Novel robotic job-shop scheduling models with deadlock and robot movement considerations

Abstract: The robotic job-shop scheduling problem (RJSP) has become increasingly important due to the wide application of robots for material delivery in modern logistics and supply chain systems. With the common assumptions of negligible material transportation procedures and infinite machine buffers, the traditional job-shop scheduling problem (JSP) models can lead to system failures due to the potential deadlock for a robot-driven production line. In this study, we propose two novel robotic job-shop scheduling models with deadlock and robot movement considerations (RJSPDT). The proposed novel models simultaneously consider the scheduling of job operations and the movement of the robot, with the objective of minimizing makespan. In order to avoid deadlock, the machine blocking strategy is applied and a set of tight deadlock-avoidance constraints is proposed. Two modelling approaches are applied: the traditional position-based approach and the novel network-based approach which is inspired by aviation scheduling studies. Through numerical examples, it is illustrated that our proposed models can completely avoid system conflicts by considering deadlock and robot movement. Besides, through computational experiments, the network-based RJSPDT shows higher solution efficiency (e.g., reducing the computational time by 96%) owing to the smaller model size than the position-based RJSPDT. Moreover, we explore the impacts of job settings (e.g., number of jobs, number of operations in a job) and job entrance strategies (i.e., fixed entrance and flexible entrance) on model performances. Results show that the number of jobs imposes greater impacts than the number of operations in a job, while the fixed entrance strategy can reduce the average computational time by 60% with little impact on the makespan.

Keywords*:* Robot-driven production lines; Job-shop scheduling; Deadlock; Machine blocking; Robot movement; Mixed integer linear programming.

1. Introduction

Through the coordination of diverse resources such as materials, machines, and labor, scheduling is crucial for the enhancement of operations efficiency as well as the reduction of overall operations costs in logistics and supply chain systems (Choi et al., 2019; Ma et al., 2019; Sun et al., 2020a). Due to the nature of NP-hard, the scheduling problem is well-known to be one of the most difficult optimization problems, while tremendous research efforts have been devoted to improving the related decision making during the past decade (Choi et al., 2018; Sun et al., 2018; Xi et al., 2020; Zhou et al., 2020).

The workshop scheduling problem is generally divided into two major research streams: The flow-shop scheduling problem (FSP) and the job-shop scheduling problem (JSP) (Özgüven et al., 2012; Mascis and Pacciarelli, 2002; Vital-Soto et al., 2020). The FSP usually deals with the production scheduling problem for homogeneous products with the same production flow on a series of machines, with the aim of enhancing production capacity. However, nowadays, manufacturers are challenged by dynamic market demands as well as the increasingly complicated production processes. The traditional FSP thus fails to characterize the scheduling problems in the modern production systems. On the other hand, the JSP is formulated to schedule a set of jobs that involve various operations which are required to be handled on specific machines in a particular sequence, with the objective of minimizing makespan or maximizing throughput (Liu et al., 2018). Therefore, the JSP has become increasingly popular in logistics and supply chain systems due to the high operations flexibility.

Over the past decade, Industry 4.0 has greatly advanced business intelligence which is facilitated by the wide application of automated techniques in various industries like automotive, electronics, chemistry, and food (Khan et al., 2019a, 2019b; Ren et al., 2020; Wang, et al., 2020b). Among these disruptive technologies, mobile robots are commonly used for a variety of activities like material delivery in automated production lines (named as *robotic cells*) (Yan et al., 2018). Robots are advantageous over human beings in doing simple and repetitive jobs, with higher accuracy, stability, and flexibility, which enables manufacturers to adjust production capacity dynamically according to the fluctuating market demands. Besides, the utilization of robots in material transportation can achieve a remarkable cost saving as the traditional labor-intensive material handling system constitutes 30%-70% of the total production cost (Rahman and Nielsen, 2019). Therefore, the manufacturing industry has witnessed a rapid growth in the installation of robots. For example, Alibaba, the largest Chinese online retailer, has established an automated warehouse, where 700 robots are employed to replace humans in production delivery on the Singles Day[1]. Its competitor, JD.com, launches the first 5G-powered smart logistics park, which also enables automated supply chain solutions[2]. It is reported that the global robot installation increased by 6% to 422,271 units in 2018, which is worth of 16.6 billion US dollars

[1] Detailed information is available at: https://www.cnbc.com/2018/10/30/alibaba-cainiao-chinas-biggest-robot-warehouse-for-singles-day.html.

