Balancing and Cyclically Sequencing Synchronous, Asynchronous, and Hybrid Unpaced Assembly Lines

Thiago Cantos Lopes[a], Adalberto Sato Michels[a], Celso Gustavo Stall Sikora[b], Rafael Gobbi Molina[a], Leandro Magatão[a*]

[a] ***Graduate Program in Electrical and Computer Engineering (CPGEI)***

[a] *Federal University of Technology - Paraná (UTFPR), Curitiba, Brazil, 80230-901*

[b] ***Institute for Operations Research***

[b] *Hamburg Business School - Moorweidenstr. 18, Hamburg, Germany, 20148*

Abstract

Mixed-model assembly lines are product-oriented production layouts often employed for large scale manufacturing of similar products. The unpaced variant of these lines employs a conveyor to discretely move pieces between stations either synchronously or asynchronously. Workload balancing and product sequencing are common optimization problems associated with these lines. While many works detail balancing and sequencing separately, few explicitly combine these degrees of freedom. Furthermore, hybrid (i.e. partly synchronous and partly asynchronous) lines are not explicitly described by previous optimization models. This paper presents a mixed-integer linear programming model capable of representing such unpaced lines and explicitly combine balancing, sequencing and cyclical scheduling. The application of the proposed method to a new dataset demonstrates the advantages of simultaneously balancing and sequencing lines, generating more efficient solutions than previously described models for 238 out of 240 instances. These results implied, however, in greater computational costs required to combine the degrees of freedom. Furthermore, a direct performance comparison study between synchronous, asynchronous, and hybrid lines is conducted. This allows line designers and managers to explicitly evaluate economical trade-offs between these line types.

Keywords: Unpaced Assembly Line Balancing, Mixed-Model Sequencing, Cyclical Scheduling, Mixed-Integer Linear Programming

---

DOI: 10.1016/j.ijpe.2018.06.012

1. Introduction

Assembly lines are product oriented layouts commonly used in industry for large-scale production of similar product models (Scholl, 1999). These lines are associated to many optimization problems. Some have been studied for a long time such as workload balancing (Salveson, 1955), model sequencing and scheduling (Thomopoulos, 1967). Others were more recently introduced such as material supply scheduling (Sternatz, 2014).

There are various types of assembly lines, which can be classified according to multiple criteria (Battaïa & Dolgui, 2013), such as product diversity. This paper will focus on mixed-model assembly lines, in which products

*Corresponding author

Email address: magatao@utfpr.edu.br (Leandro Magatão[a])

flow through the line without set-up times required between models. These lines can be further classified in terms of line pace (Boysen et al., 2007, 2009a): in paced lines, the conveyor moves all pieces continuously, while in unpaced lines movement is discrete. This paper will focus on unpaced lines, which are usually classified as synchronous or asynchronous. Because synchronous lines move forward all pieces together, they are cheaper and simpler, yet more restrictive. On the other hand, asynchronous ones allow independent movements for pieces and, consequently, they are more flexible and expensive. Hybrid lines (partly synchronous and partly asynchronous) can be reasonable intermediaries between the two (Kouvelis & Karabati, 1999). To the best of the authors' knowledge, however, hybrid lines have not been deeply studied in the literature.

Line pace is particularly relevant when mixed-model sequencing (and therefore scheduling) is considered (Boysen et al., 2009b). Some relevant scheduling problems in mixed-model unpaced assembly lines are starvations and blockages (Boysen et al., 2008). When producing at large scale with relatively stable demands, it is common to seek a cyclical schedule for the line (Levner et al., 2010). Such schedules are often based on the Minimal Part Set (MPS) concept (Bard et al., 1992), which is the smallest set of products that will reach the target demand if produced multiple times.

General and cyclic synchronous scheduling has been studied for quite some time (Karabati et al., 1992; Karabati & Kouvelis, 1996; Kouvelis & Karabati, 1999), including variants such as: lines with stochastic processing times (Doerr et al., 2000; Chiang et al., 2012; Urban & Chiang, 2016), order release in synchronous manufacturing (Riezebos, 2010, 2011), and flowshops problems with ressources and set-up constraints (Waldherr & Knust, 2017). Asynchronous lines have also been studied for long: initially as a "formulation irregularity" (Johnson, 1983), but later as a research subject on its own with both stochastic (Nakade et al., 1997) and deterministic variants (Sawik, 2000, 2002, 2004, 2012; Öztürk et al., 2013).

When combining degrees of freedom, such as balancing, sequencing, and scheduling, authors often employ decompositions (Karabati & Sayin, 2003; Battini et al., 2009) or meta-heuristic procedures (Özcan et al., 2010). Such is the case because, while the combined problem can allow better solutions, complexity and difficulty also tend to increase substantially. Furthermore, works that do combine these degrees of freedom (e.g. Karabati & Sayin (2003)) often operate some simplifications or use indirect performance measures for the throughput (Tiacci, 2015a).

A review of recent works (Section 2) shows that, while balancing and scheduling problems are sometimes addressed simultaneously, cyclical steady-state optimization of unpaced lines is still little explored. Most works on mixed-model balancing impose a common cycle time for all product models rather than explicitly considering how balancing and model sequencing interact in terms of blockages and starvations in the steady-state. This paper extends a previous formulation (Lopes et al., 2018) to allow simultaneous balancing, sequencing, and cyclical scheduling of unpaced (synchronous, asynchronous, and hybrid) lines.

The paper is organized as follows: Section 2 presents a review of recent related works. Section 3 presents the context, concepts, and hypotheses of the studied problem. Section 4 presents the new MILP model. Section 5 presents the results of the comparison between the new formulation and previously described ones. Section 6 discusses the obtained results, drawing insights from differences of solutions and computational times found in the tested formulations, as well as its implications. Lastly, the main conclusions drawn from this paper are summarized and future research directions are provided in Section 7.

2. Related Works

There are many research topics tied to mixed-model assembly lines. This review presents recent works and compares them in regard to how they measure steady-state performance (cycle time), whether they incorporate balancing, sequencing or both, and what are the goal functions.

Alghazi & Kurz (2018) introduced mathematical models that incorporate practical ergonomic and zoning constraints. In this formulation, the goal is to minimize the cycle time, measured as the weighted average of processing times across product models. Such measure can be optimistic depending on the line pace, buffer layout and product sequence (Lopes et al., 2018). A much more common approach is to impose the cycle time as a limit for all models. For instance, Akpinar & Baykasoglu (2014); Akpinar et al. (2017) present formulations and Benders cuts for mixed-model balancing with set-up times between tasks. In sequence dependent formulations, tasks are sequenced rather than product models and the task-wise scheduling of each model must respect a given cycle time. Delice et al. (2017); Roshani & Nezami (2017) presented formulations and meta-heuristics for another class of problems in which this task-scheduling is necessary, the multi-manned mixed-model assembly lines. In these lines, workstations have multiple workers or machines in order to reduce line length and, as a consequence, it is necessary to sequence tasks for each model. Cycle time is respected by the task-scheduling of every product model and the goal is to minimize the number of workers and stations (line length). Kucukkoc & Zhang (2014, 2015, 2016) present further variants and meta-heuristics for balancing-sequencing problems with parallel assembly lines. Due to possible differences in cycle times in each line, the minimal common multiple (MCM) of the cycle times is used to define a production cycle. Task times are weighted by the MCM divisors of the assembly line they are processed at and the total weighted processing time of each operator is bounded by the MCM of cycle times. These works have goal functions tied to the minimization of the number of workstations, line length and smoothness indexes of workload distribution.

