Operational Research in Engineering Sciences: Theory and Applications
Vol. 6, Issue 4, 2023, pp.1-16
ISSN: 2620-1607
eISSN: 2620-1747
DOI: https://doi.org/10.31181/oresta/060401

A SCHEDULING HEURISTIC FOR A CONVEYOR BELTING TWO-STAGE UNIFORM MACHINES HYBRID FLOW SHOP

Domingo Pavolo[1*], Delson Chikobvu[1], Louis Lagrange[2]

[1] Department of Mathematical Statistics and Actuarial Sciences, Faculty of Natural andAgricultural Sciences, University of the Free State, P.O. Box 339, Bloemfontein 9300, South Africa.
[2]Department of Engineering Sciences, Faculty of Natural and Agricultural Sciences, University of the Free State, P.O. Box 339, Bloemfontein 9300, South Africa.

Received: 05 August 2023
Accepted: 10 September2023
First Online: 02 December2023

Research Paper

Abstract: *Most production planning focuses on allocating resources to jobs in unoptimised schedules. In this work, a bi-weekly job scheduling ensemble of heuristics for optimising makespan is developed for a two-stage Hybrid-Flow-Shop (HFS) with two similar machines in the first and four similar machines in the second. The HFS problem is NP-hard. An empirical experiment to investigate the performance of four heuristics in literature versus modifications by switching the shortest with the longest processing time job before scheduling was performed using a set of seven jobs. The seven jobs were Jackknifed to create sets of six jobs each to validate heuristic performances. Eight sets of four jobs randomly chosen from the seven were scheduled to investigate the performance of the heuristics when the number of jobs is equal to or less than the number of second-stage machines. Heuristic performance was measured using makespan and percentage deviation of the makespan from a selected lower bound. Results recommend an ensemble of three heuristics, the best makespan heuristic for jobs less than or equal to four and the two that begin by ordering jobs in increasing processing times, switch the shortest with the longest processing time job then list schedule jobs to machines.*

Keywords: *Hybrid-Flow-Shop Scheduling Problem, Optimisation of Makespan, Scheduling Heuristic, Ensemble.*

[*]Corresponding Author: dpavolo68@gmail.com (D. Pavolo)
Chikobvu@ufs.ac.za (D. Chikobvu), LagrangeLF@ufs.ac.za (L. Lagrange)

1. Introduction

A conveyor belting manufacturing entity did not have a conveyor belting orders production scheduling policy. A scheduling policy was required to ensure timeous delivery performance of orders to customers as the manufacturing entity's strategic competitive advantage is on fast and flexible delivery performance. The entity applies an Assemble-To-Order (ATO) production philosophy. Rubber compounds and conveyor reinforcement fabrics are imported and stocked awaiting precise conveyor belt order specifications from customers. The entity, thus, operates on a high inventory cost policy in order to achieve a fast and flexible delivery performance.

The conveyor belt manufacturing process model of the entity is a two-stage Hybrid-Flow-Shop (HFS). A hybrid flow shop has a set of parallel machines in at least one of its stages (Ruiz & Vázquez-Rodríguez, 2010). The first stage of the conveyor belt manufacturing process consists of two parallel calendaring machines for conveyor belting assembly and construction according to order specifications. The second stage is for curing or vulcanizing rubber covered conveyor belts and consists of four parallel vulcanizing presses. Each machine in the two stages works a single conveyor belt at any one time.

Figure 1 shows the two-stage HFS structure of the manufacturing process and the parallel machines in each stage. The objective is to minimize the total production time of orders at hand (makespan) at scheduling which is represented by **F2/P(m1=2, m2=4)/** C_{max} according to Graham et al. notation (Graham et al., 1979).

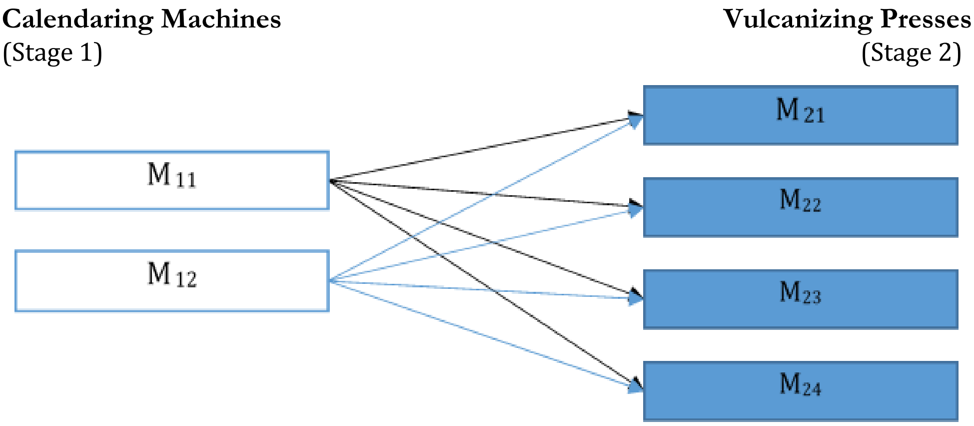

Figure 1: The Hybrid Floor Shop Model Structure of The Conveyor Belt Manufacturing Entity Process.

The basic principle of the solution methodology is to bi-weekly schedule available orders by:

i) Minimising both machine and job downtimes,
➢ Sum of machine downtimes in 1st stage machines should be = zero
➢ Sum of 1st stage job downtimes = 2nd stage machine downtime
ii) Spread jobs on second stage machines to minimise makespan

Let the HFS be described by the set $\boldsymbol{S} = \{\boldsymbol{S_1}, \boldsymbol{S_2}\}$(Gupta, 1988) of processing stages, in which stage $\boldsymbol{S_1}$ contains two similar machines $\boldsymbol{M_{1i}} = \{\boldsymbol{M_{11}}, \boldsymbol{M_{12}}\}$ and stage $\boldsymbol{S_2}$ contains four similar machines $\boldsymbol{M_{2i}} = \{\boldsymbol{M_{21}}, \boldsymbol{M_{22}}, \boldsymbol{M_{23}}, \boldsymbol{M_{24}}\}$. Similar machines are different from identical machines in that whereas identical machines have the same processing time for a given job, similar machines have different processing times for the same job although the times are related by a factor (Ruiz & Vázquez-Rodríguez, 2010).