[2] Detailed information is available at: https://www.supplychaindigital.com/logistics/jd-logistics-enabling-smart-supply-chain-solutions.

(International Federation of Robotic, 2019). Accordingly, with the wide application of robots in modern logistics and supply chain systems, the traditional JSP is extended to the robotic job-shop scheduling problem (RJSP) with the robot functions of material delivery. The advancements in RJSP studies are of great importance for the efficiency and productivity enhancement for modern logistics and supply chain systems, which further imposes great impacts on the economic development of the society (Gharehgozli and Zaerpour, 2020; Roy et al., 2019; Simoni et al., 2020).

However, the integration of the mobile robot into the manufacturing system remarkably increases the problem complexity and the computational difficulty due to the involvement of robotic transportation (Nouri et al., 2016). In fact, the proper coordination of the sequencing and time allocation of the required resources in the robotic job-shop scheduling problem is one of the most difficult operational problems (Caumond et al., 2009). The two sub-problems of the RJSP (the job scheduling and the robot routing planning) are traditionally treated independently and each component is NP-hard (Nouri et al., 2016). However, these two sub-problems are inherently interrelated, and the integration brings challenges in terms of modelling and algorithm complexity (Deroussi et al., 2008; Zheng et al., 2014). In traditional JSP studies, a typical assumption is that the material transportation time between two machines is negligible or could be incorporated into the processing time of an operation. Therefore, the material transportation process is not considered in the traditional JSP research (Soukhal and Martineau, 2005). Accordingly, the capacity and availability of the material transportation tool are ignored. However, with the application of the mobile robot, the material transportation process becomes a crucial consideration in the job-shop scheduling problem. It is thus essential to consider the movement procedures and the robot capacity in a robot-driven production line. The *robot capacity* represents the number of products that a robot can carry, which depends on the configuration of the robot. In our problem setting, we adopt a single mobile robot with the capacity of 1 (the robot can hold one product each time) because such robot is common and cost-saving.

Another commonly seen assumption in the traditional JSP studies is that machines have infinite buffers (Soukhal and Martineau, 2005). According to the JSP literature, *buffers* are storage spaces in machines where items stay and wait for delivery after the processing of operations (Mascis and Pacciarelli, 2002). Under the infinite buffer assumption, the traditional JSP studies allow machines to continue the next operation (i.e., other jobs can be transported to the machine) as long as they become idle without checking the availability of machine buffers. Although this assumption is common in the traditional JSP studies, it makes the RJSP solutions impractical as most equipment in real operations has no buffer or a limited buffer due to technique requirements and machine capabilities (Liu et al., 2018). In this paper, we investigate the machines with no-buffer constraints (i.e. no intermediate buffers between machines) or blocking restrictions. The *blocking* implies a period that a product remains on the processing machine between the time point of the completion of the operation and the time point to be removed. During such periods, the machine cannot perform other jobs due to the blocking of the current job. In fact, such a situation is common and practical in real-world productions. For example,

in the chemical industry, a product may need blocking on the processing machine to keep a certain temperature before the next movement (Liu et al., 2018).

With the machine no-buffer constraint and the robot capacity restriction, the robot is not allowed to deliver a job to its destination machine when that machine is occupied by another job. Otherwise, a deadlock situation happens, which is the conflicted and unsolvable situation that leads to the robot movement failure. In such circumstances, the system suffers from an emergent suspension and may need assistance of manual operators, which highly impairs the automation level of robotic manufacturing (Caumond et al., 2009). Figure 1 uses an example to demonstrate a deadlock situation. It can be seen that when job 1 has been processed by Machine 1, the robot attempts to deliver job 1 to the next processing machine (i.e., Machine 2). However, Machine 2 is occupied by job 2, which needs to be transported to Machine 3 after the execution. In this circumstance, the system suffers from a deadlock situation since the robot cannot move successfully (job 1 cannot be loaded to Machine 2 and job 2 cannot be unloaded from the same machine). On the other hand, Figure 2 presents a deadlock-free movement, where job 2 has been removed from Machine 2 and been delivered to Machine 3 before the delivery of job 1 to Machine 2.

