Trade-off Balancing between Maximum and Total Completion Times for No-wait Flow Shop Production

Honghan Ye[a], Wei Li[b] and Barrie R. Nault[c]

[a]Department of Industrial & Systems Engineering, University of Wisconsin-Madison, Madison, WI, USA; [b]Department of Mechanical Engineering, University of Kentucky, Lexington, KY, USA; [c]Haskayne School of Business, University of Calgary, Calgary, AB, Canada

ARTICLE HISTORY
Compiled May 26, 2019

ABSTRACT
We propose a trade-off balancing (TOB) heuristic in a no-wait flow shop to minimise the weighted sum of maximum completion time (C_{max}) and total completion time (TCT) based on machine idle times. We introduce a factorization scheme to construct the initial sequence based on current and future idle times at the operational level. In addition, we propose a novel estimation method to establish the mathematical relationship between the objectives min(C_{max}) and min(TCT) at the production line level. To evaluate the performance of the TOB heuristic, computational experiments are conducted on the classic Taillard's benchmark and one-year historical data from University of Kentucky HealthCare (UKHC). The computational results show that minimisations of C_{max} and TCT yield inconsistent scheduling sequences, and these two sequences are relatively uncorrelated. We also show that our TOB heuristic performs better than the best existing heuristics with the same computational complexity and generates stable performances in balancing trade-offs.

KEYWORDS
Scheduling; No-wait flow shop; Trade-off balancing; Maximum completion time; Total completion time.

1. Introduction

No-wait flow shops are critical in manufacturing systems, such as bakery process (Shao, Pi, and Shao 2019), metal processing (Framinan and Nagano 2008) and semiconductor manufacturing (Chien et al. 2011). In no-wait flow shop scheduling, all n jobs are processed in the same order on all m machines, and no job is allowed to wait between any two consecutive operations until the process is finished. Therefore, the start time of a job on the first machine may be delayed due to the no-wait constraint. For more details about applications of no-wait flow shop scheduling, see Allahverdi (2016).

There are two fundamental performance measures in no-wait flow shop scheduling: maximum completion time (C_{max}) and total completion time (TCT). C_{max}, or makespan, is defined as the completion time of the last job on the last machine, which reflects utilisation of the production line or production cost of the system. TCT is defined as the sum of completion times of all jobs on the last machine, which reflects material flows or holding cost in the system. Minimising makespan or min(C_{max}) can

CONTACT Wei Li. Email: wei.mike.li@uky.edu

improve the utilisation and reduce production cost, and minimising total completion time or $\min(TCT)$ can smooth material flows and reduce holding cost. Therefore, both performance measures are of great importance to operations management and have been intensively studied in the literature.

However, to $\min(C_{max})$ is NP-hard in the strong sense for a line with ≥ 3 machines (Röck 1984), and to $\min(TCT)$ is also strongly NP-hard (Johnson and Garey 1979) in no-wait flow shop scheduling. Due to the NP-hardness in no-wait flow shop scheduling, it is extremely time-consuming to seek optimal solutions by using exact methods such as branch-and-bound algorithms even for moderate size problems. Therefore, it is practical to seek near-optimal solutions by using heuristics with acceptable computation times. Instead of single-objective optimizations, balancing trade-offs among multi-objectives is suitable in many different applications, such as trade-offs between queue time and utlisation in manufacturing systems with multiple servers (Wu and McGinnis 2012) or between cost and time of clinical trials in healthcare systems (Zhao, Wu, and Huang 2018). In no-wait flow shop scheduling, there is a gap in the current literature whereby most articles consider trade-off balancing at the production line level, but not at the operational level, which destabilize system performances. More specifically, the production line level refers to performance of each job on the last machine, such as C_{max}, whereas the operational level refers to performance of each job on each machine, such as completion times of each job on each machine. In our work, we sequence jobs based on completion times of job j on machine i ($C_{j,i}$, where $j = 1, ..., n$ and $i = 1, ..., m$) at the operational level, which reveals more information about the system than the C_{max} or TCT at the production line level.

Considering the importance of bi-objective optimisation and its difficulties, we propose a trade-off balancing (TOB) heuristic to minimise trade-offs between C_{max} and TCT in no-wait flow shop scheduling. We make three contributions. First, we show the trade-offs between C_{max} and TCT in no-wait flow shop scheduling through our computational experiments, which means minimisation of one objective does not necessarily minimise the other. Second, we take four different factors into consideration at the operational level to construct the initial sequence in our TOB heuristic. The four factors are objectives, idle time, lever effect on jobs and lever effect on machines. We find the best combination for each objective and use different preferences (α) to combine these two objectives together at operational level to construct the initial sequence. Third, in the improvement phase of our TOB heuristic, we use the same preference (α) on C_{max} to reconstruct a sequence based on C_{max} and TCT at the production line level. To tackle different magnitudes of C_{max} and TCT, we propose an estimation method to normalise the impact of different magnitudes.

We test the performance of our TOB heuristic on both simulated and real data sets: 120 instances in Taillard's benchmarks (Taillard 1993) and one-year historical data from University of Kentucky HealthCare (UKHC). Based on computational results on these data sets, we make the following observations. (1) After we consider the impact of trade-off balancing at the operational level, the TOB heuristic performs better than the Average Idle Time (AIT) and Current and Future Idle time (CFI) heuristics, which are the best single-objective heuristics in the literature for $\min(C_{max})$ and $\min(TCT)$, respectively. (2) Our TOB heuristic generates stable performances on balancing trade-offs based on our UKHC data set, which means the process is better under control if our TOB heuristic is used for operating room (OR) scheduling. (3) A trade-off between C_{max} and TCT exists in no-wait flow shop scheduling as evidenced by computing Spearman rank order correlations among the sequences for $\min(C_{max})$ and $\min(TCT)$ for both data sets. The correlations are close to zero and empirically demonstrate the inconsistency between $\min(C_{max})$ and $\min(TCT)$.

The rest of this paper is organised as follows. Section 2 provides literature review on heuristics of single-objective optimisation and trade-off balancing in no-wait flow shop scheduling. After formulating the problems in section 3, section 4 presents the programming logic and detailed steps of our TOB heuristic with an illustrative example. Section 5 provides the computational results and analysis. The conclusion and future work are discussed in section 6.

2. Literature Review

We first present the current status of heuristics for C_{max} and TCT minimisations, respectively, in no-wait flow shop scheduling problems. Afterwards, we provide our literature review on trade-off balancing related with no-wait flow shop scheduing problems. For further details about prior research, please see Ruiz and Allahverdi (2009) and Samarghandi and ElMekkawy (2012) for no-wait flow shop scheduling, and see Cortés, García, and Hernández (2012), Cohn et al. (2010) and Jungwattanakit et al. (2009) for trade-off balancing.

2.1. *Single-objective heuristics*

In general, meta-heuristics take long computation times to generate near-optimal solutions, which prevents their application in industry. We focus on the development of constructive heuristics, which provide good solutions with shorter computation times.

For no-wait flow shop scheduling problems to $\min(C_{max})$, based on the objective increment method, Li, Wang, and Wu (2008) proposed a composite heuristic (CH), and experimental results showed that the CH heuristic performed better than the GR heuristic by Gangadharan and Rajendran (1993) and RAJ heuristic by Rajendran (1994). Laha and Chakraborty (2009) proposed a constructive heuristic (LC) to solve the same problem. The computational results showed that the LC heuristic was significantly better than the GR, RAJ and two other compared heuristics. Ye, Li, and Miao (2016) proposed a constructive heuristic (the ADT heuristic) to minimise makespan, and their experimental results showed that the ADT heuristic performed better than the GR, RAJ and the modified NEH heuristics (Nawaz, Enscore, and Ham 1983). Recently, Ye, Li, and Abedini (2017) proposed an effective heuristic (the AIT heuristic) to $\min(C_{max})$ for no-wait flow shop scheduling problems. In the AIT heuristic, they treated current and future idle time differently to generate the initial sequence, and use insertion and neighborhood exchanging techniques to further improve the performance. Based on their case studies, the AIT heuristic performed better than the ADT, CH and LC heuristics.

For no-wait flow shop scheduling problems to $\min(TCT)$, Aldowaisan and Allahverdi (2004) proposed six improved heuristics by using three different search methods, first by the same insertion scheme as in the NEH heuristic, second by the same insertion technique as in Rajendran and Ziegler (1997), and third by the adjacent pair-wise neighborhood exchanging method. Among the six improved heuristics, the proposed heuristic with NEH-insertion and pair-wise exchange techniques (PH1(p)) performed significantly better than the RC heuristic proposed by Rajendran and Chaudhuri (1990) and the genetic algorithm proposed by Chen, Neppalli, and Aljaber (1996). Framinan, Nagano, and Moccellin (2010) proposed an FNM constructive heuristic to $\min(TCT)$, and the results of their case studies showed that the FNM heuristic performed better than the PH1(p) heurisitc, and the heuristics proposed by Bertolissi (2000), and by Fink and Voß (2003). In 2014, Laha, Gupta, and Sapkal (2014) proposed a penalty-shift-insertion-based (PSI) algorithm and the PSI algorithm performed better than the RC, PH1(p) heuristics, and the heuristic proposed by Bertolissi (2000). Meanwhile, Laha and Sapkal (2014) proposed an improved LS heuristic, and results showed that the LS heuristic performed better than the PH1(p) and FNM heuristics. More recently, Ye et al. (2017) proposed the current and future idletime (CFI) heuristic to min(TCT) in a no-wait flow shop. Their computational results showed that the CFI heuristic performed better than the PH1(p), FNM and LS heuristics.