Let the set of jobs to be processed at any one given time be $\boldsymbol{J_1}$ = {$\boldsymbol{J_1}$, ... , $\boldsymbol{J_n}$} and each job $\boldsymbol{J_1}$ is processed at stage $\boldsymbol{S_1}$ by machine $\boldsymbol{M_{1i}}$ for $\boldsymbol{a_{ji}}$ time and at stage $\boldsymbol{S_2}$ by machine

$\boldsymbol{M_{2i}}$ for $\boldsymbol{b_{ji}}$ time**.** Let the following assumptions hold for the HFS model:

i) The number of jobs, release times, and processing times are known and fixed.
ii) All jobs follow the same job sequence.
iii) No job may be cancelled, split, or pre-empted once in the process.
iv) No two operations of the same job maybe processed simultaneously.
v) Set-up time is independent of the job sequence and is therefore considered as part of the processing time.
vi) All machines of the same stage are similar.
vii) No machine can process more than one job at a time.

Let the makespan for machine $\boldsymbol{M_{21}}$ be $\boldsymbol{C^{21}_{max}}$ **,** then.

$$C^{21}_{max} = d_{21} + \sum_{j=1}^{n}(d_{1j} * X_{21j} + d_{21(j-1)}\} \quad \textbf{(1)}$$

Where

- $\boldsymbol{d_{21}}$ is the initial delay time of machine $\boldsymbol{M_{21}}$ before the first job arrives after beginning of processing at stage 1.
- $\boldsymbol{X_{21}}$ is a {1,0} variable, 1 if job $\boldsymbol{j_j}$ is allocated to machine $\boldsymbol{M_{21}}$ and 0 otherwise.
- $\boldsymbol{d_{21(j-1)}}$ is machine $\boldsymbol{M_{21}}$ downtime between job $\boldsymbol{j_j}$ and its predecessor on the same machine.

The equation (1) holds for all the other machines and can be generalised to:

$$C^{21}_{max} = d_{2i} + \sum_{j=1}^{n}(d_{1j} * X_{21j} + d_{2i(j-1)}\} \quad (2)$$

Where i = 1,2,3 or 4.

The objective of the algorithm is, thus, to sequence the jobs such that the maximum $\boldsymbol{C^{21}_{max}}$ is minimized. Therefore, the objective function of the problem is:

$$Minimize \{\max_{1 \le i \le 4}(C^{21}_{max})\} \quad (3)$$

Subject to the following first stage assignment constraint

$$\sum_{i=1}^{2} x_{1j} = 1 \quad (4)$$

And the following second stage assignment constraint

$$\sum_{i=1}^{4} x_{2ij} = 1 \quad (5)$$

And the single route per job constraint

$$\sum_{i=1}^{2} x_{1ij} * \sum_{i=1}^{4} x_{2ij} = 1 \text{ ....(j=1,2,.. n)} \quad (6)$$

A constraint restricting the starting time of job $\boldsymbol{j_j}$ on machine $\boldsymbol{M_{2i}}$ to be greater or equal to its release time from the 1st stage.

$$c^{21} - b_{ij} = d_{2i} + b_{ij}(X_{2ij} - 1) + d_{2i(j-1)} + \sum_{j=1}^{n-1}(b_{i(j-1)} * X_{2i(j-1)} + d_{2i(j-2)}\} \quad (7)$$

Finally, the second stage machine downtime limit constraint

$$d_{2i(j-1)} \le a_{ij} \quad (8)$$

The other assumption is that there are no other eligibility constraints like the width of conveyor belts.

2. Literature Review

The HFS problem has attracted a lot of attention because of its complexity and practical relevance (Colak & Keskin, 2021). It is being used in manufacturing (Hasani & Hosseini, 2020; Hwang & Lin, 2018; Peng et al., 2018), healthcare management () (Chabouh et al., 2018; Zhou et al., 2016), transportation (Boroun et al., 2020; Yong & Huizhen, 2017), cloud computing (Li & Han, 2020) and agriculture (Guan et al., 2017). It is a combination of the Parallel Machine Shop (PMS) and the classic Flow Shop (FS) problems with at least one stage with parallel identical, uniform or unrelated machines (Chen, 2023). The case at hand consists of similar parallel machines and is the least studied of the three.

On complexity, (Gupta, 1988) showed that the two stage HFS is NP-hard even if the problem has two identical machines at the first stage and one at the second stage. If identical parallel machines are NP-hard, it leads to the cases of similar and unrelated parallel machines also being NP-hard.

The most utilised objectives have been minimising makespan, minimising total tardiness or optimising total earliness (Colak & Keskin, 2021). This implies that effective resource and system utilisation and customer satisfaction are the most targeted. In this work, minimising makespan is the objective as it optimises the use of limited resources. Constraints have mainly been job or machine related (Colak & Keskin, 2021). Energy related constraints considered include variable speed levels, electricity/energy costs and turn on/off strategies in order to minimise energy consumption with the intention to increase more environmental friendly production (An et al., 2020; Li et al., 2023; Luo et al., 2019; Peng et al., 2021; Wu & Sun, 2018; Zheng & Wang, 2016). In this work, constraints are job/machine related. Optimising makespan, also optimises energy use per square metre of rubber covered conveyor belting since both the 1st stage and 2nd stage machines are heated with steam from a coal fired boiler (Thompson et al., 2015).

Table 2.1: Showing Some Relevant Algorithms for Minimizing Makespan in HFS Systems.

(Johnson, 1954)	Step 1: Partition the jobs into two sets $N_1 = \{j \in N: aj \geq bj\}$, $N_2 = \{j \in N: aj < bj\}$
F2/P(m_1=1,m_2=1)/ C_{max}	Step 2: The jobs from set N_1 go first in increasing order of a_j. (**SPT** on M1).
	Step 3: The jobs from set N_2follow in decreasing order of b_j. (**LPT** on M2).
List Scheduling	Step 1: Arrange the jobs in some pre-specified list.
	Step 2: Assign the first unscheduled job to the first available machine
(Gupta, 1988)	Step 1: Find a Johnson sequence L for jobs.
F2/P(m_1=2, m_2=1)/ C_{max}	Step 2: Modify L, if necessary, by bringing the job with minimum a_j to the first sequence position.
	Step 3: Fix the sequence L for jobs on M_2 (second stage machine).
	Step 4: For the first stage, assign the first unassigned job in the sequence L to the latest available machine so that no additional idle time is incurred on machine M_2. (If it is not possible, then assign job to the machine such that minimum additional idle time is incurred on M_2)
(Sriskandarajah & Sethi, 1989)	Step 1: Find a Johnson sequence L for the jobs.
	Step 2: Apply List Scheduling at the 1st stage machines (1st stage times are equal)
F2/P(m1=1,m2=1)/ C_{max}	Step 3: Apply the jobs to the 2nd stage machine in the order they finish at the 1st stage
	Step 1: Find a Johnson sequence for $\frac{a_j}{m_1}$ and $\frac{b_j}{m_2}$.
(Buten & Shen, 1973)	Step 2: Apply list scheduling to the 1st stage machine.
F2/P(m1=1,m2=1)/ C_{max}	Step 3: Apply the jobs to the 2nd stage machine in the order in which they finish at the 1st stage.