The cycle time limit for all models is often used as a restriction by authors whose formulations use multiple secondary objectives: Zhong (2017)'s balancing formulation optimizes the number of stations, horizontal and vertical balancing; Saif et al. (2014)'s balancing-sequencing one also optimizes vertical and horizontal balancing, and the total flow time of the product models (without considering blockages or starvations).

Stochastic variants, in which task times are random variables were also discussed with two main strategies being recently employed: Dong et al. (2018)'s balancing formulation presents a chance-constraint for cycle time that requires the cycle time for each model to be respected with a given probability; Tiacci (2015a,b, 2017) employs direct simulation-based performance measure for balancing and balancing with buffer allocation problems. Furthermore, he argues that some common performance measures (workload smoothing (Thomopoulos, 1970), vertical and horizontal balancing (Becker & Scholl, 2006), weighted average processing times (Alghazi & Kurz, 2018), and demanding a cycle time for all models Karabati & Sayin (2003)) are not goals in themselves, but rather supposed means to achieve a high and stable throughput.

Some authors, however, do employ direct performance measures that take product sequence into account: Sawik (2002, 2004) presents balancing-sequencing models with a makespan minimization goal. A similar approach was employed by Li et al. (2017), who present a balancing-sequencing and robot selection formulation to minimize makespan. These models, however, do not produce cyclical schedules, meaning that extending such formulations for large numbers of products tend to be difficult. Sawik (2012); Öztürk et al. (2015) present balancing-sequencing formulations with makespan minimization goals and cyclical constraints (between minimal part sets) in order to better approximate steady-state optimization.

A recent work by Lopes et al. (2018) presents a balancing formulation for asynchronous lines with a steady-state performance measure (realized cycle time) that explicitly takes into account a given product sequence. Such performance measure is shown to be accurate and lead to better results than common surrogates, such as vertical balancing, horizontal balancing, and workload smoothing. However, Lopes et al. (2018)'s formulation does not incorporate sequencing decisions and is restricted to asynchronous lines. This paper extends such formulation to incorporate cyclical sequencing and to be applicable to synchronous and hybrid lines as well.

3. Problem Statement

The efficiency of unpaced mixed-model assembly lines is heavily influenced by the planning of such lines, including the task allocation, product sequence, and also how pieces are transported between each station. In respect to the transport of workpieces, these lines can either be synchronous, asynchronous or a combination of both (hybrid lines).

A station s has a synchronous pace ($s \in S_{\text{sync}}$) when the departure of the current product model must coincide with the arrival of the next one. A station s has an asynchronous pace ($s \in S_{\text{async}}$) when the arrival of the next product model can happen after the departure of the current one.

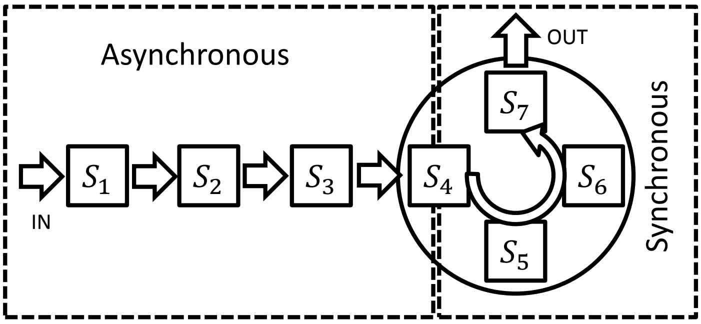

Figure 1: Hybrid unpaced line layout ($S_{\text{async}} = \{1, 2, 3, 4\}$, $S_{\text{sync}} = \{5, 6, 7\}$)

A hybrid line example is illustrated by Figure 1, in which product flow is asynchronous in stations 1 to 4 and synchronous[1] in stations 5 to 7 due to a circular rotating platform. In this case, approximating the line to an asynchronous one might lead to unfeasible schedules due to the synchronous part. Similarly, approximating the line to a purely synchronous one might eliminate good feasible schedules that take advantage of asynchronous possibilities from the first part of the line.

Synchronous transfer constraints are more restrictive than asynchronous ones and hybrid lines are expected to be intermediaries. An illustrative numerical example is hereby presented to demonstrate these differences: Consider an assembly line with four workstations and three product models with equal demand rates (MPS=$[1, 1, 1]$). The instance contains four tasks with no precedence relations between them and processing times given by Table 1.

The optimal answers for synchronous and asynchronous lines are illustrated by Gantt schedules in Figure 2 and Figure 3, respectively. In these figures, two replications of the MPS are represented to ease the cyclical behavior visualization. Lighter colors represent blockages and blank spaces are starvations; both can be considered idle times. On the cyclical schedule, red dashed lines mark the synchronous transfers of products in Figure 2.

[1]Notice that station 4 is the interface between the synchronous and the asynchronous part of the line. This means that entry times are not bound by synchronism constraints, but departure times are.

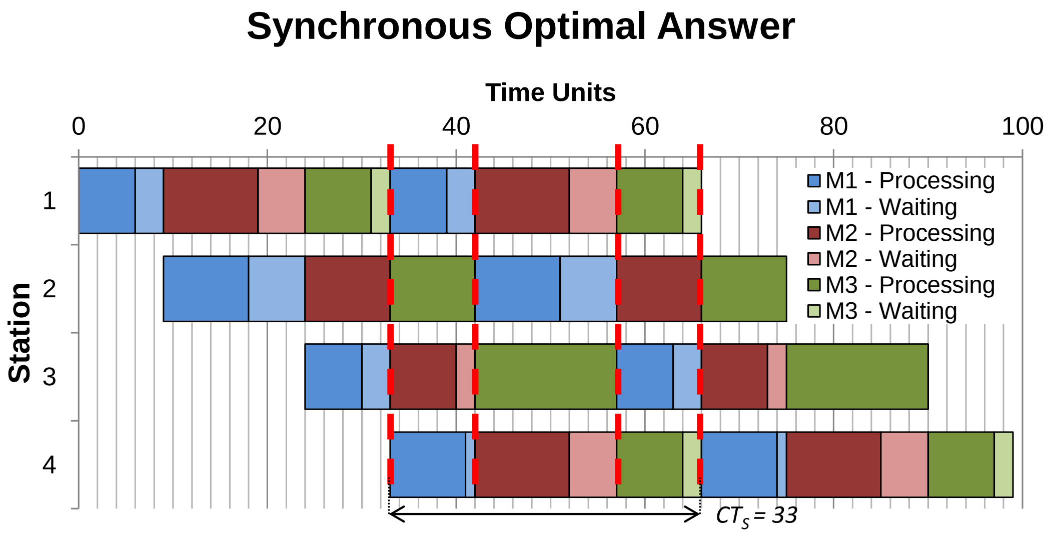

Figure 2: Optimal Cyclical Schedule for a Synchronous Line ($S_{\text{sync}} = \{1, 2, 3, 4\}$), with CT_S=33.