Based on the literature review of single-objective heuristics, the AIT and CFI heuristics are currently the best-known heuristics to min(C_{max}) and min(TCT), respectively. Both AIT and CFI heuristics consider the current and future idle times together to construct the initial sequence, and they introduce the sequence lever effect to their heuristics which can generate better performances. However, neither of them takes machine lever effects into consideration in their heuristics, which is good to min(C_{max}) or min(TCT). Therefore, to systematically analyse the effects of different factors on the trade-off balancing, we introduce a factorization scheme to handle those different factors in section 4.1, including objectives, idle times, sequence lever and machine lever effects, and to find the best combination to min(C_{max}), min(TCT), and min(TO).

2.2. *Trade-off balancing between C_{max} and TCT*

The literauture on multi-objective combinatorial optimisation for no-wait flow shop scheduling is vast. Details about total tardiness and C_{max} minimisations can be found in Allahverdi, Aydilek, and Aydilek (2018), details about minimising both weighted mean completion time and weighted mean taridness can be found in Tavakkoli-Moghaddam, Rahimi-Vahed, and Mirzaei (2008), and details about C_{max} and maximum tardiness minimisation in no-wait flow shop scheduling can be found in Pan, Wang, and Qian (2009). To focus on the problem we study, we provide a limited review of bi-objectives for C_{max} and TCT. For trade-off balancing between C_{max} and TCT, Li, Mitchell, and Nault (2014) have proved the inconsistency between maximum and total completion time in permutation flow shop scheduling, which means minimisation of C_{max} does not necessarily minimise TCT, and vice versa. Dang (2017) proposed a current and future deviation heuristic to balance trade-offs beteen C_{max} and TCT in the general permutation flow shop. In no-wait flow shop scheduling, Aydilek and Allahverdi (2012) and Allahverdi and Aydilek (2013) used the constrained optimisation approach to minimise both C_{max} and TCT for no-wait flow shop scheduling. Specifically, Aydilek and Allahverdi (2012) considered min(C_{max}) subject to a mean completion time constraint. In contrast, Allahverdi and Aydilek (2013) considered min(TCT) subject to a C_{max} constraint. More recently, Allahverdi and Aydilek (2014) address the same problem as in Allahverdi and Aydilek (2013) except that the assumption of zero setup times is relaxed. The heuristic proposed by Allahverdi and Aydilek (2014) is significantly better than the heuristic of Allahverdi and Aydilek (2013) when the setup time is zero. However, such constrained optimisation approach has some fundamental limitations. For example, those constraints may affect the solution space where optimal solutions exist.

Apart from the constrained optimisation approach, the utility approach has been adopted by converting two different objectives to a single criterion. Allahverdi and Aldowaisan (2002) proposed the PAAH heuristic to address the problem of minimising maximum and total completion time by reducing the problem into a single criterion. In the PAAH heuristic, a trade-off function $TO = \alpha \cdot C_{max} + (1 - \alpha) \cdot TCT$ was developed, where α is the weight on C_{max} changing from 0 to 1, and the sequence with minimum TO is selected and attached to the current partial sequence. The PAAH heuristic performed better than the best existing heuristics at that time. In the above TO function, the authors directly operate on completion times to sequence jobs at production line level. However, the magnitudes of C_{max} and TCT are not the same, which affects the accuracy of the trade-off function. Laha and Gupta (2016) proposed a Hungarian penalty-based construction method to minimise C_{max} and TCT. The computational experiment shows that the proposed method performs better than the state-of-the-art heuristics. However, in their method, they did not address the relationship between the C_{max} and TCT and did not optimise two objectives together when constructing sequences. More recently, Li, Nault, and Ye (2019) proposed a current and future deviations (CFD) heuristic to balance the trade-off between C_{max} and TCT in permutation flow shop scheduling. The differences between the CFD heuristic and our TOB heuristic are threefold: (1) In the CFD heuristic, sequencing is mainly based on normalised deviations, and deviations are normalised based on calculated lower and upper bounds, which makes the CFD heuristic sensitive to the accuracy of bound calculations. In our TOB heuristic, sequencing is based on the mathematical relationship between C_{max} and TCT, which makes our TOB heuristic robust. (2) When generating the inital sequence, the CFD heuristic uses the index function based on normalised deviations while our TOB heuristic uses the index function directly based on idle times, which is inspired by Framinan, Leisten, and Ruiz-Usano (2002) and Ye et al. (2017). (3) The CFD heuristic only adopts NEH insertion technique to further improve the solution quality, while our TOB heuristic uses both NEH insertion and neighborhood exchanging techniques to improve the solution quality without increasing computational complexity.

In summary, to effectively evaluate the trade-off between C_{max} and TCT, we adopt the utility approach and propose a novel estimation method to normalise C_{max} and TCT to the same magnitude for no-wait flow shop scheduling.

3. Problem Formulation

We first introduce our notation and describe the mathematical formulations of C_{max} and TCT. In addition, the assumptions in our problem setting are presented.

For n-job m-machine no-wait flow shop scheduling, we use the following notation:

n:	the number of jobs;
m:	the number of machines;
π:	a sequence of n jobs, $\pi = [J_1, J_2, \ldots, J_{j-1}, J_j, \ldots, J_n]$;
$p_{j,i}$:	the processing time of job j on machine i, where j=1,…,n and i=1,…,m;
$C_{j,i}$:	the completion time of job j on machine i;
$d_{j-1,j}$:	the distance between the completion times of two adjacent jobs on the last machine.

The completion time of job j on machine i ($C_{j,i}$) can be calculated by Eq. (1)(Reddi and Ramamoorthy 1972):

$$C_{j,i} = C_{j-1,m} + \sum_{k=1}^{i} p_{j,k} - \min_{i=1,\dots,m} \Big(\sum_{k=1}^{i-1} p_{j,k} + \sum_{k=i+1}^{m} p_{j-1,k} \Big), \tag{1}$$

where $\sum_{k=1}^{0} p_{j,k} = 0$ and $\sum_{k=m+1}^{m} p_{j,k} = 0$. The distance between the completion times of two adjacent jobs on the last machine ($d_{j-1,j}$) can be calculated by Eq. (2):

$$\begin{aligned} d^{i}_{j-1,j} = C_{j,m} - C_{j-1,m} &= \sum_{k=1}^{m} p_{j,k} - \min_{i=1,\dots,m} \Big(\sum_{k=1}^{i-1} p_{j,k} + \sum_{k=i+1}^{m} p_{j-1,k} \Big) \\ &= \max_{i=1,\dots,m} \Big(\sum_{k=1}^{m} p_{j,k} - \sum_{k=1}^{i-1} p_{j,k} - \sum_{k=i+1}^{m} p_{j-1,k} \Big) \\ &= \max_{i=1,\dots,m} \Big(\sum_{k=1}^{m} p_{j,k} - \sum_{k=i+1}^{m} p_{j-1,k} \Big). \end{aligned} \tag{2}$$

From Eq. (2), the $d_{j-1,j}$ depends only on two adjacent jobs, but not on the positions of other jobs in the sequence, thus a pre-calculated matrix $D_{n\times n}$ can provide values of $d_{j-1,j}$ for any two adjacent jobs. Although the calculation of matrix $D_{n\times n}$ is not sequence dependent, the calculations of C_{max} and TCT are sequence dependent. Therefore, the C_{max} and TCT for a given sequence π can be calculated by Eq. (3) and Eq. (4):

$$C_{max,\pi} = \sum_{i=1}^{m} p_{\pi(1),i} + \sum_{j=2}^{n} D_{\pi(j-1),\pi(j)}, \tag{3}$$

$$\begin{aligned} TCT_{\pi} &= C_{\pi(1),m} + C_{\pi(2),m} + \dots + C_{\pi(n),m} \\ &= \sum_{i=1}^{m} p_{\pi(1),i} + \sum_{j=2}^{n} \Big(\sum_{i=1}^{m} p_{\pi(1),i} + \sum_{k=2}^{j} D_{\pi(k-1),\pi(k)} \Big) \\ &= n \sum_{i=1}^{m} p_{\pi(1),i} + \sum_{j=2}^{n} \sum_{k=2}^{j} D_{\pi(k-1),\pi(k)} \\ &= n \sum_{i=1}^{m} p_{\pi(1),i} + \sum_{k=2}^{2} D_{\pi(k-1),\pi(k)} + \dots + \sum_{k=2}^{n} D_{\pi(k-1),\pi(k)} \\ &= n \sum_{i=1}^{m} p_{\pi(1),i} + (n-1) D_{\pi(1),\pi(2)} + \dots + D_{\pi(n-1),\pi(n)} \\ &= n \sum_{i=1}^{m} p_{\pi(1),i} + \sum_{j=2}^{n} (n-j+1) D_{\pi(j-1),\pi(j)}. \end{aligned} \tag{4}$$

The following assumptions are introduced to address the no-wait flow shop scheduling problem. The processing time of job j on machine i, $p_{j,i}$, is known. The set-up times and the transportation times are included in the processing times. In addition, all jobs are available to be processed at time zero on the first machine, each job can only be processed once and only once on each machine, each machine can process only one job at a time, and we assume there is no machine breakdown. Once a job starts to be processed, it cannot be interrupted before completion, which means no preemption. Based on these assumptions, our objective is to construct a heuristic which generates a sequence of jobs such that trade-offs between C_{max} and TCT are minimised.