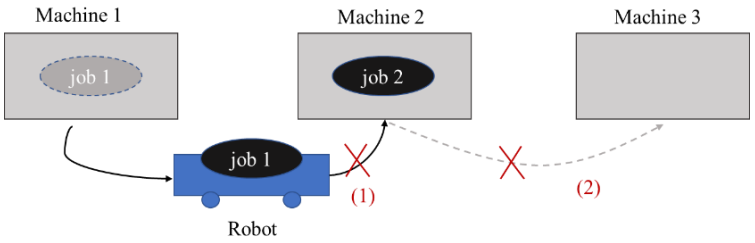

Figure 1. An example of a deadlock situation.

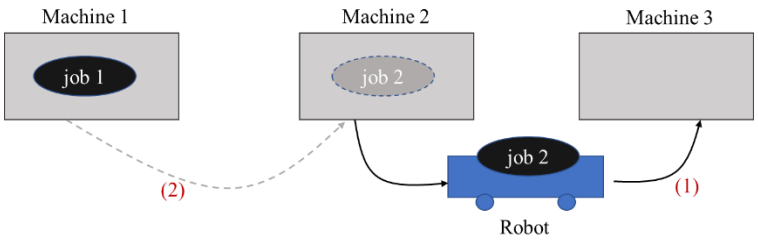

Figure 2. An example of a deadlock-free movement.

Therefore, the machine buffer and robot capacity restrictions are pivotal for the scheduling problem of a robot-driven production line (Brucker, et al., 2012b). However, how to characterize the features of deadlocks in the modelling process and describe the deadlock-avoidance strategy with mathematical formulations are very challenging. To be specific, diverse scenarios in the transportation process may lead to a deadlock situation. For example, the most straightforward deadlock happens when the robot attempts to place an item on an occupied machine. Besides, potential conflicts might occur even if the previous operations are normal and smooth, which makes the deadlock unforeseeable.

Moreover, the different features of the first operation and the last stock operation from other intermediate operations require different modelling methods for avoiding deadlock for these two operation types, which further increases the difficulty of deadlock modelling. Therefore, deadlock avoidance strategies are seldom modelled as constraints mathematically in the existing literature.

Motivated by the significance towards modern automated logistics and supply chain systems and real challenges revealed from the robot-driven production lines, in this work, we study a robotic job-shop scheduling problem with deadlock and robot movement considerations (RJSPDT). To be specific, we consider the job-shop scheduling in a robotic cell, where a robot with the capacity of 1 is responsible for the delivery of jobs among a series of machines with no-buffer constraints. Our problem setting is based on the following considerations: (i) such a robotic cell is common in real productions since this configuration could simplify the systematic control or perform as a component for more complex systems (Caumond et al., 2009); and (ii) a large body of related studies solve similar problems with heuristic approaches (e.g. Liu and Kozan, 2017), but rare attention has been paid to the mathematical modelling of the problems due to the high complexity (Hurink and Knust, 2001).

We propose two novel models simultaneously consider the schedules of operations in jobs, as well as the movement procedures of the robot. Our objective is to minimize the *makespan* (i.e., the completion time of the last operation for the system) with the considerations of machine blocking and deadlock. Our models face with the following two major challenges. First of all, the robot movement process should be determined with the job-shop scheduling at the same time in order to achieve the optimal coordination of jobs, machines, and the robot. Second, the deadlock dilemma which is caused by the no-buffer restriction, as well as the machine blocking strategy should be considered in the decision framework.

