ARTICLE IN PRESS

Journal of King Saud University – Engineering Sciences xxx (xxxx) xxx

Contents lists available at ScienceDirect

Journal of King Saud University – Engineering Sciences

journal homepage: www.sciencedirect.com

Original article

The job sequencing and tool switching problem with sequence-dependent setup time

Setyo Tri Windras Mara[a], Edi Sutoyo[b], Rachmadi Norcahyo[a], Achmad Pratama Rifai[a,*]

[a] Department of Mechanical and Industrial Engineering, Faculty of Engineering, Universitas Gadjah Mada, Indonesia
[b] Department of Information Systems, School of Industrial Engineering, Telkom University, Indonesia

ARTICLE INFO

Article history:
Received 12 November 2020
Accepted 27 February 2021
Available online xxxx

Keywords:
Flexible manufacturing system
Tool management
Job sequencing and tool switching
Sequence-dependent setup time

ABSTRACT

The job sequencing and tool switching (SSP) is a well-known combinatorial optimization problem that aims to minimize the number of tool switches. The problem mainly arises in a flexible manufacturing system, in which machines can be configured with various tools to process different jobs and tool switches will correspond to a reduction of productivity. While the general studies assume uniform setup time for tool switching, applications on the industries indicate that the setup time of a tool might depend on the previously installed tool of the same slot. Therefore, this article proposes two integer linear programming models for the sequence-dependent SSP, namely the five-index formulation and the multicommodity flow formulation. Experimental tests are executed to compare the sequence-dependent SSP with the uniform SSP and the results show the effectiveness of multicommodity flow formulation. Finally, several managerial discussions are derived from the results.

1. Introduction

One of the prerequisites to build a smart factory is high flexibility, in which factory can react to changes and demand in producing a large variety of products. The highly-flexible technology in the production operations have long been manifested in the concept of Flexible Manufacturing System (FMS) (El-Tamimi et al., 2012). The FMS allows the machines and system to handle a large variety of product, even those with complex machining requirements. Generally, FMS is constituted by a number of computer numerical control (CNC) machining centers that have the flexibility to perform various kind of jobs as long as the required tools are available in the tool magazine. The ability of this flexible machine to perform different types of operations (e.g. drilling, milling, turning) without requiring an abrupt change in between makes the production more dynamic and raises the competitiveness of the company (Paiva and Carvalho, 2017; Milind, 2018).

The limitation of the possible operations that a machine can handle is bounded by the available tools installed in its magazine. Subsequently, the magazine capacity then limits the number of tools that a CNC machining center can carry. Meanwhile, the demand to process various type of products requires the machining centers to be equipped with more tools. In this condition, the problem of tool switching arises as the number of required tools to process a group of products exceeds the magazine capacity. Tool switching is performed by replacing the currently installed tools in the magazine with other types of tool required to process the next jobs. When performing tool switching, the machine should be in a non-processing mode to ensure safety. In some conditions, such as in a bottleneck workstation, the tool switching can interrupt the production line, and reduce the production rate of the FMS. It also requires more human labor as installing tools in the magazine may be performed manually. Because it is a non-value added and time-consuming task, the tool switching operations must be minimized to attain the full advantage of FMS and optimize the production process (Paiva and Carvalho, 2017).

Since the switching operation is decided by the currently installed tools and the required tools for the next jobs, the tool switching problem is affected by the job sequencing. Thus, the problem is then known as the Job Sequencing and Tool Switching Problem (SSP) which traditionally occurs in the metalworking industry (Amaya et al., 2008; Catanzaro et al., 2015). Nevertheless, the SSP can also be found in other industries, such as in a mailroom inserting systems planning (Furrer and Mütze, 2017; Adjiashvili

* Corresponding author.
E-mail addresses: setyotriw@ugm.ac.id (S.T. Windras Mara), edisutoyo@telkomuniversity.ac.id (E. Sutoyo), rachmadinorcahyo@ugm.ac.id (R. Norcahyo), achmad.p.rifai@ugm.ac.id (A.P. Rifai).

Peer review under responsibility of King Saud University.

Production and hosting by Elsevier

https://doi.org/10.1016/j.jksues.2021.02.015
1018-3639/

et al., 2015), allocation management of a computer memory (Ghiani et al., 2007), an order picking system with crane (Schwerdfeger and Boysen, 2017), and pharmaceutical packaging in chemical manufacturing plant (Mütze, 2014).

Tang and Denardo (1988) proved the intertwined relationship between the job sequencing problem (TP) and the tool switching problem (SP). With a known job sequence and uniform required time for switching operation, the TP can be solved in polynomial time by the Keep Tool Needed Soonest (KTNS) policy (Tang and Denardo, 1988). Meanwhile, the SP sub-problem has been proven as *NP*-hard (Crama et al., 1994). Hence, the majority of literature on SSP has been dedicated to optimizing the SP sub-problem. Laporte et al. (2004) presented an integer linear programming (ILP) formulation as an alternative to the model from Tang and Denardo (1988), in which a Travelling Salesman Problem (TSP)-based formulation was used due to the similarity of SP with the classical routing problem. Then, a branch-and-cut and branch-and-bound algorithms were developed to determine the optimal solution of the SSP. The analogy of SP as a TSP was also employed by Ghiani et al. (2010) who formulated that when the number of tools needed to process each job equals to the capacity of the magazine, the problem was reduced to a TSP. Catanzaro et al. (2015) proposed novel ILP formulations for the SSP which have a tighter Linear Programming (LP) relaxation than the existing models. They investigated the combinatorial interpretation of their formulations and strengthened inequalities to improve the lower bounds provided by the LP relaxation.

Current research on the uniform SSP presents the development of both the mathematical model and heuristics method for solving the problem. Chaves et al. (2016) combined the Biased Random Key Genetic Algorithm (BRKGA) and the clustering search for the job sequencing sub-problem. Baykasoğlu and Ozsoydan (2017) developed a simulated annealing algorithm, while also considering the Automatic Tool Changer Indexing problem which aims to find the optimum allocation of the required cutting tools on the slots of a turret magazine. Ahmadi et al. (2018) modeled the job sequencing sub-problem as a Traveling Salesman Problem of Second Order (2-TSP) and proposed dynamic Q-learning-based genetic algorithm to solve the problem. Atta et al. (2019) proposed an improvement of harmony search (HS) algorithm for solving the tool indexing problem. Amaya et al. (2019) developed a novel memetic model by integrating cooperative optimization with the idea of deep metaheuristics resulting in a unified model for tackling complex optimization problems. In a recent study, da Silva et al. (2020) presented a new mathematical model of SSP based on a multicommodity flow formulation and proved that the lower bound of their proposed model is equal to the number of tools minus the capacity of tool magazine.