4. Methodology

In this section, we first present the factorization scheme including four different factors in our programming logic, second provide the steps of the initial sequence algorithm, and third propose our TOB heuristic. Finally, an example is used to illustrate the proposed method.

4.1. *Factors in our programming logic*

Four factors in our heuristic development, including objectives, idle times, sequence lever, and machine lever, have impacts on balancing trade-offs at the operational level between C_{max} and TCT. The options for each factor are summarized in Table 1.

[Table 1 about here.]

For Objective, we would like to see how combinations of other factors affect C_{max} and TCT in no-wait flow shops.

For Idle Time, we would like to see how idle times at current position and future position affect trade-off balancing. The mathematical definitions of current idle time (CI) and future idle times (FI) for a job j are:

$$CI(j) = \sum_{i=1}^{m}(C_{j,i} - p_{j,i} - C_{k-1,i}), \tag{5}$$

$$FI(j) = \sum_{i=1}^{m}(m - i + 1)(C_{k+1,i} - APT_i - C_{k,i}), \tag{6}$$

where k is the position index, j is the job index, i is the machine index. $C_{j,i}$ is the completion time of job j on machine i, $C_{k,i}$ is the completion time of the job at position k on machine i, $p_{j,i}$ is the processing time of job j on machine i, and APT_i is the average processing time of all unscheduled jobs on machine i.

The CI means we only consider the current idle times and FI means we only consider the future idle times when we construct the initial sequence. $CI \cup FI$ means we consider both current and future idle times when we construct the initial sequence.

For Sequence Lever, we would like to see how idle times on different parts of a sequence affect trade-off balancing. The S_1 means we put higher weight on idle times in the head of the sequence than those in the tail of the sequence. The S_n means we put higher weight on idle times in the tail of the sequence than those in the head of the sequence. The $\emptyset$ means we put equal weight on idle times in both head and tail of the sequence.

For Machine Lever, we would like to see how idle times on different machines affect trade-off balancing. There are two ways to put a fulcrum on an m-machine production line. Putting a fulcrum on the first machine, M_1, increases the impact of idle times on succeeding machines relative to those on preceding machines. Putting a fulcrum on the last machine, M_m, means the relative impacts of idle times are reversed. The $\emptyset$ means we assume the idle times have the same impact on each machine.

In total, there are $54 = 2 \cdot 3 \cdot 3 \cdot 3$ possible combinations of four factors. Define a 4-tuple as $[$*Obj*, *Idle time*, *Sequence*$(\cdot)$, *Machine*$(\cdot)]$, where *Obj* indicates an Objective, *idle time* indicates an option in Idle Time, *Sequence* indicates an option in Sequence Lever, its $(\cdot)$ means an option in Idle Time, *Machine* indicates an option in Machine Lever, its $(\cdot)$ means an option in Idle Time.

Based on this format, we recommend the following factor combination for initial sequence algorithm that, using an index function for makespan $(IF_j^{C_{max}})$, is generated by $[C_{max}, CI, S_1(CI), \emptyset]$. This means for $\min(C_{max})$ only current idle times are considered with a lever effect of S_1. The index function for total completion time (IF_j^{TCT}) is generated by $[TCT, CI \cup FI, S_1(CI), M_m(FI)]$. This means for $\min(TCT)$ both current and future idle times are considered and with lever effects of S_1 on CI and of M_m on FI.

Mathematically, calculations of $IF_j^{C_{max}}$ and IF_j^{TCT} for our TOB heuristic are summarized as follows:

$$IF_j^{C_{max}} = (n-k) \cdot CI(j), \tag{7}$$

$$IF_j^{TCT} = (n-k) \cdot CI(j) + FI(j), \tag{8}$$

where the k is the position index while sequencing jobs. By assigning a weighting factor $\alpha = 0.0 : 0.1 : 1.0$ on $IF_j^{C_{max}}$ to minimise C_{max}, we can choose a job with the minimum sum of weighted idle times while constructing the initial sequence, which is defined as:

$$\begin{aligned} IF_j^{TO} &= \alpha IF_j^{C_{max}} + (1-\alpha) IF_j^{TCT} \\ &= \alpha(n-k)CI(j) + (1-\alpha)\big((n-k)CI(j) + FI(j)\big) \\ &= (n-k)CI(j) + (1-\alpha)FI(j). \end{aligned} \tag{9}$$

4.2. *Initial Sequence Algorithm*

Based on the factors in heuristic development at the operational level, the steps for the initial sequence algorithm (ISA) are as follows:

(1) Set the position index k=1, the set of sequenced jobs S=$\emptyset$ and the set of unsequenced jobs U=$\{$*all jobs*$\}$.

(2) Select the jth job (denoted as $J_{[j]}$ in U (j=1,$\cdots$,n−k+1), place it into the position k in S, and calculate the average processing time (APT_i) of all jobs in U except the selected $J_{[j]}$ on each machine, where $APT_i = \big(\sum_{z \in U} p_{z,i} - p_{[j],i}\big) / \big(\mid U \mid -1\big)$, $i = 1, ..., m$, and $p_{[j],i}$ is the processing time of jth job in the sequence on machine

i. Set up an artificial job, and its processing time on each machine equals to APT_i. Append this artificial job to $J_{[j]}$, which means the artificial job is temporarily located on the $(k+1)$th position in S.

(3) Calculate the idle time between $(k-1)$th and $J_{[j]}$ job in S, which is considered as the current idle time $CI(j)=\sum_{i=1}^{m}(C_{j,i} - p_{j,i} - C_{k-1,i})$, where $C_{0,i} = 0 \; \forall \; i$. Calculate the idle time between $J_{[j]}$ and the artificial job, which is regarded as the future idle time $FI(j)= \sum_{i=1}^{m}(m - i + 1)(C_{k+1,i} - APT_i - C_{k,i})$, adding the machine lever (M_m) effect.

(4) For j=1,$\cdots$,$n-k+1$, each job in U has its own index function value calculated by Eq.(9), and we remove the job which has the minimum value of IF_j^{TO} from U and put it into the kth position in S. Set k=k+1.

(5) If $k<n$, go to Step 2, otherwise, append the last one job in U to the last position in S, and output S as the initial sequence π_0.

4.3. *TOB heuristic*

The techniques of insertion and neighborhood exchanging are used to improve solutions found by the ISA. In addition, when using the neighborhood exchanging technique, objective increment methods are used to calculate the increment of makespan (ΔC_{max}) and total completion time (ΔTCT). These methods reduce the computational complexity of calculating C_{max} and TCT from $O(n)$ to $O(1)$. The calculations for ΔC_{max} and ΔTCT can be found in Li, Wang, and Wu (2008) and in Ye et al. (2017), respectively.

After generating an initial sequence, we reconstruct the sequence based on the following in Eq. (10):

$$\Omega = \alpha \cdot \frac{(C_{max} \cdot l + \sum_{k=1}^{l}\sum_{i=1}^{m} p_{[k],i})}{2} + (1-\alpha)\cdot TCT, \tag{10}$$

where l is the number of jobs in the given sequence, and $\alpha = 0.0 : 0.1 : 1.0$ represents a preference for $\min(C_{max})$ at the production line level. We consider that preferences at both operational and production line levels should be consistent to obtain good solutions. In addition, C_{max} is the maximum completion time for the given sequence, $\sum_{k=1}^{l}\sum_{i=1}^{m} p_{[k],i}$ is the sum of processing times for the given sequence on all machines, and TCT is the total completion time for the given sequence. The given sequence could be the partial sequence when reconstructing the sequence with insertion and exchange techniques or the sequence with all scheduled jobs.

Recall that values of both C_{max} and TCT are not in the same magnitude at the production line level. Therefore, to normalise both C_{max} and TCT to the same scale, we estimate the sum of total processing time on all machines $(\sum_{k=1}^{l}\sum_{i=1}^{m} p_{k,i})$ and l times the makespan (C_{max}) is around a half of the value for a sequence's total completion time (TCT). With this estimation, we expect equivalent scales with respect to both C_{max} and TCT. The derivation of Eq. (10) can be found in Appendix A.

The steps of the proposed TOB heuristic in our study are as follows:

(1) Compute the distance matrix $D_{n\times n}$ and obtain the initial sequence π_0 using ISA. Given the value of α, let Ω_0 be the estimated value of sequence π_0. Set the current best estimated value $\Omega_b = \Omega_0$, the current best sequence π_b=π_0, and the number of iterations $r = 1$.

(2) Select the first two jobs from π_b, and choose the partial sequence with a smaller

Ω based on Eq. (10).

(3) Apply the NEH insertion technique (Nawaz, Enscore, and Ham 1983) to the obtained partial sequences, select the best partial sequence with minimum Ω as current sequence.