In terms of the modelling approach, we explore two distinct ideas for the novel RJSPDT, namely the position-based approach and the network-based approach. The two proposed models are thus named as the Position-RJSPDT and the Network-RJSPDT, respectively. In the following sections, we first develop the Position-RJSPDT with the typical position-based modelling approach, which has been widely applied in the JSP literature (Roshanaei et al., 2013; Demir et al. 2013; Meng et al. 2020). Then, we propose the model Network-RJSPDT based on the novel network modelling idea. The adoption of the network modelling idea is mainly based on the following two reasons. On one hand, the network modelling idea has a number of advantages: (i) Wang et al. (2020a) show that the network flow based models are superior than the traditional mixed integer programming modelling approaches in terms of computational time and optimality; (ii) Liang et al. (2011) suggest that models based on the network modelling idea are scalable and have a better nature of extension, which implies that the number of constraints increases much slightly along with the growth in problem sizes (e.g. the number of jobs and the number of operations in a job) compared with other modelling ideas. On the other hand, the network modelling idea has been widely adopted in aviation-related optimization problems (e.g., the aircraft routing problem and the crew scheduling problem) since the various components in the aviation system

(like aircraft, crew members) are inter-connected and affect each other, which can be represented by a network (Chung et al., 2020; Khan et al., 2019c; Qin et al., 2018, 2020a, 2020b). Similar to the aviation optimization problems, the connection and interaction of different resources in the RJSP (e.g., job operations are completed in a sequenced manner (i.e., nodes), and the robot moves from one machine to another to deliver job operations (i.e., paths)) provide a great opportunity for the application of the network modelling idea in the RJSP (Özgüven et al., 2012). But differently, as operations and machines are coordinated by only one robot, a return arc is needed for picking up the job before the next delivery, which brings modelling challenges for this problem.

To the best of our knowledge, our proposed Network-RJSPDT is the first mathematical model applying the network-based approach to formulate the RJSP with the considerations of deadlock and robot movement in the literature. The logic behind the network-based approach is to transform the production line which is featured by a set of machines, the input depot, and the output depot into a network flow model, while the operations of jobs act as the nodes in the network and the routes of robots are formulated as paths in the network. Furthermore, a set of deadlock-avoidance strategies (i.e., for machine blocking) is developed to deal with the deadlock dilemma by considering all the four scenarios with possible occurrence of deadlock. Therefore, a set of novel tight deadlock-avoidance constraints is proposed, which is shown to efficiently avoid system conflicts in the scheduling process.

Through experiments, several important insights are generated as follows. First of all, through the comparison with a benchmark case where the deadlock dilemma is not considered, it is illustrated that our proposed models can completely avoid the conflicts among machines, robots, and materials. The significance of the proposed tight deadlock-avoidance constraints is thus verified. Second, comparing the Network-RJSPDT and the Position-RJSPDT, it is found that the Network-RJSPDT can achieve higher solution efficiency owing to the smaller model size (i.e., fewer constraints and decision variables) than the Position-RJSPDT. Accordingly, for small-scale problems, the Network-RJSPDT spends 96% less computational time than the Position-RJSPDT averagely, while for most of the instances with large scales, the Network-RJSPDT is able to identify solutions within reasonable computational time while the Position-RJSPDT fails. Third, through sensitivity analysis, it is demonstrated that the number of jobs imposes greater impact on the performances of the two models than the number of operations in a job. Last, the exploration on different job entrance strategies (i.e., fixed entrance and flexible entrance) shows that the adoption of the fixed entrance strategy can reduce the average computational time by 60% with little impact on the makespan.

Contributions & Paper Structure

By integrating the deadlock and robot movement considerations into the decision framework, this study contributes to the JSP literature by proposing two novel and practical mathematical models which can better capture the distinctive characteristics of a robot-driven production line. Besides, to the best of our knowledge, this research is the first study applying the network-based modelling approach which is widely applied in aviation scheduling problems with the considerations of deadlock and robot

movement in the JSP literature, thus providing a new modelling idea for the domain. Moreover, a set of novel deadlock-avoidance constraints is developed which is shown to be capable to greatly enhance the system efficiency.

The rest of this paper is organized as follows. Section 2 reviews the related literature. Section 3 gives problem description. Then, the Position-RJSPDT and the Network-RJSPDT are established in Section 4 and Section 5, respectively. Next, Section 6 demonstrates the significance of the proposed tight deadlock-avoidance constraints through a numerical example. In Section 7, computational experiments are carried out to evaluate the performances of the two proposed models. Last, Section 8 concludes for this work.

2. Literature Review

In this section, we review the existing literature from three perspectives: (i) JSP, (ii) RJSP, and (iii) the existing modelling approaches for scheduling problems (e.g., JSP and aviation-related problems).

