Mixed-model sequencing problem with overload minimization considering workstations dependencies

Joaquín Bautista and Raúl Suárez

Abstract—This paper reviews the formulation of the Mixed Model Sequencing Problem with Workload minimization (MMSP-W). Two significant models already presented in the literature are described, showing that they are valid for the case of serial workstations, but do not properly solve the case of parallel workstations. After that, a new model is introduced that is valid for the case of serial workstations. An example is used to illustrate the performance of all the models, and a computational experience was done to verify the applicability of the proposed model using the solver CPLEX and a set of problem instances of small dimension adapted from the literature.

I. INTRODUCTION

Properly sequencing the products is a key point in large number of productive processes with manufacturing lines, particularly when the Just-in-Time and Douki-Seisan concepts are applied. There are several problems related with sequencing, ranking, for instance, from particular problems like the determination of the optimal sequence of robotized part feeding in a set of parallel machines [14] to the determination of an optimal sequence of mixed products in a manufacturing line, which is the topic of this work.

Assembly lines of mixed products allow the manufacturing of a number of different products, avoiding stock related problems. This type of lines can be frequently find in car manufacturing factories, and the proper sequence of products in the line is a relevant point when looking for the optimal efficiency of the line. There are different efficiency measures proposed in the literature, depending basically on the administrative policies of the company [11], [10], [1], [9] [3]. One of these criteria is the minimization of the overload, which is a measurement, in units of time, of the remaining work on a product that is not done in a particular workstation. This overload may appear when the needed time to finish the work on given product is larger than the assigned cycle time [18], which is an average of the time needed by the different products manufactured in the line; thus, if several products with high processing time are consecutively fed in the line at some point one workstation will not have time enough to finish its expected work and, either the line is stopped to allow the workers finishing the product before it leaves the workstation (called conveyor stoppage [16]), or the product exits the workstation partially manufactured. The amount of work not done on a product in a workstation is called with different names: work overload [18] [2], remaining work [4], or utility work [15].

Minimizing the overload is a NP-hard problem [18], and different approaches have been proposed to solve it, like: exact procedures based on branch and bound [6] and dynamic programming [18]. Other procedures are based on a local search [17], greedy with priority rules [5], meta-heuristics [13], or based on beam search [8]. There are also procedures that consider multi-criteria [9], [12]. A recent review of the literature related with the sequencing problem can be found in [7].

II. EXISTING MODELS

In order to solve the sequencing problem of mixed products in a production line with multiple workstations minimizing the work overload, we take as reference two models already presented in the literature, namely Model M1, introduced by Yano and Rachamadugu [18], and Model M2, introduced by Scholl, Klein and Domschke [13]. These two models are described in the following subsections.

A. Model M1

The goal of this model is the maximization of the profit of the performed work. The involved parameters are:

\( K \): number of workstations.

\( I \): number of different products.

\( d_i \): demand of product \( i \).

\( p_{i,k} \): processing time required by product \( i \) in the workstation \( k \).

\( T \): number of products to be sequentially processed (i.e. \( \sum_{i=1}^{I} d_i = T \)).

\( b_k \): profit per time of the work done in the workstation \( k \).

\( c \): cycle time (i.e. time assigned to the workstations to process any product).

\( L_k > c \): maximum time that the workstation \( k \) is allowed to work on any product.

And the involved variables are:

\( x_{i,t} \): binary variable equal to 1 when a product \( i \) is assigned to position \( t \) in the sequence, and equal 0 otherwise.

\( s_{k,t} \): start time of the work done in the workstation \( k \) on the \( t \)-th product (i.e. \( t \)-th element in the sequence).
\( v_{k,t} \): processing time really used to work in the workstation \( k \) on the \( t \)-th product.

Using these parameters and variables and calling \( V \) to the total profit of performed work, we can write the following mathematical programming problem,

**Problem M1:**

\[
\text{Max } V = \sum_{k=1}^{K} \left( b_k \sum_{t=1}^{T} v_{k,t} \right)
\]

subject to:

\[
\begin{align*}
\sum_{t=1}^{T} x_{i,t} &= d_i, \quad i = 1, ..., I \\
\sum_{i=1}^{I} x_{i,t} &= 1, \quad t = 1, ..., T \\
\end{align*}
\]

\[
\begin{align*}
v_{k,t} &\leq \sum_{i=1}^{I} p_{i,k} x_{i,t}, \quad k = 1, ..., K, \quad t = 1, ..., T \\
s_{t,k} &\geq (t-1)c, \quad k = 1, ..., K, \quad t = 1, ..., T \\
s_{t,k} &\geq s_{k,t-1} + v_{k,t-1}, \quad k = 1, ..., K, \quad t = 2, ..., T \\
s_{t,k} + v_{k,t} &\leq (t-1)c + L_k, \quad k = 1, ..., K, \quad t = 1, ..., T \\
v_{k,t} &\geq 0; s_{k,t} \geq 0; \quad k = 1, ..., K, \quad t = 1, ..., T \\
x_{i,t} &\in \{0,1\}, \quad i = 1, ..., I, \quad t = 1, ..., T
\end{align*}
\]