(4) Exchange each job with the remaining jobs on all positions in the current sequence. Among sequences generated by exchanging, the objective increment method is used to calculate ΔC_{max} and ΔTCT. If one sequence yields the smallest negative $\Delta\Omega{=}\alpha \cdot \frac{\Delta C_{max} \cdot l}{2} + (1-\alpha) \cdot \Delta TCT$ (where l is the number of jobs in the current sequence), then set this sequence as the current sequence. Otherwise, keep the current sequence.

(5) Repeat Steps 3 and 4 until all jobs are scheduled, and set the current sequence as π_r with Ω_r.

(6) If $\Omega_r < \Omega_b$, then set $\Omega_b{=}\Omega_r$ and $\pi_b{=}\pi_r$.

(7) For $j{=}1$ to $n{-}1$, insert the jth job in π_r into $n{-}j$ possible positions in the forward direction. If these sequences generate a lower Ω than Ω_b, then update π_b and Ω_b.

(8) Update $r = r + 1$. If $r \leq 6$, then return to Step 2; otherwise, go to Step 9.

(9) Output the final π_b.

The main computational burden of the TOB heuristic is determined by the NEH insertion and neighborhood exchanging techniques in Step 3. The computational complexity for the NEH insertion is $O(n^3)$ including calculating Ω when selecting the best insertion position. The computational complexity for the neighborhood exchanging technique is also $O(n^3)$ including calculating $\Delta\Omega$ with $O(1)$ when selecting the best exchanged pair. Therefore, the overall computational complexity of the TOB heuristic is $O(n^3)$, which is the same as those in the AIT and CFI heuristics.

4.4. *An illustrative example*

In this subsection, we provide an illustrative example to elaborate our proposed methodology. We consider a scenario in which a task including five jobs is processed on four machines. The processing times for each job on each machine are provided in Table 2 and α is assumed to be 0.5.

[Table 2 about here.]

4.4.1. Initial Sequence Algorithm

(1) Set $S =$ and the $U = \{J_1, J_2, J_3, J_4, J_5\}$.

(2) Consider J_1 in the first position of S, and average processing times of J_2, J_3, J_4, and J_5 on each machine are computed as $APT_i = [18, 15.25, 13.75, 15.5]$, which is equal to the artificial job. Append this artificial job to J_1, and we obtain the current idle time of 47 and future idle time 30.5 by adding the machine lever effects. The index function value for J_1, namely IF_1^{TO}, is 203.25. Similarly, we can consider other jobs in the first position of S and obtain $IF_2^{TO} = 209.375, IF_3^{TO} = 195.5, IF_4^{TO} = 231.25$, and $IF_5^{TO} = 243.375$. Hence, we remove J_3 that has the minimum IF value from U and place it into the first position of S. The updated $S = \{J_3\}$ and $U = \{J_1, J_2, J_4, J_5\}$.

(3) For the second position in S, we do the similar procedure as Steps 2 and 3 in ISA, and obtain the index function values for each job in U, which are

$IF = [150.33, 52, 162.17, 111.17]$. Hence, we remove J_2 from U and place it into the second position of S. Following the same procedure, we generate the initial sequence π_0 as $\{J_3, J_2, J_1, J_5, J_4\}$.

4.4.2. *TOB heuristic*

(1) From ISA, we obtained the initial sequence $\pi_0 = \{J_3, J_2, J_1, J_5, J_4\}$ and Ω_0 is 504. Set $\Omega_b = 504$, $\pi_b = \{J_3, J_2, J_1, J_5, J_4\}$ and $r = 1$.
(2) The sequence from Steps 2 -7 in the TOB heuristic is $\pi_1 = \{J_1, J_5, J_4, J_3, J_2\}$ and $\Omega_1 = 495.25$. Because 495.25 is smaller than 504, we update $\Omega_b = 492.25, \pi_b = \{J_1, J_5, J_4, J_3, J_2\}$.
(3) In the iterations from 2 to 6, both Ω_b and π_b remain unchanged. Therefore, the final sequence is $\{J_1, J_5, J_4, J_3, J_2\}$ with Ω value of 495.25.

5. Computational Experiments

In this section, computational experiments are conducted to verify the effectiveness of our TOB heuristic on balancing trade-offs between $\min(C_{max})$ and $\min(TCT)$, based on the classic Taillard's benchmarks and one-year historical data from UKHC. We present the results for each dataset, and then we discuss the inconsistency between C_{max} and TCT among different sequences.

5.1. ***Taillard's Benchmarks***

Taillard's benchmarks (Taillard 1993) are classic in flow shop scheduling, and are commonly used to test scheduling methods on $\min(C_{max})$ and on $\min(TCT)$. There are 12 scales in Taillard's benchmarks, ranging from a 20-job 5-machine flow line to a 500-job 20-machine flow line, 10 instances in each scale, and 120 instances in total.

Because $\min(C_{max})$ and $\min(TCT)$ are NP-hard problems and current literature only provides the best solutions for 110 instances, the best and worst solutions are generated based on actual performances of the AIT, CFI, and TOB(α) heuristics, as references R. Given the best and worst values of $f_{o,k}^{MIN}$ and $f_{o,k}^{MAX}$ for each instance k and for each objective o, with $o = 1$ for $\min(C_{max})$ and $o = 2$ for $\min(TCT)$, we can define a normalised deviation (ND) as follows:

$$ND_{o,k}(q, R) = \frac{f_{o,k}(q) - f_{o,k}^{MIN}(R)}{f_{o,k}^{MAX}(R) - f_{o,k}^{MIN}(R)} \cdot 100, \tag{11}$$

where $f_{o,k}(q)$ is the solution generated by a sequencing method q. This $ND_{o,k}(q, R)$ is normalised in the same way as $d_o(x, y)$ in Czyzżak and Jaszkiewicz (1998). Accordingly, an average normalised deviation for a sequencing method q with the objective o ($AND_o(q, R)$) is defined as follows:

$$AND_o(q, R) = \frac{1}{K} \cdot \sum_{k=1}^{K} ND_{o,k}(q, R), \tag{12}$$

where K is the number of instances, which is 120 in Taillard's benchmarks.

In addition to $ANDs$ for each objective, a bi-objective indicator is introduced to evaluate the performance of our TOB heuristic. Based on the $ND_{o,k}(q,R)$, we define an indicator $d_k(q,R)$ as follows, which is similar to the Pareto distance metric in Xu et al. (2015), Li et al. (2018) and Tian et al. (2018):

$$d_k(q,R) = \sqrt{(ND_{1,k}(q,R))^2 + (ND_{2,k}(q,R))^2}. \tag{13}$$

Similarly, an average distance $D1(q,R)$ is defined as follows:

$$D1(q,R) = \frac{1}{K} \cdot \sum_{k=1}^{K} d_k(q,R). \tag{14}$$

The $D1(q,R)$ metric measures the average normalised Euclidean distance between solutions generated by a sequence method q and the reference R. The smaller the value, the closer to the reference, and the better the performance.

The $ANDs$ on C_{max} and TCT objectives and $D1(q,R)$ are provided in Table 3, for all 120 instances. The minimum values for each metrics are in bold. The p-values of two-tailed t-tests on AND are also provided in Table 3 against the best sequencing method, which is TOB(1.0) for C_{max} and TOB(0.0) for TCT.

[Table 3 about here.]

For $\min(C_{max})$, the TOB (1.0) heuristic has the least average normalised deviation value of 13.13%, but based on the t-test, the AIT heuristic (13.20%) and TOB(0.9) have statistically indifferent performance from the TOB (1.0) heuristic on a 95% confidence interval. In this sense, their performances are equivalent on $\min(C_{max})$, although the AND from TOB(1.0) is smaller than those in the AIT and TOB(0.9) heuristics. For $\min(TCT)$, the TOB (0.0) heuristic (5.46%) outperforms the CFI heuristic (14.50%), and the difference is statistically significant at all reasonable p-values. For the $D1$ metric, when α changes from 0.0 to 0.9, TOB(α) can always achieve smaller distance $D1$ than those in the AIT and CFI heuristics. $D1$ values for both TOB(1.0) and AIT heuristics are almost the same, which is consistent with the t-test results that both heuristics are not statistically different.

Using the posteriori approach (Ciavotta, Minella, and Ruiz 2013), we generate a frontier in Figure 1 based on AND values generated by the TOB(α) across a range of α, AIT and CFI heuristics. We can clearly see that TOB(0.0) dominates the CFI heuristic with [75.74, 5.46] for TOB(0.0) and [79.31, 14.50] for the CFI heuristic. In addition, the TOB(1.0) dominates the AIT heuristic. In Figure 1, the points between TOB(1.0) and AIT heuristics are very close but such slight dominance can be found from Table 3: [13.13, 88.24] for TOB(1.0) and [13.20, 88.25] for the AIT heuristic.

[Figure 1 about here.]

Average computation times of the TOB(0.5), AIT and CFI heuristics are also presented in Table 4 based on each scale of 10 instances in Taillard's benchmarks. The Matlab code of all three heuristics were run on a Dell personal computer with 16.0 GB RAM and a CPU of 2.40GHz. We can see that the AIT heuristic took the least CPU time and average computation times for both CFI and TOB(0.5) heuristics are comparable. Recall that all three heursitics have the same computational complexity; the difference between the AIT and TOB(0.5) or CFI heuristic can be explained by the objective increment method since it takes more time to check the conditions when minimising TCT or Ω.