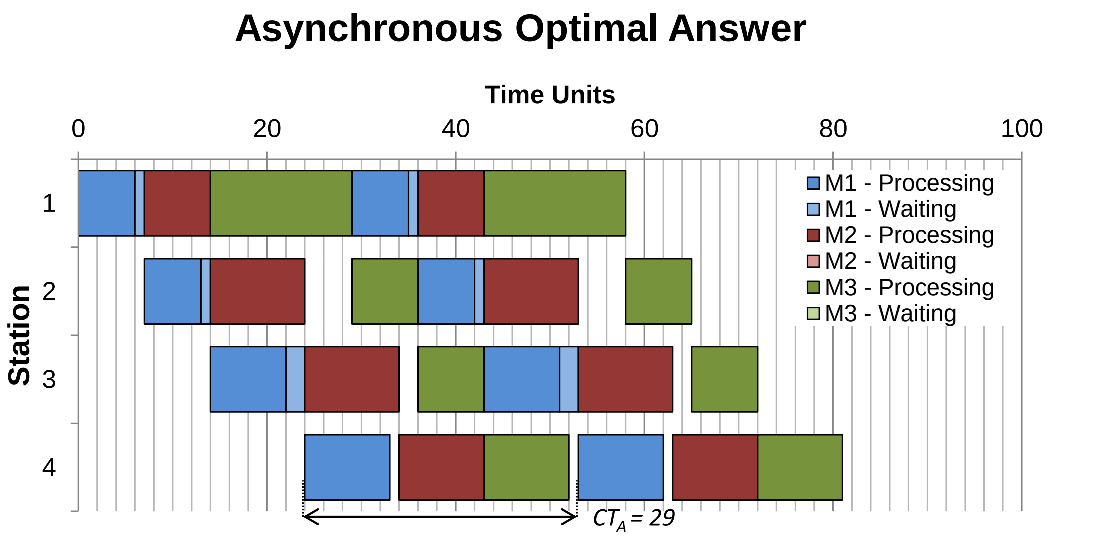

Figure 3: Optimal Cyclical Schedule for an Asynchronous ($S_{\text{async}} = \{1, 2, 3, 4\}$) Line, with CT_A=29.

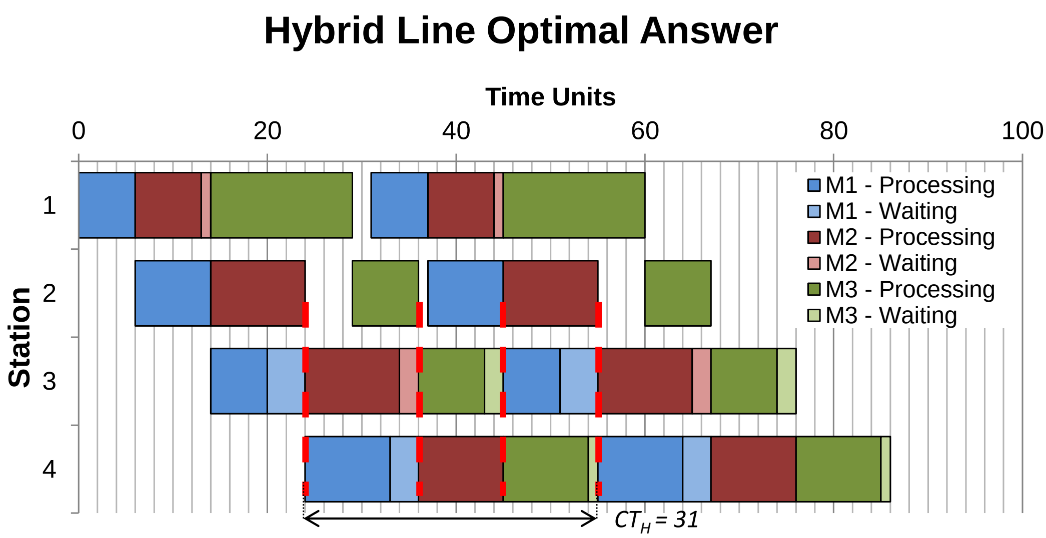

Figure 4: Optimal Cyclical Schedule for a Hybrid Line ($S_{\text{async}} = \{1, 2\}$, $S_{\text{sync}} = \{3, 4\}$), with CT_H=31.

Model	M1	M2	M3
Task 1	6	7	15
Task 2	6	10	7
Task 3	8	10	7
Task 4	9	9	9

Table 1: Numerical Example Data - Processing Times (in Time Units, TU)

If the synchronism constraints are applied to stations three and four, and the asynchronism possibilities are kept at stations one and two, a hybrid line is defined. Figure 4 presents the optimal answer for that line. Notice that synchronous transfers are forced on a subset of stations. Once again, two replications of the MPS are represented in Figure 4 and red dashed lines represent the synchronous transfers of products.

Comparing the optimal solutions of each formulation, one can see that the synchronous line has the highest value of cycle time (CT_S=33), the asynchronous one has the lowest one (CT_A=29), and the hybrid line has an intermediary value (CT_H=31). This occurs because, on hybrid lines, the synchronism constraints occur for only a subset of stations. This illustrates a class of problems that cannot be described (and therefore optimized) by previous literature mathematical models.

Furthermore, there are three degrees of freedom that ought to be simultaneously solved in order to achieve optimal solutions: balancing, cyclical sequencing and scheduling. The problem presented herein is based on the following assumptions:

1. Tasks are indivisible and must be performed in the same stations for all product models;
2. There are precedence relationships between some tasks, characterizing technological constraints;
3. The MPS is known, i.e. product demand is known;
4. Transfer times between stations are neglected;
5. Line Control is a station-wise parameter: pieces move to the next station either on a synchronous or an asynchronous pace.

Consequently, a new approach is required to optimally combine the balancing and (cyclical) sequencing degrees of freedom and maximize the line's efficiency by minimizing the average steady-state cycle time as the goal. Section 4 seeks to present a mixed-integer linear programming (MILP) formulation that allows to describe and optimize unpaced (synchronous, asynchronous, or hybrid) lines.

4. Mathematical Model

In this section, the notation is introduced and the problems presented. The goal is to maximize throughput of an assembly line with $|S|$ stations: $S = \{s_1, s_2, ..., s_{|S|}\}$. Each task t_i from the set $T = \{t_1, t_2, ..., t_{|T|}\}$ must be assigned to one station. A partial ordering is imposed on task allocation, with the set of precedence relations R with vectors (t_p, t_s). Each of such vector states that the preceding task t_p must be performed before the succeeding task t_s. The minimal part set contains $|P|$ pieces (jobs), each of them represents a different product model[2]. The processing time of each task t for each model m is given by the matrix $d_{t,m}$ and is not dependent on the station that the task is assigned to.

[2]If the MPS contains multiple instances of the same product model, these can be treated as different products with equal processing times.

The key binary decisions in these problems are: the allocation of tasks to stations, controlled by the binary variable set x (balancing); and the definition of a cyclical model sequence for the pieces, controlled by the binary variable set y (sequencing). The binaries $x_{t,s}$ are set to one when the task t is assigned at the station s. The binaries $y_{p,m}$ when the model m is the p^{th} piece in the sequence.

Once these key operational decisions are made, scheduling auxiliary variables are employed to measure solution quality. The key idea behind them is to represent steady-state by analyzing a single MPS. The variables sets Tin, Tx, and $Tout$ represent entry, processing, and departure times, respectively, of each piece at (or from) each station. In a hybrid line, scheduling rules of stations are affected by the transfer system. Some have synchronous transfers (S_{sync}), meaning a piece must enter when the previous piece departs. Other stations have asynchronous transfers (S_{async}), meaning a piece can enter after the previous piece departs. Naturally, these sets combined make into the full set of stations: $S = S_{\text{sync}} \cup S_{\text{async}}$.