As the uniform SSP can represent the general problem of switching problem faced by industries, the consideration of non-uniform setup time for the tool switching operations can be critical when the condition on the manufacturing floor requires the removed and installed tools to have different setup time. Such non-uniform setup time can be affected by several factors, one of them is by the type of tools to be removed and the new tools to be installed which then called as sequence-dependent setup time. Within this situation, the classic KTNS policy is not relevant to be applied solely since it is indifferent to the detail of the tool switching operations. This policy cannot decide the switching pair in which it cannot determine which currently installed tool should be replaced with a new tool so that the setup time is minimized.

Setup operations are non-productive tasks performed on the machines in order to prepare them for the next products. These operations mainly comprise of cleaning, fixing or releasing parts to machines, changing tools, and adjusting the tools to the machines (Pan et al., 2017). The significance of considering sequence-dependent setup time is that it mimics the real condition in a manufacturing environment as certain machines and tools require different setup time depending on the previously and to be installed tools. The sequence-dependent setup time has been widely explored in various scheduling problems. Some recent literature have considered the sequence-dependent setup time such as in hybrid flowshop (Pan et al., 2017), blocking flowshop (Shao et al., 2018), flexible job shop (Shen et al., 2018), single machine scheduling (Nesello et al., 2018), unrelated parallel machine scheduling (Bektur and Saraç, 2019), identical parallel machine (Kim et al., 2020), and open shop scheduling (Abreu et al., 2020). Nevertheless, the scenario of sequence-dependent setup time is seldom mentioned in the context of SSP. Meanwhile, the SSP requires detailed planning of tool switching to minimize the setup time in which the existing models for sequence-dependent setup time in those various problems are apparently not prevalent to be directly applied. Therefore, there is a necessity to develop a new model of SSP which can accommodate the issue of sequence-dependent setup time.

To the best of our knowledge, most of the previous research in SSP concerned about the uniform switching time (Calmels, 2019), in which the model is built under the assumption of uniform and sequence-independent setup time during the tool switching operations. As most studies concerned about uniform job sequencing and tool switching problem, the existing models suffer from the negligence of some authors to state the properties of setup times since for some practical applications, the assumption of uniform setup time would be too unrealistic (Calmels, 2019). Therefore, to overcome this gap, this study presents two new ILP formulations which accommodate the issues of sequence-dependent setup time.

2. Problem formulation

This section describes the sequence-dependent SSP in detail. The discussion starts with a brief introduction on the basic concept of uniform SSP in Section 2.1. Afterwards, Section 2.2 explains the sequence-dependent SSP along with the presentation of our proposed ILP formulations: the five-index formulation and the multi-commodity flow formulation.

2.1. Uniform SSP

The uniform SSP is a hard combinatorial optimization problem that commonly arises in a FMS. Let $J = \{1, 2, 3, \cdots, n\}$ be a set of jobs to be executed in the flexible machine and let $T = \{1, 2, 3, \cdots, m\}$ be a set of available tools to process the jobs in J. Each job $j \in J$ requires a specific subset of tools T_j to process and must satisfy $C \geq max_{j \in J}\{|T_j|\}$, where C is the capacity of tool magazine which expresses the maximum number of tools that can be installed in the machine at any given time.

In a common practical situation, all available tools T cannot be installed on the magazine at once since $C < m$. Therefore, a tool switch which is executed by removing tools in the magazine slots and installing the new ones might be necessary to process the next job. The switching cost translates to machine idleness. Hence, minimizing the switching cost through optimizing both SP and TP simultaneously as an SSP is important for increasing the productivity of the production line.

There are two factors that determine the switching cost: (i) the sequence of job processing, and (ii) tool switching schedule, which respectively correspond to the SP and TP. These factors are taken into account in the uniform SSP by simultaneously determining the sequence of job processing and the subsets of tools in the magazine for each processed job with the main goal to minimize the total switching cost. Since it is assumed that all pairs of tools have the same switching cost, thus, the order of tool in the magazine is irrelevant in the uniform SSP and the objective function can be set for minimizing the total number of tool switches. For the mathematical formulations of uniform SSP, readers can consult the works of Tang and Denardo (1988), Laporte et al. (2004), Catanzaro et al. (2015), and da Silva et al. (2020).

2.2. Sequence-Dependent SSP

This study proposes two mathematical models for the SSP with non-uniform and sequence-dependent setup times of the tool switches. These models are developed based on the state-of-the-art formulations of uniform SSP. The first model is a five-index formulation which extends the model from Catanzaro et al. (2015), while the second proposal is a multicommodity flow formulation which is adapted from the work of da Silva et al. (2020).

In order to describe the sequence-dependent SSP, let $V = \{1, 2, 3, \cdots, C\}$ be a set of magazine slots of the flexible machine. In the uniform SSP, the order of the tools in a magazine is irrelevant (Laporte et al., 2004) and the information of the currently installed tool$t \in T$ in magazine slot $v \in V$ can be neglected. On the other hand, this information is extremely necessary in the sequence-dependent SSP since the value of setup time $s_{t,u}$ is depending on the removed tool $t \in T$ and inserted tool $u \in T$ at the same magazine slot. Hence, the main goal of SSP to minimize switching cost cannot be attained by only considering the number of tool switches and the created tool switching schedule must determine at which slot v the new tool u should be installed by considering the existing tool t at that slot. Further, several basic assumptions of uniform SSP are held in the sequence-dependent SSP, such as (i) a single flexible machine is used, (ii) job processing times are unitary, (iii) each tool can be loaded into any magazine slot, (iv) the workpiece setup time is uniform, (v) the size of the tool is uniform, and (vi) a tool only occupies a single magazine slot.

2.2.1. Five-index formulation

Here, the five-index formulation is presented as our first proposal in this study. Similar to the works of Laporte et al. (2004) and Catanzaro et al. (2015), this formulation models the SP subproblem according to the classical TSP. In order to describe the formulation in detail, let us denote $\mathscr{J} = J \cup \{0\}$. The dummy job $\{0\}$ is introduced to represent the start and stop operations, which is identical to the role of a depot in TSP. Consequently, we can set $T_0 = \emptyset$. Further, several properties of an optimal Hamiltonian circuit also must be followed here, these are: (i) started and ended at $\{0\}$, (ii) visiting all remaining vertices (jobs) exactly once. Similarly, let us define $\mathscr{T} = T \cup \{0\}$ to capture the initial installations of tools for the first job.

This formulation requires three decision variables. First, x_{ij} is a binary decision variable that has the value of 1 if job j follows job i immediately, and 0 otherwise. Second, $y_j^{t,v}$ is a binary decision variable that has the value of 1 if tool t is plugged into slot v when job j is processed, and 0 otherwise. Third, $z_{i,j}^{t,u,v}$ is a binary variable which has the value of 1 if tool u is inserted to slot v in order to process job j by replacing tool t which used to process job i and 0 otherwise. Then, using these notations, the formulation can be presented as follow.

