, \dots, \pi_T)$ from A1.
3: **initialize** $T = D, t = 0, quota = false,\ \lambda_i = d_i/D\ \forall i \in I$
4: // *Quota Property*
5: **while** $(t \le T)$ **do**
6: **set** $t = t + 1$
7: **for all** $(i \in I)$ **do**
8: **set** $X_{i,t} = |\{\pi_\tau \in \pi(t) = (\pi_1, \dots, \pi_t) \subseteq \pi(T): \pi_\tau = i \in I\}|$ –see formula (5) -
9: **if** $\lfloor \lambda_i t \rfloor \le X_{i,t} \le \lceil \lambda_i t \rceil$ **then**
10: **set** $quota = true$
11: **else**
12: **set** $quota = false$
13: **exit while**
14: **end if**
15: **end for**
16: **set** $\hat{\pi}(t) = \pi(t)$
17: **end while**
18: **if** $quota = false$ **then**
19: **solve** $MAXSAT$: **set** $\hat{\pi}(T) \longleftarrow$ maxsat$(\pi(T), \lfloor \lambda_i t \rfloor \le X_{i,t} \le \lceil \lambda_i t \rceil)$
20: **end if**
21: // *End Algorithm A2*

The MAXSAT procedure in A2 (Line 19 from A2) is an exchange algorithm, based on Local Search with exhaustive descent, that solves the Maxsat Q-PRV problem satisfying the constraints (14): ($|X_{i,t} - \lambda_i t| < 1,\ \forall i\, \forall t$), which provides as a solution a sequence $\hat{\pi}(T)$ that does satisfy the *Quota property* in all of the manufacturing cycles.

Specifically, MAXSAT algorithm starts from the solution $\pi(T)$ generated by Algorithm A1 and performs in each iteration the exchange of the jobs of every pair of positions of the current sequence $\hat{\pi}(T)$, consolidating, in each iteration, the Last sequence that minimizes the number of Quota constraints violated. The execution of the MAXSAT algorithm ends when $\mathcal{Z}_{\text{sum}}(\hat{\pi}(T)) = 0$ or $\mathcal{Z}'_{\text{sum}}(\hat{\pi}(T)) = |I| \times T$ (see formula (10)).

Obviously, the CPU time efficiency of the MAXSAT procedure is higher the lower the number of Quota constraints violated by the initial sequence $\pi(T)$; for this reason, the sequences provided by the A1 algorithm are used, since they comply with the Upper Quota property and tend to comply with the Lower Quota property when the values of the admission factor α are small.

Fig. 3 Nissan Pathfinder Engine. Characteristics: (i) 747 parts and 330 references, (ii) 378 elemental assembly tasks grouped in 140 production line tasks

Table 3 Daily demands by product type and plan $\left(d_{i,\varepsilon}\right)$ for the 23 instances Nissan-9Eng.I ($\varepsilon \in$ E)

$\varepsilon \in$ E	1	2	3	4	5	6	7	8	9	SUV	Van	Truck	Total
1	30	30	30	30	30	30	30	30	30	90	60	120	270
2	30	30	30	45	45	23	23	22	22	90	90	90	270
3	10	10	10	60	60	30	30	30	30	30	120	120	270
4	40	40	40	15	15	30	30	30	30	120	30	120	270
5	40	40	40	60	60	8	8	7	7	120	120	30	270
6	50	50	50	30	30	15	15	15	15	150	60	60	270
7	20	20	20	75	75	15	15	15	15	60	150	60	270
8	20	20	20	30	30	38	38	37	37	60	60	150	270
9	70	70	70	15	15	8	8	7	7	210	30	30	270
10	10	10	10	105	105	8	8	7	7	30	210	30	270
11	10	10	10	15	15	53	53	52	52	30	30	210	270
12	24	23	23	45	45	28	28	27	27	70	90	110	270
13	37	37	36	35	35	23	23	22	22	110	70	90	270
14	37	37	36	45	45	18	18	17	17	110	90	70	270
15	24	23	23	55	55	23	23	22	22	70	110	90	270
16	30	30	30	35	35	28	28	27	27	90	70	110	270
17	30	30	30	55	55	18	18	17	17	90	110	70	270
18	60	60	60	30	30	8	8	7	7	180	60	30	270
19	10	10	10	90	90	15	15	15	15	30	180	60	270
20	20	20	20	15	15	45	45	45	45	60	30	180	270
21	60	60	60	15	15	15	15	15	15	180	30	60	270
22	20	20	20	90	90	8	8	7	7	60	180	30	270
23	10	10	10	30	30	45	45	45	45	30	60	180	270

Table 4 Grouping of the 23 instances Nissan-9Eng.I into 7 categories of demand plans

Category	Plans	Type of demand plan
01	#1	Balanced demand for products
02	#2	Balanced demand for families
03	#3 to #5	Very low demand for a family
04	#6 to #8	High demand for a family
05	#9 to #11	Very high demand for a family
06	#12 to 17	Family demand in arithmetic progression
07	#18 to 23	Family demand in hypergeometric progression

4.3 Phase 2: improvement $C_{\max}$ of the quota sequences through local search

The improvement phase starts with a *Quota* sequence $\hat{\pi}(T)$ in which five descent algorithms are run consecutively and repetitively in five neighborhoods (three exchange and two insertion) until none of them improves the best solution that is achieved during the iteration. From two arbitrary *Quota* sequences, the one that offers the least total completion time $\left(C_{\max}\right)$ is selected. The descent algorithms are based on the exchange and insertion of jobs, and they are oriented to the exploration of sequence cycles in both increasing and decreasing order. The five descent algorithms are:

LS1. *Forward exchange for ranges of job types*: For all t position of the current sequence, $\hat{\pi}(T)$., the job type is determined that is in that position and the next closest locus is searched, $t' > t$, that is occupied by the same type (i.e., $\hat{\pi}_t = \hat{\pi}_{t'}$); if no such locus exists, then its value is set by making $t' = T + 1$. Just after, the tentative exchange between $\hat{\pi}_t$ and the jobs located in the range $\left[t + 1, t' - 1\right]$ of the sequence is made. The first exchange that reduces the total completion time $C_{\max} \equiv C_{m,T}$ (see (2)) is consolidated as long as the resulting sequence satisfies the Quota property.

LS2. *Backward exchange for ranges of job types*: This procedure is similar to the previous one, but in this case the search is performed for $t = T$ to 1 step -1. Obviously, if the previous closest locus, $t'(t' < t)$, with the same job type $\left(\hat{\pi}_t = \hat{\pi}_{t'}\right)$ does not exist, it is considered $t' = 0$. The first exchange that reduces $C_{\max}$ is consolidated as long as the resulting sequence satisfies the Quota property.

LS3. *Complete exchange between pairs of positions*: This procedure is used to reinforce the previous two and uses a larger neighborhood. At each iteration, for all position t of the current sequence $\hat{\pi}(T)$, the job

Table 5 Processing time under normal operation $\left(p_{i,k}\right)$ in seconds of the 9 types of engines ($i \in I$) in the 21 workstations ($k \in K$) of the set of Nissan-9Ing.I

$k \backslash i$	1	2	3	4	5	6	7	8	9	*Av*
1	104	100	97	92	100	94	103	109	101	100.0
2	103	103	105	107	101	108	106	102	110	105.0
3	165	156	164	161	148	156	154	164	155	158.1
4	166	175	172	167	168	167	168	156	173	168.0
5	111	114	114	115	117	117	115	111	111	113.9
6	126	121	122	124	127	130	120	121	134	125.0
7	97	96	96	93	96	89	94	101	92	94.9
8	100	97	95	106	94	102	103	102	100	99.9
9	179	174	173	178	178	171	177	171	174	175.0
10	178	172	172	177	178	177	175	173	175	175.2
11	161	152	168	167	167	166	172	157	177	165.2
12	96	106	105	97	101	100	96	104	96	100.1
13	99	101	102	101	99	101	96	102	99	100.0
14	147	155	142	154	146	143	154	153	155	149.9
15	163	152	156	152	153	152	154	156	156	154.9
16	163	185	183	178	169	173	172	182	171	175.1
17	173	179	178	169	173	178	174	175	175	174.9
18	176	167	181	180	172	173	173	168	184	174.9
19	162	150	152	152	160	151	155	148	167	155.2
20	164	161	157	159	162	160	162	158	157	160.0
21	177	161	154	168	172	170	167	149	169	165.2

of the *locus* t is exchanged with the job of the *locus* $t' \in [t+1, T]$, if $\hat{\pi}_t \neq \hat{\pi}_{t'}$. The last job exchange that minimizes $C_{\max} \equiv C_{m,T}$ is consolidated, provided the Quota property is satisfied.

LS4. *Forward insertion for ranges of job types*: For all t position of the current sequence, $\hat{\pi}(T)$, the job type in the t position is detected and the next closest locus $t'(t' > t)$ is searched that is occupied by the same type ($\hat{\pi}_t = \hat{\pi}_{t'}$); if these locus does not exist, it is considered $t' = T + 1$. Following, the $\hat{\pi}_t$ job is inserted in the range of sequence positions $\left[t+1, t'-1\right]$. Then, the first insertion that leads to reduce $C_{\max} \equiv C_{m,T}$ is done as long as the resulting sequence satisfies the Quota property.

LS5. *Backward insertion for ranges of job types*: This insertion procedure is similar to LS4 with respect to the neighborhood, and analogous in the search for types of jobs to LS2.

While there is improvement, the above five algorithms are repeated.

5 A case study in an engine plant

5.1 Data set

The computational experience proposed here is focused on comparing the MS-Q and MILP (Mixed Integer Linear Programming) procedures in terms of the quality of the solutions and the CPU times. As in Bautista-Valhondo and Alfaro-Pozo [7], the analysis is related to a case study of the Nissan plant in Barcelona: an assembly line of nine types of engines grouped into three families: SUVs, Vans and Trucks (see an engine example in Fig. 3). The production line under study employs 42 operators work in shifts of 8 h, and the significant data of this case are the following:

- There are 9 job types ($|I| = 9$) so that each job type corresponds to a type of engine.
- The workshop (line) has 21 workstations ($|K| = 21$) arranged in series.
- In this work, we consider 23 engine demand plans $|\mathrm{E}| = 23$ (see Table 3).
- The daily demand is 270 jobs for all demand plans $T \equiv D_{\varepsilon} = 270$ jobs ($\forall \varepsilon \in \mathrm{E}$).
- The demand plans have been grouped into 7 categories (see Table 4).
- The values of the processing times at normal work pace $p_{i,k}(\forall i \in I, \forall k \in K)$ are between 89s and 185s (see Table 5).

Table 6 Results for $C_{\max}$ (seconds) and *Gap* (in millionths) for Nissan-9Eng.I instances using MILP-1, MILP-2 and MS-Q

$\varepsilon \in$ E	MILP-1		MILP-2				MS-Q		
	$C^1_{\max}$	CPU	*LB*	$C^2_{\max}$	Gap	CPU	$C^3_{\max}$	Gap	CPU
1	50,091	45.8	50,100	50,101	20	3600.6	50,101	20	176.8
2	50,174	15.2	50,180	50,180	0	366.7	50,180	0	130.8
3	50,301	10.3	50,303	50,303	0	37.9	50,303	0	15.5
4	50,167	13.6	50,170	50,170	0	38.6	50,170	0	213.0
5	50,379	9.9	50,385	50,385	0	45.7	50,385	0	73.6
6	50,202	14.3	50,202	50,202	0	14.1	50,204	40	2.9
7	50,395	8.3	50,397	50,397	0	33.4	50,397	0	180.1
8	50,123	12.4	50,126	50,128	40	3600.3	50,130	80	233.5
9	50,378	10.4	50,378	50,378	0	17.0	50,378	0	5.3
10	50,619	7.6	50,625	50,625	0	9.0	50,625	0	15.9
11	50,078	25.3	50,084	50,084	0	162.4	50,086	40	48.7
12	50,192	17.4	50,196	50,196	0	102.4	50,196	0	176.0
13	50,123	14.8	50,126	50,136	199	3600.3	50,136	199	12.7
14	50,218	10.1	50,223	50,223	0	134.7	50,224	20	48.7
15	50,242	10.5	50,242	50,242	0	105.0	50,242	0	175.7
16	50,118	55.8	50,123	50,123	0	160.3	50,128	100	129.0
17	50,269	10.6	50,273	50,273	0	74.0	50,275	40	4.3
18	50,273	14.3	50,273	50,273	0	15.1	50,275	40	8.3
19	50,475	8.1	50,481	50,481	0	7.8	50,481	0	15.0
20	50,089	96.1	50,100	50,100	0	65.2	50,100	0	48.1
21	50,307	13.8	50,307	50,307	0	10.5	50,307	0	5.4
22	50,539	7.3	50,545	50,545	0	9.3	50,545	0	31.9
23	50,151	11.0	50,157	50,157	0	44.0	50,158	20	24.3
Av	50,256.7	19.3	50,260.7	50,261.3	11.3	532.8	50,262.0	26.0	77.2
Max	50,619	96.1	50,625	50,625	199	3600.6	50,625	199	233.5
Min	50,078	7.3	50,084	50,084	0	7.8	50,086	0	2.9

Columns *CPU* show the CPU time (seconds) spent solving each instance

All the production plans shown in Table 1 have been used to carry out the computational experimentation developed in this work. As said, the total number of engines assembled in a working day is 270 in two shifts. The 7 categories that allow the grouping of demand plans are summarized in Table 4.

Meanwhile, the values of the processing times $p_{i,k}(\forall i \in I, \forall k \in K)$ for each job type and for each workstation are shown in Table 5.

5.2 Procedures and computational analysis

The compiled codes of the procedures that we have selected in this work are MILP (1 and 2) and MS-Q (running in Intel(R) Core (TM) i7-8750H CPU @ 2.21 GHz, 16 GB RAM, ×64 Windows 10 Pro). Table 6 shows the best results with respect to $C_{\max}$ and CPU Time from MILP (1 and 2) and MS-Q procedures for the 23 datasets of the problem $\varepsilon \in$ E. In "Appendix I", the 46 best Quota-sequences obtained by MILP-2 and MS-Q are published.