The time between the entry of the first product and the departure of the last one limits the cycle time for the whole MPS. Such cycle time is represented by the performance measure variable CT[3], whose minimization is the problem's goal.

The proposed mixed-integer linear programming formulation for synchronous and asynchronous lines is hereafter presented. The Synchronous Balancing and Cyclical Sequencing Problem (SBCS-P) is defined with Expressions (1) to (11). Its Asynchronous counterpart (ABCS-P) is defined with Expressions (1) to (9) and (12) to (13).

Base Model - Common Constraints

$$\text{Minimize} \quad CT \tag{1}$$

$$\text{Subject to:} \quad \sum_{s \in S} x_{t,s} = 1 \qquad \forall\, t \in T \tag{2}$$

$$\sum_{s \in S} s \cdot x_{t_1,s} \leq \sum_{s \in S} s \cdot x_{t_2,s} \qquad \forall\, (t_1, t_2) \in R \tag{3}$$

$$\sum_{p \in P} y_{p,m} = 1 \qquad \forall\, m \in P \tag{4}$$

$$\sum_{m \in P} y_{p,m} = 1 \qquad \forall\, p \in P \tag{5}$$

$$y_{1,1} = 1 \tag{6}$$

$$Tx_{p,s} \geq \sum_{t \in T} d_{t,m} \cdot x_{t,s} - M \cdot (1 - y_{p,m}) \qquad \forall\, m,\ p \in P,\ s \in S \tag{7}$$

$$Tout_{p,s} \geq Tin_{p,s} + Tx_{p,s} \qquad \forall\, p \in P,\ s \in S \tag{8}$$

$$Tin_{p,s} = Tout_{p,s-1} \qquad \forall\, p \in P,\ s \in S :\ s > 1 \tag{9}$$

Synchronous Blockage Constraints

$$Tin_{p,s} = Tout_{p-1,s} \qquad \forall\, p \in P : p > 1,\ s \in S_{\text{sync}} \tag{10}$$

$$Tin_{1,s} + CT = Tout_{|P|,s} \qquad \forall\, s \in S_{\text{sync}} \tag{11}$$

Asynchronous Blockage Constraints

[3]Also referred to as CT_S, CT_A, and CT_H to differentiate synchronous, asynchronous, and hybrid lines, respectively.

$$Tin_{p,s} \geq Tout_{p-1,s} \qquad \forall\, p \in P : p > 1,\ s \in S_{\text{async}} \tag{12}$$

$$Tin_{1,s} + CT \geq Tout_{|P|,s} \qquad \forall\, s \in S_{\text{async}} \tag{13}$$

Constraints (2) to (5) establish that all tasks must be performed, all precedence relations must be respected, all models (jobs) must be produced and each piece must be a unique product. Constraint (6) provides a symmetry break - valid due to the cyclical nature of the product sequences. Constraint (7) determines the processing time at each piece at each station. Constraints (8) to (9) establish that pieces can only leave a station when processing is complete, and require that pieces enter the next station when they depart from the current one. Finally, the unpaced blockage feature is introduced with Equations (10) and (11) describing the synchronous flow of pieces: all pieces must move one station forward simultaneously. Furthermore, the time between the entry of the first piece and the departure of the last piece is the same at all stations: the cycle time of an MPS is directly tied to the line's productivity. Lastly, Inequalities (12) and (13) describe the asynchronous flow of pieces. The main difference to the synchronous one is that products do not need to change stations simultaneously: a piece can only enter a station after the previous one departs from it, but does not have to do it immediately. Therefore, the last two constraints for the asynchronous problem are inequalities ($\geq$) while those of synchronous lines are equalities ($=$), meaning that asynchronous lines are indeed less restrictive.

A particular strength of this formulation is its capacity to represent hybrid unpaced lines: Suppose that some stations of an unpaced line are asynchronous, but the latter ones are synchronous, as illustrated by Figure 1. By using the appropriate constraints for each set of stations[4], it is possible to represent such hybrid lines.

5. Results

In this section, the model presented in Section 4 is tested on a dataset with 140 instances. The dataset is made available in the paper's supporting information. The balancing data vectors are based on Otto et al. (2013)'s SALBP dataset. This dataset contained instances of various sizes, probabilistic distributions for the task times, and precedence diagram structures. In this paper, tests were conducted in small and medium instances (20 and 50 tasks). In Otto's dataset, task times followed one of three distributions: peak in the bottom, peak in the middle, and bimodal distributions; the latter was chosen to emulate differences between product models. Instances were generated using all precedence relation structures described by Otto et al. (2013). In order to better represent the results, instances are gathered based on their Order Strength (OS). The OS is a measure of how restricted is the task allocation due to precedence relations. The range of the OS goes from 0 to 1, where 0 represents an instance without any precedence relation, and 1 results in an instance with a fixed sequence for the task assignments.

In order to better represent differences between models, instance generation was exclusively based on the bimodal data vectors presented by Otto et al. (2013). There were grouped five by five in order to generate mixed-model instances. Each instance of Otto's dataset is considered a model in the proposed dataset. The processing times of tasks of different data vectors were converted in processing times of different models. The precedence relations are defined as the ones of the first data vector in each group of five data vectors. This leads to two sets of 35 mixed-model task property instances. The instances were grouped in accordance to their order strength (OS). Each resulting set of instances contains 15 instances with OS = 0.2, 15 instances with OS = 0.6, and 5 instances with OS = 0.9.

[4]In Figure 1's case, synchronous constraints (Eq. 10 and Eq. 11) should be applied to stations 5-7, representing synchronous departures from stations 4-7. Asynchronous constraints should be applied to the other stations.

Two sequencing problem sizes were considered, first a small minimal part set with one piece of each model (1A, 1B, 1C, 1D, 1E), named S1, and, second, a larger one with relative demands (1A, 3B, 2C, 2D, 1E), named S2. A fixed number of stations (seven) was chosen to conduct experiments with the objective of cycle time minimization.

With two problem sizes for balancing (B1 and B2), two problem sizes for sequencing (S1 and S2) and 35 instances for each combination, a total of 140 mixed-model instances are defined, grouped in four sets (S1B1, S2B1, S1B2, S2B2). The proposed dataset is tested with the two mathematical models described by Karabati & Sayin (2003) and the proposed formulation for synchronous lines. For asynchronous lines, the proposed formulation was compared with Öztürk et al. (2015) CLP formulation. These references were chosen as they were the ones that most closely matched the cyclic synchronous/asynchronous problems' definitions. However, most works reviewed on Section 2 follow (adaptations of) the performance definitions proposed by Karabati & Sayin (2003) (i.e. cycle time as either a bound on weigthed average processing times or on model-wise processing times for all stations), furthermore, the makespan minimization goal employed by Öztürk et al. (2015) is also used by other authors (Sawik, 2012; Li et al., 2017).

Hybrid unpaced lines were assumed to have a synchronous rotating platform containing four stations ($S_{\text{async}} = \{1, 2, 3, 4\}$, $S_{\text{sync}} = \{5, 6, 7\}$ - This layout is illustrated by Figure 1). Such hybrid line was optimized by the proposed formulation and its answers are compared to those of synchronous and asynchronous counterparts.