Objective function:

$$\min F = \sum_{\substack{i,j \in \mathscr{J} \\ i \neq j}} \sum_{t \in \mathscr{T}} \sum_{u \in T} \sum_{v \in V} z_{i,j}^{t,u,v} s_{t,u} \tag{1}$$

Subject to:

$$\sum_{\substack{j \in \mathscr{J} \\ i \neq j}} x_{ij} = 1 \forall i \in \mathscr{J} \tag{2}$$

$$\sum_{\substack{i \in \mathscr{J} \\ i \neq j}} x_{ij} = 1 \forall j \in \mathscr{J} \tag{3}$$

$$\sum_{\substack{i,j \in S \\ i \neq j}} x_{ij} \leq |S| - 1 \forall S \subset \mathscr{J}, 2 \leq |S| \leq n - 1 \tag{4}$$

$$\sum_{t \in T} \sum_{v \in V} y_j^{t,v} \leq C \forall j \in J \tag{5}$$

$$\sum_{v \in V} y_j^{t,v} = 1 \forall j \in J, t \in T_j \tag{6}$$

$$\sum_{v \in V} y_j^{t,v} \leq 1 \forall j \in J, t \notin T_j \tag{7}$$

$$\sum_{t \in T} y_j^{t,v} = 1 \forall j \in J, v \in V \tag{8}$$

$$x_{ij} + y_j^{u,v} - y_i^{u,v} \leq 1 + \sum_{\substack{t \in T \\ t \neq u}} z_{i,j}^{t,u,v} \forall i,j \in J, i \neq j$$

$$u \in T, v \in V \tag{9}$$

$$x_{0j} + y_j^{u,v} \leq 1 + z_{0,j}^{0,u,v} \forall j \in J, u \in T, v \in V \tag{10}$$

$$x_{ij} \geq z_{i,j}^{t,u,v} \forall i \in \mathscr{J}, j \in J, i \neq j,$$

$$t \in T, u \in T, v \in V \tag{11}$$

$$y_i^{t,v} \geq z_{i,j}^{t,u,v} \forall i,j \in J, i \neq j,$$

$$t \in T, u \in T, v \in V \tag{12}$$

$$\sum_{u \in T_j} \sum_{v \in V} z_{i,j}^{t,u,v} \leq 1 \forall i,j \in J, i \neq j, t \in T \tag{13}$$

$$\sum_{t \in T} \sum_{v \in V} z_{i,j}^{t,u,v} \leq 1 \forall i,j \in J, i \neq j, u \in T_j \tag{14}$$

$$\sum_{\substack{j \in J \\ i \neq j}} \sum_{t \in \mathscr{T}} \sum_{u \in T} \sum_{v \in V} z_{i,j}^{t,u,v} \leq C \forall i \in \mathscr{J} \tag{15}$$

$$x_{ij} \in \{0,1\} \forall i \in \mathscr{J}, j \in \mathscr{J}, i \neq j \tag{16}$$

$$y_j^{t,v} \in \{0,1\} \forall j \in J, t \in T, v \in V \tag{17}$$

$$z_{i,j}^{t,u,v} \in \{0,1\} \forall i \in \mathscr{J}, j \in J, i \neq j,$$

$$t \in \mathscr{T}, u \in T, v \in V \tag{18}$$

The objective function for the sequence-dependent SSP is minimizing total setup time, which can be represented by Equation (1). The objective function (1) is subject to a set of constraints adapted from Catanzaro et al. (2015) and Laporte et al. (2004). The first three constraints are classical TSP constraints to determine the sequence of jobs. Constraints (2) and (3) are assignment constraints to ensure that all jobs are executed only once, while Constraint (4) is the subtours elimination constraint. The schedule of tools is then handled by Constraints (5)-(7). The number of tools that can be plugged into the magazine slots at the same time is limited by the capacity constraint in Eq. (5). To guarantee that each tool is only used once for each job and only appears in one magazine slot, Constraints (6), (7), and (8) are then deployed. Constraints (9)-(10) ensure that tools for executing the job are available in the magazine at a given time. Constraint (9) allows job$j$ to be processed after job i if and only if the required tools are available either through the previous job $i$or tool switches, while Constraint (10) guarantees the availability of tools for the initial job. Constraint (11)-(12) provides a warranty that a tool switch will only be executed to the installed tools on active job transitions. Then, Constraint (13) and Constraint (14) guarantee that for each job transition (i,j), a tool will only be replaced once and by one tool only at maximum. To sum up, Constraints (9)-(14) ensure the continuity of the job sequence and tool switching schedule. The number of tool switches simultaneously executed in any given time is no more than the number of magazine slots. This limitation is assured by Constraint (15). Lastly, Constraints (16)-(18) limit the value of decision variables as binary variables.

2.2.2. Multicommodity flow formulation

The multicommodity flow formulation models the transfer flow of several commodities within a graph. Consider a graph $G = (\mathscr{J}, V_k, A)$ where V_k denotes the set of available tools to process each k-th job when $k \in J$ and a set of dummy tool $\{0\}$ when $k = 0$, which represents the origin node of all operations. The subset V_k is formally defined in Equation (19). On the other hand, set A comprises of all arcs from nodes in V_k to nodes in V_{k+1} when $k \in [0, 1, \cdots, n-1]$.

$$V_k = \begin{cases} \{0\}, \forall k = 0 \\ \{1, \cdots, m\}, \forall k \in J \end{cases} \tag{19}$$

Fig. 1 shows a graph of a simple case illustration for the sequence-dependent SSP with $n = 6, m = 5, C = 3$. From Fig. 1, one can notice that when $k \in J$, the subset V_k contains the set of all tools. Then, we can easily model the instalment of tools for each magazine slot on each k-th job as a flow of single commodity by restricting the number of flows entering all nodes in V_k as C and restricting the number of flows entering any single node in V_k as 1. From this explanation, we can decipher the commodity flows in Fig. 1 as a tool switching schedule in Table 1. The first row presents the generated job sequence. Afterwards, the subsequent rows describe the installed tools at each given time, which can be translated into the tool switching schedule. For clarity, the circled tools are the tools that are inserted to the machine by a tool switching mechanism.

Similar to the work of da Silva et al. (2020), this formulation requires two decision variables. First, p_{ik} is a binary decision variable that takes the value 1 if job i is the k-th job in the sequence, and 0 otherwise. Second, $q_{t,u,k,v}$ is a binary decision variable that takes the value 1 if tool u is plugged into slot v to replace tool t in order to process the k-th job, and 0 otherwise. Finally, the multicommodity flow formulation of sequence-dependent SSP can be fully presented as follow.