In Table 6, the column headings represent the following characteristics:

$\varepsilon \in$ E	Identification number of the instances for Plan#1 to Plan#23
$C^1_{\max}$	Optimal value of makespan for the $Fm/prmu/C_{\max}/d_i$ problem obtained for MILP-1
$C^2_{\max}$	Best makespan value for the Heijunka – $Fm/prmu/C_{\max}/d_i$ problem obtained for procedure MILP-2
$C^3_{\max}$	Best makespan value for the Heijunka – $Fm/prmu/C_{\max}/d_i$ problem obtained for procedure MS-Q
LB	C_{max} lower limit for the Heijunka – $Fm/block/C_{\max}/d_i$ problem obtained for MILP-1 or MILP-2 using the CPLEX solver
Gap	Relative gap between $C^h_{\max}$ $(h \in \{2, 3\})$ and *LB* measured in millionths

The relative gap values (measured in millionths) between $C^k_{\max}$ and *LB* is calculated using formula (25).

Table 7 Some properties of the performance of MS-Q with the set of instances Nissan-9Engine-I

$\varepsilon \in$ E	Phase 1			Phase 2			
	$\alpha^*(\varepsilon)$	$r^*_{mxs}(\varepsilon)$	$\%r_{\text{no\_}Q}(\varepsilon)$	$\text{iter}^*(\varepsilon)$	$\text{sol}^*(\varepsilon)$	$n\_\text{Sol}(\varepsilon)$	$\text{CPU}_1(\varepsilon)$
1	0.50	0	0	13	114	171	13.28
2	0.33	0.83	0.171	12	90	161	11.64
3	0.11	1.00	6.638	1	2	2	15.50
4	1.00	0.89	4.432	9	97	206	23.08
5	0.50	1.00	2.858	7	48	142	10.54
6	0.11	0	0	1	2	2	2.86
7	0.33	0.94	0.597	16	123	159	11.46
8	0.33	0.94	0.362	20	156	156	11.68
9	0.11	1.00	0.006	1	2	2	5.28
10	1.00	1.00	12.442	2	8	129	11.94
11	0.50	1.00	2.372	3	36	195	14.83
12	1.00	0.90	6.798	10	75	161	18.74
13	0.11	0	0	1	8	8	12.69
14	0.50	0.75	1.409	4	32	164	12.88
15	0.50	0.90	1.488	10	128	285	18.31
16	0.20	0	0	12	115	176	10.04
17	0.11	0	0	1	2	2	4.34
18	0.11	1.00	0.796	1	3	3	8.30
19	0.33	1.00	1.529	2	9	128	9.07
20	0.33	1.00	0.434	3	30	208	16.93
21	0.11	1.00	0.541	1	3	3	5.39
22	0.33	1.00	1.996	4	20	109	8.83
23	0.33	1.00	0.422	2	15	250	14.87
Average	–	0.75	1.969	6	49	123	11.85
Maximum	–	1.00	12.442	20	156	285	23.08
Minimum	–	0	0	1	2	2	2.86

$$Gap(h,\varepsilon) = 10^6 \times \frac{C^h_{\max}(\varepsilon) - LB(\varepsilon)}{LB(\varepsilon)} \quad \forall h \in \{2,3\}, \forall \varepsilon \in \mathrm{E} \tag{25}$$

The characteristics of the procedures are:

- MILP-1: Model $Fm/prmu/C_{\max}/d_i$: (i) Objective function for minimizing the $C_{\max}$ value of the production sequence; (ii) implementation for IBM ILOG CPLEX solver (Optimization Studio v.12.2, win-×86–64); (iii) maximum CPU time of 180 s allowed for solving each instance (23 instances). The average CPU time used by each demand plan to find the optimal solution is equal to 19.3 s. This procedure is used to determine adjusted lower bounds for the problem under study.
- MILP-2: Model $Hejunka - Fm/prmu/C_{\max}/d_i$ (this work): (i) Objective function for minimizing the C_{max} value of the Quota production sequence; (ii) implementation for IBM ILOG CPLEX solver (Optimization Studio v.12.2, win-×86–64); (iii) maximum CPU time of 3600 s allowed for solving each instance (23 instances). The average CPU time used by each demand plan to find the best solution is equal to 532.8 s.
- MS-Q: Is the Multistart algorithm presented in this work, which is focused on minimizing the total completion time $C_{\max}$ in Quota manufacturing sequences. The maximum number of iterations for each demand plan from Nissan-9Eng.I instances is equal to 20 with five candidate admission factors $\alpha = (0.11,\ 0.20,\ 0.33,\ 0.50,\ 1)$, which generates in the constructive phase 1863 solutions and 14,110 improved solutions (improvement phase) in 115 executions. MS-Q uses on average a CPU time equal to 77.2 s to find the best solution for each demand plan and each admission factor α.

On the other hand, an analysis of Table 6 reveals the following:

- Procedure MILP-1 obtains and ensures optimal solutions in all instances with 270 jobs (23 instances Nissan-9Eng.I) when the $Fm/prmu/C_{\max}/d_i$ problem is solved (see column $C^1_{\max}$ in Table 6). The solutions obtained by MILP-1 do not necessarily satisfy the Quota property:

MILP-1 violates the Quota property in 18 of 23 demand plans.

- Procedure MILP-2 obtains and ensures optimal solutions in 20 of the 23 instances with 270 jobs when the *Heijunka* – $Fm/prmu/C_{\max}/d_i$ problem is solved (see column $C^2_{\max}$ in Table 6). All the solutions obtained by MILP-2 satisfy the Quota property.
- Procedure MS-Q obtains optimal solutions in 13 of the 23 instances with 270 jobs when the *Heijunka* – $Fm/prmu/C_{\max}/d_i$ problem is solved (see column $C^3_{\max}$ in Table 6). All the solutions obtained by MS-Q satisfy the Quota property.
- Regarding the value of objective $C_{\max}$, on average, MS-Q solutions differ by 0.7 s from MILP-2, in a range of values between 0 and 5 s (see columns $C^2_{\max}$ and $C^3_{\max}$ in Table 6), when considering a 50,770 s workday to build 270 engines. Consequently, MS-Q solutions can be considered equivalent to MILP-2 from the perspective of the management of productive operations.
- The average value of the relative gap between $C^2_{\max}$ and *LB* achieved by MILP-2 is 1.13E-05 in a range of values between 0 and 1.99E-04.
- The average value of the relative gap between $C^3_{\max}$ and *LB* achieved by MS-Q is 2.60E-05 in a range of values between 0 and 1.99E-04.
- The average CPU times used by MILP-1 (to determine lower bounds for the problem under study) are approximately 19.3 s for each instance of 270 jobs in a range of values between 7.3 and 96.1 s, when a maximum CPU time equal to 180 s is imposed on CPLEX to solve each instance for $Fm/prmu/C_{\max}/d_i$ problem.
- The average CPU times used by MILP-2 are approximately 532.8 s for each instance of 270 jobs in a range of values between 7.8 and 3600.6 s, when a maximum CPU time equal to 3600 s is imposed on CPLEX to solve each instance of the problem under study.
- The average CPU time used by MS-Q is equal to 77.2 s within a range of values between 2.9 and 233.5 s, when 20 iterations are performed with the algorithm.
- In average CPU times, MS-Q is 6.902 times faster than MILP-2.

Table 8 Results corresponding to the savings in euros $G(\cdot)$ and the increase in engine production $\Delta P(\cdot)$ for Nissan-9Eng.I instances using procedures MILP-1, MILP-2 and MS-Q

$\varepsilon \in$ E	MILP-1		MILP-2		MS-Q	
	$G(1,\varepsilon)$	$\Delta P(1,\varepsilon)$	$G(2,\varepsilon)$	$\Delta P(2,\varepsilon)$	$G(3,\varepsilon)$	$\Delta P(3,\varepsilon)$
1	1552.00	3.88	1529.14	3.82	1529.14	3.82
2	1362.29	3.41	1348.57	3.37	1348.57	3.37
3	1072.00	2.68	1067.43	2.67	1067.43	2.67
4	1378.29	3.45	1371.43	3.43	1371.43	3.43
5	893.71	2.23	880.00	2.20	880.00	2.20
6	1298.29	3.25	1298.29	3.25	1293.71	3.23
7	857.14	2.14	852.57	2.13	852.57	2.13
8	1478.86	3.70	1467.43	3.67	1462.86	3.66
9	896.00	2.24	896.00	2.24	896.00	2.24
10	345.14	0.86	331.43	0.83	331.43	0.83
11	1581.71	3.95	1568.00	3.92	1563.43	3.91
12	1321.14	3.30	1312.00	3.28	1312.00	3.28
13	1478.86	3.70	1449.14	3.62	1449.14	3.62
14	1261.71	3.15	1250.29	3.13	1248.00	3.12
15	1206.86	3.02	1206.86	3.02	1206.86	3.02
16	1490.29	3.73	1478.86	3.70	1467.43	3.67
17	1145.14	2.86	1136.00	2.84	1131.43	2.83
18	1136.00	2.84	1136.00	2.84	1131.43	2.83
19	674.29	1.69	660.57	1.65	660.57	1.65
20	1556.57	3.89	1531.43	3.83	1531.43	3.83
21	1058.29	2.65	1058.29	2.65	1058.29	2.65
22	528.00	1.32	514.29	1.29	514.29	1.29
23	1414.86	3.54	1401.14	3.50	1398.86	3.50
Average	1173.37	2.93	1162.83	2.91	1161.14	2.90
Maximum	1581.71	3.95	1568.00	3.92	1563.43	3.91
Minimum	345.14	0.86	331.43	0.83	331.43	0.83

- In average relative gap, MILP-2 solutions are at 1.13E-05 of the lower bound while MS-Q solutions are at 2.60E-05 of that bound, which constitutes a technical tie.

For its part, Table 7 shows some properties on the performance of the MS-Q procedure, both in its construction phase and in its improvement phase, when the set of Nissan-9Eng.I instances is solved.

In Table 7, the column headings represent the following characteristics:

$\varepsilon \in \mathrm{E}$	Identification number of the instances for Plan#1 to Plan#23
$\alpha^*(\varepsilon)$	Best admission factor in A1, $\alpha \in \{0.11, 0.20, 0.33, 0.50, 1\}$, for each $\varepsilon \in \mathrm{E}$
$r^*_{mxs}(\varepsilon)$	Utilization rate of MAXSAT procedure in A2 for the best solutions of each demand plan $\varepsilon \in \mathrm{E}$
$r_{no\_Q}(\varepsilon)$	Rate dissatisfaction of the Quota constraints (from A1) for the best solutions of each demand plan $\varepsilon \in \mathrm{E}$. It is measured as a percentage: $\%r_{no\_Q}(\varepsilon)$. The maximum number of Quota constraints is: $\|I\| \times T \equiv \|I\| \times D$
iter$^*(\varepsilon)$	Iteration corresponding to the best solution of each demand plan $\varepsilon \in \mathrm{E}$
sol$^*(\varepsilon)$	Number of solutions improved by Local Search (BL1 to BL5) to get the best solution locally optimal of each demand plan $\varepsilon \in \mathrm{E}$
n_Sol(ε)	Number of solutions improved by Local Search (BL1 to BL5) limiting the MS-Q procedure to a maximum 20 iterations, for each demand plan $\varepsilon \in \mathrm{E}$
$\varepsilon \in \mathrm{E}$	Identification number of the instances for Plan#1 to Plan#23
$CPU_1(\varepsilon)$	CPU time (seconds) per iteration, limiting the MS-Q procedure to a maximum 20 iterations, for each demand plan $\varepsilon \in \mathrm{E}$

5.3 Economic-productive feasibility study

In this subsection, we carry out an analysis of the results considering two aspects: economic and productive.

The first aspect aims to evaluate the economic savings in euros that result from transforming the original assembly line with a cycle time $c = 175\ s$ into a regular flow workshop in the context of the $Fm/prmu/C_{\max}/d_i$ problem.

The second aspect of the productive type is intended to measure the drop in engine production generated by the use of Heijunka concept, used in Just in Time production systems, when imposed on manufacturing sequences that satisfy the Quota property; in this case, we will use the $Heijunka - Fm/prmu/C_{\max}/d_i$ model.

To carry out this analysis, the following hypotheses are taken into account:

h1. The current engine assembly line is made up of 21 workstations arranged in series. At each workstation, a team consisting of two operators operates (42 operators in total).