Solution methodologies have been exact, heuristic or metaheuristic (Chen, 2023). The use of exact methodologies such as branch and bound has been limited because of the complexity of the HFS scheduling problem, and has been favoured when the scale of the problem is small or to provide an initial solution to a heuristic (Ruiz & Vázquez-Rodríguez, 2010; Wang et al., 2015). For middle scale problems exact algorithms have large limitations and give way to approximate or heuristic solution methodologies which can lead to suboptimal solutions in limited time (Chen, 2023). Heuristics are thus limited to small and medium range problems or used in conjunction with metaheuristics. Metaheuristics have been applied in a majority of cases due to their effectiveness and performance in large scale problems, the most popular being genetic algorithms. Because of the size of the conveyor belt scheduling problem, exact or heuristic solution methodologies suffice. Bi-weekly scheduling of conveyor jobs keeps the number of jobs for scheduling small, thus keeping the problem within the capability of heuristics (Alqahtani et al., 2019).

Some relevant two-stage flow shop heuristic algorithms for minimising makespan found in literature are worth noting and are listed in Table 2.1.

Analysis of the above heuristics shows that their key characteristics can be grouped as follows:

1. What is used to construct the Johnson sequence (minimums, maximums or averages).
2. The final sequence type (Gupta, 1988; Johnson, 1954).
3. The 2nd stage job allocation method (Buten & Shen, 1973; Gupta, 1988)

Twelve different types of heuristics can therefore be developed and tested for the general two-stage HFS scheduling problem at hand from the four heuristics in Table 2.1 as shown in Table 2.3. The best heuristic algorithm for the problem at hand can, thus, be selected and developed from these variants.

Table 2.3: The Twelve Types of Heuristic That Can Be Developed.

#	Johnson Construction Using	Final Sequence Type	2nd Stage Job Allocation
1	Minimums	Johnson	Buten & Shen
2	Minimums	Gupta	Buten & Shen
3	Minimums	Johnson	Gupta
4	Minimums	Gupta	Gupta
5	Maximums	Johnson	Buten & Shen
6	Maximums	Gupta	Buten & Shen
7	Maximums	Johnson	Gupta
8	Maximums	Gupta	Gupta
9	Averages	Johnson	Buten & Shen
10	Averages	Gupta	Buten & Shen
11	Averages	Johnson	Gupta
12	Averages	Gupta	Gupta

3. Research Methodology

3.1 The Experimental Dataset

Processing times of seven conveyor belt job orders in the system, of the manufacturing entity, were estimated for all the machines in both the calendaring stage (stage 1) and vulcanization stage (stage 2) as shown in Table 3.1.

Table 3.1: Seven Jobs and Their Processing Times at Each Stage Machine.

	Stage 1	Machines	Stage 2	Machines		
	M_{11}	M_{12}	M_{21}	M_{22}	M_{23}	M_{24}
J_1	5	8	30	45	45	62
J_2	7	12	34	51	51	70
J_3	5	9	29	47	47	65
J_4	8	13	35	53	53	72
J_5	10	16	39	59	59	81
J_6	4	6	24	35	35	48
J_7	7	11	33	49	49	66

M_{ij} stands for a machine *j* in stage *i*, i.e. M_{12} is machine 2 in stage 1. J_1 stands for job number 1 in the set of jobs to be scheduled. It can therefore be deduced that the processing time of J_1 on machine M_{11} is 5 hours, 8 hrs on machine M_{12}, 30 hours on machine M_{21}, 45 hours on machine M_{22}, 45 hours on machine M_{23} and 62 hours on machine M_{24}. The basic principle is to minimize $\sum\sum \boldsymbol{a_{ji}} + \sum\sum \boldsymbol{b_{ji}}$ at every allocation of a job to a machine.

3.2 The Experimental Design

The twelve heuristics of Table 2.3 collapse into the four heuristics of Table 3.2 below. Jobs in Table 3.1 were scheduled using the heuristics in Table 3.2 to estimate their makespans.

Table 3.2: The Four Heuristics That Were First Tested.

Heuristic		Description
SPT-LS$_1$-LS$_2$(SPT)	i)	Sort jobs according to machine M_{11} shortest processing time first (**SPT**).
	ii)	List schedule jobs to 1st **stage** machines (**LS$_1$**), minimizing cumulative processing times.
	iii)	List schedule jobs to **2nd stage** machines starting from the **fastest** processing time machine (**LS$_2$(SPT)**).
SPT-LS$_1$-LS$_2$(LPT)	i)	Sort jobs according to machine M_{11} shortest processing time first (**SPT**).
	ii)	List schedule jobs to 1st **stage** machines (**LS$_1$**), minimizing cumulative processing times.
	iii)	Allocate jobs to **2nd stage** machines starting from the **slowest** processing time machine (**LS$_2$(LPT)**).
LPT-LS$_1$-LS$_2$(SPT)	i)	Sort jobs according to machine M_{11} **longest** processing time first (**LPT**).
	ii)	List schedule jobs to 1st **stage** machines (**LS$_1$**), minimizing cumulative processing times.
	iii)	List schedule jobs to **2nd stage** machines starting from the **fastest** processing time machine (**LS$_2$(SPT)**).
LPT-LS$_1$-LS$_2$(LPT)	i)	Sort jobs according to machine M_{11} **longest** processing time first (**LPT**).
	ii)	List schedule jobs to 1st **stage** machines (**LS$_1$**), minimizing cumulative processing times.
	iii)	List schedule jobs to **2nd stage** machines starting from the **slowest** processing time machine (**LS$_2$(LPT)**).