Taking as start time reference for the operations \( s_{k,1} = 0 \) \( \forall k \), the minimum start time for each operation is \( s_{k,1}^{min} = (t-1)c \) \( \forall k \). Constraints (2) impose the satisfaction of the demand, constraints (3) indicate that in each position of the sequence only one product unit can be assigned, constraints (4) indicate that the processing time really used to work on each product unit is constrained by its required processing time, constraints (5), (6) and (7) set the limits for the operation start times and for the processing times for each product, constraints (8) indicate that the processing times and the start times are non negatives, and, finally, constraints (9) indicate the binary condition of the assignment variables.

Note that for \( b_k = 1 \) \( \forall k \) maximizing \( V \) is equivalent to maximize the completed work, and therefore equivalent to minimize the overload.

**B. Model M2**

The goal of this model is the minimization of the total overload \( W \). Additional variables needed in this model are:

\( \hat{s}_{k,t} \): difference between the actual start time \( s_{k,t} \) and the minimum start time \( s_{k,t}^{min} \) of the \( t \)-th operation in the workstation \( k \) (i.e. \( \hat{s}_{k,t} = s_{k,t} - (t-1)c \)).

\( w_{k,t} \): overload time generated by the \( t \)-th product of the sequence in the workstation \( k \).

\( \rho_{k,t} \): processing time of the work on the \( t \)-th product in the sequence in the workstation \( k \).

Now we can write the following mathematical programming problem,

**Problem M2:**

\[
\text{Min } W = \sum_{k=1}^{K} \sum_{t=1}^{T} w_{k,t}
\]

subject to:

\[
\begin{align*}
\sum_{i=1}^{I} x_{i,t} &= d_i, \quad i = 1, ..., I \\
\sum_{i=1}^{I} x_{i,t} &= 1, \quad t = 1, ..., T \\
\rho_{k,t} &= \sum_{i=1}^{I} p_{i,k} x_{i,t}, \quad k = 1, ..., K, \quad t = 1, ..., T \\
\hat{s}_{k,t+1} - \hat{s}_{k,t} + \rho_{k,t} - w_{k,t} - c &\geq k = 1, ..., K, \quad t = 1, ..., T \\
\hat{s}_{k,t} + \rho_{k,t} - w_{k,t} &\leq L_k, \quad k = 1, ..., K, \quad t = 1, ..., T \\
\hat{s}_{k,t}, \rho_{k,t}, w_{k,t} &\geq 0; \quad k = 1, ..., K, \quad t = 1, ..., T \\
x_{i,t} &\in \{0,1\}, \quad i = 1, ..., I, \quad t = 1, ..., T
\end{align*}
\]

The objective function (10) represents the minimization of the total work overload, constraints (13) link the processing time required by each product type with the processing time required by the units in the sequence, and, finally, constraints (14), (15), (16) and (17) set the lower and upper limits of \( \hat{s}_{k,t} \) as well as their relation with required processing times and the overloads.

**III. Example**

The following example illustrates the performance of the models presented above and the potential points for improvements.

Consider that there are 6 products to be manufactured \( (T = 6) \), 3 of them of the type A, 1 of type B and 2 of type C. All the products are manufactured in 3 workstations \( m_1, m_2 \) and \( m_3 \) (i.e. \( K = 3 \)), with the processing times \( p_{i,k} \) for each one given in Table I. Note that, according to the demand of each product and the total processing time per product, the total amount of work is equivalent to 77 units of time. With this information the cycle time was set to \( c = 4 \) and the maximum working window set to \( L_k = 6, \forall k \).

The optimal solution obtained using the model M1 and considering \( b_k = 1, \forall k \) is shown in Figure 1. The sequence of products with maximum completed work is: A-C-B-A-C-A. It can be seen that the total amount of performed work is \( W = 76 \), and the total overload is \( W = 1 \) (the overload is produced in the workstation \( m_2 \) when the second product, of type C, is being manufactured).

The optimal solution obtained using the model M2 is shown in Figure 2. The sequence of products with maximum completed work is: A-A-A-C-C-B. It can be seen that the

|\begin{array}{ccc}
\text{TABLE I} \\
\text{Processing times } p_{i,k} \text{ for each product type in the workstation } m_i, i = 1, ..., 3
\end{array}|
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>( m_1 )</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>( m_2 )</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>( m_3 )</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>total time</td>
<td>14</td>
<td>11</td>
</tr>
</tbody>
</table>
total amount of performed work is $V = 72$, and the total overload is $W = 5$.