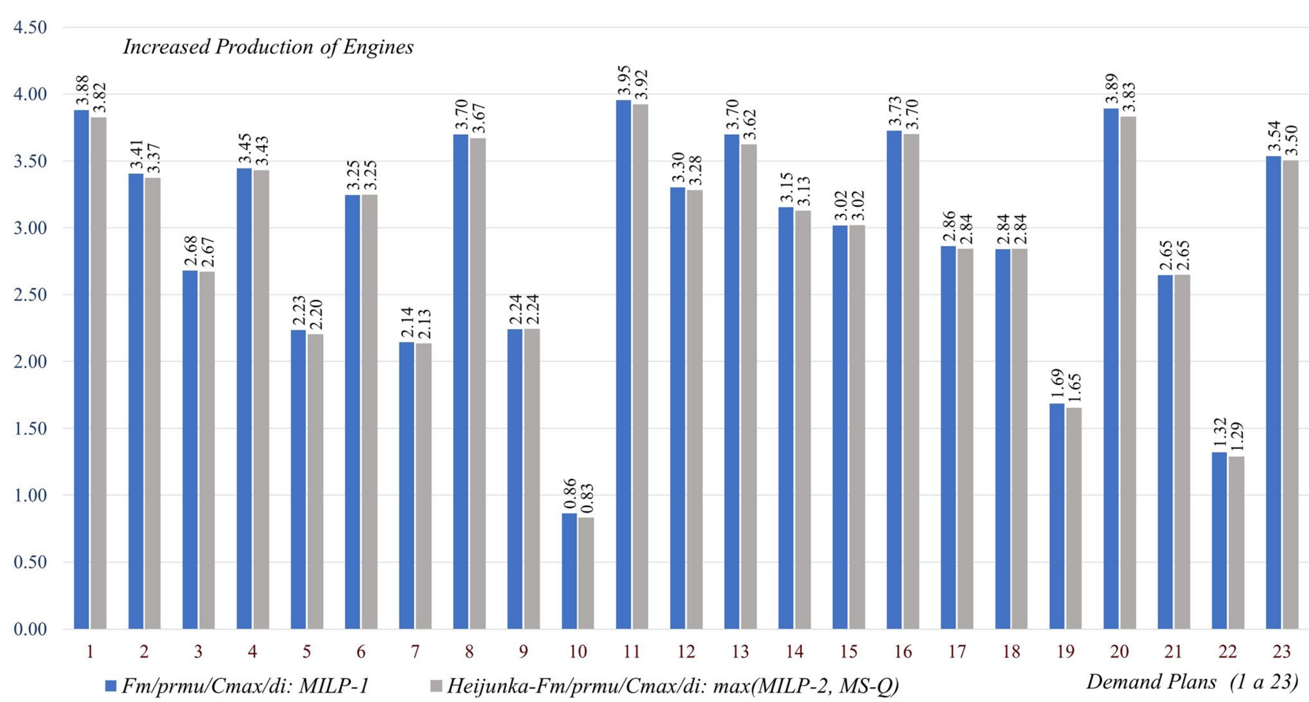

Fig. 4 Daily increased in engine production, over that of the current assembly line, obtained with procedures MILP-1 and MILP-2 or MS-Q for the Nissan-9Eng.I instance set

h2. The current assembly line has a daily production capacity equal to 270 engines. Each production day is divided into two work shifts and each work shift has a productive time equal to 7.05 h, after subtracting the scheduled rest times during the work day, the full duration of which is equal to 8 h per shift.

h3. The cost of loss engine production [6] has been valued at $\varphi = 2.28757$ euros per productive second. The cost φ is calculated taking into account three factors: (i) the average value of a motor that is equal to € 4,000, (ii) the value added to the product by the assembly line that is equal to 10% of the value of the motor, and (iii) the cycle time of the line that is equal to 175 s, that is, $c = 175\ s$ and the temporary window is $l_k = 175$ s.

h4. Assuming a cycle time $c = 175\ s$ and that the assembly line is made up of 21 stations arranged in series, the manufacture of the 270 engines requires a time equal to $C^0_{\max} = 50770$ seconds to complete the 270 jobs when there is no work in progress on the line (no-WIP). Therefore, the direct benefit provided by the line is equivalent to € 108,000 per day.

Under these conditions, the daily savings in euros $G(\cdot)$ and the daily increases in the production of $\Delta P(\cdot)$ motors, achieved with the transformation of the current assembly line into a regular flow workshop, are shown in Table 8.

In Table 8, the $G(\cdot)$ and $\Delta P(\cdot)$ values are determined according to (26) and (27).

$$G(h,\varepsilon) = \varphi \times \left(C^0_{\max} - C^h_{\max}(\varepsilon)\right) \quad \forall h \in \{1,2,3\}, \forall \varepsilon \in \mathrm{E} \tag{26}$$

$$\Delta P(h,\varepsilon) = \frac{C^0_{\max} - C^h_{\max}(\varepsilon)}{c} \quad \forall h \in \{1,2,3\}, \forall \varepsilon \in \mathrm{E} \tag{27}$$

The analysis of Table 8 allows to obtain the following conclusions:

- The daily saving in euros, $G(1,\varepsilon)$, achieved with the transformation of the current line in a flow shop $Fm/prmu/C_{\max}/d_i$, manufacturing 270 engines per day, is equal to € 1173.37 on average. Such savings are included in the interval [345.14, 1581.71], and their values depend on the demand plan used ($\varepsilon \in \mathrm{E}$).
- In case of having the same time to produce as the current one (i.e., $C^0_{\max} = 50770$ s). , the estimate of the average daily increase in engine production is $\Delta P(1,\varepsilon) = 2.93$, by transforming the line into a flow shop and assuming that the demand plans ($\varepsilon \in \mathrm{E}$) do no vary. These increases are included in the interval [0.86, 3.95], and their values depend on the demand plan used (see Fig. 4).
- The transformation of the current line into a regular flow shop subject to Heijunka concept ($Heijunka - Fm/prmu/C_{max}/d_i$) leads to a maximum average saving equal to € 1162.83 per day (see average maximum between the columns $G(2,\varepsilon)$ and $G(3,\varepsilon)$ in Table 8) when 270 engines are manufactured per day. In this case, said savings are included in the interval [331.43, 1568.00] and their values depend on the demand plan used.
- In the case of the Heijunka-flow shop, the estimate of the daily increase in engine production with respect to the current line is equal to 2.91 engines on average (see average maximum between columns $\Delta P(2,\varepsilon)$ and $\Delta P(3,\varepsilon)$ in Table 8), provided that the original production mix does not vary in the demand plans. Here, these increases (engines per day) are included in the interval [0.83, 3.92] (see Fig. 4).

Figure 4 reveals similar performance between the two types of flow shops analyzed, with respect to increased productivity on the assembly line.

In fact, for all demand plans, MILP-1 solutions (18 of which do not satisfy the Quota property) correspond to increases in productivity, since the values of $\Delta P(1,\varepsilon)$ are all positive. Taking into account that the average daily increase is equal to $\Delta P(1) = 2.93$ engines, it turns out that the average increase in productivity is 1.09% when the current assembly line becomes a regular flow shop ($Fm/prmu/C_{\max}/d_i$).

For its part, if it is imposed in the previous flow shop that also conforms to some requests of the Heijunka concept ($\text{Heijunka} - Fm/prmu/C_{\max}/d_i$), it also turns out that all the solutions obtained with MILP-2 and MS-Q correspond to positive values $\Delta P(2,\varepsilon)$ y $\Delta P(3,\varepsilon)$ for all demand plans. Therefore, the Heijunka-flow shop also promotes an increase in productivity with respect to the current assembly line, with an average increase in the order of 1.08%, considering the value $\Delta P(2) = 2.91$ engines per day corresponding to MILP-2, or $\Delta P(3) = 2.90$ engines per day corresponding to MS-Q.

Note that the solutions offered by MS-Q and MILP-2 (see Tables 6 and 8) are equivalent from a technical-productive point of view, since on average the difference between their respective times required to manufacture a total of 270 engines is equal to 0.7 s (i.e., 50,262.0–50,261.3) using sequences that satisfy the Quota property, this value is negligible compared to the current available time $\left(C^0_{\max} = 50770\ \text{s}\right)$ to manufacture 270 engines, which corresponds to a working day with just over 14 operating hours equally distributed between two work shifts.

5.4 Advantages and disadvantages of using MILP and MS-Q procedures

The solutions offered by MILP-2 and MS-Q, for the set of Nissan-9Eng.I instances, can be considered technically equivalent in terms of the value of C_{max} (see "Appendix I"); therefore, we can conclude that both procedures are equally valid to solve the problem $Hejunka - Fm/prmu/C_{\max}/d_i$. However, since these procedures are of a different nature, it is necessary to highlight some advantages and disadvantages in relation to the application of each of them.

- In the specialized literature, most articles on the $Fm/prmu/C_{\max}$ use heuristic and metaheuristic methods. The new problem proposed in this paper is more complex than the $Fm/prmu/C_{\max}$, so there is no reason to rule out the use of heuristics to solve the $Hejunka - Fm/prmu/C_{\max}/d_i$ problem (*Laplace's principle of insufficient reason*). The use of Mixed Integer Linear Programming (MILP) for flow shop problems is less widespread for both reference instances and realistic cases.
- MILP-2 uses the IBM CPLEX solver, which is commercial software that requires a license in its professional version. CPLEX incorporates the most efficient optimization techniques related to MILP, and its efficiency is widely recognized in the scientific field, requiring knowledge of modeling techniques. In contrast, MQ-S is simple and easy to implement.
- MILP-2 is an application based on an exact procedure (MILP), while MS-Q is an approximation algorithm.
- MS-Q is on average about 7 times faster than MILP-2 for the Nissan-9Eng.I instance set with 270 jobs and 21 machines.
- MS-Q operates on the set of feasible solutions, and the CPU time to converge to a local optimum is predictable based on neighborhoods. For its part, the MILP technique, as a branch and bound procedure, operates with non-feasible solutions (non-integer solutions) and, therefore, the CPU time used to reach a local (or global) optimum is much less predictable. In fact, in this work, the standard deviation of the CPU time spent by MILP-2 is $\sigma^{CPU}_{MILP-2} = 1190.63$, while for MS-Q it is $\sigma^{\mathrm{CPU}}_{MS-Q} = 77.24$.

5.5 Implementation of solutions in a production line

Taking into account the previous results, our proposal is to transform the current assembly line, with fixed cycle time and closed stations, into a regular flow workshop with open workstations within the framework of the Heijunka concept; this proposal is based on two evidences: one of an economic nature and the other of an organizational and management nature.

The first evidence is the possibility of saving an average of € 1162.83 per day by manufacturing 270 engines of various types or, alternatively, the possibility of producing on average 272.91 engines per day (instead of 270) while maintaining the current working hours (14.103 h).

The second evidence is in the organizational advantages for the management offered by the level production both in the plans and in the sequences of mixed models.

The level production concept is inherent in Heijunka's ideology, and we have applied it here by enforcing the conformity of the Quota property with manufacturing sequences. Among the productive and administrative advantages offered by Heijunka are the following:

(1) Reduction of the stock level of the types of engines and engine components (Parts).
(2) Adjustment of production capacity to the demand for engines.
(3) Reduction of delivery times in all phases of the production system.
(4) Reduction of the volume of information to direct the operations of the production system.
(5) Ability to react to fluctuations in demand, since the preservation of the production mix means keeping the manufacturing system at its center of gravity from the production-demand point of view.

Having seen the advantages that a Heijunka-flow shop offers compared to a mixed model assembly line from a productive point of view, it is worth asking how to implement a solution $(\pi(T) = (\pi_1, ., \pi_T))$ when the virtual barrier of setting the manufacturing rate by cycle time c is removed.

This seemingly harmless fact involves converting current workstations to open stations, leading to a release such that both the start and end of jobs on each workstation do not occur periodically according to the value of the cycle time c (v.gr. 175 s), but they occur at irregular intervals that will depend on the duration of each job and the times of completion of the jobs in the current station and in the previous one.

To implement a $\pi(T)$ solution in the workshop, it is necessary that at least the following conditions are met:

c1. The manufacturing sequence must comply with the standards established in the collective agreement between the employee and the company. Compliance with this condition is guaranteed because all processing times (see Table 5) have been calculated at normal work pace and the productive time to manufacturing 270 engines (14.103 h using two shifts) takes into account the scheduled rest and forced stop times within the law.

c2. Workshop operators must be kept informed about the rhythm and the progress of production at their workstations: every operator should know the following data at the all times: (i) the engine type that reaches your workstation; (ii) the subset of tasks that makes up the job in progress; (iii) the start instant of the job in progress; (iv); the processing time required to complete the job in progress at normal work pace; and (v) the time available to carry out the job in progress.

Condition c2 can be easily achieved using technologies of Internet of Things (IoT) within the framework of Industry 4.0, implementing an information system assisted by wireless connection between the central computer from production management and a set of customized tablets (42 tablets to cover the 21 workstations).

In this way, the set of tablets will visually and acoustically report on production progress at all times and on all workstations. Consequently, all operators will automatically receive the following personalized signals:

1. Audible and visual warning that indicates the beginning of a job.
2. Accelerated audible and visual warning when the time available to complete a job is ending.
3. Visual warning of the dynamic list of pending tasks on a job with the possibility that operator validates the concluded tasks and actualizes the list of tasks.

Updating activities are possible in our case, since a job is made up of 6 tasks on average and the processing times of the jobs are between 89 and 185 s (see Table 5), these times are sufficiently large to update the information in each workstation.

6 Conclusions

In this work, a new manufacturing sequence model is presented which incorporates some Heijunka properties from Just-in-Time into the $Fm/prmu/C_{max}/d_i$ problem. This extension (*Heijunka* – $Fm/prmu/C_{max}/d_i$) arises from our concern to adapt academic problems to problems closest to industrial reality in the automotive sector.

The dimension of the mathematical model corresponding to the problem presented depends on the number of types of jobs, the number of workstations and the total demand for products (engines) in a sequencing horizon. For example, the MILP formulation requires at least 13,770 variables (2430 of them binary) and 25,682 constraints, for 9 types of jobs, 21 workstations and 270 products to be manufactured.

Two methods have been used to solve the new problem applied to a case study based on an engine assembly line. The first of them is based on Mixed Integer Linear Programming, and the CPLEX solver has been used solving all 23 realistic instances from the Nissan-9Eng.I set. The second method, with which the same instances have been solved, is a multistart procedure in whose constructive phase initial solutions are generated satisfying the Quota property, while in the second phase the solutions are improved using five neighborhood (three exchange and two insertion) and attending to the criterion of minimum total completion time (C_{max}).

Both procedures have been highly competitive with the new problem, since they have been able to optimally solve a high percentage of the instances using reasonable CPU times. Specifically, procedure MILP-2 obtains and ensures optimal solutions in 20 of the 23 instances with 270 engines using an average CPU time equal to 532.8 s for each instance with an average value of the relative gap between C_{max} and the best lower bound equal to 11.3 millionths. For its part, MS-Q has been able to obtain 13 optimum within 23 instances using an average CPU time equal to 77.2 s for each instance with an average *Gap* equal to 26.0 millionths. Therefore, it can concluded that both procedures are valid to solve the Heijunka – $Fm/prmu/C_{max}/d_i$ problem with a dimension adjusted to the automotive industry. However, although the solutions offered by MILP-2 and MS-Q can be considered equivalent in terms of the value of the objective function, it can be stated that MS-Q beats MILP computationally, being 6.902 times faster in CPU time in the experimental framework of the present case study.