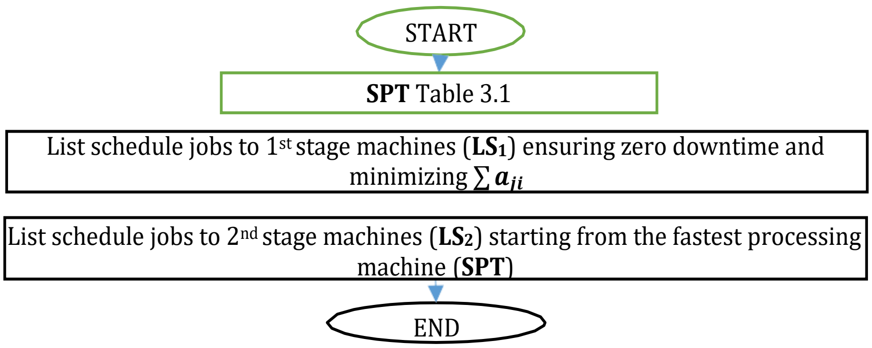

*Figure 2: Showing the flow diagram of the heuristic **SPT-LS1-LS2(SPT)** shown in Table 3.2s*

(a) The **first schedule runs** of all the seven jobs by the four heuristics is shown in Figure 3 below.

SPT

	M_{11}	M_{12}	M_{21}	M_{22}	M_{23}	M_{24}
J_6	4	6	24	35	35	48
J_1	5	8	30	45	45	62
J_3	5	9	29	47	47	65
J_2	7	12	34	51	51	70
J_7	7	11	33	49	49	66
J_4	8	13	35	53	53	72
J_5	10	16	39	59	59	81

LS_1 / LS_2(SPT)

	M_{11}	M_{12}	M_{21}	M_{22}	M_{23}	M_{24}
J_6	4		28			
J_1		8		53		
J_3	9				56	
J_2	16					86
J_7		19			105	
J_4	24			106		
J_5	34		73			

SPT

	M_{11}	M_{12}	M_{21}	M_{22}	M_{23}	M_{24}
J_6	4	6	24	35	35	48
J_1	5	8	30	45	45	62
J_3	5	9	29	47	47	65
J_2	7	12	34	51	51	70
J_7	7	11	33	49	49	66
J_4	8	13	35	53	53	72
J_5	10	16	39	59	59	81

LS_1 / LS_2(LPT)

	M_{11}	M_{12}	M_{21}	M_{22}	M_{23}	M_{24}
J_6	4					52
J_1		8			53	
J_3	9			56		
J_2	16		50			
J_7		19		105		
J_4	24				106	
J_5	34		89			

LPT

	M_{11}	M_{12}	M_{21}	M_{22}	M_{23}	M_{24}
J_5	10	16	39	59	59	81
J_4	8	13	35	53	53	72
J_2	7	12	34	51	51	70
J_7	7	11	33	49	49	66
J_3	5	9	29	47	47	65
J_1	5	8	30	45	45	62
J_6	4	6	24	35	35	48

LS_1 / LS_2(SPT)

	M_{11}	M_{12}	M_{21}	M_{22}	M_{23}	M_{24}
J_5	10		49			
J_4		13		66		
J_2	17				68	
J_7		24				90
J_3	22		78			
J_1	27			111		
J_6		30			103	

LPT

	M_{11}	M_{12}	M_{21}	M_{22}	M_{23}	M_{24}
J_5	10	16	39	59	59	81
J_4	8	13	35	53	53	72
J_2	7	12	34	51	51	70
J_7	7	11	33	49	49	66
J_3	5	9	29	47	47	65
J_1	5	8	30	45	45	62
J_6	4	6	24	35	35	48

LS_1 / LS_2(LPT)

	M_{11}	M_{12}	M_{21}	M_{22}	M_{23}	M_{24}
J_5	10					91
J_4		13			66	
J_2	17			68		
J_7		24	57			
J_3	22		86			
J_1	27				111	
J_6		30		103		

Figure 3: Results of the First Heuristic Runs on Excel Showing The Steps of Each Heuristic and Resultant Schedule.

Two tables are shown in a row for each heuristic algorithm in Figure 3. The first table of each run shows the ordering by shortest processing time (SPT) or longest processing time (LPT) of Table 3.1 depending on the heuristic algorithm used. The second table shows the list scheduling of the jobs to the machines, both for the 1st and 2nd stages. The machine allocation to a job by a heuristic is shown by the red processing time in the column of the machine and the second table shows the subsequent job completion times. The red-boxed 2nd-stage highest time is the heuristic's makes pan $\{max(C^{21}_{max})\}$. Recalling equation (3)

$$\boldsymbol{C^{21}_{max} = d_{2i} + \sum_{j=1}^{n}\{b_{ij} * X_{2ij} + d_{2i(j-1)}\} = d_{2i} + b_{16} * X_{21(6)} + b_{14} * X_{21(4)} + d_{21(4)}}$$

For M21, which first processes J6 then J4:

The time machine M21 spends waiting for job J4: $\boldsymbol{d_{21(6)}} = \mathbf{4}$
For job J6: $\boldsymbol{b_{16} * X_{21(6)}} = (\mathbf{24 * 1}) = \mathbf{24}$
For job J4: $\boldsymbol{b_{14} * X_{21(4)}} = (\mathbf{39 * 1}) = \mathbf{39}$
The time machine M21 spends waiting for job J4 after completing job J6: $\boldsymbol{d_{21(4)}} = [\mathbf{34} - \mathbf{28}] = \mathbf{6}$,
Hence

$$\boldsymbol{C_{max}^{21}} == \mathbf{4 + 24 + 6 + 39 = 73}$$

For the first schedule run **SPT-LS$_1$-LS$_2$(SPT)** of the seven jobs

Table 3.3: Showing The Heuristic Algorithm Components Of $\boldsymbol{C_{max}^{21}}$ = For $\boldsymbol{M_{2i}}$

M_{2i}	$d_{2i(j-1)}$	$b_{1(j-1)} * X_{21(j-1)}$	$b_{1j} * X_{21(j)}$	$d_{21(j)}$	C_{max}^{21}
M_{21}	4	24	39	6	73
M_{22}	8	45	53	0	106
M_{23}	9	47	49	0	105
M_{24}	16	70	0	0	76