In the Model M1, the start time of each operation in each workstation is referred to an absolute time (for each station), while in the model M2, each new cycle means a reset and thus a reference $t = 0$. In order to simplify the comparison of both approaches, Figures 1 and 2 represent the results using the time definition of the model M1.

Although both models look for an equivalent objective, there is an evident difference in the overload. This is justified because M1 considers the possibility of using for the last operation of each workstation the whole window ($L_k$), while M2 gives to the last operation of each workstation just the cycle time $c$. It is easy to verify that without this constraint the model M2 also arrives to an overload $W = 1$.

IV. PROPOSED MODEL WITH BONDS BETWEEN SERIAL WORKSTATIONS

A. Motivation

The models M1 and M2 do not consider any type of bond between workstations, then those models are valid only if a set of parallel lines are considered, such that they have to process a sequence of components that are going to be assembled in a common final workstation into a single product, and the goal is finding a unique sequence for the components (i.e. the components of each final product are in the same sequence in all the parallel lines) that minimizes the whole process overload. Despite the validity of these models for this type of problem, when the production line includes only serial workstations linked by a transfer system with constant velocity and without intermediate buffers, it is necessary to include some additional constraints to these models. In this type of lines, any workstation (with exception of the first one) can start working on a product (with exception of the first one) only when the previous operation in the workstation is considered finished and a new product comes from the previous workstation (even when the current product in the workstation is not already finished). In these conditions, considering that the start time of the operations on the same product are delayed one cycle between any two contiguous workstations, the solutions obtained with the models M1 and M2 may be incoherent. This effect is illustrated in Figures 3 and 4 that show, respectively, the application of the models M1 and M2. For instance, in Figure 3 there are incoherences related with the products A1 and A2 in their transitions between workstations $m_1$ and $m_2$, and in Figure 4 there is an incoherences related with the product A2 in its transition between workstations $m_1$.
and \( m_2 \), and another one related with the product \( C_1 \) in its transition between workstations \( m_2 \) and \( m_3 \).

In order to solve the problem illustrated in the example above, we propose a new model that considers the time constraints generated by consecutive workstations. This new model, described in next subsection, is a variation of the model M1 described in Subsection II-A.

### B. Model M3

Using the same parameters and variables introduced for the model M1, we can write the following mathematical programming problem,

**Problem M3:**

\[
\text{Max } V = \sum_{k=1}^{K} \left( b_k \sum_{t=1}^{T} v_{k,t} \right) 
\]

which is equivalent to (1) but, now, subject to:

\[
\sum_{t=1}^{T} x_{i,t} = d_i \quad i = 1, ..., I 
\]

\[
\sum_{i=1}^{I} x_{i,t} = 1 
\]

\[
\begin{align}
    v_{k,t} & \leq \sum_{i=1}^{I} p_{i,k} x_{i,t} & k = 1, ..., K & t = 1, ..., T \\
    s_{k,t} & \geq (t + k - 2)c & k = 1, ..., K & t = 1, ..., T \\
    s_{k,t} & \geq s_{k,t-1} + v_{k,t-1} & k = 1, ..., K & t = 2, ..., T \\
    s_{k,t} & \geq s_{k-1,t} + v_{k-1,t} & k = 2, ..., K & t = 1, ..., T \\
    s_{k,t} + v_{k,t} & \leq (t + k - 2)c + L_k & k = 1, ..., K & t = 1, ..., T \\
    v_{k,t} & \geq 0; s_{k,t} & \geq 0 & k = 1, ..., K & t = 1, ..., T \\
    x_{i,t} & \in \{0, 1\} & i = 1, ..., I & t = 1, ..., T 
\end{align}
\]

Taking as start time reference for the operations \( s_{1,1} = 0 \), the minimum start time for each operation in the model M3 is \( s_{k,t}^{min} = (t + k - 2)c \) \( \forall k, \forall t \).

Note that in M3, the constraints (23) and (26), replace the original constraints (5) and (7), respectively. These new constraints include the proper delays in the start working times in each workstation according to their positions in the manufacturing line. On the other hand, constraints (25) force that no workstation can start the work before the product has really left the previous workstation.

Figure 5 illustrates the result of applying model M3 to the example in Section III with the same unitary profit for the work done in each the workstations (i.e. \( b_k = 1, \forall k \)). The sequence of products with maximum completed work
is: C-C-A-A-B-A, which is different to that obtained using models M1 or M2.

If a constraint to the window working time of the last operation of all the workstations has to be finished within the nominal cycle time (as it is also considered in the model M2), i.e.

\[ s_{k,T} + v_{k,T} \leq (T + k - 1)c \quad k = 1, ..., K \]  

then the optimal solution changes, as it is illustrated in Figure 6 for the considered example.