Objective function:

$$\min F = \sum_{\substack{t \in V_{k-1} \\ t \neq u}} \sum_{u \in V_k} \sum_{k \in J} \sum_{v \in V} q_{t,u,k,v} s_{t,u} \tag{19}$$

Subject to:

$$\sum_{k \in J} p_{ik} = 1 \forall i \in J \tag{20}$$

$$\sum_{i \in J} p_{ik} = 1 \forall k \in J \tag{21}$$

$$\sum_{t \in V_{k-1}} \sum_{u \in V_k} \sum_{v \in V} q_{t,u,k,v} = C \forall k \in J \tag{22}$$

$$\sum_{t \in V_{k-1}} \sum_{v \in V} q_{t,u,k,v} \geq p_{ik} \forall i, k \in J, u \in T_i \tag{23}$$

$$\sum_{s \in V_{k-1}} q_{s,t,k,v} = \sum_{u \in V_{k+1}} q_{t,u,k,v} \forall t \in T, v \in V,$$

$$k \in [1, \cdots, n-1] \tag{24}$$

$$\sum_{t \in V_{k-1}} \sum_{v \in V} q_{t,u,k,v} \leq 1 \forall u \in T, k \in J \tag{25}$$

$$\sum_{t \in V_{k-1}} \sum_{u \in T} q_{t,u,k,v} \leq 1 \forall v \in V, k \in J \tag{26}$$

$$p_{ik} \in \{0, 1\} \forall i, k \in J \tag{27}$$

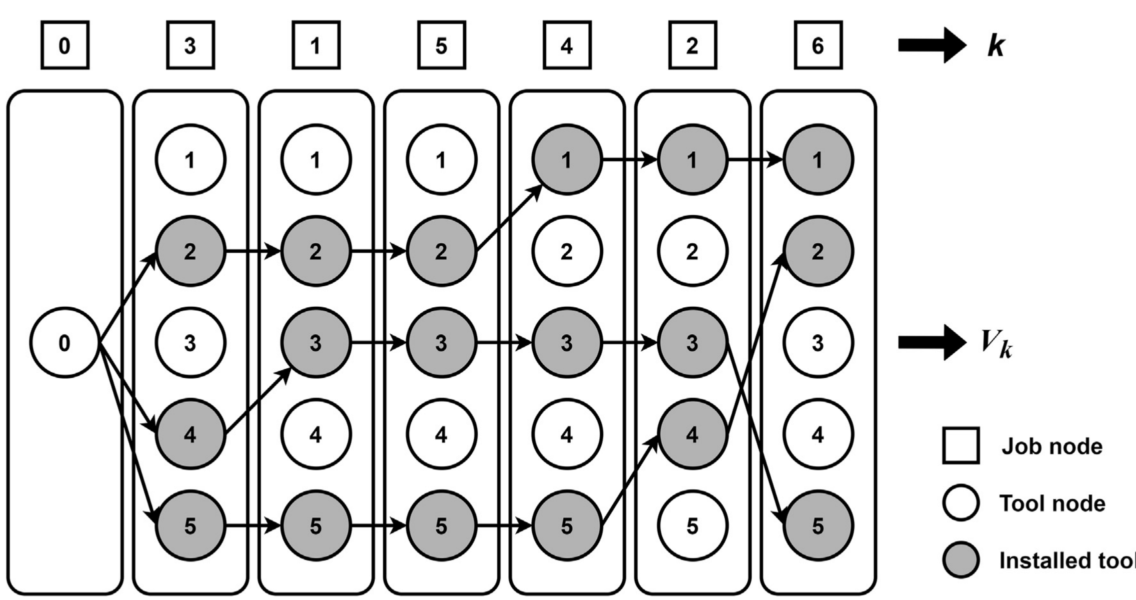

Fig. 1. A case illustration of multicommodity flow graph for the sequence-dependent SSP with $n = 6$, $m = 5$,$C = 3$

Table 1
The tool switching schedule derived from Fig. 1.

Jobs	3	1	5	4	2	6
Tools	②	2	2	①	1	⑤
	④	③	3	3	3	②
	⑤	5	5	5	④	4

$$q_{t,u,k,v} \in \{0,1\} \forall t \in V_{k-1}, u \in V_k,$$

$$k \in J, v \in V \tag{28}$$

The objective function (19) calculates the product between tool switches and its corresponding sequence-dependent setup times. Constraints (20)-(21) determine the sequence of all available jobs. Constraint (22) serves as a capacity constraint of the tool magazine for each job and Constraint (23) ensures the availability of tools required to execute each job. Afterwards, the flows of commodities in graph G are regulated by Constraints (24)-(26). Equality (24) is the flow conservation constraint to ensures the continuity of the flows. For each job, Constraint (25) guarantees that only one commodity enters each tool node at most, while Constraint (26) provides a similar warranty for each magazine slot. Lastly, Equations (27)-(28) define the value of the decision variables.

3. Numerical experiments

In this subsection, a comparison of the uniform SSP and the sequence-dependent SSP is executed to show how relaxing the assumption of uniform setup time can lead to different schedules of job sequencing and tool switching tasks. For this purpose, the multicommodity flow formulation of uniform SSP from da Silva et al. (2020) is implemented in Section 3.1 with four hours cut off time (Fernández et al., 2019) as a benchmark to showcase the difference between uniform and sequence-dependent SSP. Subsequently, the performances of our proposed models are also evaluated in Section 3.1 to find the better formulation for the sequence-dependent SSP and further experimental discussions are provided in Section 3.2 based on the results.

All codes are written and executed in Python 3.7 with GUROBI 9.0.2. Experiments are carried out in a personal computer with Intel® Core™ i7-10770 CPU 2.90 GHz with 32 GB of RAM and a Windows 10 operating system.

3.1. Comparison of uniform SSP and sequence-dependent SSP

First, let us consider an example of SSP instance from Catanzaro et al. (2015) in Table 2. The first row represents the jobs J with $n = 10$, and the next four rows depict the required tool to process each job T_j with $C = 4$. Meanwhile, the empty entries in each column show that job requires a smaller number of tools than the magazine capacity. Then, let us assume that the sequence-dependent setup time is asymmetric as shown in Table 3. The row depicts the removed tools, while the column presents the installed tool, i.e. replacing t_1 with t_5 requires a setup time $s_{1,5} = 3$. Because the setup time is assumed asymmetric, then the reverse process has a different value, i.e. $s_{5,1} = 2$. The initial loading is the condition where the tools are installed into the empty slot which happens during the setup for the first job. The setup cost for this case is expressed as initial setup time $s_{0,u}$, where u is the index of the installed tool. The values of the initial setup time are presented in Table 4.

Table 5 shows the optimal solutions of the instance from Table 2 under two conditions: (a) the uniform SSP and (b) the sequence-

Table 2
An example of SSP instance (Catanzaro et al., 2015).

Jobs	1	2	3	4	5	6	7	8	9	10
Tools	1	1	2	1	3	1	1	6	3	5
	4	3	6	5	5	2				7
	8	5	7	7	8	4				
	9		8	9						

Table 3
The sequence-dependent setup time.

t	Tool	1	2	3	4	5	6	7	8	9
$t-1$	1	–	1	5	1	3	2	5	3	2
	2	1	–	1	5	3	4	2	1	3
	3	2	2	–	5	2	4	5	3	5
	4	2	1	1	–	4	4	5	4	5
	5	2	2	1	3	–	1	3	3	4
	6	1	4	4	5	4	–	5	2	2
	7	2	4	4	4	2	3	–	5	4
	8	1	3	5	5	1	2	3	–	1
	9	5	4	3	3	3	5	4	5	–

Table 4
The initial setup time.