Therefore, for **SPT-LS$_1$-LS$_2$(SPT),** $max(\boldsymbol{C_{max}^{21}}) = \mathbf{106}$**.** For the other heuristic algorithms**:**
SPT-LS$_1$-LS$_2$(SPT), $max(\boldsymbol{C_{max}^{21}}) = \mathbf{106}$
LPT-LS$_1$-LS$_2$(SPT), $max(\boldsymbol{C_{max}^{21}}) = \mathbf{1}11$
LPT-LS$_1$-LS$_2$(LPT), $max(\boldsymbol{C_{max}^{21}}) = \mathbf{1}11$
Therefore**,** $\boldsymbol{min}\{max(\boldsymbol{C_{max}^{21}}) = \mathbf{106}$ for the four heuristic algorithms**.**

(b) The **second schedule runs** involved dropping one job at a time from the seven jobs and determining the new heuristic algorithm makespans of the remaining six jobs by all the four heuristic algorithms, returning the dropped job and dropping the next one (a form of Jackknifing). This was meant to increase the number of scheduling experimental runs from one to eight so as to have a better understanding of the behaviour of the HFS system.

(c) After the **first** and **second scheduling runs**, the shortest processing time job and the longest processing time job were switched after ordering by shortest processing time (SPT) before scheduling runs for the eight sets of jobs. This was done to investigate the effect of scheduling the shortest processing time job last after accounting for the longest processing time job by either the fastest or slowest time machine.

(d) Finally, the **last scheduling runs** involved arbitrarily selecting sets of four jobs each from the seven jobs and determining their makespans using all the four heuristic algorithms. This was to investigate the effect of moving from scheduling the number of jobs below or equal to the number of second stage machines.

In all the runs a comparative measure of optimality is used which computes the percentage deviation of the optimal makespan from the makespan lower bound (LB) which in this case is the makespan of the second stage machine M24 which is determined by the **SPT-LS$_1$-LS$_2$(SPT)** heuristic for a given set of jobs for job sets greater than 4 and **SPT-LS$_1$-LS$_2$(LPT)** for job sets equal or less than 4.

3.3 Summary of Experimental Design

The summary of the empirical experimental design is presented in Table 3.4 below, which shows the three types of job datasets used, the two types of heuristics tested and the performance measures used. It is important to note that the Jackknifed sets were more for the validation of the relative heuristic performance noted in the experiments involving the seven jobs in the order book.

Table 3.4: Showing The Summary of The Empirical Experimental Design.

#	Size of Sets of Jobs Scheduled	Four Popular Heuristics in Literature	The Same Four Heuristics with Switching
1	7 jobs in order book	Performance Measure - Makespan - % deviation from LB	Performance Measure - Makespan - % deviation from LB
2	7 sets of 6 jobs each **Jackknifed** from the 7 jobs	Validation of Performance Performance Measure - Makespan - % deviation from LB	Validation of Performance Performance Measure - Makespan - % deviation from LB
3	8 sets of 4 jobs each randomly selected from the 7	Performance Measure - Makespan - % deviation from LB	Performance Measure - Makespan - % deviation from LB

4. Results

The results from the empirical experiments in Section 3 are provided in this Section. The results for scheduling the full set of 7 jobs are provided together with those for the Jackknifed sets. The results for heuristics with switching are in a different table to highlight the impact of switching. The results of investigating scheduling of the number of jobs equal to or less than the number of second stage machines are also presented in a separate table.

4.1 Results of Scheduling Tests Runs (a) and (b)

Table 4.1: Makespans and % deviations from LB of both the full and the jackknifed sets of orders

	M_{24} Makespan (Lower Bound)	SPT-LS_1-LS_2(SPT)	SPT-LS_1-LS_2(LPT)	LPT-LS_1-LS_2(SPT)	LPT-LS_1-LS_2(LPT)	{(M-LB)/LB} %
FULL	90	106	106	111	111	18%
J-1	91	109	109	101	101	11%
J-2	85	106	106	101	101	19%
J-3	85	106	106	101	101	19%
J-4	85	104	104	98	98	15%
J-5	82	102	102	98	98	20%
J-6	86	109	109	111	111	27%
J-7	90	106	106	101	101	12%
AVERAGE		106	106	102.75	102.75	

NB: J-*i* indicates the full list of jobs minus job number *i.* The last column indicates the deviation from the makespan lower bound (LB) shown in the 2nd column of the optimum makespan M shaded in yellow.

The results show the following:

- There is agreement in the makespan results between the two shortest processing time (SPT) heuristics despite the difference in list scheduling methods (**SPT** against **LPT)** to the 2nd stage machines.
- Similarly, the two longest processing time (LPT) heuristic algorithm agree on the optimal makespans despite the difference in list scheduling methods (**SPT** against **LPT)** to the 2nd stage machines.

- Comparing the SPT heuristics against the LPT heuristics, out of the eight schedule runs, the LPT heuristics have better makespans 75% of the times.

4.2 Results of Switching the Shortest Processing Time Job with The Longest Processing Time Job (See Annexure)

Table 4.2: Makespan Results of Switching Shortest Job and Longest Job After SPT.

	M24	SPT-LS$_1$-LS$_2$(SPT)	SPT-S-LS$_1$-LS$_2$(SPT)	SPT-LS$_1$-LS$_2$(LPT)	SPT-S-LS$_1$-LS$_2$(LPT)	{(M-LB)/LB} %
FULL	90	106	102	106	102	14%
J-1	91	109	91	109	91	0%
J-2	85	106	88	106	91	3%
J-3	85	106	88	106	91	3%
J-4	85	104	88	104	91	3%
J-5	82	102	90	102	88	7%
J-6	86	109	101	109	101	17%
J-7	90	106	90	106	91	0%
AVE.		106	**92.25**	106	**93.25**	

Note: The added S in SPT-S-LS1-LS2(SPT) and SPT-S-LS1-LS2(LPT) is for switching the first and last job after SPT. In the formula {(M-LB)/LB} %, M is for the makespan and LB is the Lower Bound which is the M24 makespan for the SPT-S-LS1-LS2(SPT) heuristic.

Table 4.2 shows that switching the longest and the shortest processing time jobs after arranging the jobs according to shortest processing time first (SPT) before list scheduling jobs to the HFS system improves the makespan significantly. Note the significant drop in the % deviations from the lower bound between Tables 4.1 and 4.2. The LPT heuristic algorithms could not be improved by switching since the longest processing time job will already be at the start and the shortest at the end.