2.1 Job-shop Scheduling Problem (JSP)

In recent years, the studies in the FSP and the JSP have proposed diverse variants for the ever-changing manufacturing industry, in order to meet the practical requirements arising from the real production environment. For example, the FSP in a cyclic manner is proposed to minimize the cycle time for repetitive works in Crama et al. (2000). Besides, many studies extend the JSP to flexible JSP (FJSP) with the aim of maximizing the flexibility of the manufacturing system. Based on the FJSP, the dynamic FJSP is further developed to consider the impact of the arrival of new jobs on the existing system (Yan et al., 2018; Shahgholi et al., 2019). Moreover, some research even explores the flexible shop scheduling and super shop scheduling problems (Abreu et al., 2020; Koulamas and Panwalkar, 2019).

With the rapid evolution of JSP variants, an increasing number of new constraints are proposed to better capture the characteristics of the practical manufacturing system, such as the buffer limitations, material pickup operations, setup times, and machine availabilities. Buffer limitation is a crucial consideration which depends on machine capacity. Liu et al. (2018) study a generalized JSP problem with a combination of four buffering constraints (i.e., no-wait, no-buffer, limited-buffer and infinite-buffer constraints). Buffer management strategy is highly related to material pickup operations (i.e., the free-pickup strategy and the no-wait strategy). The free-pickup strategy allows blocking in machines. For instance, Mati et al. (2011) model an FJSP that takes blocking into account and solve the problem with Genetic Algorithm (GA). On the other hand, the no-wait strategy forbids delays between two successive operations in a job. For instance, both wait and no-wait strategies are considered in a mixed flow-shop problem proposed by Wang et al. (2017) which is solved by a heuristic greedy algorithm. In terms of the setup times, Bektur and Saraç (2019) model it as a crucial restriction for an unrelated parallel machine scheduling problem which is solved by a tabu-search and simulated annealing

algorithm. Moreover, the machine availability constraint is formulated in Hasan et al. (2011), in which disruptions like machine breakdown are considered to enhance the system robustness.

Various objectives are considered in the existing JSP studies. For example, Arroyo and Armentano (2005) pursue makespan minimization and tardiness minimization in a flow-shop scheduling problem which is solved by a genetic local search algorithm, while Zhang and Chiong (2016) study a bi-objective energy-efficient JSP which minimizes the weighted tardiness and energy consumption.

2.2 Robotic Job-shop Scheduling Problem (RJSP)

The introduction of robots for material transportation further complicates the job-shop scheduling problem as the transportation process should be incorporated into the decision framework (Deroussi et al., 2008; Zheng et al., 2014). In fact, a two-machine flowshop scheduling problem with transportation considerations and a single robot is NP-hard (Hurink and Knust, 2001). In the literature, traditional studies divide the job-shop scheduling problem with transportation considerations into two stages in which the job scheduling and the transportation routing are determined sequentially (Egbelu and Tanchoco, 1984; Hurink and Knust, 2005; Nouri et al., 2016). Few scholars treat the two sub-problems as an integrated system, where the job scheduling and the robot routing are considered simultaneously. Besides, most previous research attentions are paid to the flowshop robotic scheduling problem due to the natural relationship between automation and mass production (Crama et al., 2000; Soukhal and Martineau, 2005; Zhou et al., 2012). Dawande et al. (2005) carry out a survey on the research development of robotic flowshop cells consisting of one or more robots and develop a comprehensive classification scheme from three characters, namely machine environments, processing restrictions, and objective functions. However, few research efforts have dealt with the robotic job-shop scheduling problems.

In addition, meta-heuristic algorithms are developed to solve the related robotic scheduling problems. For example, Liu and Kozan (2017) define a blocking job shop scheduling problem with robotic transportation (BJSSRT) and solve it with a meta-heuristic local search algorithm. Besides, Li et al. (2019) investigate a JSP in a robotic cell with time window constraints which aims to minimize the total earliness and tardiness. A metaheuristic approach that combines a memetic algorithm and local search is proposed to solve the problem (Li et al., 2019). However, although such intelligent algorithms can solve large instances within reasonable time limits, they fail to guarantee the solution optimality (Nouri et al., 2016). Only a few attentions have been paid to the optimization theories of the RJSP. Ham (2020) solves a robotic JSP by a constraint programming methodology, while Dai et al. (2019) present an optimization model for an FJSP with transportation and energy saving considerations. Quinton et al. (2020) apply the Benders decomposition technique together with a heuristic algorithm to solve a sequence-based cyclic FJSP model with the application of robotic technologies.