Tool	1	2	3	4	5	6	7	8	9
Initial setup time	2	3	2	4	5	5	3	3	1

Table 5
The optimal solutions for the instance shown in Table 2.

(a) solution for uniform SSP										
Jobs	7	10	4	2	5	9	1	6	8	3
Tools	1	1	1	1	1	1	1	1	1	7
	9	9	9	9	8	8	8	8	8	8
	7	7	7	3	3	3	4	4	6	6
	5	5	5	5	5	5	9	2	2	2
(b) solution for sequence-dependent SSP										
Jobs	6	1	7	4	2	9	5	10	3	8
Tools	1	1	1	1	1	2	8	8	8	8
	2	8	5	5	5	5	5	5	6	6
	4	4	2	7	7	7	7	7	7	7
	9	9	9	9	3	3	3	2	2	2

dependent SSP. Here, the readers may notice that not all tools installed in the magazine are the tools required to execute the currently-processed job. These tools are kept in the magazine to avoid unnecessary switches by the KTNS policy. For example, the job $j = 10$ only requires two tools $\{5, 7\}$ to be processed, but in the solution (a), tools $\{1, 9\}$ are kept in the magazine since those tools will be required soon to process the next jobs.

The uniform SSP and the sequence-dependent SSP generate two substantially different optimal solutions. In the solution (a), the optimal job sequence $P(a) = \{7, 10, 4, 2, 5, 9, 1, 6, 8, 3\}$ is generated. There are totally 11 switches for the generated switching schedule: four switches classified as the initial tool insertion while the rests are switches in between two consecutive jobs processing. On the other hand, by relaxing the assumption of setup time uniformity, the solution (b) generates different job sequence $P(b) = \{6, 1, 7, 4, 2, 9, 5, 10, 3, 8\}$ with a higher number of switches $Z = 13$ where four switches classified as the initial tool insertion. In terms of setup time, the total setup time of solution (b) is 23 s. Then, let us assume that the solution (a) from the uniform SSP model also shares similar information on setup times in Table 3 and Table 4 (relaxing the assumption of uniformity). Using this information, these 11 tool switches from solution (a) produce 42 s of total setup time. Thus, although may result in a higher number of tool switches, it is observed that considering the sequence-dependent issue can be beneficial for the decision-maker for composing a better tool switching schedule.

Here, computational tests are performed on a set of 22 sequence-dependent SSP instances generated with various combinations of n, m, and C. The aim of this evaluation is to highlight the difficulty of solving the sequence-dependent SSP, as well as comparing the performance of our proposed formulations. For comparison purpose, the mathematical formulation of uniform SSP from da Silva et al. (2020) is again borrowed.

Table 6 presents the computational experiment results, in which column $F(s)$ stands for the total setup time and column Z shows the total switches. For the uniform SSP, it is shown that within four hours of running time in a state-of-the-art computing hardware, the mathematical formulation from da Silva et al. (2020) can find the optimal solutions for 19 from 22 instances. On the other hand, for the sequence-dependent SSP, the five-index formulation can only optimally solve instances with equal or less than $n = 10$, $m = 10$, and $C = 5$, while the multicommodity flow formulation performs better and can solve larger instances until $n = 10$, $m = 15$, and $C = 12$.

Overall, the results show that by considering the sequence-dependent setup time, the SSP becomes a more complex combinatorial problem. This stems from the consideration to seek an optimum pair replacement schedule, which enlarges the search space

Table 6
Computational experiment results.

Instance	n	m	C	Uniform SSP (da Silva et al., 2020)			Sequence-dependent SSP: Five-index Formulation				Sequence-dependent SSP: Multicommodity Flow Formulation			
				Z	Gap (%)	Runtime (s)	$F(s)$	Z	Gap (%)	Runtime (s)	$F(s)$	Z	Gap (%)	Runtime (s)
datA1	5	5	2	5	0	0.01	8	5	0	0.09	8	5	0	0.11
datA2	5	5	3	6	0	0.01	9	6	0	0.21	9	6	0	0.19
datA3	5	8	4	8	0	0.03	14	7	0	6.3	14	7	0	0.27
datA4	5	8	6	8	0	0.19	22	7	0	10.81	22	7	0	0.13
datB1	8	8	3	10	0	0.12	16	13	0	2.14	16	11	0	7.32
datB2	8	8	4	9	0	0.17	13	11	0	22.73	13	10	0	2.77
datB3	8	10	4	10	0	0.09	18	12	0	779.93	18	12	0	5.8
datB4	8	10	6	10	0	0.15	15	11	0	1553.76	15	11	0	9.31
datB5	8	10	8	10	0	0.08	17	11	0	456.44	17	11	0	1.72
datC1	10	10	4	12	0	2.64	18	16	0	607.17	18	16	0	388.99
datC2	10	10	5	9	0	0.08	10	10	0	5305.18	10	10	0	23.02
datC3	10	10	6	10	0	0.05	12	11	33.33	14,400	12	10	0	16.76
datC4	10	10	7	10	0	0.02	15	10	20	14,400	15	10	0	1.76
datC5	10	15	6	16	0	3.81	26	21	57.69	14,400	25	23	0	4841.57
datC6	10	15	12	17	0	76.61	49	24	28.57	14,400	43	19	0	3421.53
datD1	12	20	8	22	0	169.57	71	51	83.1	14,400	27	22	3.7	14,400
datD2	12	20	12	24	0	353.91	128	72	78.91	14,400	56	31	26.79	14,400
datD3	12	20	16	23	0	187.99	120	80	75.83	14,400	50	28	14	14,400
datD4	15	20	6	29	0	1757.11	58	46	84.48	14,400	42	41	45.24	14,400
datD5	15	20	8	22	9.09	14,400	–	–	–	14,400	32	30	31.25	14,400
datE1	30	40	15	130	69.23	14,400	–	–	–	14,400	401	315	94.01	14,400
datF1	40	60	20	284	78.87	14,400	–	–	–	14,400	–	–	–	14,400

of the tooling subproblem. The results also support the previous discussion that the optimal solution from uniform SSP model that minimizing the number of switches may be a suboptimal solution for the sequence-dependent SSP instances, and the optimal solution for the sequence-dependent SSP instances may result in a higher number of switches but with a lower total setup time.