4.3 Results of Scheduling of Four Jobs at A Time for The Different Heuristics

Table 4.3: Makespans for 8 sets of 4 jobs.

	SPT-LS$_1$-LS$_2$(SPT)	SPT-LS$_1$-LS$_2$(LPT)	LPT-LS$_1$-LS$_2$(SPT)	LPT-LS$_1$-LS$_2$(LPT)	{(M-LB)/LB} %
JOBS					
J2,J1,J3,J6	86	56	62	77	8%
J5,J1,J3,J7	82	56	62	73	8%
J1,J3,J2,J7	85	67	82	77	0%
J6,J3,J2,J7	82	63	64	77	21%
J6,J1,J2,J7	81	62	64	77	21%
J6,J1,J3,J2	86	56	62	77	8%
J6,J1,J3,J2	86	56	64	77	8%
J6,J1,J3,J7	82	56	62	73	8%

NOTE: LB is M_{24} of SPT-LS$_1$-LS$_2$(LPT) and M is for makespan in the formula **[(M-LB)/LB} %.**

The best makespans for each schedule of jobs are shown shaded in yellow. It is observed that when only four jobs (equal to the number of second stage machines) are considered at a time, the SPT-LS1-LS2(LPT) performs better than all the other three heuristics.

5. Discussion of Results

The results are analysed in three stages: (i) when the number of orders for scheduling is equal to or less than the number of 2nd stage machines, (ii) when the number of orders is above the number of 2nd stage machines but equal to or less than eight, then (iii) performance according to % deviation from lower bound. The final discussion is whether a best single heuristic or an ensemble of heuristics should be recommended for use.

5.1 Conveyor Belt Orders Are 4 Or less.

SPT-LS1-LS2(LPT), see Table 4.3 above, gives the best makespan results, when the number of orders for scheduling is 4 or less. Four is the number of the 2nd stage machines.

Table 5.1: The SPT-LS1-LS2(LPT) Heuristic Algorithm.

	i)	Sort jobs, using machine M_{11} times, in increasing order of processing time (SPT).
SPT-LS_1-LS_2(LPT)	ii)	List schedule jobs to 1st **stage** machines, minimizing cumulative processing times.
	iii)	List schedule jobs to **2nd stage** machines starting from the **slowest** machine (LPT).

When the shortest job is allocated first to the slowest machine, the longest processing time job is reserved for the fastest machine. This gives the SPT-LS-LS(LPT) heuristic the effect of averaging 2nd stage machine makespans, thus achieving the best performance in comparison to the other three heuristics, when the number of jobs is equal to or below the number of 2nd stage machines.

5.2 Conveyor Belt Orders Are Above 4 But Below 8

Table 4.1 shows the makespan results when the number of jobs is higher than the number of 2nd stage machines. SPT heuristics agree on the optimum makespan, just as the LPT heuristics do. However, the LPT heuristics have better optimum makespans for 75% of the time than the SPT heuristics. There is, thus, a shift from SPT heuristics to dominantly LPT heuristics when the number of orders for scheduling are more than the number of 2nd stage machines.

The addition of the step of **switching** the shortest processing time job with the longest processing time job after SPT, but before list scheduling of jobs to machines, improves the makespans from the SPT heuristics to values better than those of the LTP heuristics. In fact, the heuristic SPT-S-LS1-LS2(SPT) performs better most of the time compared to the rest. This suggests that **switching** is a necessary step and should be part of the final scheduling heuristic for this HFS problem.

5.3 % Deviation from Lower Bound

Table 5.2: Comparing Heuristic Performances.

#	Heuristic	Best % Deviation	Worst % Deviation
1	LPT- LS_1-LS_2(SPT) {4<Jobs<8]	11%	27%
2	SPT-LS_1-LS_2(LPT) {Jobs $\leq$ 4}	0%	21%
3	SPT-S-LS_1-LS_2(SPT) {4<Jobs<8	0%	17%

Table 5.2 shows the computed best and worst percentage deviations from the lower bound (LB) makespan for comparison of heuristic performances. The deviation from the LB of the makespans is computed using the formula {(M-LB)/LB} %, where M represents the makespan and LB the lower bound of the makespans for the heuristic under consideration.

Heuristic #1, in red, is the best among the four popular heuristics in literature. Heuristic #2 has the best performance when the number of jobs is equal to or less than the number of 2nd stage machines. Heuristic #3 has the best performance of the two heuristics improved by **switching** the shortest processing time job with the longest processing time job after ordering the jobs in increasing processing times (SPT). Table 5.2 shows that heuristics #2 and #3 perform better than heuristic #1 when judged using the % deviation from the LB. Heuristics #1 and #3 use the same makespan LB, the makespan of the machine M24 when the **SPT-LS$_1$-LS$_2$(SPT)** heuristic is run. The LB used for the case of jobs less than or equal to the number of 2nd stage machines is the makespan of M24 for the heuristic **SPT-LS$_1$-LS$_2$(LPT).**

5.4 Best Heuristic Selection

Having observed the dynamics in Sections 5.1 and 5.2, the question of best heuristic becomes necessary. The objective of the work is to recommend a scheduling methodology which identifies an optimum makespan schedule for available belting orders. It has been observed that SPT-LS1-LS2(LPT) performs better than all the other three when the number of orders are equal to or less than the number of 2nd stage machines but this changes when the number of orders increase above the number of 2nd stage machines. At any scheduling occasion, orders can be below or above the number of 2nd stage machines. It is, therefore, not prudent to recommend a single best heuristic among the ones considered. The ensembling theory permits the use of several heuristics to obtain a number of solutions which is then combined or fused to obtain the best answer. The best option was, therefore, to keep the two SPT heuristics, SPT-S-LS1-LS2(SPT) and SPT-S-LS1-LS2(LPT), plus SPT-LS1- LS2(LPT) so as not to lose any possible optimal makespans.

6. Conclusion

The proposed ensemble of heuristics has three SPT base heuristics shown in Table 6.1. Any other competitive heuristic or metaheuristic can be added to the ensemble as long as the optimal makespan is the least of the heuristics and or metaheuristic makespans.

Table 6.1: Three SPT Heuristics Make Up the Final Ensemble.