V. Computational Results

In order to validate the proposed model M3, we use 225 problem instances (similar to those presented in [2]) built from the 45 production programs and the 5 structures of processing times shown in Tables II and III. The 45 production programs has been grouped into 5 blocks with the following properties: B1) one product with significant higher demand; B2) half of the products with significant higher demand; B3) all the products with similar demand; B4) one product with significant smaller demand; and B5) all the products with different demands (regularly distributed). The 5 structures of processing times are characterized with the following properties: S1) similar processing times and close to the cycle time; S2) half of the processing times are close to the allowed window time; S3) half of the products have high processing times in the first workstations and the other half in the last workstations, S4) half of the products have high processing times and the other half have short processing times in all the workstations; and S5) each product has a high processing time in a different workstations.

The optimal solution for each of the 225 instances, according to the model M3, was found using SOLVER CPLEX V11.0 with a single processor license running in a computer MacPro with CPU Intel Xeon 3 GHz and 2 Gb RAM under Windows XP. Table IV shows the results.

The following comments regarding the CPU time can be derived from the results in Table IV: 1) the average time of the 225 instances was 32.92 s; 2) the actual range of time goes from 0.03 to 1336.69 s; 3) the block B1 has the minimal average time, 0.31 s, and the block B3 the highest one, 111.13 s, showing that the instances with large demand of one product are solved easier, while the instances with similar demand of different product are more difficult to be solved; 4) the structure S4 (half of the products have high processing times and the other half have short processing times in all the workstations) has the highest
average time, 88.39 s, while the other structures have similar and significantly smaller average times, 19.05 s.

VI. CONCLUSIONS

By means of an example, it was shown that existing models to minimize the total overload in a manufacturing line are valid for synchronized parallel lines, but when serial workstations are considered they present some incoherences and the solutions cannot be used in practice. In order to solve this problem a new model was presented as an extension of that introduced in [18]. The performance of the new model was validated through a computational experience, using a set of 225 instances and using the solver CPLEX. The computational times required to find the optimal solutions are acceptable for very small dimension problems.

REFERENCES

<table>
<thead>
<tr>
<th>Instance</th>
<th>Y^0</th>
<th>Y^1</th>
<th>R</th>
<th>CPU</th>
<th>Instance</th>
<th>Y^0</th>
<th>Y^1</th>
<th>R</th>
<th>CPU</th>
<th>Instance</th>
<th>Y^0</th>
<th>Y^1</th>
<th>R</th>
<th>CPU</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/4</td>
<td>6392</td>
<td>6352</td>
<td>30</td>
<td>0.08</td>
<td>1/2</td>
<td>6201</td>
<td>6250</td>
<td>0.27</td>
<td>16/1</td>
<td>5605</td>
<td>5675</td>
<td>32</td>
<td>16.63</td>
<td>3/1</td>
</tr>
<tr>
<td>2/4</td>
<td>6391</td>
<td>6201</td>
<td>250</td>
<td>0.27</td>
<td>16/2</td>
<td>5670</td>
<td>5755</td>
<td>117</td>
<td>17.84</td>
<td>3/2</td>
<td>6249</td>
<td>6276</td>
<td>153</td>
<td>6.42</td>
</tr>
<tr>
<td>3/4</td>
<td>6407</td>
<td>6074</td>
<td>369</td>
<td>0.11</td>
<td>20/3</td>
<td>6705</td>
<td>5599</td>
<td>196</td>
<td>5.63</td>
<td>3/3</td>
<td>6723</td>
<td>6278</td>
<td>243</td>
<td>49.77</td>
</tr>
<tr>
<td>4/4</td>
<td>6417</td>
<td>6461</td>
<td>710</td>
<td>0.03</td>
<td>16/4</td>
<td>5540</td>
<td>5500</td>
<td>240</td>
<td>21.19</td>
<td>3/4</td>
<td>6397</td>
<td>6083</td>
<td>314</td>
<td>258.34</td>
</tr>
<tr>
<td>5/4</td>
<td>6280</td>
<td>6369</td>
<td>211</td>
<td>0.31</td>
<td>16/5</td>
<td>5609</td>
<td>5544</td>
<td>65</td>
<td>4.61</td>
<td>3/5</td>
<td>6468</td>
<td>6320</td>
<td>140</td>
<td>9.94</td>
</tr>
</tbody>
</table>

V: Completed work; V0: Total work to complete; W: Work overload; CPU: computer time to obtain the optimal solution
The instances are identified by the program number and structure number separated by ‘/’.

**TABLE IV**

Optimal results of applying the model M3 to the 225 problem instances