All experiments were conducted with the same hardware and software configurations: A 64-bit Intel TM i7 CPU (2.9 GHz) with 8 GB of RAM was employed using eight threads and the IBM ILOG CPLEX Optimization Studio 12.6.3. Time limit was set as 1800 seconds. The two models described by Karabati & Sayin (2003) were implemented according to the paper, Öztürk et al. (2015)'s model was adapted from the one made available by the authors at the paper's supporting information.

5.1. Overview of previous literature models

The proposed formulation is compared in the next sections to three alternative formulations. For synchronous lines, the two models described by Karabati & Sayin (2003) are employed. They are the Total Processing Time Problem (TPTP) model, whose goal function is equivalent to Expression (14), and the Maximum Sub-cycle Time (MST) problem, whose goal function is equivalent to Expression (15). Karabati & Sayin (2003) show that TPTP defines a lower bound for steady-state cycle time (CT), while the second (MST) defines an upper bound. Notice that these expressions do not require any sequencing ($y_{p,m}$) or scheduling variable (Tin, Tx, $Tout$).

$$\text{Minimize} \quad \max_{s \in S} \left(\sum_{m \in P} \sum_{t \in T} d_{t,m} \cdot x_{t,s} \right) \tag{14}$$

$$\text{Minimize} \quad |P| \cdot \max_{s \in S} \left(\max_{m \in P} \left(\sum_{t \in T} d_{t,m} \cdot x_{t,s} \right) \right) \tag{15}$$

For asynchronous lines, the Constraint Logical Programming (CLP) model described by Öztürk et al. (2015), hereafter referred to as OZT, is used for comparisons. In the employed notation, the makespan minimization function would be equivalent to Expression (16). Their formulation employs two MPS replications (the P set is doubled) as well as symmetry constraints between each of these replications, which are equivalent to Equation (17). Notice that OZT's equivalent MILP formulation requires sequencing and scheduling variables.

$$\text{Minimize} \quad Tout_{|P|,|S|} \tag{16}$$

$$y_{p,m} = y_{p-|P|/2,\ m-|P|/2} \qquad \forall\ m \in P,\ p \in P:\ \ m,\ p > |P|/2 \tag{17}$$

Both the TPTP and MST models were re-implemented according to Karabati & Sayin (2003)'s paper. It was not necessary however, to fully translate Öztürk et al. (2015)'s model to mixed-integer programming: their paper's supporting information includes the full implementation files in Constraint Logical Programming and can be solved by the CP solver of the IBM ILOG interface used to develop and test the proposed formulation. Only minor adaptations on input files (.dat) were necessary, in order to adapt the OZT model to the proposed dataset.

5.2. Synchronous Lines

The synchronous version of the proposed model (P_S) was tested and compared to two literature benchmarks designed to address this problem in particular: the Maximum Sub-cycle Time (MST) formulation, and the Total Processing Times Problem (TPTP), both presented by Karabati & Sayin (2003). MST seeks to minimize the worst sub-cycle time by minimizing the largest model-wise processing time at stations. TPTP seeks to optimize the best case scenario by minimizing the largest station-wise sum of model-wise processing times weighted by their demands. The optimal value of the MST model offers a valid upper bound, while the optimal value of the TPTP model offers a lower bound.

Neither TPTP nor MST explicitly considers sequencing. Therefore, in order to make a valid comparison, their answers were post-processed to report the value of steady-state efficiency generated by the best cyclical sequence. Such post-processing was not applied to the proposed formulation. Due to the combined degrees of freedom, P_S is expected to generate better answers than MST and TPTP. A comparison ratio is defined for each of the benchmark models by computing $1 - UB_{MST}/UB_{P_S}$[5] and $1 - UB_{TPTP}/UB_{P_S}$. These measures offer percentage differences obtained by the proposed model P_S. Negative values indicate that P_S was outperformed in the specific instance. *Nopt* indicates the number of instances solved to optimality for the proposed formulation. For example: 13/15 indicates that 13 out of the 15 instances in that set were solved to optimality. Table 2 presents the results obtained for synchronous lines. The number of instances P_S solved to optimality (*Nopt* (P_S)) in each sub-dataset (e.g. S1-B1 OS=0.2) is reported at the rightmost column of Table 2.

5.3. Asynchronous Lines

The asynchronous version of the proposed model (P_A) was tested and compared to literature benchmarks designed to address this problem in particular: the CLP formulation presented by Öztürk et al. (2015). OZT seeks to generate better cyclical schedules by minimizing the makespan of two replications of the minimal part set. This is an indirect performance measure and is not the model's goal in itself, but rather a method to achieve a low cycle time. OZT reported values of cycle time were defined according to the original implementation files provided by the authors' supporting information.

Just as the proposed formulation (P_A), OZT explicitly considers sequencing. Therefore, in order to make a valid comparison, no post-processing was applied to either model. The proposed formulation P_A incorporates a direct performance measure of cycle time and is, therefore, expected to generate better answers than OZT. A comparison ratio is defined for each of the benchmark models by computing $1 - UB_{OZT}/UB_{P_A}$. This measure offers percentage difference obtained by the proposed model P_A. Table 3 presents the results obtained for asynchronous lines. The number of instances P_A solved to optimality (*Nopt* (P_A)) in each sub-dataset (e.g. S1-B1 OS=0.2) is reported at the rightmost column of Table 3.

[5]Notation: UB_{method} denotes the steady-state cycle time obtained by specific method/model during experiments.

Data Info		Avg. Exec. Time (s)			Avg. Ratios to P_S		Min. Ratios to P_S		
Data Set	OS	MST	TPTP	$P(_S)$	MST	TPTP	MST	TPTP	Nopt (P_S)
S1-B1	0.2	2	1.2	469	5.21%	13.07%	1.78%	3.42%	15/15
	0.6	0.7	0.7	55.5	5.36%	12.58%	1%	8.32%	15/15
	0.9	0.6	0.4	1.8	2.87%	8.3%	1.18%	3.53%	5/5
S2-B1	0.2	2.2	1.1	1800	2.83%	11.53%	0.95%	8.57%	3/15
	0.6	0.7	0.6	922.5	4.11%	10.59%	0.61%	6.03%	15/15
	0.9	0.6	0.6	26.9	4.19%	8.29%	2.82%	5.31%	5/5
S1-B2	0.2	1683.3	1359	1800	4.82%	14.67%	1.25%	12.09%	3/15
	0.6	93.4	20.9	1031.9	3.24%	12.62%	1.43%	7.65%	13/15
	0.9	0.6	0.7	11.2	4.1%	6.15%	2.41%	4.36%	5/5
S2-B2	0.2	1491.6	1246.4	1800	1.84%	11.51%	-0.68%	7.61%	0/15
	0.6	63.4	18.4	1545.9	1.57%	9.61%	-0.47%	3.36%	8/15
	0.9	0.7	0.7	94	3.25%	6.89%	0.64%	4.78%	5/5
All	-	357.7	283.8	796.6	3.62%	10.48%	-0.68%	3.36%	92/140

Table 2: Results for synchronous lines

5.4. Hybrid Lines

Unpaced lines that combine synchronous and asynchronous parts are described as hybrid ones. To the best of the authors' knowledge, no previous paper presents a model that can be used as a comparative benchmark. Therefore, the proposed formulation for the hybrid lines P_H is hereby compared to the proposed formulations for synchronous and asynchronous lines (P_S and P_A).