SPT-LS$_1$-LS$_2$(LPT)	i) Sort jobs in order of increasing processing times (SPT) using machine M_{11}. ii) List schedule jobs to 1st **stage** machines, minimizing cumulative processing times. i) List schedule jobs to **2nd stage** machines starting from the **slowest** machine (LPT).
SPT-S-LS$_1$-LS$_2$(SPT)	ii) Sort jobs in order of increasing processing times (SPT) using machine M_{11}. iii) Switch the shortest processing time job with the longest processing time job. iv) List schedule jobs to 1st **stage** machines, minimizing cumulative processing times. v) List schedule jobs to **2nd stage** machines starting from the **shortest processing time** machine (SPT) for the first four jobs. vi) Allocate all the remaining jobs to 2nd stage machines starting from the machine with the shortest makespan and eliminate the longest makespans.
SPT-S-LS$_1$-LS$_2$(LPT)	i) Sort jobs in order of decreasing processing times (SPT) using to machine M_{11}. ii) Switch the shortest processing time job with the longest processing time job iii) List schedule jobs to 1st **stage** machines, minimizing cumulative processing times. iv) List schedule jobs to **2nd stage** machines starting from the **longest processing time** machine (LPT) for the first four jobs. v) Allocate all the remaining jobs to 2nd stage machines starting from the machine with the shortest makespan and eliminate the longest makespans.

The ensemble of heuristics is run as shown in Figure 3. The objective is to come out with the optimum makespan, that is $minimum\{max(C_{max}^{2i})\}$.

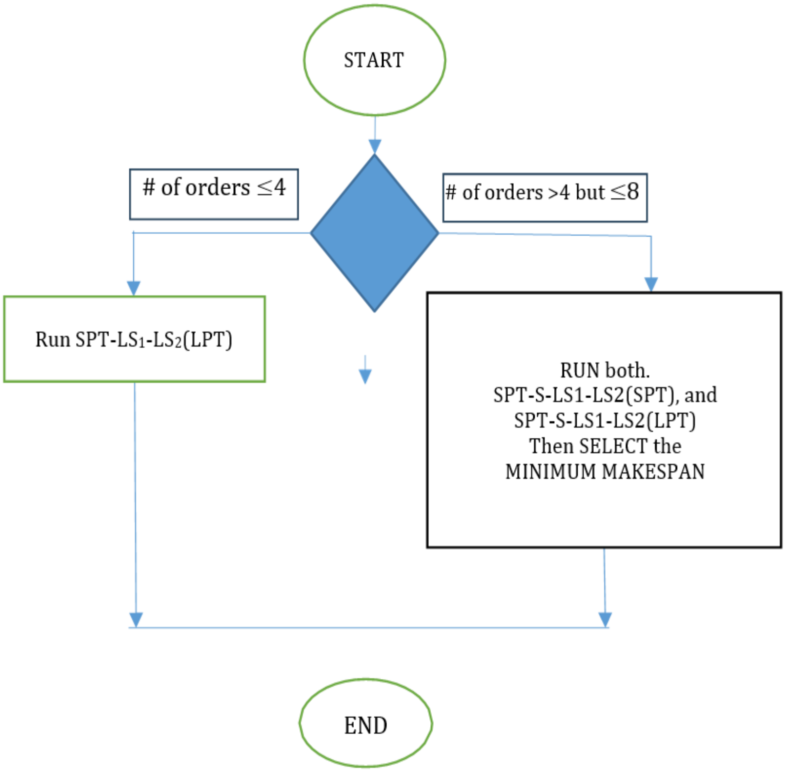

Figure 4: Showing The Process Diagram of Running the Ensemble of Heuristics of Table 6.1.

7. Limitations

The study looked at producing a scheduling heuristic of conveyor belt orders to use bi- weekly on Monday and Thursday. In the manufacturing entity considered, the number of belting orders for scheduling rarely go beyond twice the number of the 2nd stage machines. Scheduling only considers processable belting orders that have raw materials in stock. So, the study is limited to this environment.

8. Areas of Further Research

There is need to look for Metaheuristics for scheduling orders of more than eight jobs for the same HFS problem if ever they occur.

There is also need to look for the best software for coding the ensemble of heuristics for faster accurate and efficient scheduling.

- ***Supplementary Materials***

Annexure: Illustration of the effect of switching the shortest processing time job with the longest processing time job before list scheduling.

	SPT		SWITCH					LS_1		LS_2(SPT)			
	M11	M12	M21	M22	M23	M24		M11	M12	M21	M22	M23	M24
J5	10	16	39	59	59	81	J5	10		49			
J1	5	8	30	45	45	62	J1		8		53		
J3	5	9	29	47	47	65	J3	15				62	
J2	7	12	34	51	51	70	J2		20				90
J7	7	11	33	49	49	66	J7	22			102		
J4	8	13	35	53	53	72	J4	30		84			
J6	4	6	24	35	35	48	J6		26			97	

	SPT		SWITCH					LS_1		LS_2(LPT)			
	M11	M12	M21	M22	M23	M24		M11	M12	M21	M22	M23	M24
J5	10	16	39	59	59	81	J4	10					91
J1	5	8	30	45	45	62	J1		8			53	
J3	5	9	29	47	47	65	J3	15			62		
J2	7	12	34	51	51	70	J7		20	54			
J7	7	11	33	49	49	66	J2	22				102	
J4	8	13	35	53	53	72	J5	30		89			
J6	4	6	24	35	35	48	J6		26		97		

- ***Acknowledgments***: We would like to express our gratitude to the University of the Free State in South Africa for covering the publication costs.

- ***Author Contributions***: All the three authors mentioned contributed in thought and authorship of the manuscript in their various specialities.
- ***Conflicts of Interest***: The authors declare no conflict of interest.