The uniform SSP model is obviously unsuitable to be applied for the sequence-dependent problems as it lacks the ability to determine the optimum switching setup times. Therefore, from the perspective of sequence-dependent SSP, the uniform model results on an unfinished solution which needs to be further refined to determine the switching pair. However, as the uniform model aims to minimize the number of switches (Tang and Denardo, 1988; Laporte et al., 2004; da Silva et al., 2020), the job sequences provided might not deliver the best setup times, as indicated by the results in Table 4 where the job sequences that deliver the minimum number of switches are often different with the job sequences that deliver the minimum setup times. Thus, the uniform model fails to obtain the optimal solution for sequence-dependent SSP. In real-world manufacturing scenarios, the approach for minimizing setup times might be more practical than minimizing the number of switches, as setup time directly correlates to the manufacturing lead time. In addition, the uniform model generalizes an important issue that switching tools might require different time as it is affected by the type of tools to be replaced and type of tool to be installed (Adjiashvili et al., 2015). For example, switching an end milling tool with a turning tool requires a longer switching time as the two tools have different tool holders. Meanwhile, switching an end milling tool with another end milling tool with different size may require a shorter time than the previous process.

Further, these experiments highlight the efficiency of the multicommodity flow formulation adapted from da Silva et al. (2020) compared to the proposal of five-index formulation adapted from Catanzaro et al. (2015). This finding confirms the conclusion of da Silva et al. (2020) who proved the superiority of a multicommodity flow formulation for the uniform SSP.

In regard to the computation time, the experiment results show that within four hours of running time in state-of-the-art computing hardware, the commercial solver GUROBI 9.0.2 can only optimally solve instances with equal or less than $n = 10$, $m = 10$, and $C = 5$. Conversely, the uniform SSP model (da Silva et al., 2020) can find the optimal solutions for the first nine instances within four hours running time.

3.2. Discussion

This subsection presents the sensitivity analysis to further analyze the impact of density (tool usage rate) on the complexity of the problems. First, we define Ω as the ratio of the number of required tools to process a job over the overall number of tools. For this analysis, we generate new instances with three levels of density. The density of each job is generated using a uniform distribution, with the value of $U(025, 0.5)$, $U(0.5, 0.75)$, and $U(0.75, 1.0)$ for low (level 1), medium (level 2), and high density (level 3), respectively. The multicommodity flow formulation models are used here as it delivered better efficiency than the five-index formulation in the previous experiments. Table 7 presents the results for computation with the new instances.

The results indicate that the setup time increases as the density level increase. It is understandable since the higher level of density means there are fewer unused slots which can store the required tools for the next jobs. Hence, more switches occur, which correlates to the increase of setup time. Fig. 2 depicts the results of sensitivity analysis: (a) number of switches, (b) setup time, and (c) computation times. The depicted results only include up to data27 since the GUROBI often fails to find the optimality in larger datasets.

The graph in Fig. 2 indicates the increasing trend of the number of switches, setup time, and running time as the density level increases. Therefore, it shows that the complexity increases along with the increase of density, which further confirms that the search space enlarges. The higher density means a higher number of solutions since the jobs require more tool switches in a limited capacity. In contrast, in the problems with lower density, the required tools for the subsequent jobs can be installed earlier, hence reducing the number of switches. Consequently, the computational time increases as the algorithm need more time to explore the larger solution space. Further, this experiment highlights the need for an effective and efficient heuristic solution for the sequence-dependent SSP (Calmels, 2019), including solving the problem with larger density and higher complexity.

Based on the results, we also draw some insights regarding the sequence-dependent SSP model. First, the proposed sequence-dependent SSP model can capture the amplitude of the mean setup time as indicated by Fig. 3. The increase in mean setup time for each tool pair linearly increases the total setup time. Meanwhile, the objective of minimizing the number of switches fails to highlight this issue as the number of switches remains constant albeit the switching pair mean setup time is increasing. Thus, the uniform model tends to hide this waste. As the increasing on the total setup time is identified, the decision-maker can further analyze and takes decisions for optimizing the production, such as transferring some jobs to other machine cells.

Second, in the case where the sequence-dependent setup times are highly varied, the proposed model can find the optimum switching pairs which result on the minimum switching time, whereas the uniform SSP models do not capture this phenomenon. This condition might happen if there is a large number of types of tool required for processing jobs. Fig. 4 presents this situation where the increase in the tool type variation is represented by the increase in the standard deviation of sequence-dependent setup times. Thus, the higher the standard deviation (higher variation of tools types and its setup times), the determination of switching pairs is getting more significant effect on the total setup time.

Here, the proposed models can effectively arrange the switching pairs so that the total setup time is lowered, resembling the effort of the machine operator to exploit the tool switching pairs that are easier to execute. Note that the sequence-dependent SSP is reduced to the uniform SSP when the standard deviation of setup-times is zero as any currently installed tools can be replaced with any new tools for processing the next jobs without altering the setup time.

4. Summary and future research directions

This article proposed two mathematical formulations for the sequence-dependent SSP, namely the five-index formulation and the multicommodity flow formulation. A comparison to the uniform SSP model from da Silva et al. (2020) is executed to study the effect of considering the sequence-dependent setup time that commonly happens in industrial applications. Then, the performances of the proposed formulations are also compared in order to find a more suitable model for the problem considered.

Overall, the results provide three main conclusions. First, it is shown that by considering the sequence-dependent setup time, the tooling problem becomes a complex combinatorial problem that seeks the optimum pair replacement schedule between the removed and installed tools in each interlude between job processing, which is not considered in the uniform SSP. Second, the multi-

Table 7
Results of sensitivity analysis.

Instance	*n*	*m*	*C*	*C/m*	Ω	Multicommodity Flow				Instance	*n*	*m*	*C*	*C/m*	Ω	Multicommodity Flow			
						Z	*F*(s)	Gap (%)	Runtime (s)							Z	*F*(s)	Gap (%)	Runtime (s)
data1	5	8	4	0.5	1	5	11	0	0.15	data24	10	10	6	0.6	3	16	27	0	102.79
data2	5	8	4	0.5	2	7	14	0	0.27	data25	10	10	7	0.7	1	9	20	0	1.05
data3	5	8	4	0.5	3	11	21	0	0.5	data26	10	10	7	0.7	2	11	24	0	20.56
data4	5	8	6	0.75	1	6	18	0	0.05	data27	10	10	7	0.7	3	15	33	0	113.62
data5	5	8	6	0.75	2	7	22	0	0.13	data28	10	15	6	0.4	1	13	16	0	35.73
data6	5	8	6	0.75	3	10	23	0	0.58	data29	10	15	6	0.4	2	23	25	0	4841.57
data7	8	10	4	0.4	1	8	12	0	1.22	data30	10	15	6	0.4	3	27	43	0	4698.88
data8	8	10	4	0.4	2	12	18	0	5.8	data31	10	15	9	0.6	1	13	26	0	7.21
data9	8	10	4	0.4	3	15	26	0	7.74	data32	10	15	9	0.6	2	22	33	9.09	14,400
data10	8	10	6	0.6	1	9	12	0	0.37	data33	10	15	9	0.6	3	29	50	16	14,400
data11	8	10	6	0.6	2	11	15	0	9.31	data34	10	15	12	0.8	1	16	28	0	21
data12	8	10	6	0.6	3	13	26	0	13.77	data35	10	15	12	0.8	2	19	43	0	3421.53
data13	8	10	8	0.8	1	9	22	0	0.25	data36	10	15	12	0.8	3	25	47	6.38	14,400
data14	8	10	8	0.8	2	11	17	0	1.72	data37	12	20	8	0.4	1	19	20	0	7056.48
data15	8	10	8	0.8	3	14	38	0	4.13	data38	12	20	8	0.4	2	22	27	3.7	14,400
data16	10	10	4	0.4	1	8	10	0	0.89	data39	12	20	8	0.4	3	48	72	52.78	14,400
data17	10	10	4	0.4	2	14	19	0	21.89	data40	12	20	12	0.6	1	21	34	0	522.96
data18	10	10	4	0.4	3	16	26	0	95.92	data41	12	20	12	0.6	2	31	56	26.79	14,400
data19	10	10	5	0.5	1	9	12	0	1.91	data42	12	20	12	0.6	3	55	76	48.68	14,400
data20	10	10	5	0.5	2	11	17	0	23.39	data43	12	20	16	0.8	1	20	44	0	66.89
data21	10	10	5	0.5	3	15	20	0	19.74	data44	12	20	16	0.8	2	28	50	14	14,400
data22	10	10	6	0.6	1	11	19	0	11.54	data45	12	20	16	0.8	3	43	74	32.43	14,400
data23	10	10	6	0.6	2	15	19	0	30.29										