[Table 4 about here.]

5.2. *Historical Data from UKHC*

To validate our TOB heuristic for ORs scheduling across the periop process, we carry out a case study based on historical data from the UKHC in which the first-come-first-serve (FCFS) rule is applied. The periop process can be divided into three main stages, including preoperatives, intraoperatives and postoperatives (Gupta 2007). Patients are not supposed to wait during the process, especially from the intraoperatives stage to postoperatives stage, which leads to high operating room costs. Therefore, immediately after the surgery in the intraoperatives stage, patients are transferred to the wards for recovery. Therefore, the periop process can be modelled as a three-machine no-wait flow shop (Hsu, De Matta, and Lee 2003; Ye et al. 2017).

Historical data from UKHC consist of almost 30,000 cases in 365 consecutive days from 2013 to 2014. UKHC schedules ORs on weekdays, but opens emergency rooms on weekends and holidays, and thus the number of cases on weekends and holidays is much less than that on weekdays. Excluding data from weekends and holidays, we have more than 28,000 cases in 260 days used in our case study.

Utilisation of the periop process and patient flowtime across the periop process, or average completion time ($\overline{C}$), are used to evaluate performances of OR scheduling methods. The relative performances of the AIT, CFI and TOB(0.5) heuristics are provided in Table 5, compared to that of UKHC.

[Table 5 about here.]

In Table 5, we can see that the TOB (0.5) heuristic can achieve 78.48% utilisation, larger than UKHC's 75.04%. Moreover, the patient flow time for the TOB (0.5) is much smaller than UKHC's, which the improvement is calculated by (615.4 − 549.2)/615.4=10.75%. The 10.75% improvement indicates that an additional 3,000 patients could be served in a year if our TOB (0.5) heuristic was used for OR scheduling. However, there are many other factors affecting OR scheduling and control, such as emergencies, the availability of patients, equipment and resources, etc.

A different measure of performance is whether the process is "under control". Run charts of utilisation that compares our TOB (0.5) heuristic with UKHC are shown in Figure 2.

[Figure 2 about here.]

[Figure 3 about here.]

From Figure 2, we can see that utilisations of the periop process achieved by our TOB heuristic are under better control, with only one point out of control limits in the X-bar chart for the average performance, or in the R chart for performance variation ranges. In constrast, the utilisation of the periop process at UKHC sometimes is out of control, as indicated by two points below the lower control limit (LCL), two points above the upper control limit (UCL) in the X-bar chart and by two points above UCL in the R-chart.

Run charts for average patient flowtime are also shown in Figure 3. Figure 3 shows that patient flowtime is under better control with our TOB (0.5) heuristic, noting the much tighter control limits with our TOB (0.5) heuristic than the control limits from UKHC data.

In addition to run charts, we further generate process capability charts in Table 6 for TOB(0.0, 0.5, 1.0) on utilisation and patient flowtime to see if the periop process is better under control. According to Montgomery (2007), a process capability index (C_{pk}) is used to verify if the process drifts way from the average and tends to be out of control, given $C_{pk} = \min[(USL - \mu)/(3 \cdot \sigma), (\mu - LSL)/(3 \cdot \sigma)]$, where μ is the average of historical process performance, σ is the standard deviation of historical process performance, and USL and LSL are upper and lower specification limits respectively. Given USL and LSL for expected performance, the larger the C_{pk} values, the more centered the process is under control in terms of μ and σ.

To compare three methods of TOB(0.0, 0.5, 1.0) on trade-off balancing, we generate specifications according to the performance of $[78.48\%, 549.2]$ by TOB(0.5) with one standard deviation. The process capabilities of three methods are summarized in Table 6. From Table 6, we can make the following observations: the $Util$ of TOB(0.5) is most centered while the $Util$ of TOB(0.0) shifts slightly to the right and the $Util$ of TOB(1.0) shifts slightly to the left. This is supported by calculations of C_{pk}, which are 0.638 for TOB(0.0), 0.700 for TOB(0.5), and 0.695 for TOB(1.0), respectively. Similarly, the PtF of TOB(0.5) is most centered among three methods, compared to that of TOB(0.0) shifting slightly to the right and that of TOB(1.0) shifting severely to the left. This is also supported by calculations of C_{pk}, which are 0.511 for TOB(0.0), 0.733 for TOB(0.5), and -0.351 for TOB(1.0), respectively. Moreover, the negative value -0.351 of TOB(1.0) indicates that the process is out of control in terms of patient flow if the OR manager simply maximises the utilisation.

[Table 6 about here.]

Overall, through our case study on historical data from UKHC, the TOB(0.5) heuristic could improve the current practice at UKHC in terms of utilisation and patient flow. In addition, the performance of the peri-op process could be more stable based on the results of run charts and process capability of C_{pk}.

5.3. *Inconsistency among* $\min(C_{max})$, $\min(TCT)$ *and* $\min(TO)$

To confirm the inconsistency among $\min(C_{max})$, $\min(TCT)$, and $\min(TO)$ in generating sequences, we run our TOB heuristic with $\alpha = 1.0$ to generate the sequence for $\min(C_{max})$, with $\alpha = 0.0$ to generate the sequence for $\min(TCT)$, and with $\alpha = 0.5$ to generate the sequence for $\min(TO)$. We do this for both Taillard's benchmarks and UKHC historical data. For each dataset, we compute the Spearman rank correlation coefficient ρ between the sequences, and the results are shown in Table 7.

[Table 7 about here.]

In Table 7, 0.0 vs. 1.0 means the correlation between $\min(TCT)$ and $\min(C_{max})$, 0.0 vs. 0.5 means that between $\min(TCT)$ and $\min(TO)$, and 0.5 vs. 1.0 means that between $\min(TO)$ and $\min(C_{max})$. It is obvious that the inconsistency exists between $\min(TCT)$ and $\min(C_{max})$ with small Spearman rank correlation coefficients that are not significantly different from zero for both datasets. As the number of machines m

is small, but n is large, the empirical inconsistency between $\min(TO)$ and$\min(C_{max})$ is obvious for the UKHC data set with a Spearman ρ of -0.0064, but the inconsistency between $\min(TO)$ and $\min(TCT)$ is less obvious with a Spearman ρ of 0.2295. However, when both n and m are large, the inconsistencies are obvious for Taillard's benchmarks with Spearman ρ of 0.0624 and 0.0700, respectively. Thus, using our TOB heuristic with extreme weights across our datasets, 1.0 for $\min(C_{max})$ and 0.0 for $\min(TCT)$, we find that the empirical inconsistency exists not only between $\min(C_{max})$ and $\min(TCT)$, but also between $\min(TO)$ and $\min(C_{max})$ and between $\min(TCT)$ and $\min(TO)$ as well.

6. Conclusion

We propose a trade-off balancing (TOB) heuristic to minimise the weighted sum of maximum and total completion time based on machine idle times. First, we introduce a factorization scheme to construct the initial sequence at the operational level. Second, we propose the estimation method to establish the mathematical relationship between the objectives of C_{max} and TCT at the production line level, and reconstruct the sequence in the improvement phase of our TOB heuristic. Third, we demonstrate the inconsistency beween C_{max} and TCT in the no-wait flow shop by computing Spearman rank order correlations among sequences. Using the same preference α on C_{max} to model coupled machine idle times at the operational level and balance trade-offs at the production line level, we show that our TOB heuristic is sufficiently flexible to address a variety of management concerns.

Through computational experiments on 120 instances in Taillard's benchmarks, we show that our TOB heuristic outperforms the AIT and CFI heuristics on $\min(C_{max})$ and $\min(TCT)$, respectively. Based on a real case study of one-year historical data from UKHC, utilisation and patient flowtime of the periop process are improved, and the process is better under control using our TOB heuristic.

For future work, our first direction is to relax the assumption that preferences at operational and production line levels are the same. The second direction is to propose the adaptive TOB heuristic to achieve robust scheduling because variations in processing times may occur in practice. The third direction is to separate setup times from processing times.

Acknowledgements

We appreciate support from UK HeathCare, the Department of Mechanical Engineering at University of Kentucky, the Department of Industrial & Systems Engineering at University of Wisconsin-Madison, the Natural Sciences and Engineering Research Council of Canada, and the Haskayne School of Business at University of Calgary.

Disclosure statement

No potential conflict of interest was reported by the authors.

Funding

This project was supported by grant number R03HS024633 from the Agency for Healthcare Research and Quality. The content is solely the responsibility of the authors and does not necessarily represent the official views of the Agency for Healthcare Research and Quality.

References

Aldowaisan, Tariq, and Ali Allahverdi. 2004. "New heuristics for m-machine no-wait flowshop to minimize total completion time." *Omega* 32 (5): 345–52.

Allahverdi, Ali. 2016. "A Survey of Scheduling Problems with No-Wait in Process." *European Journal of Operational Research* 255: 665–86.

Allahverdi, Ali, and Tariq Aldowaisan. 2002. "No-wait flowshops with bicriteria of makespan and total completion time." *Journal of the Operational Research Society* 53 (9): 1004–15.

Allahverdi, Ali, and Harun Aydilek. 2013. "Algorithms for no-wait flowshops with total completion time subject to makespan." *International Journal of Advanced Manufacturing Technology* 2237–51.