Regarding the transformation of the current assembly line into a Heijunka-flow shop, the economic-productive feasibility study reveals that it is possible to save an average of € 1162.83 per day by manufacturing 270 engines or, alternatively, that it is possible to produce 3 more engines per day with the current working hours.

Finally, for future lines of work, we propose to incorporate in the presented model other productive concepts such as the activity factor of the operators and the possibility of blocking the productive flow between the workstations, as well as the incorporation of some desirable properties in the workloads of the manufacturing sequence.

Appendix I: Best sequences for Heijunka—$Fm/prmu/C_{max}/d_i$ with the set of instances Nissan-9Eng.I

Best Quota sequences from MILP-2. Headers (Plan#n, Cmax, Lower Bound, CPU Time in seconds)

PLAN#1Q 50101 50100 3600.586
5 8 1 4 2 3 6 7 9 7 8 9 5 1 3 6 4 2 7 8 1 6 3 2 4 9 5 8 7 1 2 6 9 4 5 3 1 8 9 7 2 6 3 4 5 3 2 7 1 8 4 9 5 6
9 3 7 8 1 5 2 4 6 2 1 8 7 6 3 4 9 5 2 7 5 3 1 8 9 6 4 8 2 7 1 3 5 9 4 6 8 1 7 2 9 4 3 6 5 3 1 8 7 9 2 4 5 6
3 1 7 2 5 8 6 4 9 1 2 8 9 6 7 3 5 4 9 8 3 2 7 4 5 6 1 2 3 8 7 4 1 9 5 6 3 1 7 2 9 8 5 4 6 8 5 3 9 7 4 6 2 1
2 5 9 8 3 4 1 7 6 2 3 6 8 4 5 7 9 1 1 2 7 3 5 8 9 6 4 2 7 9 3 8 5 1 6 4 8 1 3 2 9 7 5 4 6 2 9 6 4 3 7 8 5 1
4 3 6 9 7 2 8 1 5 7 3 9 1 2 6 4 8 5 7 3 5 8 4 9 1 2 6 3 9 7 6 8 4 1 2 5 3 7 4 9 1 2 6 8 5 9 1 3 4 5 7 6 2 8

PLAN#2Q 50180 50180 366.68
5 2 1 8 4 9 7 3 4 6 5 6 5 2 7 1 4 3 5 3 9 8 2 4 8 7 1 1 5 4 6 3 5 2 9 4 1 2 9 5 3 4 8 7 6 2 5 4 1 3 7 8 5 4
3 6 5 9 6 4 7 2 1 8 4 5 1 3 5 2 6 4 3 9 5 4 9 2 7 8 1 4 5 6 3 2 1 8 5 4 3 7 5 4 2 6 1 9 7 9 4 5 4 5 1 2 3 6
8 2 8 1 4 5 7 5 3 4 9 6 5 2 1 8 3 4 3 4 7 6 5 2 1 9 5 8 4 3 5 7 4 6 2 1 3 9 5 1 9 4 7 2 6 5 2 4 4 5 3 8 1 9
2 7 3 6 5 4 5 4 1 8 1 2 5 8 7 3 6 4 3 9 5 4 1 2 5 7 6 3 8 4 9 4 1 2 9 5 7 6 4 8 3 5 5 2 1 3 4 9 7 6 1 2 4 5
3 5 8 4 9 1 2 7 6 8 5 4 3 1 2 4 5 8 7 4 9 6 5 3 1 2 4 5 3 9 7 8 1 2 5 4 4 6 3 7 5 9 4 1 2 1 3 5 5 2 4 7 6 8

PLAN#3Q 50303 50303 37.851
5 2 6 8 4 7 5 4 9 7 5 8 4 5 3 6 4 9 7 8 5 6 4 4 1 5 9 4 7 8 5 1 9 5 4 6 2 8 7 5 4 5 4 6 9 8 5 7 4 4 6 5 9 3
5 8 9 1 4 7 5 4 6 8 4 7 5 2 9 4 5 6 7 8 5 6 4 5 9 4 3 5 7 8 9 4 1 6 5 4 7 8 5 4 5 2 9 4 6 8 4 3 7 5 9 5 4 6
2 8 4 5 7 4 5 6 9 7 3 8 4 5 6 9 5 4 1 8 4 7 5 6 9 5 4 8 7 6 4 5 4 9 5 1 3 8 7 4 5 4 5 6 9 8 7 2 5 4 5 6 9 4
5 8 4 7 5 6 9 1 4 8 7 5 4 2 9 6 5 4 9 8 7 5 4 4 3 6 5 5 7 8 4 6 1 9 5 4 5 4 7 8 5 2 9 4 6 8 9 3 5 4 7 6 5 4
5 7 2 8 4 9 5 4 6 9 7 8 4 5 4 3 6 5 9 8 4 5 7 1 4 5 6 3 5 8 7 4 6 4 9 5 2 4 9 8 5 7 4 5 6 4 9 1 5 5 4 7 6 8

PLAN#4Q 50170 50170 38.569
5 2 6 3 9 1 2 7 8 9 3 4 7 1 6 2 8 1 3 6 1 2 3 8 7 9 5 8 9 1 3 2 7 1 6 4 6 2 3 8 5 1 7 3 9 9 7 2 1 3 6 8 2 4
9 8 7 1 2 3 5 1 6 3 7 8 6 2 2 4 9 1 5 9 3 8 1 6 7 2 3 1 7 8 9 3 2 1 4 6 7 9 3 5 2 8 1 6 2 3 9 1 7 2 8 3 6 4
5 2 1 8 3 9 1 6 7 6 7 3 2 2 1 8 9 4 5 7 3 1 9 3 2 8 6 8 7 1 9 3 6 2 3 4 5 1 2 7 1 8 6 3 9 7 9 2 1 6 8 3 2 4
1 5 7 6 9 3 2 1 8 8 7 3 9 2 4 1 6 3 2 5 3 7 1 9 2 8 6 1 7 4 9 6 2 3 1 8 2 3 7 5 1 8 9 6 3 2 9 1 7 2 3 6 4 8
1 8 7 3 9 6 2 1 5 2 7 9 3 1 8 2 4 6 6 7 3 1 8 3 5 9 2 6 7 1 9 3 2 8 1 4 5 6 9 2 3 3 7 8 1 9 6 2 4 7 1 3 2 8

PLAN#5Q 50385 50385 45.693
5 1 2 8 4 5 3 2 4 4 1 3 5 1 5 3 4 9 7 2 3 4 5 5 1 2 4 5 4 6 3 2 4 1 5 3 5 4 2 8 1 2 5 9 4 3 1 5 3 4 5 2 1 4
1 3 4 5 2 5 3 4 6 7 4 5 1 2 5 2 7 4 5 3 1 3 4 2 1 5 4 1 2 8 5 4 3 1 4 5 5 2 4 6 3 2 5 4 1 5 4 3 1 2 9 5 4 3
5 3 1 4 5 2 4 2 1 8 3 5 4 1 2 5 3 4 7 5 4 2 5 6 3 4 1 6 2 5 3 4 4 1 5 1 3 5 9 2 4 5 1 4 3 2 8 2 5 4 3 5 4 1
1 2 3 4 5 7 7 4 5 5 1 2 4 3 9 5 3 4 5 2 1 2 4 3 4 5 1 2 5 3 1 4 5 8 6 4 1 5 3 4 2 2 5 4 3 9 1 5 3 4 1 5 4 2
1 3 5 4 5 2 3 4 2 7 5 6 4 1 3 2 4 5 5 1 3 2 4 5 4 6 1 2 4 5 3 1 4 3 5 1 2 4 9 5 3 7 4 2 5 4 1 3 5 1 2 5 4 8

PLAN#6Q 50202 50202 14.106
2 8 4 5 3 1 2 3 6 1 2 3 9 7 4 1 2 5 8 3 4 1 3 6 5 2 1 2 7 3 4 1 2 3 5 9 8 1 3 4 5 2 6 1 3 4 1 2 3 2 9 5 7 1
2 3 5 8 1 4 3 6 9 2 1 7 2 1 3 4 2 5 5 3 8 1 4 3 6 2 1 2 1 3 7 4 3 9 5 2 1 2 4 5 8 3 9 1 2 6 3 5 1 7 3 2 1 4
8 2 5 7 3 1 2 4 3 6 1 4 9 2 3 1 2 5 8 3 6 1 1 2 3 5 4 4 7 5 3 2 1 2 3 9 1 2 4 7 1 3 5 2 6 5 3 1 3 8 9 1 2 4
4 7 1 2 3 2 5 6 1 3 2 5 3 8 9 4 1 2 5 9 3 1 1 3 7 2 4 3 8 2 1 6 2 5 3 4 6 1 4 5 9 2 3 1 3 4 5 2 1 7 3 1 2 8
5 2 4 3 6 1 1 2 9 5 3 8 3 1 4 7 2 3 5 1 9 2 1 7 3 2 4 6 5 4 1 2 3 1 3 8 7 2 1 5 3 9 6 2 4 5 1 3 2 1 4 3 2 8

PLAN#7Q 50397 50397 33.399
5 2 4 9 3 7 5 4 5 4 5 8 1 1 4 4 6 5 3 8 4 5 2 5 4 9 7 5 4 2 5 6 4 4 5 1 8 5 4 5 3 2 6 4 5 4 4 3 9 5 1 5 4 7
5 8 4 4 5 2 3 4 9 6 5 1 4 5 4 3 5 7 7 2 5 4 5 8 1 4 9 4 5 5 6 4 5 2 4 1 5 8 3 4 1 3 5 4 9 4 5 6 5 2 4 5 4 7
5 4 3 1 4 9 5 8 4 5 5 2 4 1 5 4 6 7 3 8 4 5 2 5 6 4 4 5 4 3 5 2 4 5 9 7 8 4 5 5 1 2 4 3 5 4 4 9 1 5 6 5 4 7
3 5 4 4 1 8 9 5 5 4 4 5 2 1 6 4 5 7 9 4 5 8 3 5 2 4 6 4 5 4 3 5 2 5 4 7 1 8 5 4 1 4 5 4 5 2 5 4 9 6 4 5 3 7
3 8 4 5 5 1 2 4 6 4 5 5 9 4 5 2 4 7 9 6 5 4 4 5 3 4 1 3 5 8 5 4 4 5 2 7 9 6 5 4 1 3 5 4 4 5 1 2 4 5 4 5 7 8

PLAN#8Q 50128 50126 3600.312
5 6 3 9 7 2 8 8 4 1 6 7 9 1 3 8 4 5 2 9 7 6 7 8 4 5 9 6 6 1 7 8 4 9 5 2 8 9 7 6 3 5 2 8 4 1 4 6 7 9 7 8 3 5
6 1 3 9 7 8 4 5 6 9 7 2 8 9 5 2 6 4 1 7 3 8 4 6 9 5 7 1 6 2 9 6 3 8 4 5 9 8 7 7 2 8 5 4 6 9 7 1 3 8 9 5 6 4
1 2 7 4 9 6 3 8 5 9 7 8 6 1 7 2 4 5 6 8 9 7 5 8 3 4 6 3 6 9 7 8 4 5 9 2 7 9 1 8 4 6 7 2 5 1 6 8 7 9 5 4 8 3
6 3 1 9 7 8 5 6 4 2 7 6 9 8 3 2 5 4 7 9 6 8 4 5 7 9 1 3 6 8 4 5 6 2 7 9 4 8 7 9 1 3 5 6 6 2 9 8 7 8 4 1 5 9
6 3 9 4 7 6 7 2 5 8 9 8 4 1 6 3 7 5 7 8 9 6 2 4 9 1 5 2 7 3 5 6 8 4 9 6 4 7 8 9 1 6 3 7 5 8 9 1 2 5 7 4 6 8

PLAN#9Q 50378 50378 16.981
2 8 3 1 2 3 1 3 4 1 2 3 5 1 2 2 9 1 4 3 3 1 7 2 1 2 3 6 3 2 1 2 3 1 1 5 3 2 1 7 3 5 2 3 1 2 1 2 9 3 1 2 3 4
1 2 3 1 3 8 5 2 6 3 1 2 1 3 7 2 1 4 3 2 1 3 2 1 5 3 2 1 3 2 2 1 9 4 3 1 5 2 3 1 3 2 3 1 6 2 3 1 7 2 1 2 3 4
5 1 2 3 3 1 2 8 9 1 2 3 3 6 1 2 1 4 3 2 3 5 1 2 3 2 1 7 3 1 2 6 3 2 1 4 3 1 2 8 3 2 1 5 3 2 1 1 3 2 1 3 2 4
5 2 3 1 1 4 3 2 1 2 3 2 1 3 9 1 6 2 3 4 5 2 3 1 3 1 2 1 3 8 2 1 3 2 3 2 1 7 1 5 3 6 2 9 1 3 2 1 3 2 1 3 2 4
1 3 4 2 1 5 3 2 3 1 2 7 1 2 3 8 3 1 4 2 1 2 6 3 3 1 2 3 1 2 1 2 3 5 1 3 2 5 1 4 3 2 1 9 3 2 1 2 7 3 1 3 2 8

PLAN#10Q 50625 50625 9.044
5 4 5 8 4 4 5 1 5 4 5 6 4 5 9 4 4 5 5 4 4 3 2 5 4 5 7 5 4 4 5 5 4 7 4 5 5 4 5 8 4 5 3 4 5 4 2 5 4 4 5 5 4 1
5 4 4 5 6 5 4 4 9 2 5 5 4 7 4 5 4 5 3 5 4 5 4 5 4 8 1 5 4 5 4 4 5 4 3 5 5 1 4 5 4 5 2 4 4 6 5 4 5 5 9 4 4 5
3 4 5 6 4 5 4 5 2 5 4 5 4 9 5 4 5 4 4 5 4 5 1 5 4 4 7 5 4 5 4 6 5 4 2 5 5 4 8 5 4 4 5 4 5 3 4 5 4 5 1 5 9 4
5 3 4 5 4 7 4 5 5 6 4 5 4 2 5 4 5 4 5 4 5 1 4 8 4 5 4 3 5 5 4 4 2 5 4 5 5 4 5 4 7 4 5 4 6 5 4 5 4 5 1 5 4 9
5 4 3 4 5 4 2 5 5 4 5 8 4 4 5 7 5 4 5 4 5 6 4 5 4 1 5 3 4 4 5 2 5 4 5 4 4 9 5 5 4 5 4 7 4 5 1 5 4 5 4 5 4 8