Furthermore, the research of machine buffers (no-buffer or limited buffer) has received very limited attention. Liu et al. (2018) study the different buffer management strategies. Drobouchevitch et

al. (2010) investigate a free-pickup FSP in a robotic cell where one transporter (i.e., robot) is involved, and reveal that the introduction of machine buffers for a single robot problem is not helpful for improving the system throughput, which helps justify the significance of our problem setting. However, failing to consider machine buffers and robot capacity might lead to deadlock. As pointed out by Soukhal and Martineau (2005), the RJSP studies without deadlock constraints are impractical and lack of real-world implementation values. General methodologies to deal with deadlock situations include deadlock detection & recovery and deadlock avoidance (Caumond et al. 2009). In this paper, we adopt the deadlock avoidance strategy. The research related to deadlock-avoidance strategies are briefed in the following. Caumond et al. (2009) regulate the maximum number of jobs that can be handled at the same time in a flexible manufacturing system (FMS). In a similar manner, Yan et al. (2018) divide the whole planning horizon into a set of time intervals and assign each transportation activity into those time intervals to avoid deadlock. Besides, Ham (2020) assigns machines with sufficient input/output buffer space to alleviate system conflicts. Different from these deadlock avoidance methods, we propose a set of deadlock-avoidance constraints mathematically, which avoids all possible systematic deadlocks from the source.

In terms of the robot routing sub-problem, a related optimization research area is the vehicle routing problem (VRP), which is important to logistics and supply chain operations. The VRP investigates the efficient supply for distributed customers with diverse delivery requirements (Baker and Ayechew, 2003; Pillac et al., 2013; Pisinger and Ropke, 2007). Basically, a depot is established for each vehicle to pick up goods and the vehicle needs to return to the depot once the delivery is completed (Baker and Ayechew, 2003). Ng et al. (2017) investigate an extension of VRP and propose an Online Vehicle Routing Problem (OVRP) model, which integrates the real time traffic density estimation into the dynamic decision framework to alleviate the possible delivery delay. Similar to the VRP, the RJSP studied in this paper also needs to plan the route for the transporter (i.e., the robot). For the VRP, a transportation task finishes when the vehicle delivers goods to customers. However, differently, the problem studied in this paper has two distinctive components that need careful coordination: the scheduling of jobs (operations), and the scheduling and routing of the moving robot. Besides, a return action is required for the robot before the next movement. Therefore, the modelling process of the RJSP is different from the vehicle routing problem.

2.3 Modelling Approaches for Scheduling Problems

In the JSP literature, many classical modelling approaches are applied, among which the position-based idea proposed by Wagner (1959) is a typical and widely adopted modeling method (Roshanaei et al., 2013; Demir et al. 2013; Meng et al. 2020). For example, Hsu and Yang (2016) propose a position-based mixed integer linear programming optimization model to schedule the production of an automatic manufacturing factory. Özgüven et al. (2010) and Naderi and Azab (2014) address extended job-shop scheduling problems mainly based on the position-based modelling approach. Besides, Karimi et al.

(2017) characterize the movement of robots by combining the position- and sequence- based modelling approaches. Yan et al. (2018) apply the time interval modelling idea to solve a dynamic FJSP with a robot. Some scholars consider the comparisons among different modelling ideas. For instance, Demir et al. (2013) construct five FJSPs and Meng et al. (2020) deal with the FSP by eight models involving various modelling ideas. Different models are characterized by various decision variables which capture the relationships among operations and machines from different perspectives. Furthermore, different modelling approaches lead to the discrepancy in model sizes (e.g., the numbers of constraints and decision variables), thus affecting the computational times. Although plenty of attention has been paid to the modelling of JSP, to the best of our knowledge, the research on mathematical formulations of RJSP is very limited. The related literature is discussed in the following. Brucker et al. (2012a) formulate a cyclic JSP with robot movement and blocking considerations with a sequence-based modelling approach. Besides, Zhou et al. (2012) use a similar modelling idea to address a cyclic RFSP considering processing time windows, robot capacity, and machine availability.