Allahverdi, Ali, and Harun Aydilek. 2014. "Total completion time with makespan constraint in no-wait flowshops with setup times." *European Journal of Operational Research* 238 (3): 724–34.

Allahverdi, Ali, Harun Aydilek, and Asiye Aydilek. 2018. "No-wait flowshop scheduling problem with two criteria; total tardiness and makespan." *European Journal of Operational Research* 269 (2): 590–601.

Aydilek, Harun, and Ali Allahverdi. 2012. "Heuristics for no-wait flowshops with makespan subject to mean completion time." *Applied Mathematics and Computation* 219 (1): 351–9.

Bertolissi, Edy. 2000. "Heuristic algorithm for scheduling in the no-wait flow-shop." *Journal of Materials Processing Technology* 107 (1): 459–65.

Chen, Chuen-Lung, Ranga V Neppalli, and Nasser Aljaber. 1996. "Genetic algorithms applied to the continuous flow shop problem." *Computers & Industrial Engineering* 30 (4): 919–29.

Chien, Chen-Fu, Stéphane Dauzère-Pérès, Hans Ehm, John W Fowler, Zhibin Jiang, Shekar Krishnaswamy, Tae-Eog Lee, Lars Moench, and Reha Uzsoy. 2011. "Modelling and analysis of semiconductor manufacturing in a shrinking world: challenges and successes." *European Journal of Industrial Engineering* 5 (3): 254–71.

Ciavotta, Michele, Gerardo Minella, and Rubén Ruiz. 2013. "Multi-objective sequence dependent setup times permutation flowshop: A new algorithm and a comprehensive study." *European Journal of Operational Research* 227 (2): 301–13.

Cohn, Amy, Selin Kurnaz, Yihan Guan, and Yiwen Jiang. 2010. "Trading off between makespan and customer responsiveness in flow shop sequencing." *International Journal of Production Research* 48 (22): 6777–97.

Cortés, Beatriz Murrieta, Juan Carlos Espinoza García, and Fabiola Regis Hernández. 2012. "Multi-objective flow-shop scheduling with parallel machines." *International Journal of Production Research* 50 (10): 2796–808.

Czyzżak, Piotr, and Adrezej Jaszkiewicz. 1998. "Pareto simulated annealing—a metaheuristic technique for multiple-objective combinatorial optimization." *Journal of Multi-Criteria Decision Analysis* 7 (1): 34–47.

Dang, Feidi. 2017. "An efficient heuristic to balance trade-offs between utlization and patient flowtimes in operating room management." Master's thesis, University of Kentucky.

Fink, Andreas, and Stefan Voß. 2003. "Solving the continuous flow-shop scheduling problem by metaheuristics." *European Journal of Operational Research* 151 (2): 400–14.

Framinan, Jose Manuel, Rainer Leisten, and Rafael Ruiz-Usano. 2002. "Efficient heuristics for flowshop sequencing with the objectives of makespan and flowtime minimisation." *European*

Journal of Operational Research 141 (3): 559–569.

Framinan, José Manuel, and Marcelo Seido Nagano. 2008. “Evaluating the performance for makespan minimisation in no-wait flowshop sequencing.” *Journal of Materials Processing Technology* 197 (1): 1–9.

Framinan, José Manuel, Marcelo Seido Nagano, and João Vitor Moccellin. 2010. “An efficient heuristic for total flowtime minimisation in no-wait flowshops.” *The International Journal of Advanced Manufacturing Technology* 46 (9-12): 1049–57.

Gangadharan, Rajesh, and Chandrasekharan Rajendran. 1993. “Heuristic algorithms for scheduling in the no-wait flowshop.” *International Journal of Production Economics* 32 (3): 285–90.

Gupta, Diwakar. 2007. “Surgical suites’ operations management.” *Production and Operations Management* 16 (6): 689–700.

Hsu, Vernon Ning, Renato De Matta, and Chung-Yee Lee. 2003. “Scheduling patients in an ambulatory surgical center.” *Naval Research Logistics (NRL)* 50 (3): 218–38.

Johnson, MR Gareyand DS, and Michael R Garey. 1979. “Computers and intractability.” *W. II. Freeman and Company, New York* .

Jungwattanakit, Jitti, Manop Reodecha, Paveena Chaovalitwongse, and Frank Werner. 2009. “A comparison of scheduling algorithms for flexible flow shop problems with unrelated parallel machines, setup times, and dual criteria.” *Computers & Operations Research* 36 (2): 358–78.

Laha, Dipak, and Uday K Chakraborty. 2009. “A constructive heuristic for minimizing makespan in no-wait flow shop scheduling.” *The International Journal of Advanced Manufacturing Technology* 41 (1-2): 97–109.

Laha, Dipak, and Jatinder ND Gupta. 2016. “A Hungarian penalty-based construction algorithm to minimize makespan and total flow time in no-wait flow shops.” *Computers & Industrial Engineering* 98: 373–83.

Laha, Dipak, Jatinder ND Gupta, and Sagar U Sapkal. 2014. “A penalty-shift-insertion-based algorithm to minimize total flow time in no-wait flow shops.” *Journal of the Operational Research Society* 65 (10): 1611–24.

Laha, Dipak, and Sagar U Sapkal. 2014. “An improved heuristic to minimize total flow time for scheduling in the m-machine no-wait flow shop.” *Computers & Industrial Engineering* 67: 36–43.

Li, Junqing, Hongyan Sang, Yuyan Han, Cungang Wang, and Kaizhou Gao. 2018. “Efficient multi-objective optimization algorithm for hybrid flow shop scheduling problems with setup energy consumptions.” *Journal of Cleaner Production* 181: 584–598.

Li, Wei, Victoria L Mitchell, and Barrie R Nault. 2014. “Inconsistent Objectives in Operating Room Scheduling.” In *IIE Annual Conference. Proceedings*, 727–36. Institute of Industrial and Systems Engineers (IISE).

Li, Wei, Barrie R Nault, and Honghan Ye. 2019. “Trade-off balancing in scheduling for flow shop production and perioperative processes.” *European Journal of Operational Research* 273 (3): 817–30.

Li, Xiaoping, Qian Wang, and Cheng Wu. 2008. “Heuristic for no-wait flow shops with makespan minimization.” *International Journal of Production Research* 46 (9): 2519–30.

Montgomery, Douglas C. 2007. *Introduction to Statistical Quality Control*. John Wiley & Sons.

Nawaz, Muhammad, E Emory Enscore, and Inyong Ham. 1983. “A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem.” *Omega* 11 (1): 91–5.

Pan, Quan-Ke, Ling Wang, and Bin Qian. 2009. “A novel differential evolution algorithm for bi-criteria no-wait flow shop scheduling problems.” *Computers & Operations Research* 36 (8): 2498–511.

Rajendran, Chandrasekharan. 1994. “A no-wait flowshop scheduling heuristic to minimize makespan.” *Journal of the Operational Research Society* 45 (4): 472–8.

Rajendran, Chandrasekharan, and Dipak Chaudhuri. 1990. “Heuristic algorithms for continuous flow-shop problem.” *Naval Research Logistics (NRL)* 37 (5): 695–705.

Rajendran, Chandrasekharan, and Hans Ziegler. 1997. “An efficient heuristic for scheduling in a flowshop to minimize total weighted flowtime of jobs." *European Journal of Operational Research* 103 (1): 129–38.

Reddi, SS, and CV Ramamoorthy. 1972. "On the flow-shop sequencing problem with no wait in process." *Operational Research Quarterly* 23 (3): 323–31.

Röck, Hans. 1984. "The three-machine no-wait flow shop is NP-complete." *Journal of the ACM (JACM)* 31 (2): 336–45.

Ruiz, Rubén, and Ali Allahverdi. 2009. "New heuristics for no-wait flow shops with a linear combination of makespan and maximum lateness." *International Journal of Production Research* 47 (20): 5717–38.

Samarghandi, Hamed, and Tarek Y ElMekkawy. 2012. "A meta-heuristic approach for solving the no-wait flow-shop problem." *International Journal of Production Research* 50 (24): 7313–26.

Shao, Weishi, Dechang Pi, and Zhongshi Shao. 2019. "A Pareto-Based Estimation of Distribution Algorithm for Solving Multiobjective Distributed No-Wait Flow-Shop Scheduling Problem With Sequence-Dependent Setup Time." *IEEE Transactions on Automation Science and Engineering* .

Taillard, Eric. 1993. "Benchmarks for basic scheduling problems." *European Journal of Operational Research* 64 (2): 278–85.

Tavakkoli-Moghaddam, R, AR Rahimi-Vahed, and AH Mirzaei. 2008. "Solving a multi-objective no-wait flow shop scheduling problem with an immune algorithm." *The International Journal of Advanced Manufacturing Technology* 36 (9-10): 969–81.

Tian, Ye, Ran Cheng, Xingyi Zhang, Fan Cheng, and Yaochu Jin. 2018. "An indicator-based multiobjective evolutionary algorithm with reference point adaptation for better versatility." *IEEE Transactions on Evolutionary Computation* 22 (4): 609–622.

Wu, Kan, and Leon McGinnis. 2012. "Performance evaluation for general queueing networks in manufacturing systems: Characterizing the trade-off between queue time and utilization." *European Journal of Operational Research* 221 (2): 328–339.