PLAN#11Q 50084 50084 162.42
5 6 3 8 9 7 7 8 1 9 6 8 9 6 7 8 9 4 6 7 8 6 7 2 9 9 7 8 5 6 7 2 3 8 6 4 9 1 9 6 7 8 9 8 7 6 7 8 9 6 7 8 5 4
6 2 9 7 3 9 6 8 7 8 9 5 6 8 9 4 6 7 9 7 6 8 7 9 8 6 1 6 7 8 2 5 9 4 7 8 9 6 7 6 3 8 5 9 7 8 9 6 4 8 9 7 6 1
7 3 8 6 9 7 2 8 9 6 7 8 5 6 6 9 4 8 9 7 6 8 7 4 1 9 7 6 2 6 9 5 8 8 7 9 7 8 6 4 9 6 7 3 9 6 7 8 9 6 8 1 8 5
7 2 4 9 6 7 7 3 9 8 5 6 8 6 9 7 9 8 7 9 8 6 7 4 1 6 8 9 6 7 8 5 6 2 9 7 9 6 7 3 8 6 5 9 7 8 9 7 8 6 9 8 4 1
7 3 6 7 8 9 6 8 9 6 7 2 4 6 9 5 8 9 7 8 4 7 6 1 8 9 6 3 7 9 5 7 6 8 9 6 7 8 4 9 7 6 8 9 1 8 7 6 9 7 5 6 2 8

PLAN#12Q 50196 50196 102.445
5 9 1 8 6 4 7 2 5 3 9 4 6 1 3 8 4 5 2 7 9 8 4 5 7 2 5 4 6 3 1 8 4 7 5 2 1 6 5 9 7 4 9 8 3 1 5 4 6 2 5 9 8 4
7 6 1 3 5 4 8 2 7 5 9 4 5 6 7 4 3 8 1 6 9 3 5 4 2 4 8 7 5 9 3 1 6 2 4 5 7 8 5 6 3 4 9 1 2 5 8 4 7 1 5 6 3 4
9 7 9 8 5 4 4 6 7 2 5 2 5 9 1 8 6 4 7 3 5 1 8 4 3 4 6 2 5 9 7 4 5 2 1 9 6 8 4 3 5 1 8 5 7 6 4 2 7 9 5 6 9 4
1 7 3 8 5 4 1 2 6 8 4 5 3 7 9 2 5 4 1 9 6 8 5 4 3 5 4 2 6 3 7 9 7 4 8 5 1 3 5 8 6 4 7 9 6 5 2 4 1 8 5 3 4 6
7 2 9 5 8 4 5 4 1 2 9 3 7 6 4 9 8 5 1 3 2 5 4 6 9 7 7 8 4 5 1 3 5 6 4 8 4 7 9 1 2 5 3 6 9 6 4 5 1 2 5 7 4 8

PLAN#13Q 50136 50126 3600.349
5 2 4 3 8 7 1 9 6 1 2 3 5 8 7 4 1 2 9 6 5 3 3 4 9 1 8 2 4 5 4 1 3 6 2 7 7 9 5 8 5 1 2 6 3 4 3 2 5 1 8 4 1 2
9 5 3 6 7 4 1 3 9 8 5 2 6 4 7 1 2 3 1 4 7 8 5 2 9 6 3 5 2 6 4 1 3 8 5 2 1 7 4 9 7 3 5 2 3 8 4 1 2 9 5 6 1 4
3 2 6 8 4 5 9 1 7 2 3 4 3 5 1 8 9 5 7 6 4 1 2 6 3 1 2 8 5 4 9 7 6 3 1 2 4 5 3 8 2 7 5 4 9 1 3 2 5 1 4 8 3 6
7 9 2 1 3 4 1 5 2 8 7 9 3 6 4 5 5 6 3 1 2 4 2 5 4 7 8 1 3 9 6 1 2 3 8 7 5 1 4 6 3 4 2 9 5 2 4 1 3 7 9 6 2 5
4 1 2 8 5 9 8 7 3 6 1 2 4 5 1 7 3 3 9 8 6 5 2 4 1 3 7 2 4 9 5 1 8 3 6 5 1 2 4 7 4 3 9 1 5 6 2 4 1 2 5 7 3 8

PLAN#14Q 50223 50223 134.718
5 2 3 8 4 9 5 1 6 2 1 4 7 5 3 8 4 2 1 4 3 5 2 3 9 5 1 4 6 7 2 8 3 4 5 6 1 5 1 2 9 4 3 2 7 4 5 8 7 3 1 9 5 4
5 2 3 6 1 4 1 2 5 8 4 1 3 6 3 5 2 4 9 2 7 4 5 3 1 8 6 9 5 4 3 5 1 2 4 7 1 2 4 3 5 8 1 3 2 6 5 4 9 7 5 4 3 6
1 2 2 8 4 5 1 5 3 1 4 7 5 2 3 6 4 9 5 8 3 2 1 4 7 9 1 4 5 2 2 6 3 4 5 8 9 1 5 4 3 7 6 2 4 1 5 2 1 3 3 8 4 5
5 2 7 4 1 2 6 5 9 4 3 1 4 8 5 9 3 7 2 1 3 6 5 4 2 3 4 1 5 8 9 1 7 5 4 2 3 6 1 2 5 4 3 7 8 5 1 4 2 9 3 5 2 4
1 6 2 5 4 7 3 5 4 3 1 2 4 6 9 5 3 1 2 8 7 4 9 5 1 2 4 3 6 5 4 3 1 8 5 2 4 7 9 1 3 5 6 2 4 1 3 5 1 2 5 4 7 8

PLAN#15Q 50242 50242 105.037
5 6 1 8 4 5 2 3 9 4 7 6 2 5 4 4 5 7 1 4 8 9 5 3 5 8 9 1 4 6 7 4 2 5 5 3 9 4 3 2 6 4 1 7 5 4 8 1 5 8 3 5 2 4
6 4 7 5 5 7 4 9 8 9 5 2 6 1 4 5 3 4 1 5 2 4 6 7 9 8 3 1 5 4 7 4 6 8 5 3 5 2 4 1 7 3 9 4 5 6 4 5 8 4 2 5 9 1
4 6 3 2 5 1 7 8 4 5 2 4 3 9 5 8 5 6 4 5 4 7 9 1 2 3 4 8 7 5 6 4 9 5 2 1 4 5 3 7 6 4 8 5 2 1 5 7 3 4 9 8 5 4
6 5 4 7 1 3 4 9 5 4 6 2 9 5 1 8 2 7 5 4 3 4 5 1 6 8 3 5 4 6 7 5 4 2 9 9 5 4 3 6 1 2 4 7 5 8 5 1 4 2 4 9 5 3
5 4 8 1 4 5 7 6 3 1 2 9 4 5 6 8 7 5 2 4 4 9 3 5 5 8 4 1 7 4 6 3 9 5 1 2 5 4 7 8 4 9 6 3 5 4 1 5 2 5 4 7 6 8

PLAN#16Q 50123 50123 160.264
5 6 2 8 4 7 9 1 3 3 8 1 7 5 4 9 5 2 6 3 1 8 2 4 5 9 7 6 3 4 2 1 9 8 7 6 5 4 5 7 8 9 1 2 3 4 3 6 1 4 2 8 9 5
6 4 7 3 5 2 9 7 1 2 8 5 6 4 7 3 1 8 3 6 9 4 5 2 7 4 1 6 3 8 5 4 7 2 9 1 5 8 3 1 2 6 9 7 5 3 4 1 2 8 9 6 5 4
2 3 7 8 4 6 5 4 1 9 2 7 5 8 6 4 1 3 3 7 9 2 5 6 1 8 4 3 5 7 9 6 8 1 4 2 5 3 2 7 5 9 6 4 1 3 4 9 7 8 5 2 1 8
4 6 7 3 5 6 2 9 1 2 9 6 7 3 4 5 8 1 2 3 9 8 5 4 5 7 1 2 6 8 4 4 9 3 1 7 3 2 5 4 6 9 8 1 7 3 5 2 4 6 5 8 9 1
2 3 6 4 7 5 8 9 1 3 4 5 6 2 9 7 8 1 3 4 5 7 2 4 9 6 1 2 3 7 5 8 4 6 9 1 7 3 5 6 2 4 1 8 9 5 3 1 7 5 2 4 6 8

PLAN#17Q 50273 50273 74.047
5 2 3 8 4 7 5 1 4 6 9 4 2 5 3 8 4 1 5 3 5 9 2 6 4 1 5 4 3 7 5 8 4 9 2 1 1 3 5 4 5 6 4 2 7 3 5 8 4 4 5 2 1 9
3 2 5 4 6 7 5 4 1 2 8 9 3 5 4 5 1 6 4 4 7 6 1 2 5 8 3 9 4 5 5 3 1 2 4 7 2 1 5 4 3 8 4 5 4 6 5 9 7 1 5 3 4 2
3 6 5 8 4 9 2 1 4 5 1 7 6 4 5 2 4 3 5 8 9 2 1 5 4 7 3 4 5 3 5 4 2 1 9 8 6 4 5 2 4 7 3 5 1 6 1 2 5 4 5 8 3 4
7 2 5 4 5 3 1 7 4 6 5 3 9 8 4 2 1 5 4 5 1 4 3 6 2 4 9 2 8 5 9 5 7 3 4 1 5 6 3 4 5 4 1 2 7 9 3 5 6 4 5 4 1 2
3 4 1 2 5 4 8 8 7 5 9 4 3 5 2 6 4 1 3 5 5 4 7 2 9 6 1 2 5 4 3 8 4 5 1 4 3 5 7 9 6 5 2 4 1 3 1 5 2 4 5 4 7 8

PLAN#18Q 50273 50273 15.122
2 3 8 1 2 1 3 4 5 6 2 3 1 2 3 1 4 5 2 5 3 1 3 4 1 2 7 2 3 4 1 2 3 9 1 5 2 3 8 1 3 1 5 2 4 2 7 4 3 1 1 2 3 5
3 5 6 2 1 2 4 3 1 3 1 2 9 3 2 1 5 4 3 2 4 1 2 1 8 3 5 5 3 4 2 1 3 7 2 1 6 2 3 1 1 4 5 2 3 5 3 1 2 2 1 6 3 4
3 1 5 4 2 9 3 1 2 5 7 3 2 1 3 2 1 4 5 2 3 1 1 2 3 9 4 3 1 5 2 2 4 1 3 7 2 5 3 1 3 6 1 2 4 8 5 2 3 1 1 3 2 4
5 2 1 3 3 4 2 1 8 7 5 1 3 2 4 1 3 2 1 5 6 2 3 1 3 2 4 9 1 2 3 1 5 3 2 4 2 3 5 1 1 4 3 2 6 7 5 1 3 2 4 1 3 2
5 1 3 2 3 9 1 2 4 5 1 2 3 1 3 8 2 4 3 5 1 2 1 6 3 2 4 3 1 4 9 2 3 5 1 2 1 5 2 3 7 1 3 2 4 5 1 3 2 1 3 4 2 8

PLAN#19Q 50481 50481 7.811
5 2 4 8 5 4 3 5 4 9 5 4 4 5 7 6 5 4 5 8 4 1 4 5 4 5 9 3 5 4 5 6 4 4 7 5 4 5 7 8 4 5 2 5 4 4 5 6 4 1 5 5 9 4
4 8 5 5 4 9 5 4 3 2 4 5 5 6 4 5 4 7 8 5 4 1 4 5 9 5 4 4 5 6 5 4 7 5 4 1 5 8 4 4 5 7 5 4 6 2 5 4 9 5 4 4 3 5
1 5 4 8 5 4 9 4 5 4 5 6 5 4 3 5 4 7 5 8 4 4 2 5 5 9 4 6 5 4 5 4 3 4 5 7 9 5 4 8 4 5 5 6 4 5 4 2 4 5 1 5 4 7
5 8 4 3 4 5 5 4 9 6 4 5 7 5 4 4 5 2 9 4 5 8 4 5 1 5 4 5 6 4 4 3 5 5 4 7 6 5 4 4 5 7 5 4 9 2 4 5 8 5 4 1 4 5
5 4 3 8 5 4 5 6 4 5 2 4 4 7 5 5 4 9 5 8 4 6 4 5 4 1 5 3 5 4 9 5 4 4 5 7 2 4 5 6 4 5 9 4 5 5 7 4 1 5 4 5 4 8

PLAN#20Q 50100 50100 65.203
5 2 6 8 9 7 7 8 9 6 3 1 7 3 8 9 6 4 7 9 5 6 2 8 7 1 8 6 2 9 7 6 9 8 4 1 7 8 9 6 3 1 6 2 3 9 8 7 7 5 8 6 9 4
9 6 7 2 8 1 7 3 8 6 9 5 4 9 7 8 1 6 2 8 7 3 6 9 6 8 4 5 9 7 7 8 9 2 6 3 7 5 9 8 1 6 6 8 4 2 7 9 7 3 9 1 8 6
3 7 8 1 9 6 7 8 2 9 4 6 9 3 5 7 8 6 7 8 5 9 2 6 4 1 7 8 9 6 9 8 1 6 7 2 7 8 9 3 2 6 7 8 1 5 9 6 3 4 7 8 9 6
4 9 5 7 8 6 1 3 8 7 9 6 8 2 1 7 6 9 4 7 3 6 9 8 7 2 9 6 8 2 1 8 7 5 6 9 4 8 9 3 7 6 5 7 1 8 9 6 7 8 9 6 3 2
4 5 7 8 9 6 7 8 1 3 9 6 9 2 7 8 1 6 7 8 5 4 9 6 7 2 3 8 9 6 7 8 1 6 9 3 4 8 7 2 9 6 5 1 7 8 9 6 9 6 7 3 2 8

