Journal of Global Optimization (2024) 89:435–455
https://doi.org/10.1007/s10898-023-01354-0

Single-lot, lot-streaming problem for a 1 + m hybrid flow shop

Sanchit Singh[1] · Subhash C. Sarin[1] · Ming Cheng[2]

Received: 8 October 2022 / Accepted: 5 December 2023 / Published online: 12 January 2024

Abstract
In this paper, we consider an application of lot-streaming for processing a lot of multiple items in a hybrid flow shop (HFS) for the objective of minimizing makespan. The HFS that we consider consists of two stages with a single machine available for processing in Stage 1 and m identical parallel machines in Stage 2. We call this problem a $1 + m$ TSHFS-LSP (two-stage hybrid flow shop, lot streaming problem), and show it to be NP-hard in general, except for the case when the sublot sizes are treated to be continuous. The novelty of our work is in obtaining closed-form expressions for optimal continuous sublot sizes that can be solved in polynomial time, for a given number of sublots. A fast linear search algorithm is also developed for determining the optimal number of sublots for the case of continuous sublot sizes. For the case when the sublot sizes are discrete, we propose a branch-and-bound-based heuristic to determine both the number of sublots and sublot sizes and demonstrate its efficacy by comparing its performance against that of a direct solution of a mixed-integer formulation of the problem by CPLEX®.

Keywords Scheduling · Lot-streaming · 1 + m hybrid flow shop

1 Introduction

In this paper, we consider the problem of scheduling a single lot over a hybrid flow shop. Depending upon the application, the lot is said to either have a size U or consist of U identical items. The hybrid flow shop that we consider has two stages with a single machine in Stage 1 and m parallel and identical machines in Stage 2. We implement the process of

✉ Subhash C. Sarin
sarins@vt.edu

Sanchit Singh
sanchit@vt.edu

Ming Cheng
mcheng1@suda.edu.cn

1 Grado Department of Industrial and Systems Engineering, Virginia Tech, Blacksburg, VA 24061, USA

2 School of Rail Transportation, Soochow University, Suzhou, China

lot-streaming, thereby allowing the lot to be split into multiple sub-lots for simultaneous processing over parallel machines in Stage 2. Upon processing over the Stage 1 machine, each sublot incurs a fixed removal time in Stage 1 before its transfer to one of the machines in Stage 2. We determine an optimal schedule for the objective of minimizing makespan. As such, we obtain the number of sublots, sublot sizes, and sublots' assignment to the machines in Stage 2. We consider two cases for sublot sizes-continuous and discrete (integer). We designate this problem as a 1 + m Two-Stage Hybrid Flow Shop Lot-streaming Problem (1 + m TSHFS-LSP).

The Hybrid Flow Shops (HFS) have important applications in flexible manufacturing systems that work on the principle of agile manufacturing. Devor et al. [4] mention the view shared by industrial executives that a competitive advantage in the future lies in strategies promoting speed to market and adherence to changing customer demands. Agile manufacturing is a collective expression for all such strategies geared towards thriving in a continuously changing production environment, and therefore it is becoming a system of increasing importance currently because of its ability to effectively respond to changing customers' demands. Flexible manufacturing systems have attracted a significant amount of attention from the research community in the past. The HFSs are the physical entities that define the core of such systems. They have been employed for producing discrete products in the electronics, furniture and steel industries Tang et al. [21], and also, have been used in continuous processing industries such as textile Jungwattanakit et al. [11], food-processing Yaurima-Basaldua et al. [25], and chemical and pharmaceutical Gholami et al. [5].

Some specific examples of 1 + m HFSs include the scheduling problem encountered at the blade line for Pratt and Whitney Inc. Li [14], where various part types are grouped into part families or batches in Stage 1 for processing on the machines in Stage 2, and the scheduling of a program on a parallel-task computer, where a single machine in Stage 1 performs sequential data loading from external memory. Subsequently, the program is split into tasks that are performed concurrently on parallel CPUs in Stage 2 Carpov et al. [1]. Wittrock [24] and Jin et al. [10] present a related problem for the assembly of printed circuit boards (PCB), where the lots of different sizes of PCBs move sequentially through a three-stage HFS, with each stage having several insertion machines in parallel to insert electronic components on PCBs.

An extensive overview of research on the HFSs has been presented by Ribas et al. [16] and Ruiz and Vázquez-Rodríguez [17]. The scheduling of jobs or lots on a two-stage HFS (having a single machine at Stage 1) with varied constraints and objectives has been addressed by Gupta [6], Sriskandarajah and Sethi [20], Kusiak [13], Gupta and Tunc [7], Gupta and Tunc [8], Hoogeveen et al. [9] and Carpov et al. [1]. However, their work does not consider lot-streaming. Gupta [6] has shown that the two-stage HFS scheduling problem is NP-hard for a given number of jobs and the objective of minimizing makespan. We claim that the 1 + m TSHFS-LSP is an NP hard problem as well, since it is a generalization of the problem studied by Gupta [6]. If we fix the number and sizes of sublots in 1 + m TSHFS-LSP, the resulting sublots have a role similar to that of the jobs that require scheduling over the two-stage HFS. Although, the 1 + m TSHFS-LSP is NP hard in the general sense, yet we will show later that there exists a special case for which this problem presents itself with a pseudo-polynomial time solution.

Even though lot-streaming promises benefits while minimizing completion time-based measures for flow shop configurations and its variants (including HFS), its application in the HFS-based problems has been rather limited in the literature, due to the added modeling and computational complexity. Notably, the studies in this regard include those by Tsubone et al. [22], Zhang et al. [27], Zhang et al. [28], Liu [15], Cheng et al. [2], and, more recently, by Zhang et al. [26] and Shao et al. [19]. Tsubone et al. [22] studied the impact of using lot-streaming in a $1 + m$ hybrid flow shop on makespan, total flow time, maximum work-in-process, and capacity utilization, by using a simulation model. Zhang et al. [28] considered the $1 + m$ HFS problem for the objective of minimizing mean completion times of the sublots. They study the case of equal sublot sizes and have presented two heuristics together with specifying a lower bound on the solution value. The case of a single lot m + 1 hybrid flow shop for the criterion of minimizing makespan is presented in Zhang et al. [27]. The problem is solved assuming equal sublot sizes, which are then modified to obtain an integer solution. Liu [15] studied the $m+1$ HFS problem for the objective of minimizing makespan by treating sublot sizes to be continuous. They proved a property of an optimal solution, in which the sublots can be assigned to Stage 1 machines following a round-robin assignment rule. Even though they specified sublot assignment decisions, sublot sizes were obtained using a linear program for a given number of sublots. A heuristic method was presented to determine the number of equal-sized sublots. Cheng et al. [2] made an important contribution towards reducing the complexity of the algorithm for obtaining an optimal schedule of a special case of the $1 + m$ HFS problem (where they studied $m = 2$). They not only determined an optimal number of sublots, but also determined their sizes for both continuous and discrete cases. Their key contribution lies in the development of closed-form expressions for continuous sublot sizes when the number of sublots is specified. In this paper, we have generalized this result to $1 + m$ two-stage HFS (where $m \geq 2$). Moreover, we have developed an efficient method to determine an optimal number as well as sizes of sublots for both continuous and discrete cases. Zhang et al. [26] present a collaborative variable descent neighborhood algorithm for a hybrid flow shop problem where a lot is split into consistent sublots for processing, while Shao et al. [19] address a distributed heterogeneous hybrid flow shop problem in which a job is split into sublots for assignment to factories (machines) having different capacities for the objective of minimizing the makespan.

The problem that we address in this paper can be stated as follow: Given a lot of size U (parts) to be processed in a $1 + m$ hybrid flow shop consisting of one machine in Stage 1 and m parallel machines in Stage 2 (all available at time zero), determine the number of sublots and sublot sizes where each sublot is processed first on the machine in Stage 1 and then on one of the machines in Stage 2, so as to minimize the makespan. A sublot-attached removal time is incurred for each sublot on the machine in Stage 1. We consider instances of both continuous and discrete sublot sizes. We have already presented above real-life examples of such a problem for both continuous Jungwattanakit et al. [11], Yaurima-Basaldua et al. [25], and Gholami et al. [5] and discrete Tang et al. [21], Jin et al. [10], and Wittrock et al. [24] sublot sizes. A sublot-attached removal time is incurred because of preparation of a sublot before its transfer to a machine at the second stage.

For a multi-stage production process, it is well-known that splitting a lot of a job into sublots and then processing them in an overlapping fashion over the available machines can improve a regular measure of performance Kalir and Sarin [12]. A key question that arises in implementing this idea is how to split the lot, i.e., determination of number of sublots and sublot sizes. For the problem on hand, because of the presence of parallel machines in Stage 2, the allocation of sublots to the machines in that stage is another issue that must be addressed. However, it has been shown that for the case of continuous sublot sizes Liu [15], it is optimal to allocate the sublots in rotation to the machines in Stage 2. Still, the makespan is impacted by the number of sublots and sublot sizes, which is what we determine in this paper. To that end, first, we assume the number of sublots to be given and sublot sizes to be continuous. Then, the resulting problem is only to determine sublot sizes in order to minimize makespan. Subsequently, we address the problem of determining both the number of sublots and continuous sublot sizes for the objective of minimizing makespan. Finally, we address the general problem of determining number of sublots, discrete sublot sizes and allocation of sublots to the machines in Stage 2 for the same objective function.