Xu, Zhitao, XG Ming, Maokuan Zheng, Miao Li, Lina He, and Wenyan Song. 2015. "Cross-trained workers scheduling for field service using improved NSGA-II." *International Journal of Production Research* 53 (4): 1255–1272.

Ye, Honghan, Wei Li, and Amin Abedini. 2017. "An improved heuristic for no-wait flow shop to minimize makespan." *Journal of Manufacturing Systems* 44: 273–9.

Ye, Honghan, Wei Li, Amin Abedini, and Barrie Nault. 2017. "An effective and efficient heuristic for no-wait flow shop production to minimize total completion time." *Computers & Industrial Engineering* 108: 57–69.

Ye, Honghan, Wei Li, and Enming Miao. 2016. "An effective heuristic for no-wait flow shop production to minimize makespan." *Journal of Manufacturing Systems* 40: 2–7.

Zhao, Hui, Kan Wu, and Edward Huang. 2018. "Clinical trial supply chain design based on the Pareto-optimal trade-off between time and cost." *IISE Transactions* 50 (6): 512–524.

Appendix A. Derivation of the Estimated Equation

To derive the estimated equation between C_{max} and TCT when reconstructing sequences, we denote the $p_{j,i}$ as the processing time of job j on machine i, $C_{j,i}$ as the completion time of job j on machine i, and $I_{j,m}$ as the idle time between job $j-1$ and job j on the last machine m. Here, we assume the given sequence is the sequence with all scheduled jobs. Therefore, the number of jobs is n. The calculation of TCT can be described in Eq. (A1), which is the sum of completion times on all jobs.

$$TCT = \sum_{j}^{n} C_{j,m} = C_{1,m} + C_{2,m} + ... + C_{n,m}. \tag{A1}$$

Also, the calculation of makespan can be described in Eq. (A2), which consists of the sum of processing times of the first job in the sequence on all machines, the sum of processing times from the second job to the last job of the sequence on the last machine, and the sum of idle time from the second job to the last job of the sequence on the last machine.

$$C_{max} = \sum_{i=1}^{m} p_{1,i} + \sum_{j=1}^{n} p_{j,m} + \sum_{j=2}^{n} I_{j,m}. \tag{A2}$$

As the equations (A1) and (A2) are sequence-dependent, in order to estimate the relationship between C_{max} and TCT, we have three-step estimations. The first step is to estimate the processing time on the first position of the sequence, denoted as u_p, the second step is to estimate the idle time on the last machine, denoted as u_e, and the third step is to estimate the processing time on the last machine, denoted as u_m. These three-step estimations are mathematically described in the following equations:

$$u_p = \widetilde{p}_{1,i} = \frac{\sum_{j=1}^{n} \sum_{i=1}^{m} p_{j,i}}{nm}, \tag{A3}$$

$$u_e = \widetilde{I}_{j,m} = \frac{\sum_{j=1}^{n} I_{j,m}}{n}, \tag{A4}$$

$$u_m = \widetilde{p}_{j,m} = \frac{\sum_{j=1}^{n} p_{j,m}}{n}. \tag{A5}$$

Therefore, we re-model the C_{max} using the above equations:

$$C_{max} = C_{n,m} = mu_p + (n-1)u_m + (n-1)u_e. \tag{A6}$$

where the makespan is composed of three parts: total processing times of the job on the first position of the sequence on all machines, processing times of $n-1$ jobs on the last machine, and total idle times on the last machine.

Similarly, we can re-model the TCT based on Eq. (A2) and the above equations:

$$\begin{aligned} TCT &= \sum_{j}^{n} C_{j,m} = C_{1,m} + C_{2,m} + ... + C_{n,m} \\ &= mu_p + mu_p + u_m + u_e + ... + mu_p + (n-1)u_m + (n-1)u_e \\ &= nmu_p + \frac{n(n-1)}{2}u_m + \frac{n(n-1)}{2}u_e. \end{aligned} \tag{A7}$$

By multiplying $n/2$ with Eq. (A6), we obtain

$$\frac{n}{2}C_{max} = \frac{nmu_p}{2} + \frac{n(n-1)}{2}u_m + \frac{n(n-1)}{2}u_e. \tag{A8}$$

Combining Eq. (A7) and Eq. (A8), we can obtain the relationship between C_{max} and TCT as follows:

$$TCT = \frac{n}{2}C_{max} + \frac{nm}{2}u_p = \frac{n}{2}C_{max} + \frac{\sum_{j=1}^{n}\sum_{i=1}^{m} p_{j,i}}{2}. \tag{A9}$$

Therefore, in our TOB heuristic development, we define the estimated equation as shown in Eq. (A10):

$$\Omega = \alpha \cdot \frac{(C_{max} \cdot n + \sum_{j=1}^{n}\sum_{i=1}^{m} p_{j,i})}{2} + (1-\alpha) \cdot TCT, \tag{A10}$$

where $\alpha = 0.0 : 0.1 : 1.0$ represents a preference on $\min(C_{max})$. Notice that when the number of jobs is n in the sequence, the value of $\sum_{j=1}^{n}\sum_{i=1}^{m} p_{j,i}$ is sequence-independent, and the $p_{[j],i}$ can be replaced with $p_{j,i}$.

Table 1.: Four factors in balancing trade-offs

Factors	Options
Objective	$[C_{max}, TCT]$
Idle time	$[\{CI\}, \{FI\}, \{CI \cup FI\}]$
Sequence Lever	$[S_1, S_n, \emptyset]$
Machine Lever	$[M_1, M_m, \emptyset]$

Table 2.: Processing times of a 5-job 4-machine example

	M_1	M_2	M_3	M_4
J_1	12	24	12	13
J_2	20	3	19	11
J_3	19	20	3	15
J_4	14	23	16	14
J_5	19	15	17	22

Table 3.: Performance metrics by TOB(α), AIT and CFI heuristics on Taillard's benchmarks (%)

Metrics	TOB(α)											AIT	CFI
	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0		
AND for C_{max}	75.74	46.55	42.95	38.96	32.28	26.46	26.91	21.07	16.65	14.47	**13.13**	13.20	79.31
p-value by TOB(1.0)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.02	0.33	-	0.32	0.00
AND for TCT	**5.46**	13.21	21.06	27.53	33.43	40.47	48.00	53.97	65.83	71.75	88.24	88.25	14.50
p-value by TOB(0.0)	-	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
$D1$	76.49	**50.98**	52.27	53.21	51.84	53.81	62.06	62.89	70.56	74.72	90.12	90.14	83.28

Table 4.: Average computation times for the AIT, CFI, and TOB(0.5) heuristics in Taillard's benchmarks (Units: seconds)

n by m	AIT	CFI	TOB (0.5)
20 by 5	0.09	0.15	0.08
20 by 10	0.02	0.11	0.06
20 by 20	0.03	0.08	0.07
50 by 5	0.08	0.49	0.47
50 by 10	0.08	0.55	0.48
50 by 20	0.08	0.52	0.49
100 by 5	0.32	2.81	2.93
100 by 10	0.31	2.82	2.91
100 by20	0.36	2.86	2.95
200 by 10	2.03	20.56	21.16
200 by 20	2.17	20.77	21.33
500 by 20	33.15	356.59	363.64
Average	3.23	34.02	34.71

Table 5.: Utilisation (%) and Average Completion Times (minute)

		AIT	CFI	TOB (0.5)	UKHC
$Util$	Avg.	78.52	77.91	78.48	75.04
	StD	5.95	5.89	5.89	5.30
$\overline{C}$	Avg.	560.1	544.6	549.2	615.4
	StD	43.4	41.6	41.6	56.3

Table 6.: Process Capability Charts for Trade-off Balancing on UKHC Data

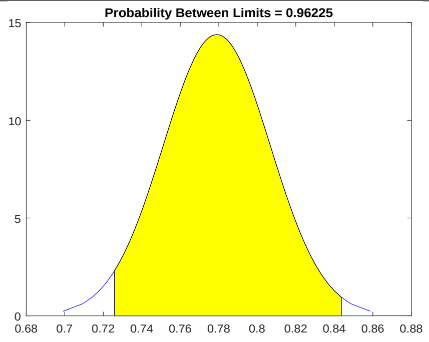
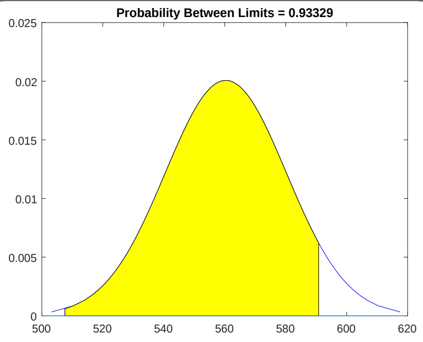
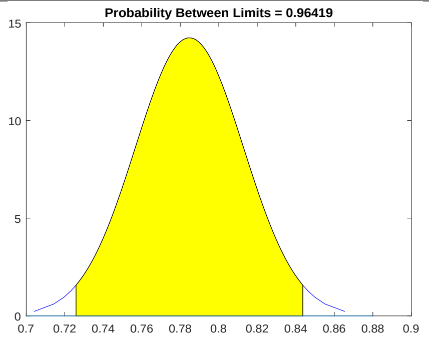
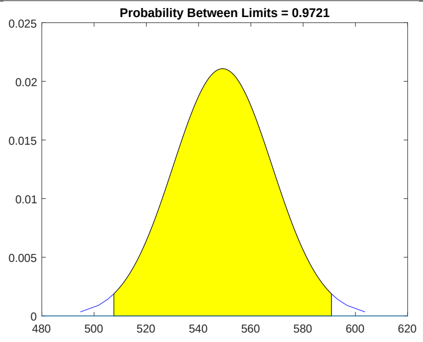
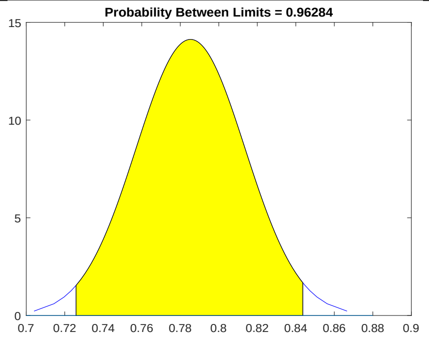
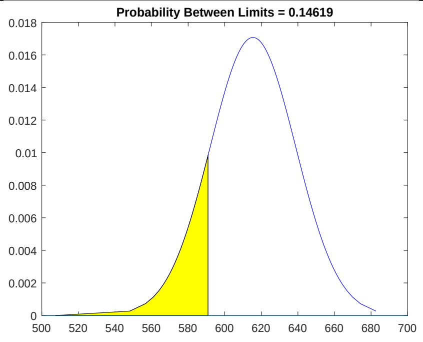
TOB(α)	Util	PtF
(0.0)		
(0.5)		
(1.0)		