PLAN#21Q 50307 50307 10.545
2 3 8 6 1 2 3 4 1 7 2 3 9 1 3 5 2 1 5 2 3 1 8 3 7 2 1 6 2 3 9 1 3 1 2 4 5 8 1 2 3 2 9 3 1 3 1 2 6 1 7 2 3 4
5 2 3 1 1 3 7 8 2 9 3 1 6 2 3 1 2 4 5 2 1 7 3 1 8 2 3 1 3 2 9 1 2 6 3 4 5 7 3 1 2 8 1 3 2 3 1 6 9 2 1 3 2 4
5 7 1 2 3 6 1 3 2 1 2 3 9 1 3 8 2 4 5 7 3 1 2 4 3 1 2 9 3 1 6 2 3 2 1 8 5 7 1 2 3 1 9 3 2 6 3 2 1 8 1 3 2 4
7 5 3 2 1 6 3 1 2 2 3 1 9 1 3 8 2 4 5 6 2 3 1 1 3 2 7 1 2 3 9 1 2 8 3 4 1 5 3 2 1 7 3 6 2 9 1 2 3 2 3 1 8 4
4 1 7 3 2 6 1 2 3 1 8 3 9 2 1 3 2 5 5 2 3 1 9 7 3 1 2 4 1 6 3 2 3 8 1 2 1 9 6 3 2 1 4 2 3 1 5 3 7 2 1 3 2 8

PLAN#22Q 50545 50545 9.294
5 2 4 8 5 4 5 7 4 4 5 3 4 1 5 5 3 4 5 2 4 4 9 5 5 4 1 2 5 4 5 3 4 6 4 5 4 5 1 8 5 4 5 9 4 4 5 3 5 2 4 5 4 1
5 2 4 4 5 3 5 4 6 7 5 4 1 5 4 4 5 2 5 4 7 5 4 1 4 5 3 5 4 8 5 4 3 1 5 4 5 2 4 4 1 5 5 4 9 5 4 6 4 2 5 4 3 5
6 5 4 5 3 4 4 8 5 5 4 2 4 1 5 2 5 4 5 3 4 1 4 5 4 5 7 5 3 4 5 6 4 4 5 2 5 4 1 5 9 4 5 4 3 4 2 5 4 1 5 5 4 9
5 4 3 4 7 5 4 6 5 2 4 5 1 5 4 3 5 4 4 2 5 5 1 4 8 5 4 4 5 2 3 4 5 5 4 9 1 5 4 7 4 5 6 4 5 2 5 4 3 4 5 5 1 4
4 3 5 4 5 2 8 5 4 4 1 5 4 3 5 2 4 5 5 7 4 4 5 6 5 1 4 3 5 4 5 2 4 5 9 4 4 5 1 5 3 4 5 2 4 4 7 5 5 1 4 5 4 8

PLAN#23Q 50157 50157 43.954
5 9 2 8 7 6 7 8 4 9 6 1 8 7 6 5 4 9 7 6 9 8 3 4 8 5 7 1 6 9 7 8 9 4 5 6 3 9 7 8 5 6 7 8 4 5 9 6 9 7 8 6 2 4
5 8 1 7 9 6 7 9 4 8 2 6 7 8 4 5 9 6 7 5 6 8 4 9 7 8 3 9 5 6 8 7 9 1 4 6 2 7 9 5 6 8 7 8 4 9 3 6 7 8 9 5 4 6
5 4 7 8 9 6 2 9 7 5 6 8 8 7 3 9 4 6 7 8 5 9 6 4 8 1 7 9 3 6 7 8 9 5 4 6 8 7 6 1 4 9 5 8 6 7 4 9 9 8 7 2 5 6
6 8 7 5 3 9 9 6 4 8 7 4 7 8 9 2 6 5 7 4 6 8 5 9 1 8 9 7 6 3 8 7 9 5 4 6 2 8 9 7 6 4 5 7 6 8 4 9 9 8 5 7 6 1
7 3 6 8 4 9 7 2 5 9 8 6 7 4 6 9 5 8 7 8 4 9 6 5 9 1 7 3 8 6 4 9 7 5 8 6 6 4 9 5 8 7 9 1 8 7 4 6 9 7 5 6 2 8

Best Quota-sequences from MS-Q. Headers (Plan#n, Cmax, Lower Bound, CPU Time in seconds)

PLAN#1Q 50101 50100 176.758
5 6 2 8 9 1 3 4 7 6 5 2 9 3 4 8 1 7 8 2 3 7 6 4 1 9 5 3 2 1 4 8 7 5 6 9 4 1 8 5 7 9 2 6 3 2 7 8 4 5 9 1 3 6
7 9 5 8 6 1 4 3 2 2 9 5 3 8 6 4 1 7 5 3 6 4 8 7 1 9 2 2 9 4 8 7 3 1 6 5 1 3 9 6 7 8 5 4 2 2 9 6 3 1 7 4 5 8
8 6 1 4 2 5 7 3 9 6 3 2 9 8 4 1 7 5 9 4 3 5 1 2 6 8 7 5 3 8 1 6 4 7 2 9 8 5 6 3 2 1 9 7 4 4 1 8 7 6 2 5 3 9
8 7 6 1 3 2 9 5 4 9 5 4 2 1 3 7 6 8 9 3 5 7 4 2 6 1 8 9 3 8 1 2 4 6 7 5 9 8 3 4 5 6 1 2 7 8 4 9 1 7 2 3 6 5
3 4 9 6 5 2 7 8 1 3 4 6 1 2 9 8 7 5 9 3 4 1 8 7 6 5 2 9 5 3 7 2 1 4 6 8 9 3 1 2 4 7 5 6 8 9 1 3 5 4 7 6 2 8

PLAN#2Q 50180 50180 130.824
5 4 7 2 1 6 8 4 3 2 5 9 6 5 4 8 3 1 5 2 9 3 4 7 8 5 1 4 9 3 4 1 7 2 5 6 5 2 4 9 3 6 1 5 4 3 7 8 5 6 1 4 2 8
7 5 4 9 1 2 7 3 9 4 6 5 8 4 2 1 5 3 5 8 4 1 7 3 9 2 4 5 6 2 5 3 6 7 1 4 8 3 5 4 2 7 9 1 9 4 5 6 3 4 2 8 5 1
3 2 5 1 4 8 6 4 7 5 1 7 6 9 3 2 5 4 9 4 1 2 3 5 7 8 5 8 4 6 3 1 5 2 9 4 7 5 3 6 1 4 2 4 5 6 9 8 1 5 2 8 3 4
5 7 4 9 7 2 3 6 1 4 5 3 2 9 5 1 8 4 5 7 4 6 3 1 2 4 6 8 5 3 2 1 9 4 5 7 8 3 5 7 9 4 1 5 2 6 2 4 9 3 4 8 1 5
7 3 6 5 4 9 1 2 5 8 3 4 6 4 1 2 7 5 3 8 4 1 9 5 6 2 4 5 3 7 4 8 9 1 5 2 4 7 6 3 5 9 1 2 5 3 6 4 1 2 5 7 4 8

PLAN#3Q 50303 50303 15.496
5 4 6 8 3 4 9 7 5 8 6 1 4 5 5 4 9 7 7 5 4 9 6 8 5 4 2 4 5 8 9 4 5 3 6 7 7 5 6 4 2 5 9 8 4 6 5 9 4 5 7 1 8 4
6 8 5 4 5 4 3 7 9 8 1 4 5 9 7 4 6 5 5 6 4 9 5 2 7 4 8 1 8 4 5 7 5 4 9 6 9 7 8 4 5 6 4 5 2 4 8 7 5 4 3 9 5 6
7 4 8 5 2 4 6 9 5 3 8 6 5 4 4 7 5 9 6 4 5 7 5 1 4 8 9 2 5 6 4 4 5 8 9 7 7 4 5 6 3 9 4 8 5 4 8 9 5 4 6 1 5 7
7 3 6 5 4 9 8 4 5 2 6 5 4 8 7 9 4 5 8 9 1 5 4 4 6 5 7 2 7 9 5 4 4 5 6 8 4 5 3 7 9 4 8 5 6 6 1 4 5 5 7 8 9 4
7 4 5 2 4 5 8 6 9 3 4 7 5 8 4 6 9 5 4 5 8 1 7 4 9 5 6 3 5 7 4 9 6 8 5 4 7 8 4 9 5 2 4 5 6 4 9 5 1 5 7 4 6 8

PLAN#4Q 50170 50170 212.965
5 2 6 3 9 8 1 3 7 8 9 1 2 6 7 3 4 1 2 7 3 1 5 2 6 8 9 6 3 9 1 2 4 3 7 8 9 7 1 5 2 2 6 8 1 7 3 1 3 8 9 6 2 4
6 7 2 1 9 3 8 3 1 2 6 9 5 7 1 3 8 4 2 9 3 6 1 5 8 2 7 6 8 9 7 3 2 1 4 1 6 9 3 7 2 2 3 1 8 7 5 2 1 3 8 6 4 9
6 9 2 8 1 7 3 5 1 7 9 2 3 1 4 3 8 6 9 7 2 6 2 5 8 3 1 4 1 7 9 3 6 2 3 8 1 9 7 2 6 8 2 1 3 5 7 3 8 6 1 9 4 2
7 8 1 9 3 6 2 3 1 8 6 5 4 2 7 1 9 2 6 3 1 2 7 3 9 5 8 3 1 2 9 8 7 1 4 6 6 9 5 3 2 8 2 7 3 1 6 8 7 2 1 9 4 3
6 1 4 9 7 3 2 1 8 3 9 2 6 1 5 7 8 2 7 3 3 9 8 6 4 1 2 6 9 1 8 5 2 3 7 3 9 1 2 6 1 5 7 8 4 9 2 3 1 7 6 3 2 8

PLAN#5Q 50385 50385 73.626
5 4 1 2 3 4 5 1 7 5 3 4 6 2 1 5 4 3 4 2 3 1 5 4 8 5 2 1 2 5 4 5 3 3 4 1 2 9 4 5 3 4 1 7 5 9 4 2 1 5 2 5 4 3
1 4 2 6 5 4 3 5 3 1 2 8 4 5 1 2 5 4 4 7 3 5 3 1 4 5 2 3 4 8 1 5 2 4 5 1 3 5 6 4 2 9 3 5 4 1 5 2 2 4 5 4 3 1
2 1 4 5 3 4 2 5 3 1 6 4 9 5 1 4 2 5 3 5 2 3 4 7 5 1 4 2 1 5 7 4 5 3 1 4 2 4 3 6 5 2 8 5 4 3 1 3 5 4 5 4 2 1
3 1 4 2 5 5 4 6 2 8 5 1 3 4 2 4 9 5 1 4 3 2 5 1 4 5 3 7 2 4 3 5 4 1 5 1 4 8 2 3 5 6 2 5 4 4 1 3 1 5 5 4 2 3
3 4 5 2 7 4 1 5 2 9 3 4 1 5 2 5 4 1 5 3 3 1 4 5 4 2 9 5 4 2 3 4 5 1 6 2 5 3 4 1 3 2 1 5 4 4 5 3 1 5 2 7 4 8

PLAN#6Q 50204 50202 2.855
5 2 3 8 1 2 4 1 3 5 2 3 4 6 9 7 1 3 8 6 2 1 4 1 2 3 5 2 1 4 3 7 3 9 1 5 2 9 6 8 3 5 1 2 4 2 1 3 7 2 4 3 1 5
3 7 6 4 1 2 3 5 2 1 5 8 2 4 9 3 1 2 3 1 7 2 4 8 1 5 3 4 2 9 6 3 1 5 1 3 2 8 6 3 2 4 1 5 9 3 2 1 1 4 5 3 7 2
2 6 5 1 3 7 3 2 4 1 1 2 4 8 3 9 5 2 8 1 3 2 3 1 6 4 5 7 1 5 2 3 1 9 4 3 2 9 1 6 8 3 4 2 5 7 1 2 3 3 1 4 2 5
1 2 3 8 5 1 2 4 3 4 5 1 2 3 7 6 1 9 2 3 9 1 4 3 8 5 2 4 1 5 2 3 7 1 6 2 3 9 1 7 6 3 4 2 5 8 1 2 3 3 1 4 2 5
2 3 1 5 9 2 4 3 7 1 2 5 8 6 4 3 1 1 3 2 5 4 1 9 6 3 2 1 2 4 3 5 7 1 3 8 9 2 3 5 1 4 2 7 1 2 5 4 3 1 6 3 2 8

PLAN#7Q 50397 50397 180.110
5 3 4 8 9 5 4 4 5 7 1 2 4 5 3 6 5 4 2 4 5 1 7 5 4 5 8 4 6 3 4 5 9 4 5 2 8 1 5 4 3 5 4 5 4 6 4 7 5 2 9 1 4 5
4 5 9 7 1 5 4 5 4 2 8 3 4 5 1 6 4 5 3 5 4 2 5 4 7 4 5 8 4 6 5 9 4 5 1 3 5 2 4 4 3 8 5 4 7 5 4 5 6 9 5 2 1 4
4 5 8 5 2 1 4 4 5 3 7 4 9 5 4 6 2 5 4 1 5 4 7 3 5 4 6 5 5 4 1 8 4 3 9 5 8 5 4 2 2 4 5 4 5 6 9 4 7 5 1 5 4 3
4 5 8 3 5 1 4 4 7 5 5 6 2 4 5 4 9 3 5 4 1 2 4 9 5 4 8 5 5 4 6 1 2 7 5 4 5 3 4 5 4 8 6 4 7 5 3 4 2 5 9 1 4 5
5 4 8 5 6 9 4 1 4 5 4 5 3 2 7 4 5 3 9 5 4 5 8 1 4 2 4 5 3 5 6 4 5 2 4 7 9 4 5 1 3 6 4 5 4 5 2 4 5 1 4 5 7 8