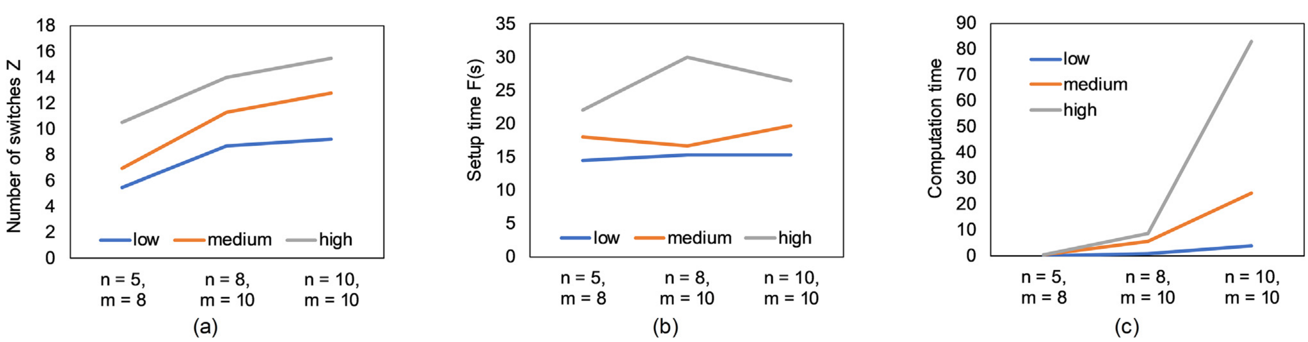

Fig. 2. The results of sensitivity analysis with three levels of density: (a) *Z*, (b) *F(s)*, (c) Running time.

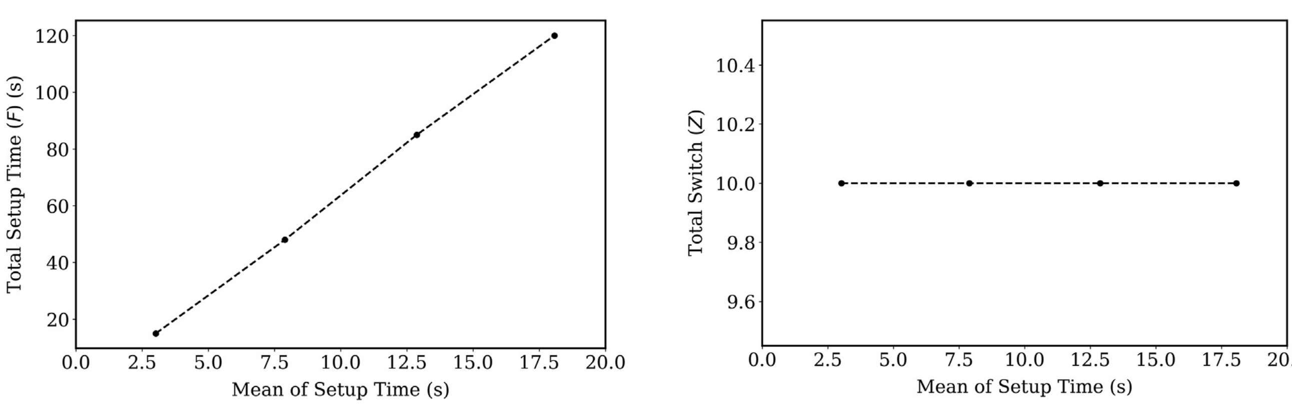

Fig. 3. Comparisons between different means of $s_{t,u}$ and its effect to the optimal *F* and *Z* of datA4.

commodity flow formulation is proven to be more effective and efficient than the five-index formulation. Our numerical experiments show that this formulation is able to obtain the optimal solution for sequence-dependent SSP instances in a generally lower running time. Third, the uniform SSP and the sequence-dependent SSP may generate two substantially different optimal solutions. It has been shown that when the assumption of setup time uniformity is relaxed, the solution of uniform SSP model can result in the higher total setup time in the same dataset. Thus, although may result in a higher number of tool switches, it is observed that considering the sequence-dependent issue can be beneficial for the decision-maker for composing a better tool switching schedule.

Finally, the research in this paper can be extended in several directions. The first direction is to relax other assumptions that hold in the uniform and sequence-dependent SSP. For instance,

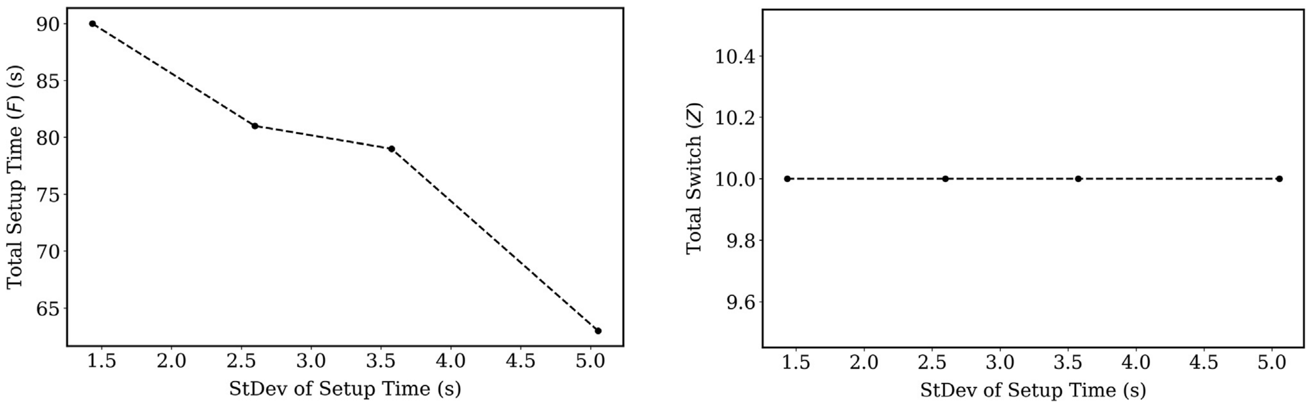

Fig. 4. Comparisons between different standard deviations of $s_{t,u}$ and its effect to the optimal F and Z of datA4.