Table 7.: Spearman's ρ among sequences for scheduling objectives

Dataset	0.0 vs. 1.0	0.5 vs. 1.0	0.0 vs. 0.5
Taillard's benchmarks	0.0267	0.0624	0.0700
UKHC historical data	-0.0288	-0.0064	0.2295

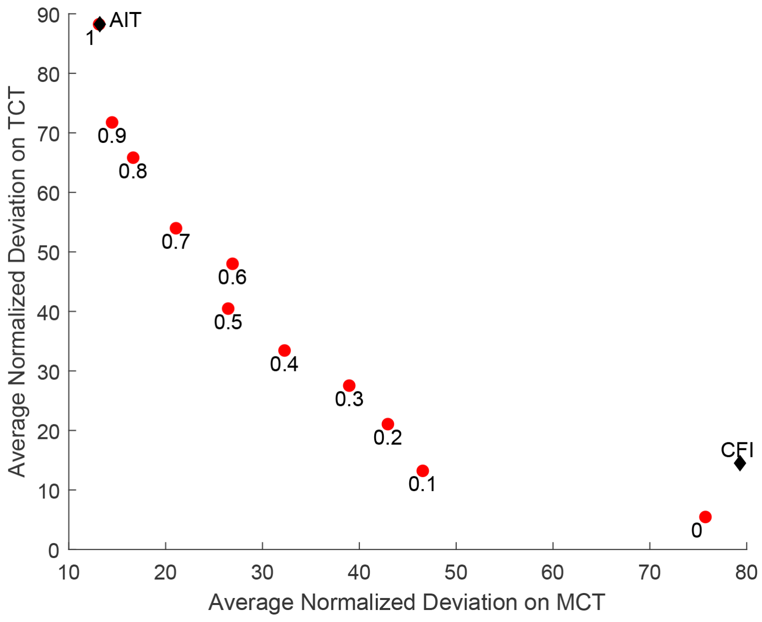
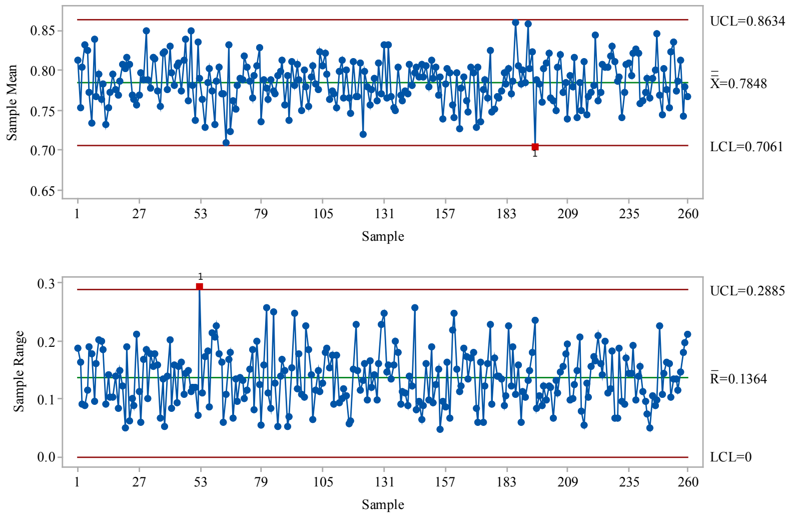
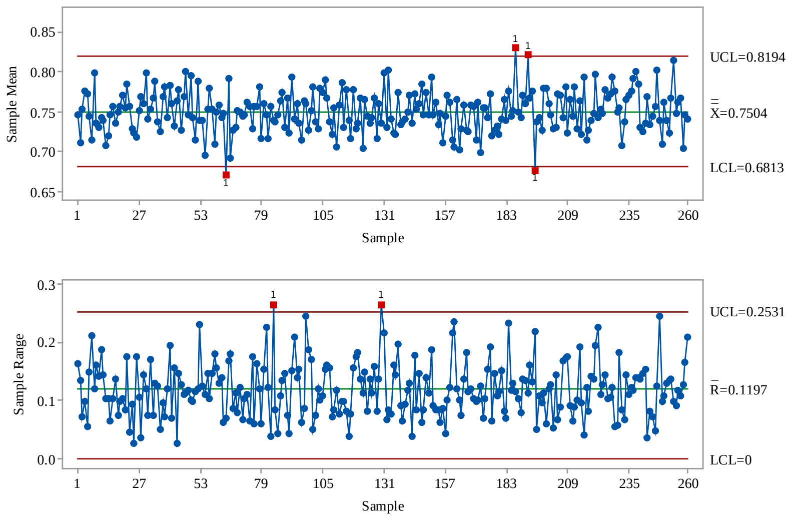


Figure 1.: A Frontier Based on AND on C_{max} and TCT for Taillard's benchmark

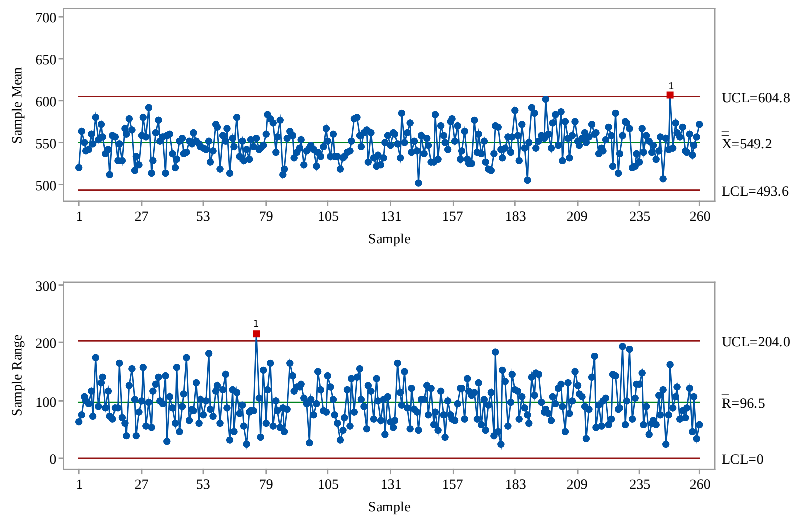


(a) TOB (0.5)

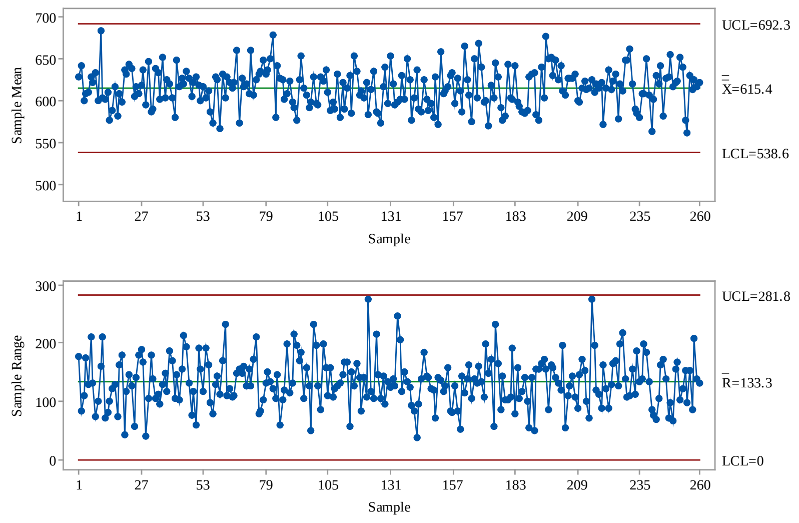


(b) UKHC

Figure 2.: Run charts of utilisation in a perioperative process



(a) TOB (0.5)



(b) UKHC

Figure 3.: Run charts of patient flowtime in a perioperative process

List of Figures