PLAN#8Q 50130 50126 233.543
5 7 6 9 1 3 8 9 4 2 6 7 8 5 7 4 6 1 2 5 9 8 7 3 9 6 4 8 5 1 2 6 3 7 4 8 9 7 6 4 8 9 7 8 5 6 4 1 3 5 9 2 7 8
3 6 9 5 2 9 7 4 6 6 8 4 9 1 7 2 5 8 9 8 5 1 7 4 6 7 3 8 9 5 6 7 6 8 9 4 3 2 4 6 1 5 2 7 3 5 8 6 9 8 1 7 4 9
1 7 6 3 4 8 9 5 7 5 9 6 2 6 4 8 8 7 3 6 7 9 2 9 4 1 5 5 1 6 8 7 4 2 8 9 3 5 8 4 6 7 1 9 7 8 6 3 2 5 9 4 8 6
7 9 6 3 2 4 5 9 7 7 1 8 5 4 6 3 1 8 4 2 9 6 7 8 6 9 5 3 9 7 8 1 5 7 6 4 8 5 2 9 7 3 4 6 1 2 9 7 6 8 8 4 9 5
7 4 5 3 6 2 1 7 8 9 9 6 7 3 4 5 8 9 6 5 2 6 4 7 8 1 8 9 3 7 4 6 9 6 8 5 4 1 2 7 3 5 7 9 6 8 1 9 7 5 6 2 4 8

PLAN#9Q 50378 50378 5.281
2 3 1 2 8 1 3 6 3 5 1 2 4 3 1 2 3 9 1 2 3 4 1 2 3 2 1 5 2 1 3 2 7 1 3 3 2 1 3 2 1 4 3 2 1 5 3 2 1 8 3 2 1 6
3 2 1 7 3 2 1 5 3 2 1 9 1 2 3 1 4 3 2 1 4 3 2 1 3 2 5 3 2 1 7 6 3 2 1 3 2 1 1 2 3 4 3 1 2 5 3 2 7 1 3 2 8 1
3 2 1 3 2 1 9 5 4 2 1 3 9 3 2 1 6 3 2 1 1 2 3 2 1 5 3 7 2 1 3 4 2 1 3 2 1 3 3 5 1 2 3 8 1 2 3 4 2 1 6 2 3 1
3 2 1 3 9 2 1 5 3 2 1 6 4 2 3 1 2 3 1 5 2 3 1 2 8 3 1 4 2 1 3 2 1 3 2 1 3 5 3 7 1 2 4 3 2 1 3 1 2 6 2 3 7 1
3 2 1 5 3 2 1 1 4 9 2 3 2 3 8 1 3 1 2 1 3 5 2 6 1 3 2 3 1 2 3 7 9 1 2 4 3 2 1 5 1 2 3 1 3 2 3 1 2 4 1 3 2 8

PLAN#10Q 50625 50625 15.883
5 4 4 6 5 4 5 4 1 5 4 5 3 4 5 7 4 5 4 5 5 4 2 5 4 5 4 9 5 4 4 8 5 4 6 5 5 4 7 4 5 4 5 3 4 5 4 5 1 2 4 5 4 5
4 5 4 9 5 4 5 4 1 5 8 4 5 4 5 4 2 5 4 5 5 4 6 5 4 5 3 4 5 4 9 5 4 4 1 5 5 4 2 5 4 5 4 3 7 5 4 5 4 4 5 4 7 5
5 4 4 5 8 4 5 4 5 2 5 4 6 1 5 4 4 5 3 4 5 5 4 9 5 4 4 5 5 4 7 5 4 5 2 4 5 4 5 8 4 4 5 4 1 5 3 4 5 6 4 5 4 5
5 4 9 5 4 4 5 7 3 4 5 4 5 1 5 4 5 4 2 4 5 4 5 6 5 4 8 5 4 5 4 5 4 4 5 3 1 4 5 4 5 4 5 4 2 5 8 4 5 9 4 5 4 5
7 4 5 5 4 6 5 4 4 5 1 4 5 4 5 4 2 5 4 5 5 4 3 9 4 5 4 5 4 5 1 4 5 4 3 5 2 5 4 6 5 4 5 4 4 5 5 4 5 7 4 5 4 8

PLAN#11Q 50086 50084 48.682
5 8 9 6 3 7 7 9 4 8 6 7 9 1 6 8 2 8 7 6 9 9 8 4 7 6 9 3 6 7 6 8 9 5 1 7 8 8 9 6 7 7 8 9 6 7 8 5 2 9 6 8 9 4
7 3 6 2 7 9 4 6 8 9 7 6 7 8 6 9 5 7 8 8 9 4 6 9 1 6 7 6 2 8 5 9 7 7 8 3 9 6 6 8 5 8 7 7 9 9 1 6 4 8 6 8 7 6
7 9 8 9 7 2 8 5 9 6 6 8 4 9 7 7 9 1 8 6 6 3 9 8 7 6 5 3 9 7 7 6 8 8 7 4 6 9 8 6 2 7 9 5 9 7 8 8 6 6 1 7 9 4
9 3 6 7 8 8 6 9 5 2 9 7 8 6 8 7 6 4 1 9 9 7 7 8 4 9 6 2 6 8 9 7 5 7 6 8 9 7 6 4 8 9 5 8 6 8 7 7 3 6 9 9 7 1
3 6 8 7 9 6 8 9 6 8 7 4 9 2 8 6 7 5 7 4 6 9 8 1 8 9 6 3 7 5 6 9 8 7 7 9 6 8 7 4 9 1 8 6 9 6 8 7 9 5 7 6 2 8

PLAN#12Q 50196 50196 175.992
5 7 2 6 8 4 3 5 4 9 6 1 3 1 9 5 4 2 8 7 7 6 4 5 5 9 8 2 1 4 5 9 6 3 7 4 8 1 2 5 6 4 4 5 9 8 3 7 5 7 8 4 2 3
1 9 6 1 4 5 7 5 2 6 4 9 5 8 1 4 3 7 6 3 8 9 5 4 7 5 2 9 4 6 2 3 5 4 8 1 6 4 8 5 2 7 9 6 1 7 4 5 1 4 5 3 9 3
8 2 5 8 4 6 5 7 7 3 9 4 5 1 2 8 6 4 5 6 1 9 9 4 8 5 2 7 4 1 3 5 4 6 8 7 3 6 9 4 1 5 8 2 4 7 5 9 7 4 6 2 9 5
4 3 1 8 5 2 4 5 3 7 6 7 8 1 9 4 2 5 6 3 9 4 5 8 1 7 5 2 8 4 6 6 9 5 7 4 5 3 3 4 1 2 6 4 8 5 7 9 4 1 7 5 2 6
8 9 5 4 3 9 4 1 3 8 5 7 5 2 6 8 4 9 3 1 6 1 4 5 2 7 4 8 9 5 5 3 4 6 7 8 9 4 1 2 5 7 3 4 6 9 1 5 2 5 6 7 4 8

PLAN#13Q 50136 50126 12.691
5 2 3 7 1 4 6 5 1 3 9 2 8 7 4 1 2 9 5 6 3 1 8 4 3 4 2 6 5 1 2 9 7 5 8 3 1 2 4 6 7 5 8 2 3 9 4 5 1 4 7 3 1 2
3 4 5 9 6 8 3 5 9 1 4 2 7 1 2 5 6 8 6 3 1 4 2 5 8 3 4 1 7 2 9 3 5 4 9 6 1 2 7 6 2 3 1 3 8 5 4 8 7 1 5 2 4 9
5 4 8 1 3 7 3 4 2 6 7 9 1 2 8 5 4 3 6 2 5 6 1 4 3 2 9 7 9 1 5 2 8 3 1 4 5 7 3 2 6 8 5 4 9 1 3 2 7 4 1 6 3 5
5 1 2 8 9 4 6 4 5 8 3 7 1 2 3 6 7 1 3 9 2 4 5 1 2 4 6 7 2 8 5 1 3 9 9 4 2 5 6 3 3 8 1 7 4 2 5 5 4 1 3 9 6 2
8 7 1 3 9 5 2 4 5 1 3 7 6 2 4 8 1 3 5 7 9 4 2 6 1 3 1 8 4 2 5 4 6 9 3 1 2 5 8 4 7 1 3 9 5 6 2 1 2 4 5 7 3 8

PLAN#14Q 50224 50223 48.688
5 6 1 3 7 4 5 2 9 1 4 8 3 4 2 9 2 5 3 6 5 1 1 4 8 5 3 2 4 7 4 9 6 7 1 5 2 3 2 4 3 5 5 1 8 1 3 4 9 4 5 2 2 3
5 6 1 4 1 7 3 5 4 8 6 2 9 5 7 3 1 4 2 2 5 6 3 4 4 8 1 8 1 5 9 5 7 2 3 4 5 4 2 3 1 1 4 6 3 9 2 5 5 2 7 6 3 4
8 1 4 9 2 5 4 7 1 2 5 3 5 4 3 7 1 6 2 8 3 1 4 5 2 5 9 8 7 4 1 4 5 1 3 6 4 2 8 3 5 2 3 7 4 1 2 5 4 3 9 8 1 5
1 2 6 6 4 5 5 4 2 1 3 7 9 4 8 2 5 3 9 4 1 3 6 5 7 1 4 2 8 5 4 5 3 3 1 2 9 1 4 6 5 2 2 7 4 8 5 3 6 1 9 2 5 4
1 3 5 7 3 4 5 1 2 8 9 4 4 5 6 3 2 1 5 3 7 2 4 9 1 2 5 3 4 6 5 3 1 4 8 7 1 2 9 3 5 4 6 5 2 4 1 3 1 2 5 7 4 8

PLAN#15Q 50242 50242 175.659
5 6 4 9 5 7 4 1 8 3 2 4 6 1 5 2 7 3 5 4 8 4 9 6 5 4 8 5 1 3 5 4 2 9 7 4 7 8 1 5 6 2 4 5 3 4 3 6 5 9 4 1 5 2
4 7 8 5 2 5 6 9 8 4 3 4 1 5 7 4 9 1 6 5 4 7 5 3 9 5 2 4 8 4 7 8 5 1 3 9 5 4 6 2 5 3 1 4 6 5 4 7 2 4 9 8 6 5
1 2 3 5 4 5 7 8 4 1 6 4 2 5 9 3 5 8 9 4 7 4 1 7 5 6 2 4 1 5 5 3 4 3 9 5 2 4 8 4 6 9 7 5 1 5 6 3 7 4 8 4 2 5
8 4 1 3 5 9 4 5 6 2 9 7 5 4 4 6 1 8 7 5 3 4 1 2 5 5 4 3 9 6 5 8 7 4 9 4 5 2 8 1 7 6 4 2 5 3 1 8 4 5 2 4 5 6
9 7 5 4 3 4 1 9 2 5 8 1 4 5 7 3 4 9 6 5 4 3 5 1 2 6 7 5 4 8 4 9 3 5 8 1 4 5 2 4 7 9 6 3 5 5 1 2 4 5 4 7 6 8

PLAN#16Q 50128 50123 129.008
5 7 9 8 3 4 6 2 1 4 1 7 8 6 5 5 2 3 9 3 2 4 8 4 5 1 9 7 6 6 1 2 7 8 5 3 4 9 6 1 2 8 3 5 4 7 9 3 7 2 9 5 1 4
2 1 6 7 4 8 5 3 9 2 4 5 8 6 7 1 3 8 9 1 5 2 6 4 4 5 3 2 3 9 8 7 6 7 4 1 8 5 6 4 5 3 2 1 9 7 3 9 8 1 5 4 6 2
2 1 3 5 4 9 7 6 6 2 8 4 5 8 3 9 1 7 1 6 7 4 5 3 8 2 4 3 7 5 6 9 1 9 5 2 6 4 2 8 3 5 1 7 9 4 6 2 4 1 8 3 5 7
2 9 8 6 7 3 4 5 1 2 8 9 3 4 5 4 1 7 5 3 2 6 9 4 5 1 7 8 3 6 7 4 1 2 9 8 5 3 9 6 4 5 2 1 8 3 6 1 7 2 9 4 5 8
5 6 3 4 7 7 2 9 1 3 6 2 5 8 4 5 4 1 8 9 6 3 7 2 4 1 6 5 3 8 4 9 7 1 2 9 5 6 3 8 7 1 2 4 9 1 5 3 5 4 7 6 2 8

PLAN#17Q 50275 50273 4.336
5 8 3 4 2 7 5 4 1 4 5 6 3 9 1 2 9 5 4 2 6 5 8 4 3 1 5 7 4 3 5 4 6 7 1 2 5 4 2 3 1 8 9 4 5 3 5 6 4 4 2 8 1 5
3 5 4 2 7 4 5 9 1 6 8 1 7 5 4 4 3 2 5 2 4 6 7 9 5 1 3 4 5 4 2 1 3 5 5 4 8 2 9 5 7 1 4 8 3 5 2 6 4 5 1 3 4 7
6 5 4 9 3 5 4 2 1 5 1 8 4 3 9 5 4 2 7 5 2 1 3 4 5 6 9 3 4 7 5 2 1 8 4 8 5 4 6 5 1 9 4 3 2 5 6 1 4 5 3 9 2 4
7 3 5 4 8 5 2 1 4 4 1 6 5 3 7 4 8 2 5 4 3 2 1 5 4 9 7 6 5 4 3 1 5 2 5 4 8 2 9 5 6 3 7 1 4 1 9 5 3 4 2 4 5 7
5 8 4 1 6 3 4 2 5 3 9 6 5 4 7 2 4 1 5 5 3 8 1 4 4 5 2 9 6 3 5 4 8 1 2 7 5 4 9 3 1 6 2 4 5 3 1 5 2 4 5 7 4 8