The contributions made by the paper can be summarized as follows:

1. A closed-form expression to determine continuous sublot sizes for a given number of sublots for a new hybrid flow shop lot streaming problem with applications in flexible manufacturing systems and agile manufacturing, among others, for the objective of minimizing makespan.
2. A polynomial time algorithm to determine both the number of sublots and continuous sublot sizes.
3. A method to determine both the number of sublots and discrete sublot sizes that performs better than the direct solution of a mathematical model of the problem by the commercial solver CPLEX.

Organization of the paper The remainder of this paper is organized as follows. Section 2 contains the notation used along with a model formulation for the 1+m TSHFS-LSP. Section 3 addresses the case of continuous sublot sizes. Here, we obtain closed-form expressions for sublot sizes when the number of sublots, n is specified. We also present an algorithm to determine an optimal value of n in this section besides determination of sublot sizes that minimize makespan. The case of discrete sublot sizes is addressed in Sect. 4, wherein a branch-and-bound-based heuristic is developed to obtain the number and sizes of sublots as well as the assignment of sublots to the machines in Stage 2 for the objective of minimizing makespan. The performance of this method is compared with that of directly solving the proposed model formulation for the 1 + m TSHFS-LSP by CPLEX® for which the results are presented in Sect. 5. Concluding remarks are then made in Sect. 6.

Keeping in view that we are studying a HFS, it is assumed that $m > 1$ for the material presented in Sects. 3 and 4. For the case when $m = 1$ (i.e., we have a two-machine flow shop) with multiple sublots, we refer to Vickson [23].

2 Model formulation

Consider the following notation.

Parameters:

- U Lot size (u is used as an index for lot size, $1 \leq u \leq U$).
- t Removal time for a sublot at the Stage 1 machine incurred before transferring it to any machine in Stage 2 (note that the unit processing time of the machine in Stage 1 is fixed at unity; other parameter values, p and t, and variables measuring time are scaled accordingly).
- p Time to process one part at any machine at Stage 2.

m Number of machines available at Stage 2 (note that j is used to represent an index of a machine and k is used to represent number of machines at Stage 2,$1 \leq j, k \leq m$).

$\overline{n}$ Maximum number of sublots allowed, for both continuous and integer-sized sublots (note that i is used to represent indices of a sublot and n is used to represent number of sublots, $1 \leq i, n \leq \overline{n}$).

$\rho = (p + 1)/p$.

Decision Variables:

- $\mathcal{M}(u, n, k)$ Makespan value of an optimal schedule, for a given u, n and k, where the lot of size u is split into n sublots and processed on precisely k machines in Stage 2.

Note that when the number of sublots (k) is less than m, then these sublots will be processed on k machines in Stage 2 because these machines will be available when the sublots finish processing on the machine in Stage 1. However, when $k \geq m$, the sublots will be processed on m machines. Consequently, we use the notation as indicated.

$\widehat{\mathcal{M}}(u)$ Makespan value of an optimal schedule for a given lot size u. The number of sublots in such a schedule is designated by $\hat{n}(u)$, and these sublots are assigned to $\hat{k}(u)$ machines in Stage 2 ($1 \leq \hat{k}(u) \leq m,\ 1 \leq \hat{n}(u) \leq \overline{n}$).

$\widetilde{\mathcal{M}}(u, n_o)$ Makespan value of an optimal schedule, for a given u and n_o. An optimal number of sublots in such a schedule is designated by $\tilde{n}(u, n_o)$, and these sublots are assigned to $\tilde{k}(u, n_o)$ machines in Stage 2 ($1 \leq \tilde{k}(u, n_o) \leq m$, $1 \leq \tilde{n}(u, n_o) \leq n_o \leq \overline{n}$).

$\mathcal{S}_i(u, n, k)$ Sublot size for the$i$th sublot (sublots are numbered in order of their sequence on Stage 1 machine) in a schedule, where the lot of size u is split into n sublots and processed on precisely k machines at Stage 2. We also use sublots numbered in the reverse order in which case $\overline{s}_i(u, n, k)$ represents size of the ith sublot from the end of a schedule in Stage 1; i.e. $\overline{s}_i(u, n, k) = s_{n-i+1}(u, n, k)$. Note that the sublot sizes need not be equal. They are appropriately determined for the continuous and discrete cases.

$\mathcal{C}_{1,i}(u, n, k)$ Completion time of the ith sublot in Stage 1 when the lot of size$u$ is split into$n$ sublots and processed on precisely k machines in Stage 2.

$\mathcal{C}_{2,i,j}(u, n, k)$ Completion time of the ith sublot on the jth machine in Stage 2 when the lot of size u is split into$n$ sublots and processed on precisely k machines at Stage 2 ($1 \leq j \leq k$).

We now present a mixed integer program (MIP) model for the $1 + m$ TSHFS-LSP, and designate it as Model TSHFS-LSP. We also define three sets of binary variables:$v_i = 1$, if sublot i is used, and $= 0$, otherwise, $\forall i = 2, \ldots \overline{n}$; $x_{i,j} = 1$, if sublot i is assigned to machine j in Stage 2, and $= 0$, otherwise, $\forall i = 1, \ldots \overline{n}$, $\forall j = 1, \ldots \overline{m}$; and $h_j = 1$, if jth machine is used for scheduling any sublot in Stage 2, and $= 0$, otherwise, $\forall j = 1, \ldots m$. We use an upper bound on the number of sublots. Since all of these sublots need not be used, variable v captures this fact.

2.1 Model TSHFS-LSP

Minimize: $M + \varepsilon \sum_{j=1}^{m} h_j$

$$M \geq c_{2,n,j}, \quad \forall j = 1, \ldots m \tag{1}$$

$$c_{1,1} = s_1 + t \tag{2}$$

$$c_{1,i} \geq c_{1,i-1} + s_i + tv_i,\ \forall i = 2, \ldots \widetilde{n} \tag{3}$$

$$c_{2,i,j} \geq c_{1,i} + ps_i - \theta\left(1 - x_{i,j}\right),\ \forall i = 1, \ldots \widetilde{n},\ j = 1, \ldots m \tag{4}$$

$$c_{2,i,j} \geq c_{2,i-1,j} + ps_i - \theta\left(1 - x_{i,j}\right),\ \forall i = 2, \ldots \widetilde{n},\ j = 1, \ldots m \tag{5}$$

$$c_{2,i,j} \geq c_{2,i-1,j},\ \forall i = 2, \ldots \widetilde{n},\ j = 1, \ldots m \tag{6}$$

$$\sum_{j=1}^{m} x_{i,j} = 1, \forall i, \cdots \widetilde{n} \tag{7}$$

$$\sum_{i=1}^{n} \overline{s}_i = U \tag{8}$$

$$s_i \leq U v_i, \forall i = 1, \ldots \widetilde{n} \tag{9}$$

$$v_i \leq v_{i-1}, \forall i = 2, \ldots \widetilde{n} \tag{10}$$

$$\sum_{i=1}^{n} x_{i,j} \leq \sum_{i=1}^{n} x_{i,j-1}, \forall j = 1, ...m \tag{11}$$

$$s_i \geq 0, \ \forall j = 2, \ldots m \tag{12}$$

$$\left. \begin{array}{l} c_{1,i} \geq 0, \ v_1 \in \{0, 1\}, \ h_j \in \{0, 1\} \\ c_{2,i,j} \geq 0, \ x_{i,j} \in \{0, 1\} \end{array} \right\}, \forall i = 1, \ldots \widetilde{n}, \ j = 1, \ldots m \tag{13}$$

The objective is to minimize the makespan of the schedule, along with a small penalty proportional to the total number of machines used in Stage 2 (we use a small enough value of ϵ in order to minimize the number of machines used without compromising the makespan value; here, we use ϵ). Constraints (1) enforce the makespan value to be at least as large as the last sublot's completion time on each machine in Stage 2. Note that, it is not necessary for the last sublot (or any other sublot) to have non-zero size for its completion time to be defined. The relationships among sublots' completion times are captured by Constraints (2) and (3) over the Stage 1 machine, and by Constraints (4), (5), and (6) over the machines in Stage 2. Constraints (7) assert unique assignment for each sublot in the second stage. Constraint (8) ensures all sublot sizes to add up to the lot size U. Constraints (9) ensure that the sublot size is zero when a sublot is not used. Constraints (10) strengthen the model by adding symmetry-breaking conditions. Constraints (11) ensure that no sublot is assigned to a machine when that machine is not used. Constraints (12) strengthen the model by adding another symmetry-breaking condition that enforces the number of sublots assigned to the machines to be in the non-increasing order of their indices. Constraints (13) define the domains of the variables. A standard value of θ for use in constraints (4) and (5) is taken as pU.