Given that for hybrid lines (P_H), the synchronism constraints are only applied to a subset of stations, it is expected that these lines performance will be an intermediary between the synchronous lines (P_S) and the asynchronous ones (P_A). A comparison ratio is defined for each of the line control variants by computing $1 - UB_{P_S}/UB_{P_A}$ and $1 - UB_{P_H}/UB_{P_A}$, this measure offers percentage difference in cycle time between synchronous/hybrid lines with respect to asynchronous ones. Negative values indicate that P_A was outperformed by P_S or P_H in the specific instance, this is only possible for instances in which P_A was not solved to optimality. Table 4 presents the results obtained for hybrid lines. The number of instances P_H solved to optimality (*Nopt* (P_H)) in each sub-dataset (e.g. S1-B1 OS=0.2) is reported at the rightmost column of Table 4.

Data Info		Avg. Exec. Time (s)		Ratios to P_A		
Data Set	OS	OZT	P_A	Avg.	Min.	Nopt (P_A)
S1-B1	0.2	1800	58.5	17.24%	7.6%	15/15
	0.6	1800	15.9	24.66%	6.36%	15/15
	0.9	1800	2.7	21.7%	13.41%	5/5
S2-B1	0.2	1800	1267.9	24.88%	17.56%	9/15
	0.6	1800	454	29.54%	16.58%	14/15
	0.9	1800	41.1	30.34%	22.27%	5/5
S1-B2	0.2	1800	1272.5	25.9%	12.14%	6/15
	0.6	1800	475	34.36%	14.34%	15/15
	0.9	1800	25.2	53.32%	29.75%	5/5
S2-B2	0.2	1800	1800	31.57%	25.9%	0/15
	0.6	1800	1627.7	42.72%	25.95%	4/15
	0.9	1800	171.9	63.79%	43.26%	5/5
All	-	1800	601	33.34%	6.36%	98/140

Table 3: Results for asynchronous lines

6. Discussion

Karabati & Sayin (2003)'s TPTP model performed significantly worse than their MST model (Table 2). In average, to simply minimize the sum of processing times at stations does not optimize the line. In a single instance (Dataset S1-B1, case 6), however, the TPTP model did produce a better solution than the MST. This indicates that minimizing the maximum model-wise processing time does not necessarily optimize the line either. That said, the MST did perform significantly better than the TPTP, but the proposed formulation P_S outperformed the MST one by 3.62% percent in average. This difference in performance is due to taking into account sequencing and scheduling problems explicitly. Note that this comes at a cost for the proposed formulation: in average, it took significantly longer processing times in all datasets.

Öztürk et al. (2015)'s CLP model was considerably outperformed by the proposed MILP model (Table 3). This might be due to the fact that the search field is rather large and not very restricted, leading to weaker performance on the constraint propagation and domain reduction mechanisms CLP relies rather heavily on. Furthermore, the lack of a systematic lower bounds for the OZT formulation meant it could not solve a single instance to optimality. In average, the proposed formulation produced asynchronous cyclical schedules with 33% lower cycle times than the ones generated by Ozturk's model. Notice that the proposed model also produced solutions faster, reaching optimal solutions in 98 out of 140 instances, meaning that, for asynchronous lines, the proposed formulation also displayed a better time performance.

For both synchronous and asynchronous lines, the proposed model generated better solutions in average than the tested benchmarks. The proposed formulation was outperformed in only 2 out of the 140 synchronous instances (Dataset S2-B2, cases 14 and 25 in the supporting information). Notice that cases with higher ordering strength (OS) were solved faster for all models: Even the larger instances (S2-B2) with high OS (0.9) could be optimally solved in a relatively low computation time (172s in average for P_A), while smaller ones (S2-B1) with lower OS (0.6) took much longer (454s in average for P_A). The higher solution difficulty of problems with lower OS is also reflected by the number of solutions solved to optimality: for the harder dataset (S2-B2), only one instance with low OS (0.2) was solved to optimality (by P_H). Out of the instances with medium OS (0.6), 24

Data Info		Avg. Exec. Time (s)			Avg. Ratios to P_A		Min. Ratios to P_A		
Data Set	OS	P_S	P_H	P_A	P_S/P_A	P_H/P_A	P_S/P_A	P_H/P_A	Nopt(P_H)
S1-B1	0.2	106.3	70.3	58.5	3.71%	0.73%	1.4%	0.18%	15/15
	0.6	21.9	17.7	15.9	4.44%	1.05%	2.81%	0%	15/15
	0.9	3.6	2.7	2.7	6.76%	1.75%	3.97%	0.16%	5/5
S2-B1	0.2	1579.3	1293.9	1267.9	3.86%	0.85%	2.6%	-0.1%	8/15
	0.6	559.6	532	454	4.3%	1.16%	2.04%	0.07%	15/15
	0.9	40.9	37.7	41.1	5.69%	1.45%	5.11%	0.96%	5/5
S1-B2	0.2	1690.4	1567.6	1272.5	1%	0.1%	0.31%	-0.52%	3/15
	0.6	705.6	457.6	475	1.31%	0.38%	0%	0%	14/15
	0.9	35.7	32.6	25.2	2.28%	0.69%	1.8%	0.3%	5/5
S2-B2	0.2	1800	1800	1800	1.66%	0.62%	0.69%	0.05%	1/15
	0.6	1716.5	1591.4	1627.7	1.38%	0.23%	0.48%	-0.7%	12/15
	0.9	264	226.3	171.9	2.6%	0.71%	2.08%	0%	5/5
All	-	710.3	635.8	601	3.25%	0.81%	0%	-0.7%	103/140

Table 4: Results for hybrid lines

were solved to optimality (8 by P_S, 4 by P_A, and 12 by P_H).

Notice that the number of instances solved to optimality is higher for hybrid lines (103) than for synchronous (94) or asynchronous ones (98) (Table 4). This reflects the fact that hybrid lines are, on the one hand more flexible than synchronous ones (and, hence, reach lower upper bounds), and on the other hand more restrictive than asynchronous ones (and, hence, allow higher lower bounds). This can explain why significantly more instances of the large dataset (e.g. S2-B2 OS=0.6) could be solved to optimality for hybrid lines (12/15) than for synchronous (8/15) or asynchronous ones (4/15).

The hybrid line case study demonstrated that such lines behave as intermediaries between the synchronous and asynchronous ones. In average, synchronism demanded a 3.25% higher cycle time when applied to the whole line and a 0.81% higher cycle time when applied to half the line. In three cases[6], the hybrid line reached the same value of cycle time as the asynchronous one (both the hybrid and asynchronous instances were solved optimality). If the cost of hybrid lines is also (as expected) an intermediary between the costs of synchronous and asynchronous ones, then hybrid lines can be seen as cheaper alternatives to asynchronous lines: In the computational case study, hybrid lines offer most of the productivity difference with a smaller cost.

This highlights a particular value of the proposed formulation as it allows performance evaluations of different line control systems. This, in turn, offers to managers a possibility to quantitatively measure the trade-offs allowed by the both full and partial (hybrid) asynchronism. While such evaluation was previously possible via simulation (Tiacci, 2015a), simulations would require both balancing and sequencing solutions as inputs. The presented model allows balancing and sequencing to become degrees of freedom to be optimized, leading to smaller cycle times and better economical results.