References

Alqahtani, Z., El-Shahed, M., & Mottram, N. (2019). Derivative-order-dependent stability and transient behaviour in a predator–prey system of fractional differential equations. *Letters in Biomathematics*, *6*(1), 32-49. https://doi.org/10.30707/LiB6.2Alqahtani

An, Y., Chen, X., Zhang, J., & Li, Y. (2020). A hybrid multi-objective evolutionary algorithm to integrate optimization of the production scheduling and imperfect cutting tool maintenance considering total energy consumption. *Journal of Cleaner Production*, *268*, 121540. https://doi.org/10.1016/j.jclepro.2020.121540

Boroun, M., Ramezani, S., Vasheghani Farahani, N., Hassannayebi, E., Abolmaali, S., & Shakibayifar, M. (2020). An efficient heuristic method for joint optimization of train scheduling and stop planning on double-track railway systems. *INFOR: Information Systems and Operational Research*, *58*(4), 652-679. https://doi.org/10.1080/03155986.2020.1746100

Buten, R. E., & Shen, V. Y. (1973). A scheduling model for computer systems with two classes of processors. In *Proceedings Sagamore Computer Conference on Parallel Processing* (pp. 130-138). Purdue University. https://www.proquest.com/openview/48964b3adaac16075ef9b3f4b4759502/1?pq-origsite=gscholar&cbl=18750&diss=y

Chabouh, S., Hammami, S., Marcon, E., & Bouchriha, H. (2018). Appointment scheduling of inpatients and outpatients in a multistage integrated surgical suite: application to a Tunisian ophthalmology surgery department. *Journal of Simulation*, *12*(1), 67-75. https://doi.org/10.1080/17477778.2017.1398288

Chen, H. (2023). A summary of algorithm research on hybrid flow-shop scheduling problem *CONF-CIAP 2023*, *5*. https://doi.org/10.54254/2753-8818/5/20230280

Colak, M., & Keskin, G. A. (2021). An extensive and systematic literature review for hybrid flowshop scheduling problems. *International Journal of Industrial Engineering Computations*, *13*(2). https://doi.org/10.5267/j.ijiec.2021.12.001

Graham, R. L., Lawler, E. L., Lenstra, J. K., & Kan, A. R. (1979). Optimization and approximation in deterministic sequencing and scheduling: a survey. In *Annals of discrete mathematics* (Vol. 5, pp. 287-326). Elsevier. https://doi.org/10.1016/S0167-5060(08)70356-X

Guan, S., Shikanai, T., Nakamura, M., & Fukami, K. (2017). Mathematical model and solution for land-use crop planning with cooperative work. In *2017 6th IIAI International Congress on Advanced Applied Informatics (IIAI-AAI)* (pp. 903-908). IEEE. https://doi.org/10.1109/IIAI-AAI.2017.110

Gupta, J. N. (1988). Two-stage, hybrid flowshop scheduling problem. *Journal of the Operational Research Society*, *39*(4), 359-364. https://doi.org/10.2307/2582115

Hasani, A., & Hosseini, S. M. H. (2020). A bi-objective flexible flow shop scheduling problem with machine-dependent processing stages: Trade-off between production costs and energy consumption. *Applied Mathematics and Computation*, *386*, 125533. https://doi.org/10.1016/j.amc.2020.125533

Hwang, F., & Lin, B. M. (2018). Survey and extensions of manufacturing models in two-stage flexible flow shops with dedicated machines. *Computers & Operations Research*, *98*, 103-112. https://doi.org/10.1016/j.cor.2018.05.016

Johnson, S. M. (1954). Optimal two-and three-stage production schedules with setup times included. *Naval research logistics quarterly*, *1*(1), 61-68. https://doi.org/10.1002/nav.3800010110

Li, J.-q., & Han, Y.-q. (2020). A hybrid multi-objective artificial bee colony algorithm for flexible task scheduling problems in cloud computing system. *Cluster Computing*, *23*(4), 2483-2499. https://doi.org/10.1007/s10586-019-03022-z

Li, J., Li, H., He, P., Xu, L., He, K., & Liu, S. (2023). Flexible Job Shop Scheduling Optimization for Green Manufacturing Based on Improved Multi-Objective Wolf Pack Algorithm. *Applied Sciences*, *13*(14), 8535. https://doi.org/10.3390/app13148535

Luo, S., Zhang, L., & Fan, Y. (2019). Energy-efficient scheduling for multi-objective flexible job shops with variable processing speeds by grey wolf optimization. *Journal of Cleaner Production*, *234*, 1365-1384. https://doi.org/10.1016/j.jclepro.2019.06.151

Peng, K., Pan, Q.-K., Gao, L., Zhang, B., & Pang, X. (2018). An improved artificial bee colony algorithm for real-world hybrid flowshop rescheduling in steelmaking-refining-continuous casting process. *Computers & Industrial Engineering*, *122*, 235-250. https://doi.org/10.1016/j.cie.2018.05.056

Peng, Z., Zhang, H., Tang, H., Feng, Y., & Yin, W. (2021). Research on flexible job-shop scheduling problem in green sustainable manufacturing based on learning effect. *Journal of Intelligent Manufacturing*, *33*, 1-22. https://doi.org/10.1007/s10845-020-01713-8

Ruiz, R., & Vázquez-Rodríguez, J. A. (2010). The hybrid flow shop scheduling problem.

European Journal of Operational Research, *205*(1), 1-18. https://doi.org/10.1016/j.ejor.2009.09.024

Sriskandarajah, C., & Sethi, S. P. (1989). Scheduling algorithms for flexible flowshops: worst and average case performance. *European Journal of Operational Research*, *43*(2), 143-160. https://doi.org/10.1016/0377-2217(89)90208-7

Thompson, E., Everett, J., Rowell, J. T., Rychtář, J., & Rueppell, O. (2015). The evolution of cooperation is affected by the persistence of fitness effects, the neighborhood size and their interaction. *Letters in Biomathematics*, *2*(1), 67. https://doi.org/10.30707/LiB2.1Thompson

Wang, S., Liu, M., & Chu, C. (2015). A branch-and-bound algorithm for two-stage no-wait hybrid flow-shop scheduling. *International journal of production research*, *53*(4), 1143-1167. https://doi.org/10.1080/00207543.2014.949363

Wu, X., & Sun, Y. (2018). A green scheduling algorithm for flexible job shop with energy-saving measures. *Journal of Cleaner Production*, *172*, 3249-3264. https://doi.org/10.1016/j.jclepro.2017.10.342

Yong, Y., & Huizhen, Z. (2017). Wolf algorithm for vehicle rout-ing problem with multiple dis-tribution centers *Computer Application Research*, *2017*(9), 2590-2593.

Zheng, X.-L., & Wang, L. (2016). A collaborative multiobjective fruit fly optimization algorithm for the resource constrained unrelated parallel machine green scheduling problem. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, *48*(5), 790-800. https://doi.org/10.1109/TSMC.2016.2616347

Zhou, B.-h., Yin, M., & Lu, Z.-q. (2016). An improved Lagrangian relaxation heuristic for the scheduling problem of operating theatres. *Computers & Industrial Engineering*, *101*, 490-503. https://doi.org/10.1016/j.cie.2016.09.003