2.2 Solution methodology for continuous sublot sizes

In this section, we determine an optimal schedule for the $1+m$ TSHFS-LSP for the case when the sublot sizes can take on continuous values. It is further divided into two sub-sections. Sect. 3.1 addresses the determination of sublot sizes $\mathcal{S}_i(U, n, m)$ (or $\overline{\mathcal{S}}_i(U, n, m)$), $\forall i = 1, \ldots n$, allocation of sublots to the Stage 2 machines, and makespan value $\mathcal{M}(U, n, k)$ when the number of sublots is fixed to $n(\leq \overline{n})$, while in Sect. 3.2, some key results are derived for use in the determination of an optimal schedule when the number of sublots is not specified.

2.3 Determination of optimal schedule when the number of sublots is fixed

Theorem 1 *For a given lot of size U, number of machines, m* (≥ 2)*, and number of sublots, n,* *1. if $n \leq m$, the optimal continuous sublot sizes can be obtained by only utilizing n number of machines, and assigning each sublot to a different machine in Stage 2. The sublot sizes are thus obtained by solving the following expressions.*

$$\bar{s}_i = \rho\bar{s}_{i-1} + t/p, \forall i = 2, \dots n \tag{14}$$

$$\sum_{i=1}^{n} \bar{s}_i = U \tag{15}$$

2. if $n > m$, the optimal continuous sublot sizes are obtained by solving the following expressions.

$$\bar{s}_i = \rho\bar{s}_{i-1} + t/p, \forall i = 2, \dots m \tag{16}$$

$$\bar{s}_i = (\bar{s}_{i-1} \dots + \bar{s}_{i-m})/p + mt/p, \forall i = m+1, \dots n \tag{17}$$

$$\sum_{i=1}^{n} \bar{s}_i = U \tag{18}$$

Sublots are assigned to the machines in Stage 2 following the 'round robin assignment' rule; i.e., sublots $\overline{S}_1$, $\overline{S}_{m+1}$, $\overline{S}_{2m+1}$, ... are assigned to machine 1, sublots $\overline{S}_2$, $\overline{S}_{m+2}$, $\overline{S}_{2m+2}$, ... are assigned to machine 2, and so on.

Note that $\overline{S}_i$ is used as a shorthand for $\overline{S}_i(U, n, m)$ here. In order to minimize the makespan, the completion times of the last sublots assigned to the machines in Stage 2 should be as close to each other as possible. Starting backward, Expressions (14), (16) and (17) specify sublot sizes such that the processing of a sublot overlaps those of the requisite number of subsequent sublots without creating an idle time on any machine in Stage 2.

The proof of Theorem 1 and those of other results in the sequel are presented in the Appendix section in order not to detract the reader.

Remark 1 For the case, $n \leq m$, $\mathcal{M}(U, n, m)$ is equivalent to $\mathcal{M}(U, n, n)$; similarly, $\overline{S}_i(U, n, m) \equiv \overline{S}_i(U, n, n)$, $\forall i = 1, \dots n$.

Determination of $\overline{S}_1$ (when $n \leq m$). The value of $\overline{S}_1$ is obtained by straight-forward substitutions using Eqs. (14) and (15) as follows.

$$\bar{s}_1 = \frac{U(\rho - 1) - t(\rho^n - n\rho + n - 1)}{\rho^n - 1} \tag{19}$$

$$\bar{s}_i = \rho^{i-1}\bar{s}_1 + t\left(\rho^{i-1} - 1\right), \forall i = 2, \dots n \tag{20}$$

Determination of $\overline{S}_1$ (when n > m). In matrix notation, Eqs. (16)–(18) can be represented in the form, $\mathbf{Ax} = \mathbf{b}$, where $\mathbf{x}$ denotes a vector of unknown sublot sizes of dimension n. As such, various established exact methods can be used to obtain $\mathbf{x}$, or equivalently, $\overline{S}_i$, $\forall i = 1, \dots n$. These include 'Gaussian Elimination', which is known to have arithmetic complexity of $\mathcal{O}(n^3)$. At best, the inverse operation takes $\mathcal{O}(n^{2.376})$ order of complexity using the Coppersmith-Winograd algorithm Coppersmith and Winograd [3], which involves performing 'LU decomposition' of matrix $\mathbf{A}$. Our aim is to further improve upon the time complexity of obtaining $\overline{S}_i$, $\forall i = 1, \dots n$, by utilizing the inherent structure present in (16)–(18).

By using Theorem 1 and the results from linear algebra pertaining to recurrence sequences, the value of $\overline{S}_1$ is obtained by the expression given in (21) when $p \neq m$ and by (22) when

$p = m$ (for the derivation of these expressions and generation of these two conditions, please refer to Appendix EC.2).

$$\bar{s}_1 = \frac{U - \left((\omega_n - e - nc) + t \left(\sum_{k=2}^{m-1} \left(\beta_{n,k} - b_k \right) \left(\rho^{k-1} - 1 \right) \right) \right)}{\sum_{k=1}^{m-1} \left(\beta_{n,k} - b_k \right) \rho^{k-1}} \tag{21}$$

$$\bar{s}_1 = \frac{U - \left(\left(\omega_{n'} - ne' - n(n+1)c'/2 \right) + t \left(\sum_{k=2}^{m-1} \left(\beta_{n,k'} - nb_{k'} \right) \left(\rho^{k-1} - 1 \right) \right) \right)}{\sum_{k=1}^{m-1} \left(\beta_{n,k'} - nb_{k'} \right) \rho^{k-1}} \tag{22}$$

Among the scalars, of particular interest are $\beta_{n,l}$ and $\omega_n, \forall i = 1, \ldots m-1$, when $p \neq m$, and $\beta'_{n,l}$ and $\omega'_n, \forall i = 1, \ldots m-1$ when $p = m$. These are obtained as follows.

$$\left[\beta_{n,1} \ \beta_{n,2} \ \ldots \ \beta_{n,m-1} \right]^T = \mathbf{B}_{m,m-1}^T \mathbf{R}_{m,m}^{-1} \mathbf{r}_m^n \tag{23}$$

$$\omega_n = \mathbf{W}_m \mathbf{R}_{m,m}^{-1} \mathbf{r}_m^n \tag{24}$$

$$\left[\beta_{n,1'} \beta_{n,2'} \ldots \beta_{n,m-1'} \right]^T = \mathbf{B}_{m,m-1'}{}^T \mathbf{R}_{m,m}^{-1} \mathbf{r}_m^n, \textit{ and} \tag{25}$$

$$\omega'_n = \mathbf{w}'_m \mathbf{R}_{m,m}^{-1} \mathbf{r}_m^n, \tag{26}$$

where $\mathbf{B}_{m,m-1}$, $\mathbf{R}_{m,m}$ and $\mathbf{B}'_{m,m-1}$ are scalar matrices, and $\mathbf{r}_m^n$, $\mathbf{w}_m$ and $\mathbf{w}'_m$ are scalar vectors. Note that the calculations in Eqs. (23)–(26) involve computation for the inverse of matrix $\mathbf{R}_{m,m}$, which has an order of complexity, $\mathcal{O}\left(m^{2.376}\right)$ based on the method described in Coppersmith and Winograd [3]. Also, it takes $\mathcal{O}(mn)$ time to obtain $\mathbf{r}_m^n$, thus requiring $\mathcal{O}\left(m^{2.376} + mn\right)$ time to obtain $\overline{S}_1$ alone. Finally, we can determine sizes of the remaining n-1 sublots in no more than $\mathcal{O}(n)$ time, after having computed the value of $\overline{S}_1$, as shown next. Overall, the complete schedule can be obtained in polynomial time, $\mathcal{O}\left(m^{2.376} + mn + n\right)$ or $\mathcal{O}\left(m^{2.376}\right)$. Since, $m < \ < n$ typically, our approach offers a significant reduction in time complexity over the conventional approach as discussed earlier; i.e., compare $\mathcal{O}\left(m^{2.376}\right)$ v/s $\mathcal{O}\left(n^{2.376}\right)$, respectively.

The rest of the sublot sizes, $\overline{S}_i$, $\forall i = 2, \ldots m$, follow from the $\overline{S}_1$ value, and they can be obtained using (16) as follows.

$$\bar{s}_i = \rho^{i-1} \bar{s}_1 + t\left(\rho^{i-1} - 1 \right) \forall i = 2, \ldots m \tag{27}$$

The value of $\overline{S}_{m+1}$ can be obtained by using (17). For $i = m+2, \ldots n$, $\overline{S}_i$ is obtained as follows.

$$\bar{s}_i = \rho \bar{s}_{i-1} - \bar{s}_{i-m-1}/p, \ \forall i = m+2, \ldots n \tag{28}$$

Equations (16)–(18) do not guarantee attainment of non-negative values for all sublot sizes. The next result provides necessary and sufficient conditions that ensure the schedule obtained using Theorem 1 is feasible; i.e. all the sublot sizes are non-negative.