[6] S1-B1 instance 28, S1-B2 instance 26, S2-B2 instance 34

7. Conclusion

This paper presented an MILP formulation that successfully optimizes small and medium sizes of mixed-model unpaced assembly lines. The proposed model simultaneously solves the balancing of mixed-model lines along with the cyclical sequencing and scheduling of products in the line. The formulation is flexible enough to consider both synchronous and asynchronous assembly lines, with the possibility of treating hybrid (partly synchronous, partly asynchronous) lines as well.

A dataset for the simultaneously balancing and cyclical sequencing of unpaced Assembly Lines containing 140 instances is proposed. Instances are based on Otto et al. (2013)'s SALBP instances. The dataset can be solved for both synchronous and asynchronous lines, and is used as a benchmark to compare the proposed formulation to the former best methods for both synchronous and asynchronous lines. With a time limit of 1800 seconds, the proposed model outperformed previously described alternatives in 138 out of the 140 instances of synchronous lines and all 140 instances for the asynchronous lines. In average, the proposed formulation found answers with 3.62% smaller cycle time for the synchronous cases and 33.3% for the asynchronous lines.

The structure of the proposed model allows efficiency comparisons of line control in assembly lines. To the best of the authors' knowledge, no other previous work on assembly line balancing considers the optimization of hybrid synchronous/asynchronous lines. As expected, the independent movement between the stations of asynchronous lines results in lower cycle times when compared to the synchronous counterpart. In average, for the proposed dataset, the asynchronous lines can produce 3.25% faster and in only one instance the optimal answers of both synchronous and asynchronous were the same. A hybrid line consisting of half asynchronous and half synchronous stations achieved an intermediary result with cycle times slightly higher than the fully asynchronous line (0.81%).

Although the efficiency of asynchronous lines are greater or equal to the synchronous lines, the latter have smaller implementing costs. With the flexibility of the proposed formulation, the output of lines can be determined and used in an economic evaluation of the transport systems. Hybrid lines are especially advantageous when asynchronous stations can be installed to take advantage of the more flexible piece transportation, while simpler synchronous stations are kept where these benefits are not present.

The proposed model can be extended to include the economic pricing of transport systems or by considering further practical characteristics. Another aspect that can be incorporated as a problem's feature is buffer allocation. Although buffers tend not to be very relevant for synchronous lines (Boysen et al., 2008) and might even worsen the performance of said lines (Kouvelis & Karabati, 1999), in asynchronous lines (and in the asynchronous part of hybrid lines) buffers might be instrumental in allowing better performance by reducing blockages and starvations associated to the asynchronous flow of pieces. Furthermore, other characteristics such as parallel stations or U-shaped assembly lines are relevant to the industrial production and are pointed as directions for future works.

Funding

This work was supported by Fundação Araucária (Agreements 141/2015, 06/2016 and 041/2017) and CNPq (Grants 406507/2016-3 and 307211/2017-7).

References

Akpinar, S., & Baykasoglu, A. (2014). Modeling and solving mixed-model assembly line balancing problem with setups. Part I: A mixed integer linear programming model. *Journal of Manufacturing Systems*, *33*, 177–187. doi:10.1016/j.jmsy.2013.11.004.

Akpinar, S., Elmi, A., Bektaş, T., & Bektas, T. (2017). Combinatorial Benders cuts for assembly line balancing problems with setups. *European Journal of Operational Research*, *259*, 527–537. doi:10.1016/j.ejor.2016.11.001.

Alghazi, A., & Kurz, M. E. (2018). Mixed model line balancing with parallel stations, zoning constraints, and ergonomics. *Constraints*, *23*, 123–153. doi:10.1007/s10601-017-9279-9.

Bard, J. F., Dar-elj, E., & Shtub, A. (1992). An analytic framework for sequencing mixed model assembly lines. *International Journal of Production Research*, *30*, 35–48. doi:10.1080/00207549208942876.

Battaïa, O., & Dolgui, A. (2013). A taxonomy of line balancing problems and their solution approaches. *International Journal of Production Economics*, *142*, 259–277. doi:10.1016/j.ijpe.2012.10.020.

Battini, D., Faccio, M., Persona, A., & Sgarbossa, F. (2009). Balancing - sequencing procedure for a mixed model assembly system in case of finite buffer capacity. *International Journal of Advanced Manufacturing Technology*, *44*, 345–359. doi:10.1007/s00170-008-1823-8.

Becker, C., & Scholl, A. (2006). A survey on problems and methods in generalized assembly line balancing. *European Journal of Operational Research*, *168*, 694–715. doi:10.1016/j.ejor.2004.07.023.

Boysen, N., Fliedner, M., & Scholl, A. (2007). A classification of assembly line balancing problems. *European Journal of Operational Research*, *183*, 674–693. doi:10.1016/j.ejor.2006.10.010. arXiv:1503.07717.

Boysen, N., Fliedner, M., & Scholl, A. (2008). Assembly line balancing: Which model to use when? *International Journal of Production Economics*, *111*, 509–528. doi:10.1016/j.ijpe.2007.02.026. arXiv:arXiv:1011.1669v3.

Boysen, N., Fliedner, M., & Scholl, A. (2009a). Production planning of mixed-model assembly lines: Overview and extensions. *Production Planning and Control*, *20*, 455–471. doi:10.1080/09537280903011626.

Boysen, N., Fliedner, M., & Scholl, A. (2009b). Sequencing mixed-model assembly lines: Survey, classification and model critique. *European Journal of Operational Research*, *192*, 349–373. doi:10.1016/j.ejor.2007.09.013.

Chiang, W.-C. C., Urban, T. L., & Xu, X. (2012). A bi-objective metaheuristic approach to unpaced synchronous production line-balancing problems. *International Journal of Production Research*, *50*, 293–306. doi:10.1080/00207543.2011.571460.

Delice, Y., Aydogan, E. K., Özcan, U., & Ilkay, M. S. (2017). A modified particle swarm optimization algorithm to mixed-model two-sided assembly line balancing. *Journal of Intelligent Manufacturing*, *28*, 23–36. doi:10.1007/s10845-014-0959-7.

Doerr, K. H., Klastorin, T. D., & Magazine, M. J. (2000). Synchronous Unpaced Flow Lines with Worker Differences and Overtime Cost. *Management Science*, *46*, 421–435. doi:10.1287/mnsc.46.3.421.12064.

Dong, J., Zhang, L., & Xiao, T. (2018). A hybrid PSO / SA algorithm for bi-criteria stochastic line balancing with flexible task times and zoning constraints. *Journal of Intelligent Manufacturing*, *29*, 737–751. doi:10.1007/s10845-015-1126-5.

Johnson, R. V. (1983). A Branch and Bound Algorithm for Assembly Line Balancing Problems with Formulation Irregularities. *Management Science*, *29*, 1309–1324. doi:10.1287/mnsc.29.11.1309.

Karabati, S., & Kouvelis, P. (1996). Cyclic Scheduling in Flow Lines: Modeling Observations, Effective Heuristics and a Cycle Time Minimization Procedure. *Naval Research Logistics*, *43*, 211–231. doi:10.1002/(SICI)1520-6750(199603)43:2<211::AID-NAV4>3.0.CO;2-D.

Karabati, S., Kouvelis, P., & Kiran, A. S. (1992). Games, Critical Paths and Assignment Problems in Permutation Flow Shops and Cyclic Scheduling Flow Line Environments. *Journal of Operational Research Society*, *43*, 241–258. doi:10.2307/2583714.