PLAN#18Q 50275 50273 8.297
5 2 4 3 1 6 2 3 1 3 4 8 2 1 3 2 5 1 1 2 4 9 3 3 1 2 5 7 3 4 2 1 3 1 5 2 2 4 1 3 2 3 5 9 1 5 2 3 1 4 3 2 1 8
3 7 1 4 2 1 3 5 2 3 2 5 6 1 2 1 3 4 5 2 3 4 1 2 3 6 1 2 3 4 1 5 1 3 2 7 4 1 3 8 2 3 1 2 5 1 2 4 3 1 3 2 5 9
1 4 5 2 3 3 2 8 1 1 5 4 2 3 7 3 1 2 1 4 2 5 3 6 3 1 2 1 5 4 2 3 9 3 1 2 1 4 2 5 3 7 3 1 2 1 4 2 5 3 9 3 1 2
1 4 6 2 3 3 5 1 2 1 4 2 5 3 7 3 1 2 1 4 2 5 3 6 3 1 2 8 1 2 4 3 5 3 1 2 4 2 3 1 2 3 6 1 5 1 4 2 5 3 8 3 1 2
3 7 4 1 2 3 5 1 2 1 9 4 2 3 5 3 1 2 9 3 2 4 1 2 5 3 1 3 1 7 5 2 3 2 4 1 3 2 1 4 5 3 6 2 1 1 5 4 3 2 1 3 2 8

PLAN#19Q 50481 50481 15.000
5 9 4 4 5 8 5 7 4 2 5 4 6 4 5 3 4 5 5 1 4 8 4 5 9 4 5 4 5 6 4 5 1 4 7 5 7 4 5 4 2 5 6 5 4 8 4 5 5 9 4 3 4 5
7 4 5 5 4 6 8 5 4 1 4 5 3 4 5 5 4 9 4 5 8 4 5 6 2 5 4 4 5 9 3 5 4 5 4 7 4 2 5 5 1 4 7 4 5 6 4 5 5 9 4 8 4 5
4 5 8 4 6 5 7 5 4 3 5 4 2 4 5 9 4 5 8 4 5 7 4 5 1 4 5 5 4 6 4 1 5 9 4 5 9 5 4 6 4 5 8 5 4 7 4 5 5 3 4 2 4 5
8 5 4 4 7 5 4 5 6 3 5 4 1 4 5 9 4 5 2 5 4 4 5 6 4 5 7 4 5 9 5 1 4 4 8 5 2 4 5 4 5 3 9 4 5 8 5 4 7 5 4 6 4 5
5 8 4 5 4 7 6 4 5 3 4 5 2 4 5 9 4 5 4 5 1 5 4 8 6 5 4 3 4 5 9 4 5 7 5 4 4 5 9 6 4 5 1 4 5 7 4 5 5 2 4 5 4 8

PLAN#20Q 50100 50100 48.132
5 2 6 8 9 7 7 3 9 8 6 4 9 1 6 2 8 7 1 8 6 3 9 7 9 6 8 7 3 5 9 7 6 8 2 4 9 1 6 7 8 2 7 9 3 6 8 1 9 4 7 5 8 6
2 6 9 7 8 3 4 9 1 8 7 6 7 5 8 9 3 6 7 9 6 2 8 5 1 7 8 6 3 9 8 9 6 2 7 4 8 9 7 1 6 4 8 3 9 1 6 7 9 7 5 2 6 8
7 8 4 9 6 2 6 1 8 7 9 3 5 7 8 6 2 9 1 3 6 9 7 8 4 9 6 5 8 7 7 2 3 8 9 6 7 6 8 9 1 3 7 1 9 4 8 6 6 7 9 8 2 5
6 7 4 5 9 8 1 2 7 6 9 8 6 3 2 7 9 8 1 7 8 6 3 9 5 6 8 1 7 9 3 7 6 8 9 4 9 5 8 7 2 6 8 7 4 1 9 6 6 3 2 9 8 7
9 5 7 3 6 8 7 2 1 6 9 8 8 1 4 7 9 6 2 9 7 8 3 6 4 8 7 9 6 1 8 7 3 6 5 9 8 9 4 7 2 6 8 9 1 5 7 6 9 6 7 3 2 8

PLAN#21Q 50307 50307 5.385
2 8 3 1 2 3 9 1 4 2 3 1 6 3 5 7 2 1 8 2 3 4 1 2 3 9 1 3 2 1 7 3 2 1 5 6 6 3 1 2 1 3 8 2 4 1 7 5 3 2 9 3 1 2
3 1 2 8 1 2 3 9 5 1 6 4 3 2 7 3 1 2 8 2 3 1 2 3 5 1 6 4 3 1 2 7 3 1 2 9 8 2 3 1 2 3 5 1 6 4 3 1 2 7 3 1 2 9
8 2 3 1 2 3 5 1 6 4 3 1 2 7 3 1 2 9 8 2 3 1 2 3 5 1 6 4 3 1 2 7 3 1 2 9 3 1 2 8 1 2 3 9 5 1 6 4 3 2 7 3 1 2
3 1 2 9 1 2 3 8 5 1 6 7 3 2 4 3 1 2 3 1 2 9 1 2 3 5 8 1 6 7 2 3 3 1 4 2 3 1 2 9 1 2 3 8 6 3 7 1 4 2 5 3 1 2
2 9 5 1 3 2 3 4 1 1 8 2 6 3 1 2 7 3 7 3 1 2 9 3 6 2 1 2 4 1 5 3 1 8 2 3 5 9 1 2 3 1 3 2 4 7 1 2 6 3 1 3 2 8

PLAN#22Q 50545 50545 31.922
5 1 4 4 5 3 4 5 9 2 4 5 8 5 4 4 5 1 5 2 4 4 3 5 4 7 5 5 3 4 6 4 5 4 5 2 1 5 4 4 5 3 1 4 5 5 4 9 4 2 5 5 4 7
5 4 3 5 6 4 5 4 2 4 1 5 8 5 4 4 2 5 5 4 3 5 4 7 5 4 1 5 4 3 9 5 4 5 4 6 4 5 2 1 5 4 8 4 5 1 4 5 5 4 3 4 5 2
4 5 1 4 5 6 5 2 4 4 7 5 5 3 4 3 5 4 1 5 4 5 4 2 5 4 9 5 4 2 4 5 8 1 4 5 5 3 4 5 3 4 5 4 2 4 5 7 5 4 1 4 5 9
3 4 5 5 6 4 8 5 4 4 1 5 5 2 4 5 7 4 5 4 2 4 1 5 4 3 5 3 5 4 2 5 4 4 1 5 5 4 6 5 4 1 4 5 3 5 4 6 4 2 5 5 4 7
9 5 4 4 1 5 5 4 8 4 3 5 2 4 5 4 5 3 4 1 5 5 6 4 4 2 5 3 4 5 1 4 5 5 4 9 4 2 5 4 5 3 1 5 4 2 5 4 7 5 4 5 4 8

PLAN#23Q 50158 50157 24.340
5 8 9 7 2 6 6 4 7 9 1 8 9 4 5 6 7 8 6 4 7 3 9 8 5 6 7 8 9 4 6 7 9 5 3 8 2 4 6 8 7 9 6 5 8 7 9 1 5 6 4 7 8 9
8 6 7 4 9 2 7 3 5 6 9 8 9 4 5 6 7 8 1 6 8 9 7 4 5 7 6 9 8 5 4 8 6 9 1 7 4 7 8 9 6 2 5 6 7 8 9 3 4 6 5 7 8 9
7 9 8 4 5 6 7 6 8 4 1 9 2 5 9 6 7 8 3 6 8 7 9 4 5 6 7 9 8 3 6 8 7 5 9 4 4 6 7 8 9 2 6 5 7 8 5 9 4 1 7 6 8 9
6 8 9 4 7 5 6 7 9 4 8 5 2 9 6 3 7 8 1 7 6 8 9 4 6 9 5 8 7 5 3 7 8 4 6 9 4 2 6 7 8 9 6 5 1 8 9 7 4 6 8 9 5 7
7 3 6 9 5 8 8 4 7 4 9 6 6 7 8 5 9 2 9 8 7 6 4 5 1 8 7 9 3 6 9 5 6 4 8 7 9 1 2 6 7 8 4 9 5 7 6 8 9 5 7 6 4 8

Acknowledgements This work has been funded by the Ministry of Economy and Competitiveness of the Government of Spain through project OPTHEUS (ref. PGC2018-095080-B-I00), including European Regional Development Funds (ERDF).

Funding Open Access funding provided thanks to the CRUE-CSIC agreement with Springer Nature.

References

1. Aggoune, R.: Minimizing the makespan for the flow shop scheduling problem with availability constraints. Eur. J. Oper. Res. **153**(3), 534–543 (2004). https://doi.org/10.1016/S0377-2217(03)00261-3
2. Bautista, J., Alfaro, R.: A GRASP algorithm for Quota sequences with minimum work overload and forced interruption of operations in a mixed product assembly line. Prog. Artif. Intell. **7**, 197–211 (2018). https://doi.org/10.1007/s13748-018-0144-x
3. Bautista, J., Alfaro, R.: Transformación de una línea de montaje de modelos mixtos en un taller de flujo regular: caso de estudio en la factoría Nissan de Barcelona. Dirección y Organización. **69**, 99–110 (2019). https://doi.org/10.37610/dyo.v0i69.562
4. Bautista, J., Cano, A., Companys, R., Ribas, I.: Solving the Fm|block|Cmax problem using bounded dynamic programming. Eng. Appl. Artif. Intell. **25**(6), 1235–1245 (2012). https://doi.org/10.1016/j.engappai.2011.09.001
5. Bautista, J., Companys, R., Corominas, A.: Heuristics and exact algorithms for solving the Monden problem. Eur. J. Oper. Res. **88**(1), 495–513 (1996). https://doi.org/10.1016/0377-2217(94)00165-0
6. Bautista-Valhondo, J., Alfaro-Pozo, R.: An expert system to minimize operational costs in mixed-model sequencing problems with activity factor. Expert Syst. Appl. **104**, 185–201 (2018). https://doi.org/10.1016/j.eswa.2018.03.031
7. Bautista-Valhondo, J., Alfaro-Pozo, R.: Mixed integer linear programming models for flow shop scheduling with a demand plan of job types. CEJOR **28**, 5–23 (2020). https://doi.org/10.1007/s10100-018-0553-8
8. Caraffa, V., Ianes, S., Bagchi, T.P., Sriskandarajah, C.: Minimizing makespan in a blocking flowshop using genetic algorithms. Int. J. Prod. Econ. **70**(2), 101–115 (2001). https://doi.org/10.1016/S0925-5273(99)00104-8
9. Choi, K., Kim, C., Kim, H.J.: Multi-Period efficiency and productivity changes in global Automobile: A VRS-VRM and SML productivity index approach. Expert Syst. Appl. **86**, 77–86 (2017). https://doi.org/10.1016/j.eswa.2017.05.022
10. Gmys, J., Mezmaz, M., Melab, N., Tuyttens, D.: A computationally efficient branch-and-bound algorithm for the permutation flow-shop scheduling problem. Eur. J. Oper. Res. **284**(3), 814–833 (2020). https://doi.org/10.1016/j.ejor.2020.01.039
11. Graham, R.L., Lawler, E.L., Lenstra, J.K., Rinnooy Kan, A.H.G.: Optimization and approximation in deterministic sequencing and scheduling: a survey. Ann. Discrete Math. **5**, 287–326 (1979). https://doi.org/10.1016/S0167-5060(08)70356-X
12. Hüttmeir, A., De Treville, S., Van Ackere, A., Monnier, L., Prenninger, J.: Trading off between heijunka and just-in-sequence. Int. J. Prod. Econ. **118**(2), 501–507 (2009). https://doi.org/10.1016/j.ijpe.2008.12.014
13. Lei, D., Guo, X.: A parallel neighborhood search for order acceptance and scheduling in flow shop environment. Int. J. Prod. Econ. **165**, 12–18 (2015). https://doi.org/10.1016/j.ijpe.2015.03.013
14. Matzka, J., Di Mascolo, M., Furmans, K.: Buffer sizing of a Heijunka Kanban system. J. Intell. Manuf. **23**, 49–60 (2012). https://doi.org/10.1007/s10845-009-0317-3
15. Miltenburg, J.: Level schedules for mixed-model assembly lines in just-in-time production systems. Manage. Sci. **35**(2), 192–207 (1989). https://doi.org/10.1287/mnsc.35.2.192
16. Miyata, H.H., Nagano, M.S.: The blocking flow shop scheduling problem: a comprehensive and conceptual review. Expert Syst. Appl. **137**, 130–156 (2019). https://doi.org/10.1016/j.eswa.2019.06.069
17. Monden, Y.: Toyota production system: an integrated approach to just-in-time, 4th edn. Productivity Press, New York (2011)
18. Ozolins, A.: Improved bounded dynamic programming algorithm for solving the blocking flow shop problem. CEJOR (2017). https://doi.org/10.1007/s10100-017-0488-5
19. Pinedo, M.L.: Scheduling (theory, algorithms, and systems). Springer International Publishing, Berlin (2016). https://doi.org/10.1007/978-3-319-26580-3
20. Reeves, C.R.: A genetic algorithm for flowshop sequencing. Comput. Oper. Res. **22**(1), 5–13 (1995). https://doi.org/10.1016/0305-0548(93)E0014-K
21. Ronconi, D.P.: A note on constructive heuristics for the flowshop problem with blocking. Int. J. Prod. Econ. **87**(1), 39–48 (2004). https://doi.org/10.1016/S0925-5273(03)00065-3
22. Taillard, E.: Some efficient heuristic methods for the flow shop sequencing problem. Eur. J. Oper. Res. **47**(1), 65–74 (1990). https://doi.org/10.1016/0377-2217(90)90090-X
23. Yagmahan, B., Yenisey, M.M.: A multi-objective ant colony system algorithm for flow shop scheduling problem. Expert Syst. Appl. **37**(2), 1361–1368 (2010). https://doi.org/10.1016/j.eswa.2009.06.105