Corollary 1 *If the last sublot size is non-negative, then all other sublot sizes are strictly positive; i.e.,*$\overline{S}_i \geq 0 \Rightarrow \overline{S}_i > 0, \forall i = 2, \ldots, n$, *and the complete schedule is a feasible one.*

Corollary 2 *The schedule given by the sublots is critical (i.e., no sublot waits for processing over the machines in Stage 2, once it starts processing on the machine in Stage 1, and there is no idle time on the machines in Stage 2), and the completion times of the last sublots on Stage 2 machines are the same. In such a case, the makespan value,* $\mathcal{M}(U, n, m)$ *is given by,*

$$\mathcal{M}(U, n, m) = U + nt + p\bar{s}_1(U, n, m) \tag{29}$$

The next question of interest is: how does $\overline{S}_1$ vary with U and n, and when in fact can it become negative? Indeed, we show in Theorem 3 below that $\overline{S}_1$ decreases monotonically with increment in n, and thus it can become negative beyond certain value of n that we call n_f, for n feasible. This result is helpful in determining optimal n, a topic that we address in Sect. 3.2. We present a result in Theorem 4 that directly aids in the determination of optimal n as well. We also present below an algorithm to findn_f. But first, we show in Theorem 2 how sublot sizes vary with variation in U for a given n.

Theorem 2 *For a given number of machines, $m(\geq 2)$, and the number of sublots, n, the values of $\overline{S}_i(U, n, m)$, $\forall i = 1, \ldots, n$, increase (decrease) with increment (decrement) in lot size, U.*

Theorem 3 *For a given lot of size, U, and number of machines, $m(\geq 2)$,*

1. *$\overline{S}_1(U, n, m)$ decreases monotonically with increment in the number of sublots, n.*
2. *There exists an integer, n_i, s.t.$\forall n > n_i$, the solution given by* Theorem 1*is infeasible; i.e., $\overline{S}_1(U, n, m)$ is negative (we designate by $n_f$the smallest of all such n_i* values *and $\overline{n}$.)*

Corollary 3 *For a given U, $n_o$and $m(\geq 2)$,*

1. if $\overline{S}_1(U, n_o, m) < 0$, then a solution given by Theorem 1 is not feasible $\forall n \geq n_o$.
2. if $\overline{S}_1(U, n_o, m) \geq 0$, then a solution given by Theorem 1 is feasible $\forall n \leq n_o$.

Remark 2 The definition of n_f together with Corollary 3 imply that $\overline{S}_1(U, n, m) \geq 0$, $\forall n \leq n_f$ and that if $n_f \neq \overline{n}$, then $\overline{S}_1(U, n, m) < 0$, $\forall n > n_f$.

We now present an algorithm to obtain n_f, based on the binary search method.

Algorithm 1 Determination of n_f, for a given U and $m(\geq 2)$.

1. Initialize $n_l = 2$, and $n_u = \overline{n}$, at the start.
2. If $\overline{S}_1(U, n_u, m) < 0$, go to Step 3, else if $\overline{S}_1(U, n_u, m) \geq 0$, let $n_f = n_u$ and stop.
3. If $\overline{S}_1(U, n_l, m) \geq 0$, go to Step 4, else if $\overline{S}_1(U, n_l, m) < 0$, let $n_f = n_l - 1$ and stop.
4. Let $n_z = \lfloor (n_l + n_u)/2 \rfloor$, if $\overline{S}_1(U, n_z, m) < 0$ then $n_u = n_z - 1$, else, $n_l = n_z + 1$. Return to Step 2.

Theorem 4 *For a given U,n$\left(\leq n_f\right)$and $m(\geq 2)$, there exists an integer, n_s, s.t. the makespan for the lot of size U obtained using the number of sublots larger than n_s, where $n_s = n + \left\lfloor p\overline{S}_1(U, n, m)/t \right\rfloor$, does not improve over the makespan obtained using n sublots.*

Another interesting question to investigate is how does $\overline{s}_1$ (and accordingly other sublot sizes) vary with p. To understand this behavior, consider the case of $n \leq m$. We can re-write Expression 19 for $\overline{s}_1$, as follows:

$$\overline{s}_1 = \frac{U(\rho - 1)}{\rho^n - 1} - t\left[1 - \frac{n(\rho - 1)}{\rho^n - 1}\right]$$

Recall $\rho = \frac{1+p}{p}$. Note that as $p \to \infty$, $\rho \to 1$, and when $p \to 0$, $\rho \to \infty$. These values of p represent the situations where the processing time on the machines in Stage 2 is very large or very small. Note that

$$\lim_{p\to\infty} \overline{s}_1 = -t, \text{ and also } \lim_{p\to 0} \overline{s}_1 = -t$$

That is, in the limit, the solution becomes infeasible basically due to the presence of removal time t. The expression for $\overline{s}_1$, for the case when $n > m$, albeit more complex, has a similar structure, and will lead to infeasible solutions as well under these conditions.

2.4 Determination of an optimal schedule when the number of sublots is not specified

By Remark 1, we have the following expression for optimal makespan value,

$$\hat{\mathcal{M}}(U) = \min_{\substack{1 n \tilde{n} \\ 1 k m}} \mathcal{M}(U, n, k) = \min_{\substack{1 n \tilde{n} \\ 1 k \min(n, m)}} \mathcal{M}(U, n, k) \tag{30}$$

Note that, for $1 \leq m_1 < m_2 \leq n$, $\mathcal{M}(U, n, m_2) \leq \mathcal{M}(U, n, m_1)$, since we can always transfer $m_2 - m_1$ sublots from the optimal schedule on m_1 machines to each one of the extra $m_2 - m_1$ machines in Stage 2 that become available when using m_2 machines in total. Therefore, Expression (30) reduces to the following.

$$\widehat{\mathcal{M}}(U) = \min_{1 n \tilde{n}} \mathcal{M}(U, n, \min(n, m)) \tag{31}$$

For a given set of problem parameters, U, $\overline{n}$, m, t, and p, we next present an algorithm to determine the optimal makespan value, $\widehat{\mathcal{M}}(U)$, following (31), which leverages upon the results given in Remark 2 and Theorem 4 in order to reduce the search space for n. Even though, an overwhelming number of experimental results show that the optimal makespan value for a specified number of sublots, n, is a convex function of n (in the discrete sense), we only state this as a conjecture at this point and do not use it in the design of the algorithm. As such, in the worst-case scenario, we perform a linear search from 1 to $\overline{n}$ to determine optimal n. We designate the algorithm as Linear Search Algorithm (LSA). We also determine, $\widetilde{\mathcal{M}}(U, n)$, $\forall 1 \leq n \leq \overline{n}$, within this algorithm, which will be used later in Heuristic 1. Algorithm LSA has a pseudo-polynomial time complexity of order $\mathcal{O}\left(m^{2.376} + m\overline{n}\right)$ (note that the parameter, $\overline{n}$, is usually expressed as a linear function of U). The steps of LSA are as follows.

Algorithm LSA Determination of best makespan value (continuous sublot sizes).

1. Let $\widehat{\mathcal{M}}(U) \leftarrow \widehat{\mathcal{M}}(U, 1, 1) \leftarrow U + t + pU$, $\hat{n}(U) \leftarrow 1$, and $\hat{m}(U) \leftarrow 1$. $\widetilde{\mathcal{M}}(U, 1) \leftarrow \mathcal{M}(U, 1, 1)$. Let $k \leftarrow 2$.
2. If $k < m$, continue, else go to Step 5.
3. Let $n \leftarrow k$. If $n \leq n_f(U, m)$, continue, else go to Step 7.
4. Obtain $\mathcal{M}(U, n, k)$ by (29). If $\widehat{\mathcal{M}}(U)\ominus > \mathcal{M}(U, n, k)$, then let $\widehat{\mathcal{M}}(U) \leftarrow \mathcal{M}(U, n, k)$, $\hat{n}(U) \leftarrow n$, and $\hat{m}(U) \leftarrow k$. Let $\widetilde{\mathcal{M}}(U) \leftarrow \min\{\widetilde{\mathcal{M}}(U, n-1), \mathcal{M}(U, n, k)\}$. Let $k \leftarrow k + 1$. Return to Step 2.
5. Let $n \leftarrow m$. If $n \leq n_f(U, m)$, let $\overline{\overline{n}} \leftarrow n_f(U, m)$, and continue, else go to Step 7.
6. Obtain $\mathcal{M}(U, n, m)$ by (29).If $\widehat{\mathcal{M}}(U)\ominus > \mathcal{M}(U, n, m)$, then let $\widehat{\mathcal{M}}(U) \leftarrow \mathcal{M}(U, n, m)$,$\hat{n}(U) \leftarrow n$, and $\widehat{m}(U) \leftarrow m$. Let $\widetilde{\mathcal{M}}(U, n) \leftarrow \min\{\widetilde{\mathcal{M}}(U, n-1), \mathcal{M}(U, n, m)\}$. Let $\overline{\overline{n}} \leftarrow \min\left(\overline{\overline{n}}, n_s(U, n, m)\right)$. Let $n \leftarrow n + 1$. If $n \leq \overline{\overline{n}}$, then repeat Step 6, else go to Step 7.
7. If $n \leq \overline{n}$, let $\widetilde{\mathcal{M}}\left(U, n'\right) \leftarrow \widetilde{\mathcal{M}}(U, n-1)$, $\forall n' = n, \ldots \overline{n}$. Stop.