Karabati, S., & Sayin, S. (2003). Assembly line balancing in a mixed-model sequencing environment with synchronous transfers. *European Journal of Operational Research*, *149*, 417–429. doi:10.1016/S0377-2217(02)00764-6.

Kouvelis, P. P., & Karabati, S. S. (1999). Cyclic scheduling in synchronous production lines. *IIE Transactions*, *31*, 709–719. doi:10.1080/07408179908969870.

Kucukkoc, I., & Zhang, D. Z. (2014). Mathematical model and agent based solution approach for the simultaneous balancing and sequencing of mixed-model parallel two-sided assembly lines. *International Journal of Production Economics*, *158*, 314–333. doi:10.1016/j.ijpe.2014.08.010.

Kucukkoc, I., & Zhang, D. Z. (2015). Balancing of parallel U-shaped assembly lines. *Computers and Operations Research*, *64*, 233–244. doi:10.1016/j.cor.2015.05.014.

Kucukkoc, I., & Zhang, D. Z. (2016). Integrating ant colony and genetic algorithms in the balancing and scheduling of complex assembly lines. *International Journal of Advanced Manufacturing Technology*, *82*, 265–285. doi:10.1007/s00170-015-7320-y.

Levner, E., Kats, V., de Pablo, D. A. L., Cheng, T. C., & de Pablo, D. A. L. (2010). Complexity of cyclic scheduling problems: A state-of-the-art survey. *Computers and Industrial Engineering*, *59*, 352–361. doi:10.1016/j.cie.2010.03.013.

Li, Z., Janardhanan, M. N., Tang, Q., & Nielsen, P. (2017). Mathematical model and metaheuristics for simultaneous balancing and sequencing of a robotic mixed-model assembly line. *Engineering Optimization*, *50*, 877–893. doi:10.1080/0305215X.2017.1351963.

Lopes, T. C., Sikora, C. G. S., Michels, A. S., & Magatão, L. (2018). Mixed-Model Assembly Line Balancing with Given Buffers and Product Sequence: Model, Formulation Comparisons and Case Study. *Annals of Operations Research*, *1*, 1–26. doi:10.1007/s10479-017-2711-0.

Nakade, K., Ohno, K., & Shanthikumar, J. G. (1997). Bounds and approximations for cycle times of a U-shaped production line. *Operations Research Letters*, *21*, 191–200. doi:10.1016/s0167-6377(97)00037-0.

Otto, A., Otto, C., & Scholl, A. (2013). Systematic data generation and test design for solution algorithms on the example of SALBPGen for assembly line balancing. *European Journal of Operational Research*, *228*, 33–45. doi:10.1016/j.ejor.2012.12.029.

Özcan, U., Çerçioğlu, H., Gökçen, H., & Toklu, B. (2010). Balancing and sequencing of parallel mixed-model assembly lines. *International Journal of Production Research*, *48*, 5089–5113. doi:10.1080/00207540903055735.

Öztürk, C., Tunali, S., Hnich, B., Örnek, A., & Tunal, S. (2015). Cyclic scheduling of flexible mixed model assembly lines with parallel stations. *Journal of Manufacturing Systems*, *36*, 147–158. doi:10.1007/s00170-012-4675-1.

Öztürk, C., Tunali, S., Hnich, B., & Örnek, M. A. (2013). Balancing and scheduling of flexible mixed model assembly lines. *Constraints*, *18*, 434–469. doi:10.1007/s10601-013-9142-6.

Riezebos, J. (2010). Order release in synchronous manufacturing. *Production Planning & Control: The Management of Operations*, *21*, 347–358. doi:10.1080/09537280903453877.

Riezebos, J. (2011). Order sequencing and capacity balancing in synchronous manufacturing. *International Journal of Production Research*, *49*, 531–552. doi:10.1080/00207540903067185.

Roshani, A., & Nezami, F. G. (2017). Mixed-model multi-manned assembly line balancing problem: a mathematical model and a simulated annealing approach. *Assembly Automation*, *37*, 34–50. doi:10.1108/AA-02-2016-016.

Saif, U., Guan, Z., Liu, W., Wang, B., & Zhang, C. (2014). Multi-objective artificial bee colony algorithm for simultaneous sequencing and balancing of mixed model assembly line. *International Journal of Advanced Manufacturing Technology*, *75*, 1809–1827. doi:10.1007/s00170-014-6153-4.

Salveson, M. E. (1955). The assembly line balancing problem. *The Journal of Industrial Engineering*, *6*, 18–25.

Sawik, T. (2000). Simultaneous versus sequential loading and scheduling of flexible assembly systems. *International Journal of Production Research*, *34*, 3267–3282. doi:10.1080/002075400418252.

Sawik, T. (2002). Monolithic vs. hierarchical balancing and scheduling of a flexible assembly line. *European Journal of Operational Research*, *143*, 115–124. doi:10.1016/S0377-2217(01)00328-9. arXiv:arXiv:1011.1669v3.

Sawik, T. (2004). Loading and scheduling of a flexible assembly system by mixed integer programming. *European Journal of Operational Research*, *154*, 1–19. doi:10.1016/S0377-2217(02)00795-6.

Sawik, T. (2012). Batch versus cyclic scheduling of flexible flow shops by mixed-integer programming. *International Journal of Production Research*, *50*, 5017–5034. doi:10.1080/00207543.2011.627388.

Scholl, A. (1999). *Balancing and sequencing assembly lines*. (2nd ed.). Heidelberg: Physica.

Sternatz, J. (2014). The joint line balancing and material supply problem. *International Journal of Production Economics*, *159*, 304–318. doi:10.1016/j.ijpe.2014.07.022.

Thomopoulos, N. T. (1967). Line Balancing-Sequencing for Mixed-Model Assembly. *Management Science*, *14*, B—-59—-B75.

Thomopoulos, N. T. (1970). Mixed Model Line Balancing with Smoothed Station Assignments. *Management Science*, *16*, 593–603. doi:10.1287/mnsc.16.9.593.

Tiacci, L. (2015a). Coupling a genetic algorithm approach and a discrete event simulator to design mixed-model un-paced assembly lines with parallel workstations and stochastic task times. *International Journal of Production Economics*, *159*, 319–333. doi:10.1016/j.ijpe.2014.05.005.

Tiacci, L. (2015b). Simultaneous balancing and buffer allocation decisions for the design of mixed-model assembly lines with parallel workstations and stochastic task times. *International Journal of Production Economics*, *162*, 201–215. doi:10.1016/j.ijpe.2015.01.022.

Tiacci, L. (2017). Mixed-model U-shaped assembly lines: Balancing and comparing with straight lines with buffers and parallel workstations. *Journal of Manufacturing Systems*, *45*, 286–305. doi:10.1016/j.jmsy.2017.07.005.

Urban, T. L., & Chiang, W. C. (2016). Designing energy-efficient serial production lines: The unpaced synchronous line-balancing problem. *European Journal of Operational Research*, *248*, 789–801. doi:10.1016/j.ejor.2015.07.015.

Waldherr, S., & Knust, S. (2017). Decomposition algorithms for synchronous flow shop problems with additional resources and setup times. *European Journal of Operational Research*, *259*, 847–863. doi:10.1016/j.ejor.2016.11.015.

Zhong, Y.-g. (2017). Hull mixed-model assembly line balancing using a multi-objective genetic algorithm simulated annealing optimization approach. *Concurrent Engineering: Resarch and Applications*, *25*, 30–40. doi:10.1177/1063293X16666204.