the SPP assumes that the size of tools is uniform and can be loaded into any magazine slot, however, this assumption can be violated in the real-life situation. Another interesting direction is to design a heuristic solution for the sequence-dependent SSP. Our computational experiments have shown that the multicommodity flow formulation cannot solve instances with 15 jobs and 20 tools even after four hours running time of the commercial software GUROBI. A proposal on a decomposition method for the sequence-dependent SSP can be an exciting starting point since the SSP is a combination of two sub-problems. Then, regarding the previous point, future works should also focus on the development of exact approaches that can exploit the problem structure of sequence-dependent SSP.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

Abreu, L.R., Cunha, J.O., Prata, B.A., Framinan, J.M., 2020. A genetic algorithm for scheduling open shops with sequence-dependent setup times. Comput. Oper. Res. 113, 104793.
Adjiashvili, D., Bosio, S., Zemmer, K., 2015. Minimizing the number of switch instances on a flexible machine in polynomial time. Oper. Res. Lett. 43, 317–322.
Ahmadi, E., Goldengorin, B., Süer, G.A., Mosadegh, H., 2018. A hybrid method of 2-TSP and novel learning-based GA for job sequencing and tool switching problem. Appl. Soft. Comput. 65, 214–229.
Amaya, J.E., Cotta, C., Fernández, A.J., 2008. A memetic algorithm for the tool switching problem. Springer Verlag, pp. 190–202.
Amaya, J.E., Cotta, C., Fernández-Leiva, A.J., García-Sánchez, P., 2019. Deep memetic models for combinatorial optimization problems: Application to the tool switching problem. Memetic Comput. 12 (1), 3–22. https://doi.org/10.1007/s12293-019-00294-1.
Atta, S., Sinha Mahapatra, P.R., Mukhopadhyay, A., 2019. Solving tool indexing problem using harmony search algorithm with harmony refinement. Soft Comput. 23 (16), 7407–7423.
Baykasoğlu, A., Ozsoydan, F.B., 2017. Minimizing tool switching and indexing times with tool duplications in automatic machines. Int. J. Adv. Manuf. Technol. 89 (5–8), 1775–1789.
Bektur, G., Saraç, T., 2019. A mathematical model and heuristic algorithms for an unrelated parallel machine scheduling problem with sequence-dependent setup times, machine eligibility restrictions and a common server. Comput. Oper. Res. 103, 46–63.
Calmels, D., 2019. The job sequencing and tool switching problem: state-of-the-art literature review, classification, and trends. Int. J. Prod. Res. 57, 5005–5025.
Catanzaro, D., Gouveia, L., Labbé, M., 2015. Improved integer linear programming formulations for the job Sequencing and tool Switching Problem. Eur. J. Oper. Res. 244, 766–777.
Chaves, A.A., Lorena, L.A.N., Senne, E.L.F., Resende, M.G.C., 2016. Hybrid method with CS and BRKGA applied to the minimization of tool switches problem. Comput. Oper. Res. 67, 174–183.
Crama, Y., Oerlemans, A.G., Spieksma, F.C.R., 1994. Minimizing the number of tool switches on a flexible machine. Springer, Berlin, Heidelberg, pp. 165–195.
da Silva, T.T., Chaves, A.A., Yanasse, H.H., 2020. A new multicommodity flow model for the job sequencing and tool switching problem. Int. J. Prod. Res., 1–16
El-Tamimi, A.M., Abidi, M.H., Mian, S.H., Aalam, J., 2012. Analysis of performance measures of flexible manufacturing system. J. King Saud Univ. Eng. Sci. 24, 115–129.
Fernández, E., Laporte, G., Rodríguez-Pereira, J., 2019. Exact solution of several families of location-arc routing problems. Transp. Sci. 53, 1313–1333.
Furrer, M., Mütze, T., 2017. An algorithmic framework for tool switching problems with multiple objectives. Eur. J. Oper. Res. 259, 1003–1016.
Ghiani, G., Grieco, A., Guerriero, E., 2007. An exact solution to the TLP problem in an NC machine. Robot. Comput. Integr. Manuf. 23, 645–649.
Ghiani, G., Grieco, A., Guerriero, E., 2010. Solving the job sequencing and tool switching problem as a nonlinear least cost hamiltonian cycle problem. Networks 55, 379–385.
Kim, J.G., Song, S., Jeong, B., 2020. Minimising total tardiness for the identical parallel machine scheduling problem with splitting jobs and sequence-dependent setup times. Int. J. Prod. Res. 58 (6), 1628–1643.
Laporte, G., Salazar-González, J.J., Semet, F., 2004. Exact algorithms for the job sequencing and tool switching problem. IIE Trans. 36, 37–45.
Milind, M.A., 2018. Understanding of industry performance from the perspective of manufacturing strategies. J. King Saud Univ. Eng. Sci. 3, 206.
Mütze, T., 2014. Scheduling with few changes. Eur. J. Oper. Res. 236, 37–50.
Nesello, V., Subramanian, A., Battarra, M., Laporte, G., 2018. Exact solution of the single-machine scheduling problem with periodic maintenances and sequence-dependent setup times. Eur. J. Oper. Res. 266 (2), 498–507.
Paiva, G.S., Carvalho, M.A.M., 2017. Improved heuristic algorithms for the Job Sequencing and Tool Switching Problem. Comput. Oper. Res. 88, 208–219.
Pan, Q.K., Gao, L., Li, X.Y., Gao, K.Z., 2017. Effective metaheuristics for scheduling a hybrid flowshop with sequence-dependent setup times. Appl. Math. Comput. 303, 89–112.
Schwerdfeger, S., Boysen, N., 2017. Order picking along a crane-supplied pick face: The SKU switching problem. Eur. J. Oper. Res. 260, 534–545.
Shao, Z., Pi, D., Shao, W., 2018. A novel discrete water wave optimization algorithm for blocking flow-shop scheduling problem with sequence-dependent setup times. Swarm Evol. Comput. 40, 53–75.
Shen, L., Dauzère-Pérès, S., Neufeld, J.S., 2018. Solving the flexible job shop scheduling problem with sequence-dependent setup times. Eur. J. Oper. Res. 265 (2), 503–516.
Tang, C.S., Denardo, E.V., 1988. Models arising from a flexible manufacturing machine, part I: minimization of the number of tool switches. Oper. Res. 36, 767–777.