3 Case of discrete sublot sizes

In this section, we present a Branch-and-Bound-based Heuristic (B&BH) to solve the $1 + m$ TSHFS-LSP, for the case when the sublot sizes are discrete (integer). However, for comparative purposes, we also solve Model $1 + m$ TSHFS-LSP directly by using a state-of-the-art commercial solver, for a given U, $\overline{n}$, m, t and p, after imposing the integrality restrictions on sublot sizes. The results of an experimental investigation on the performance of these methods that provide information on the makespan value, $\widehat{\mathcal{M}}(U) = \mathcal{M}^* - \epsilon \sum_{j=1}^{m} h_j^*$, number of sublots, $\hat{n}(U) = \sum_{i=1}^{\overline{n}} v_i^*$, and the number of machines required,$\hat{m}(U) = \sum_{j=1}^{m} h_j^*$, corresponding to the best MIP solution, are presented in Sect. 5. Although we describe the proposed B&BH in detail in Appendix EC.9, we briefly discuss here some of its key features. It is a constructive procedure, wherein a tree structure evolves for a given set of problem parameters, $U,\overline{n},\ m,\ t$, and p, starting from a single root node. At any time during the procedure, the tree structure may contain multiple nodes with each node in the proposed branch-and-bound scheme representing a partial schedule (note that we use the word 'partial' in a general sense to encompass a complete schedule as well). Using the results derived for the case of continuous sublot sizes, we impose the following at every node generated in the tree: (1) the maximum number of sublots in a schedule to be smaller than $\overline{n}$ (because of which we call it a heuristic method), and (2) a tight lower bound on the makespan value attained by using a sublot-based bound and a machine-based bound, which helps in reducing the size of the search tree. During the course of this procedure, a node corresponding to the incumbent solution is maintained, which yields the best schedule so far for the $1+m$ TSHFS-LSP at its termination.

4 Computational investigation

In this section, we investigate the performances of methods LSA and B&BH against the direct solution of the MIP Model $1+m$ TSHFS-LSP (the sublot integrality constraints are relaxed for comparing its performance with that of LSA) using CPLEX®. Both LSA and B&BH were implemented and solved using C++ on Xcode (v10.0), whereas the MIP model was solved using C++ Concert library of CPLEX® (v12.8) in multi-thread mode. All tests were conducted on a 2.6 GHz Intel Core i5 processor, with a maximum available RAM of 8GB.

We present six tables of comparative results—Table 1 for the continuous case, and Tables 2, 3, 4, 5, 6 and 7 for the discrete case, one for each unique pair of U and t values. In all of these tables, the maximum number of sublots,$\overline{n}$, is fixed at half of the lot size, U, which is fixed at 1000 for the continuous case, and it takes the value of either 100 or 1000 for the discrete case. For both cases, the value of t is fixed at either 0.20, 1.00, or 5.00 secs, whereas the value of p is varied. A tolerance value of $\tau = 0.1\%$ is set for the optimality gap for the direct solution method CPLEX® as well as for B&BH. As such, a test run was terminated either when the allowable CPU time limit of 1800.00 secs was reached, or when the lower bound on the makespan value and the best incumbent solution's makespan value fell within the tolerance value set for the optimality gap.

For the continuous case, we observe that the best makespan values obtained by the direct solution method and LSA are comparable in almost all the test cases. However, the striking difference is in the time taken by the two methods before termination. In all the test cases, LSA achieves an optimal solution in a matter of milliseconds, amounting to nearly a 100.00% drop compared to the time taken by the direct solution method.

For the discrete case, first, we make remarks on the general trends observed for the $1+m$ TSHFS-LSP based on the results presented in Tables 2, 3, 4, 5, 6 and 7: (1) For a given U and m, the computational difficulty (time taken to achieve optimality within the prescribed tolerance value) increases for both methods either with an increment in the value of p for a fixed t value, or with a decrement in the value of t for a fixed p value. This is attributed to

Table 1 Results for the 1 + m TSHFS-LSP—continuous sublot sizes

t	m	p	Direct solution by CPLEX®					LSA				% drop with LSA	
			Time (sec)	$\widehat{\mathcal{M}}(U)$	$\hat{n}(U)$	$\hat{m}(U)$	gap%	time (millisec)	$\widehat{\mathcal{M}}(U)$	$\hat{n}(U)$	$\hat{m}(U)$	time	Makespan
0.2	2	0.2	5.2	1001.1	5	2	0.1	**3.1**	1001.1	5	2	99.9	0.0
		1.0	221.6	1003.1	15	2	0.1	**1.6**	1003.1	15	2	100.0	0.0
		5.0	1800.0*	2500.5	35	2	59.9	**1.8**	2500.5	35	2	100.0	0.0
	5	0.2	81.1	1001.2	6	4	0.1	**3.1**	1001.2	6	4	100.0	0.0
		1.0	1800.0*	1002.5	12	5	0.2	**2.1**	1002.5	12	5	100.0	0.0
		5.0	1800.0*	1040.2	138	5	3.8	**3.1**	1040.2	138	5	100.0	0.0
1.0	2	0.2	5.1	1004.0	4	2	0.1	**4.8**	1004.0	4	2	99.9	0.0
		1.0	72.8	1012.1	12	2	0.1	**2.0**	1012.1	12	2	100.0	0.0
		5.0	1800.0*	2502.5	46	2	59.6	**3.3**	2502.5	74	2	100.0	0.0
	5	0.2	71.3	1004.6	4	4	0.1	**3.3**	1004.6	4	4	100.0	0.0
		1.0	1800.0*	1010.0	10	5	0.6	**3.6**	1010.0	10	5	100.0	0.0
		5.0	1800.0*	1082.7	82	5	7.2	**2.7**	**1078.6**	78	5	100.0	0.4
5.0	2	0.2	4.6	1018.8	3	2	0.0	**3.6**	1018.8	3	2	99.9	0.0
		1.0	50.1	1044.3	8	2	0.0	**1.9**	1044.3	8	2	100.0	0.0
		5.0	1800.0*	2512.5	36	2	56.0	**2.0**	2512.5	63	2	100.0	0.0
	5	0.2	52.1	1018.7	3	3	0.1	**2.9**	1018.7	3	3	99.9	0.0
		1.0	1304.9	1038.3	7	5	0.0	**2.8**	1038.3	7	5	100.0	0.0
		5.0	1800.0*	1180.8	36	5	12.4	**1.9**	1179.0	35	5	100.0	0.2

*For these cases, the direct solution method was forced to terminate early with optimality gap > 0:1%. Among both the methods, the lower values of best makespan and time before termination are marked in bold

Table 2 Results for the 1 + m TSHFS-LSP—discrete sublot sizes ($U = 100$; $t = 0.20$ secs)

Row	M	p	Direct solution by CPLEX®					B&BH					% drop with B&BH	
			time (sec)	$\widehat{\mathcal{M}}(U)$	$\hat{n}(U)$	$\hat{m}(U)$	Gap% (L.B.)	time (sec)	$\widehat{\mathcal{M}}(U)$	$\hat{n}(U)$	$\hat{m}(U)$	gap% (L.B.)	time	makespan
1	2	0.2	1.0	101.0	4	1	0.1 (100.9)	**1.1**	101.0	4	2	0.0 (101.0)	– 14.3	0.0
2	2	0.6	1.5	101.8	6	2	0.1 (101.7)	**1.4**	101.8	6	2	0.0 (101.8)	2.7	0.0
3	2	1.0	15.5	102.8	9	2	0.1 (102.7)	**1.5**	102.8	9	2	0.1 (102.7)	90.1	0.0
4	2	3.0	**20.0***	152.4*	26	2	32.6 (102.8)	22.5	152.4	21	2	0.0 (152.4)	98.8	0.0
5	2	5.0	**15.0***	252.4*	32	2	59.2 (103.0)	33.2	252.4	12	2	0.1 (252.2)	98.2	0.0
6	5	0.2	11.0	**101.0**	4	1	0.0 (101.0)	**3.6**	101.2	3	2	0.0 (101.2)	66.9	– 0.2
7	5	0.6	48.0*	102.0*	6	2	0.2 (101.6)	**4.0**	102.0	5	5	0.1 (102.0)	99.8	– 0.2
8	5	1.0	**2.3***	**102.4***	7	4	1.2 (101.2)	**21.1**	102.6	8	5	0.1 (102.5)	98.8	– 0.2
9	5	3.0	700.0*	106.8*	17	5	5.6 (100.8)	**73.6**	**106.6**	16	5	0.1 (106.5)	95.9	0.2
10	5	5.0	1748.0*	116.0*	48	5	12.8 (101.2)	**254.0**	**115.2**	37	5	0.0 (115.2)	85.9	0.7
11	10	0.2	174.0	**101.0**	4	1	0.1 (100.9)	**15.6**	101.2	3	3	0.1 (101.1)	91.1	– 0.2
12	10	0.6	**16.0***	102.0*	6	2	1.2 (100.6)	**19.1**	102.0	5	5	0.1 (101.9)	98.9	– 0.2
13	10	1.0	**60.0***	**102.4***	7	4	1.8 (100.6)	**66.0**	102.6	7	7	0.1 (102.5)	96.3	– 0.2
14	10	3.0	1600.0*	106.0*	12	7	4.9 (100.8)	**300.3**	106.0	13	10	0.1 (105.7)	83.3	0.2
15	10	5.0	1632.0*	109.2*	21	10	7.3 (101.2)	**612.6**	**109.0**	20	10	0.1 (108.9)	66.0	0.2

*For these cases, (i) a test run was forced to terminate at 1800.00 secs, (ii) the optimality gap at termination was > 0.1%, (iii) the values of makespan $\hat{M}(U)$, $\hat{n}(U)$, and $\hat{m}(U)$ reported belong to the best incumbent solution, and time (secs) represents the time when the best incumbent solution was observed during a run, and (iv) the optimality gap (%) and the lower bound value reported at termination point. Among both the methods, the lower values of best makespan and time before termination are marked in bold

Table 3 Results for the 1 + m TSHFS-LSP—discrete sublot sizes ($U = 100$; $t = 1.00$ secs)

Row	M	p	Direct solution by CPLEX®					B&BH					% Drop with B&BH	
			time (sec)	$\widehat{\mathcal{M}}(U)$	$\hat{n}(U)$	$\hat{m}(U)$	Gap% (L.B.)	time (sec)	$\widehat{\mathcal{M}}(U)$	$\hat{n}(U)$	$\hat{m}(U)$	Gap% (L.B.)	Time	Makespan
1	2	0.2	0.5	103.4	3	2	0.0 (103.4)	**0.3**	103.4	3	2	0.0 (103.4)	32.6	0.0
2	2	0.6	0.8	105.6	5	2	0.0 (105.6)	**0.5**	105.6	5	2	0.0 (105.6)	30.7	0.0
3	2	1.0	0.7	108.0	7	2	0.0 (108.0)	**0.8**	108.0	6	2	0.0 (108.0)	– 8.5	0.0
4	2	3.0	11.0*	157.0*	42	2	23.4 (120.2)	**6.2**	157.0	15	2	0.1 (156.9)	99.7	0.0
5	2	5.0	**1.5***	254.0*	12	2	49.4 (128.5)	12.2	254.0	12	2	0.1 (253.8)	99.3	0.0
6	5	0.2	5.4	103.4	3	2	0.1 (103.3)	**1.9**	103.4	3	3	0.1 (103.3)	65.1	0.0
7	5	0.6	80.2	105.6	5	2	0.1 (105.5)	**1.9**	105.6	5	5	0.1 (105.6)	97.6	0.0
8	5	1.0	281.5	**107.0**	6	3	0.1 (106.9)	**8.6**	108.0	5	5	0.0 (108.0)	97.0	– 0.9
9	5	3.0	56.0*	117.0*	12	5	6.9 (108.0)	**22.1**	**116.0**	12	5	0.0 (116.0)	98.8	0.9
10	5	5.0	1610.0*	131.0*	18	5	17.6 (108.0)	**49.6**	**128.0**	22	5	0.1 (127.9)	97.3	2.3
11	10	0.2	17.1	103.4	3	2	0.0 (103.4)	**6.8**	103.4	3	3	0.0 (103.4)	60.2	0.0
12	10	0.6	**6.0***	105.6*	5	2	1.5 (104.0)	8.9	105.6	5	5	0.0 (105.6)	99.5	0.0
13	10	1.0	190.0*	107.0*	6	3	3.7 (103.0)	**33.3**	107.0	6	6	0.1 (106.9)	98.2	0.0
14	10	3.0	308.0*	115.0*	11	7	8.7 (105.0)	**148.7**	115.0	11	10	0.1 (114.9)	91.7	0.0
15	10	5.0	1670.0*	126.0*	21	10	16.7 (105.0)	**246.7**	123.0	15	10	0.0 (123.0)	86.3	2.4

*For these cases, (i) a test run was forced to terminate at 1800.00 secs, (ii) the optimality gap at termination was > 0.1%, (iii) the values of makespan $\hat{M}(U)$, $\hat{n}(U)$, and $\hat{m}(U)$ reported belong to the best incumbent solution, and time (secs) represents the time when the best incumbent solution was observed during a run, and (iv) the optimality gap (%) and the lower bound value reported at termination point. Among both the methods, the lower values of best makespan and time before termination are marked in bold

Table 4 Results for the 1 + m TSHFS-LSP—discrete sublot sizes ($U = 100$; $t = 5.00$ secs)

Row	M	p	Direct solution by CPLEX®					B&BH					% drop with B&BH	
			time (sec)	$\widehat{\mathcal{M}}(U)$	$\hat{n}(U)$	$\hat{m}(U)$	Gap% (L.B.)	time (sec)	$\widehat{\mathcal{M}}(U)$	$\hat{n}(U)$	$\hat{m}(U)$	Gap% (L.B.)	Time	Makespan
1	2	0.2	0.3	112.0	2	2	0.0 (112.0)	**0.2**	112.2	2	2	0.1 (112.1)	33.3	– 0.2
2	2	0.6	0.5	119.0	3	2	0.0 (119.0)	**0.3**	119.0	3	2	0.1 (118.9)	36.7	0.0
3	2	1.0	0.5	125.0	4	2	0.0 (125.0)	**0.5**	125.0	4	2	0.1 (124.9)	1.9	0.0
4	2	3.0	50.7	173.0	14	2	0.1 (172.8)	**3.8**	173.0	11	2	0.0 (173.0)	92.5	0.0
5	2	5.0	12.0*	264.0*	21	2	4.0 (253.5)	**8.1**	264.0	11	2	0.1 (263.8)	99.6	0.0
6	5	0.2	6.3	112.2	2	2	0.0 (112.2)	**1.2**	112.2	2	2	0.1 (112.1)	81.0	0.0
7	5	0.6	4.8	118.6	3	3	0.0 (118.6)	**1.5**	118.6	3	3	0.1 (118.5)	69.5	0.0
8	5	1.0	7.5	123.0	4	4	0.0 (123.0)	**3.8**	123.0	4	4	0.1 (122.9)	49.9	0.0
9	5	3.0	408.1	**143.0**	8	5	0.1 (142.9)	**18.3**	144.0	7	5	0.1 (143.9)	95.5	– 0.7
10	5	5.0	70.0*	164.0*	11	5	14.1 (140.0)	**31.2**	**163.0**	11	5	0.1 (162.9)	98.3	0.6
11	10	0.2	27.5	112.2	2	2	0.0 (112.2)	**4.0**	112.2	2	2	0.0 (112.2)	85.6	0.0
12	10	0.6	13.7	118.6	3	3	0.0 (118.6)	**5.9**	118.6	3	3	0.1 (118.5)	57.0	0.0
13	10	1.0	54.7	123.0	4	4	0.0 (123.0)	**20.0**	123.0	4	4	0.1 (122.9)	63.4	0.0
14	10	3.0	**15.0***	143.0*	8	5	9.8 (129.0)	102.0	143.0	8	8	0.1 (142.9)	94.3	0.0
15	10	5.0	243.0*	156.0*	10	10	16.7 (130.0)	163.5	156.0	10	10	0.0 (155.9)	90.9	0.0

*For these cases, (i) a test run was forced to terminate at 1800.00 secs, (ii) the optimality gap at termination was > 0.1%, (iii) the values of makespan $\hat{M}(U)$, $\hat{n}(U)$, and $\hat{m}(U)$ reported belong to the best incumbent solution, and time (secs) represents the time when the best incumbent solution was observed during a run, and (iv) the optimality gap (%) and the lower bound value reported at termination point. Among both the methods, the lower values of best makespan and time before termination are marked in bold

Table 5 Results for the 1 + m TSHFS-LSP—discrete sublot sizes ($U = 1000$; $t = 0.20$ secs)

Row	M	p	Direct solution by CPLEX®					B&BH					% drop with B&BH	
			Time (sec)	$\widehat{\mathcal{M}}(U)$	$\hat{n}(U)$	$\hat{m}(U)$	Gap% (L.B.)	Time (sec)	$\widehat{\mathcal{M}}(U)$	$\hat{n}(U)$	$\hat{m}(U)$	Gap% (L.B.)	Time	Makespan
1	2	0.2	**8.8**	1001.2	5	2	0.1 (1000.6)	18.4	1001.6	4	2	0.1 (1000.9)	– 109.6	0.0
2	2	0.6	89.0	1002.4	9	2	0.1 (1001.5)	**23.3**	1002.8	8	2	0.0 (1002.5)	73.9	0.0
3	2	1.0	783.0	1003.8	14	2	0.1 (1002.8)	**39.8**	1003.6	13	2	0.0 (1003.2)	94.9	0.0
4	2	3.0	927.0*	**1502.4***	90	2	33.2 (1004.2)	**311.5**	1504.0	22	2	0.0 (1503.7)	82.7	– 0.1
5	2	5.0	**142.0***	2502.4*	73	2	59.9 (1001.6)	602.2	2502.4	17	2	0.1 (2500.2)	66.5	0.0
6	5	0.2	176.0	1001.4	6	2	0.1 (1000.5)	**91.2**	1001.6	4	4	0.0 (1001.3)	48.2	0.0
7	5	0.6	509.0*	1002.2*	8	4	0.1 (1000.8)	**134.7**	1002.2	8	5	0.0 (1002.0)	92.5	0.0
8	5	1.0	645.0*	**1003.2***	11	5	0.3 (1000.6)	**282.7**	1004.2	9	5	0.0 (1003.9)	84.3	– 0.1
9	5	3.0	**1101**.0*	1013.8*	53	5	75.8 (245.6)	1378.1	**1011.8**	25	5	0.1 (1010.8)	23.4	0.2
10	5	5.0	1789.0*	1124.4*	79	5	99.1 (9.8)	**1440.0***	**1051.2***	83	5	0.8 (1043.3)	0.0	6.5
11	10	0.2	830.0	1001.2	5	3	0.1 (1000.4)	**332.3**	1001.6	4	4	0.1 (1001.1)	60.0	0.0
12	10	0.6	1123.0*	1002.6*	10	3	0.2 (1000.4)	**443.9**	1002.6	7	7	0.1 (1001.6)	75.3	0.0
13	10	1.0	**869.0***	1004.2*	16	10	0.4 (1000.4)	1489.3	**1003.4**	10	10	0.1 (1002.4)	17.3	0.1
14	10	3.0	**897.0***	1151.6*	346	6	99.9 (1.2)	1207.0*	**1010.4***	18	10	0.8 (1002.5)	0.0	12.3
15	10	5.0	**436.0***	1050.2*	101	10	99.2 (8.6)	1335.0*	**1013.6***	33	10	1.2 (1001.4)	0.0	3.5

*For these cases, (i) a test run was forced to terminate at 1800.00 secs, (ii) the optimality gap at termination was > 0.1%, (iii) the values of makespan $\hat{M}(U)$, $\hat{n}(U)$, and $\hat{m}(U)$ reported belong to the best incumbent solution, and time (secs) represents the time when the best incumbent solution was observed during a run, and (iv) the optimality gap (%) and the lower bound value reported at termination point. Among both the methods, the lower values of best makespan and time before termination are marked in bold

Table 6 Results for the 1 + m TSHFS-LSP—discrete sublot sizes ($U = 1000; t = 1.00$ secs)

Row	M	p	Direct solution by CPLEX®					B&BH					% drop with B&BH	
			time (sec)	$\widehat{\mathcal{M}}(U)$	$\hat{n}(U)$	$\hat{m}(U)$	Gap% (L.B.)	Time (sec)	$\widehat{\mathcal{M}}(U)$	$\hat{n}(U)$	$\hat{m}(U)$	Gap% (L.B.)	Time	Makespan
1	2	0.2	**6.4**	1004.8	4	2	0.1 (1003.9)	12.6	1004.8	4	2	0.0 (1004.7)	− 96.4	0.0
2	2	0.6	**13.0**	1008.2	7	2	0.1 (1007.3)	22.3	1008.2	7	2	0.1 (1007.2)	− 71.9	0.0
3	2	1.0	87.0	1013.0	12	2	0.1 (1012.1)	**27.2**	1013.0	10	2	0.1 (1012.1)	68.7	0.0
4	2	3.0	1675.0*	1567.0*	29	2	33.1 (1009.0)	**190.5**	**1507.0**	23	2	0.1 (1505.6)	89.4	3.8
5	2	5.0	**196.0***	2504.0*	41	2	59.7 (1010.0)	394.9	2504.0	17	2	0.0 (2503.8)	78.1	0.0
6	5	0.2	225.0	1004.6	4	3	0.1 (1003.8)	**62.4**	1004.8	4	4	0.1 (1004.2)	72.3	0.0
7	5	0.6	1359.0	**1007.6**	7	4	0.1 (1006.7)	**88.6**	1008.4	6	5	0.1 (1007.7)	93.5	− 0.1
8	5	1.0	698.0*	1011.0*	10	4	0.8 (1003.0)	**202.0**	1011.0	9	5	0.1 (1010.0)	88.8	0.0
9	5	3.0	1740.0*	1041.0*	24	5	99.3 (7.0)	**1012.8**	**1028.0**	22	5	0.1 (1027.4)	43.7	1.3
10	5	5.0	1661.0*	1283.0*	258	5	71.3 (368.8)	**1603.2**	**1089.0**	53	5	0.0 (1088.6)	10.9	15.1
11	10	0.2	578.0*	1004.6*	4	4	0.3 (1002.0)	**203.4**	1004.8	4	4	0.1 (1004.3)	88.7	0.0
12	10	0.6	470.0*	1007.6*	7	7	0.6 (1002.0)	**288.0**	1007.8	7	7	0.1 (1007.1)	84.0	0.0
13	10	1.0	1716.0*	1011.0*	9	8	0.9 (1002.0)	**1200.8**	**1010.0**	9	9	0.1 (1009.3)	33.3	0.1
14	10	3.0	**923.0***	1031.0*	24	10	2.4 (1005.0)	1026.0*	**1023.0***	17	10	0.5 (1017.7)	0.0	0.8
15	10	5.0	1780.0*	1144.0*	47	8	90.1 (113.5)	**887.0***	**1041.0***	26	10	0.6 (1034.7)	0.0	9.0

*For these cases, (i) a test run was forced to terminate at 1800.00 secs, (ii) the optimality gap at termination was > 0.1%, (iii) the values of makespan $\hat{M}(U)$, $\hat{n}(U)$, and $\hat{m}(U)$ reported belong to the best incumbent solution, and time (secs) represents the time when the best incumbent solution was observed during a run, and (iv) the optimality gap (%) and the lower bound value reported at termination point. Among both the methods, the lower values of best makespan and time before termination are marked in bold

Table 7 Results for the 1 + m TSHFS-LSP—discrete sublot sizes ($U = 1000$; $t = 5.00$ secs)

Row	M	p	Direct solution by CPLEX®					B&BH					% drop with B&BH	
			Time (sec)	$\widehat{\mathcal{M}}(U)$	$\hat{n}(U)$	$\hat{m}(U)$	Gap% (L.B.)	time (sec)	$\widehat{\mathcal{M}}(U)$	$\hat{n}(U)$	$\hat{m}(U)$	Gap% (L.B.)	Time	Makespan
1	2	0.2	16.7	**1019.0**	3	2	0.0 (1019.0)	**7.2**	1019.2	3	2	0.1 (1018.5)	56.8	0.0
2	2	0.6	23.0	**1030.2**	6	2	0.1 (1029.3)	**9.3**	1031.8	6	2	0.0 (1031.5)	59.4	– 0.2
3	2	1.0	71.0	1045.0	8	2	0.1 (1044.1)	**16.2**	1045.0	8	2	0.1 (1044.1)	77.1	0.0
4	2	3.0	610.0*	1525.0*	35	2	30.1 (1066.0)	**156.9**	1525.0	23	2	0.1 (1524.2)	91.3	0.0
5	2	5.0	**74.0***	2514.0*	15	2	57.6 (1065.0)	235.5	2514.0	14	2	0.1 (2512.7)	86.9	0.0
6	5	0.2	165.0	**1018.8**	3	3	0.0 (1018.8)	**34.2**	1019.4	3	3	0.0 (1019.0)	79.3	– 0.1
7	5	0.6	225.0	1029.8	5	5	0.1 (1028.9)	**44.9**	1029.8	5	5	0.0 (1029.7)	80.0	0.0
8	5	1.0	1735.0	1039.0	7	4	0.1 (1038.1)	**112.7**	1039.0	7	5	0.1 (1038.4)	93.5	0.0
9	5	3.0	1645.0*	1088.0*	17	5	5.3 (1030.0)	**590.1**	1088.0	15	5	0.1 (1087.0)	67.2	0.0
10	5	5.0	1756.0*	1215.0*	27	5	15.4 (1027.4)	**966.1**	**1184.0**	33	5	0.1 (1182.9)	46.3	2.6
11	10	0.2	1638.0	**1018.8**	3	3	0.1 (1018.0)	**120.4**	1019.4	3	3	0.1 (1018.4)	92.7	– 0.1
12	10	0.6	896.0*	1029.8*	6	6	1.9 (1010.0)	**177.4**	1029.8	5	5	0.1 (1028.8)	90.2	0.0
13	10	1.0	1562.0*	1039.0*	7	7	2.3 (1015.0)	**502.5**	1039.0	7	7	0.0 (1038.9)	72.1	0.0
14	10	3.0	987.0*	1174.0*	34	5	99.5 (5.5)	**201.0***	**1077.0***	14	10	0.2 (1074.4)	0.0	8.3
15	10	5.0	1792.0*	1291.0*	28	6	99.6 (6.0)	**1007.8***	**1115.0***	20	10	0.4 (1111.0)	0.0	13.6

*For these cases, (i) a test run was forced to terminate at 1800.00 secs, (ii) the optimality gap at termination was > 0.1%, (iii) the values of makespan $\hat{M}(U)$, $\hat{n}(U)$, and $\hat{m}(U)$ reported belong to the best incumbent solution, and time (secs) represents the time when the best incumbent solution was observed during a run, and (iv) the optimality gap (%) and the lower bound value reported at termination point. Among both the methods, the lower values of best makespan and time before termination are marked in bold

an increment in the value of the ratio p/t, which, in turn, has a direct correlation with the number of sublots, $\hat{n}(U)$, specified in an optimal schedule, and the number of machines, $\hat{m}$ (U), used for allocating these sublots in Stage 2,$\hat{m}$. (2) For a given U and m, the makespan value, $\mathcal{M}(U)$, corresponding to an optimal schedule increases for both methods either with an increment in the value of p for a fixed t value, or with an increment in the value of t for a fixed p-value, as expected. It also decreases with an increment in the number of machines available in Stage 2, m, for a given U, p and t. Regarding comparative performances of the two methods: (i) B&BH is able to achieve optimality in 83 out of the 90 total cases tested. In the remaining 7 cases, it terminates with the maximum optimality gap of only 1.20% from its lower bound value. Even so, in such cases, it reports a better makespan value than that for the direct solution method. (ii) The direct solution method is not able to achieve optimality in 50 out of the 90 cases tested, with the maximum optimality gap of as high as 99.89% (note that the direct solution method struggles with ramping up the lower bound value for higher values of any combination of U, m, p, or $1/t$ in general). Still, in 20 such cases, the makespan value matches that obtained using the B&BH, and in only 8 of these cases, it reports an improvement of 0.13% on an average over B&BH in makespan value (note that even though the B&BH achieved optimality within the set tolerance of 0.1% in all of these 8 cases, the lower bound reported for the B&BH is slightly higher than the makespan value observed for the direct solution method in 6 out of these 8 cases, which is due to the fact that B&BH is actually a heuristic method, therefore the lower bounds observed for the two methods are not directly comparable). However, in the remaining 22 cases, the B&BH reports a smaller makespan value compared with that obtained by the direct solution method by an average of 3.84% with a maximum as high as 15.12%. In fact, the standard deviation in the improvement is 4.67% which points to a strong positive skewness in the distribution of makespan improvement percentage for these 22 cases. (iii) For all the 90 test cases, the B&BH reports an improvement in makespan value of 0.90% on average compared with that obtained by the direct solution method. Even though, this improvement in the makespan value is not relatively large, the difference in time required before termination by both methods is a clear indication of the superiority of the B&BH, where the CPLEX requires close to 1095.00 secs compared to 315.00 secs for the B&BH on average.

5 Concluding remarks

In this paper, we addressed scheduling of a production lot over a two-stage HFS with $1 + m$ configuration using lot-streaming, for the objective of minimizing the makespan. A HFS configuration is encountered in a variety of practical situations such as continuous processing industries, flexible manufacturing systems and parallel computing environments. We have generalized the work of Cheng et al. [2] to an $1 + m$ HFS. The novelty of our work is in obtaining an optimal schedule in polynomial time, $\mathcal{O}\left(m^{2.376} + mn\right)$, where n is the specified number of sublots the lot is to be split into for scheduling over m machines in Stage 2, for the case when the sublot sizes are relaxed to be continuous. A branch-and-bound-based heuristic is also developed for the case in which sublot sizes are discrete; it relies on the use of a tight lower bound on makespan. Its efficacy is revealed after testing its performance against that of the direct solution of a MIP formulation for the $1+m$ TSHFS-LSP by CPLEX®. The former is able to obtain solutions within a 0.10% optimality gap in 83 out of 90 instances, and requires an average of 315.00 secs before termination, whereas the latter obtains solutions within 0.10% optimality gap in only 40 out of 90 instances, and requires as much as 1095.00 s on average before termination. The results clearly show the superiority of our proposed method over the direct solution approach.

Having analyzed the application of lot streaming to a $1 + m$ hybrid flow shop problem, a potential direction for further research includes consideration of more than one parallel machine in Stage 1, both for the case of continuous and discrete sublot sizes. In particular, it would be interesting to investigate if the alternate assignment rule applies to the Stage 1 machines as well. Another interesting situation to consider is the one obtained by relaxing the assumption of parallel machines in Stage 2 and instead considering uniform or unrelated machines. In the same vein, this situation can be considered for the machines in Stage 1 as well.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s10898-023-01354-0.

References

1. Carpov, S., Carlier, J., Nace, D., Sirdey, R.: Two-stage hybrid flow shop with precedence constraints and parallel machines at second stage. Comput. Oper. Res. **39**(3), 736–745 (2012)
2. Cheng, M., Sarin, S.C., Singh, S.: Two-stage, single-lot, lot streaming problem for a 1+2 hybrid flow shop. J. Global Optim. **66**, 263–290 (2016)
3. Coppersmith, D., Winograd, S.: Matrix multiplication via arithmetic progressions. J. Symb. Comput. **9**(3), 251–280 (1990)
4. Devor, R., Graves, R., Mills, J.J.: Agile manufacturing research: accomplishments and opportunities. IIE Trans. **29**(10), 813–823 (1997)
5. Gholami, M., Zandieh, M., Alem-Tabriz, A.: Scheduling hybrid flow shop with sequence-dependent setup times and machines with random breakdowns. Int. J. Adv. Manuf. Technol. **42**(1–2), 189–201 (2009)
6. Gupta, J.N.: Two-stage, hybrid flowshop scheduling problem. J. Oper. Res. Soc. **39**(4), 359–364 (1988)
7. Gupta, J.N., Tunc, E.A.: Schedules for a two-stage hybrid flowshop with parallel machines at the second stage. Int. J. Prod. Res. **29**(7), 1489–1502 (1991)
8. Gupta, J.N., Tunc, E.A.: Scheduling a two-stage hybrid flowshop with separable setup and removal times. Eur. J. Oper. Res. **77**(3), 415–428 (1994)
9. Hoogeveen, J., Lenstra, J.K., Veltman, B.: Preemptive scheduling in a two-stage multiprocessor flow shop is np-hard. Eur. J. Oper. Res. **89**(1), 172–175 (1996)
10. Jin, Z., Ohno, K., Ito, T., Elmaghraby, S.E.: Scheduling hybrid flow shops in printed circuit board assembly lines. Prod. Oper. Manag. **11**(2), 216–230 (2002)
11. Jungwattanakit, J., Reodecha, M., Chaovalitwongse, P., Werner, F.: Algorithms for flexible flow shop problems with unrelated parallel machines, setup times, and dual criteria. Int. J. Adv. Manuf. Technol. **37**(3–4), 354–370 (2008)
12. Kalir, A.A., Sarin, S.C.: Evaluation of the potential benefits of lot streaming in flow shop systems. Int. J. Prod. Econ. **66**, 131–142 (2000)
13. Kusiak, A.: Aggregate scheduling of a flexible machining and assembly system. IEEE Trans. Robot. Autom. **5**(4), 451–459 (1989)
14. Li, S.: A hybrid two-stage flow shop with part family, batch production, major and minor set-ups. Eur. J. Oper. Res. **102**(1), 142–156 (1997)
15. Liu, J.: Single-job lot streaming in m- 1 two-stage hybrid flow shops. Eur. J. Oper. Res. **187**(3), 1171–1183 (2008)

16. Ribas, I., Leisten, R., Framiñan, J.M.: Review and classification of hybrid flow shop scheduling problems from a production system and a solutions procedure perspective. Comput. Oper. Res. **37**(8), 1439–1454 (2010)
17. Ruiz, R., Vázquez-Rodríguez, J.A.: The hybrid flow shop scheduling problem. Eur. J. Oper. Res. **205**(1), 1–18 (2010)
18. Sedgewick, R., Wayne, K.: Algorithms (Deluxe): Book and 24-Part Lecture Series. Addison-Wesley Professional (2015)
19. Shao, W., Shao, Q., Pi, D.: Modeling and optimization of distributed heterogeneous hybrid flow shop lot streaming scheduling problem. Expert Syst. Appl. **214**, 119151 (2023)
20. Sriskandarajah, C., Sethi, S.P.: Scheduling algorithms for flexible flow shops: worst and average case performance. Eur. J. Oper. Res. **43**(2), 143–160 (1989)
21. Tang, L., Liu, W., Liu, J.: A neural network model and algorithm for the hybrid flow shop scheduling problem in a dynamic environment. J. Intell. Manuf. **16**(3), 361–370 (2005)
22. Tsubone, H., Ohba, M., Uetake, T.: The impact of lot sizing and sequencing on manufacturing performance in a two-stage hybrid flow shop. Int. J. Prod. Res. **34**(11), 3037–3053 (1996)
23. Vickson, R.: Optimal lot streaming for multiple products in a two-machine flow shop. Eur. J. Oper. Res. **85**(3), 556–575 (1995)
24. Wittrock, R.J.: An adaptable scheduling algorithm for flexible flow lines. Oper. Res. **36**(3), 445–453 (1988)
25. Yaurima-Basaldua, V., Tchernykh, A., Villalobos-Rodr´ıguez, F., Salomon-Torres, R.: Hybrid flow shop with unrelated machines, setup time, and work in progress buffers for bi-objective optimization of tortilla manufacturing. Algorithms **11**(5), 68 (2018)
26. Zhang, B., Pan, Q., Meng, L.L., Zhang, X.L., Ren, Y.P., Li, J.Q., Jiang, S.C.: A collaborative variable neighborhood descent algorithm for the hybrid flow shop scheduling problem with consistent sublots. Appl. Soft Comput. **106**, 107305 (2021)
27. Zhang, W., Liu, J., Linn, R.J.: Model and heuristics for lot streaming of one job in m+1 hybrid flow shop. Int. J. Oper. Quant. Manag. **9**(1), 49–64 (2003)
28. Zhang, W., Yin, C., Liu, J., Linn, R.J.: Multi-job lot streaming to minimize the mean completion time in m+1 hybrid flow shop. Int. J. Prod. Econ. **96**(2), 189–200 (2005)