The network-based modelling approach has achieved wide application, especially in the domain of aviation-related scheduling problems, in which the decision variables can be related to paths, nodes, or arcs in the generated network. For example, Wen et al. (2020) state that a network derived from a flight schedule is able to characterize the specific features of the scheduling problem in the domain of airline crew planning. Similarly, Sun et al. (2020b) apply a duty-based flight network to capture the operational risks caused by flight delays for a robust air crew scheduling problem. Focusing on the aircraft routing problem, Basdere and Bilge (2014) employ a modified connection network representation to provide maintenance-feasible aircraft routes with the objective of fleet utilization maximization in a quick manner. In a similar study, Liang et al. (2011) develop a modified time-space network presentation for the aircraft maintenance routing problem and suggest that the size of their developed network can remain small even for realistic problems.

Research gaps

From the above discussions, several important research gaps can be revealed. First of all, despite the abundant studies on the JSP, the research on the RJSP with robot movement considerations is very limited. However, with the fast development of the robotic technology, the robot-driven logistics and supply chain systems become increasingly popular, which deserves more research attention. Second, the deadlock problem can greatly impair the overall production efficiency of the system and thus should be avoided. Although different deadlock solutions have been proposed in the literature, none of these approaches mathematically develops efficient and tight deadlock-avoidance constraints and integrate them into the modelling process, which can help prevent the possible deadlock from the source. Third, even though the network-based modelling approach has great potential to be applied for the robot-driven job-shop scheduling problems, its application in the related area has not been fully explored. In conclusion, the JSP literature will be greatly benefited by bridging these crucial research gaps.

3. Problem Description

The robotic job-shop scheduling problem with deadlock and robot movement considerations (RJSPDT) studied in this paper is described as follows. A robotic cell is a unit (e.g., a job shop) that consists of $|M|$ linearly placed stationary machines (the set of machines is denoted by M) and a robot for delivering materials among machines. For a job assignment consisting of a set of jobs ($i \in I$), each job i involves a set of operations $J_i = \{O_{i1}, O_{i2}, \ldots, O_{i|Ji|}\}$ which should be completed in a specific sequence (known as *operation sequence*). Each machine in the robotic cell could execute a particular type of operation. Besides, each operation is a nonstop process that will be processed by a designated machine M_{ij} with a certain processing time denoted by PT_{ij}. One machine can only execute one operation at any time and pre-emption is not allowed. Each job will enter the production process through an input depot D. Then, jobs (materials) will be processed on and transported among machines for the execution of operations. Finally, a finished job will be delivered to an output depot S for stocking. The RJSPDT aims to finish all the operations involved in this assignment by using the minimum makespan.

Although operations within a job should be performed in a certain order, there is no sequence restriction for operations from different jobs, which means that the robot can move to an operation from any job as long as each operation sequence is not violated. Once the route of the robot is determined, the sequence of the operations assigned to each machine can be decided (known as *machine sequence*). The robotic cell specified in this context is depicted in Figure 3.

The machine buffer and robot movement considerations in the robotic cell context are highlighted as follows. The machines have no buffer and the single mobile robot has a capacity to hold one product at each time (i.e., the robot also has no buffer), and there is no other space to stock intermediates within the robotic cell. Consequently, the robot or each machine can only be occupied by one job at any time. Since machines are linearly placed, the travelling times between any pair of machines are symmetric and determined by the absolute distance between their locations. Moreover, the robotic cell obeys the free-pick up criteria, which means that a job can be blocked on a machine until the robot comes for unloading (i.e., the so-called machine blocking). Thus, there are two types of movements in the process of transportation, namely the loaded movements and empty movements. A *loaded movement* occurs when the robot moves with carrying a product, while an *empty movement* refers to the case when the robot moves without carrying any product. In addition, in the RJSPDT, the predecessors or successors for an operation can be classified into three categories, namely the *same-job* (the predecessors or successors are the operations involved in the same job), the *same-machine* (the predecessors or successors are the operations assigned to the same machine), and the *robot-routing* (the predecessors or successors are the operations assigned in the robot route) ones.

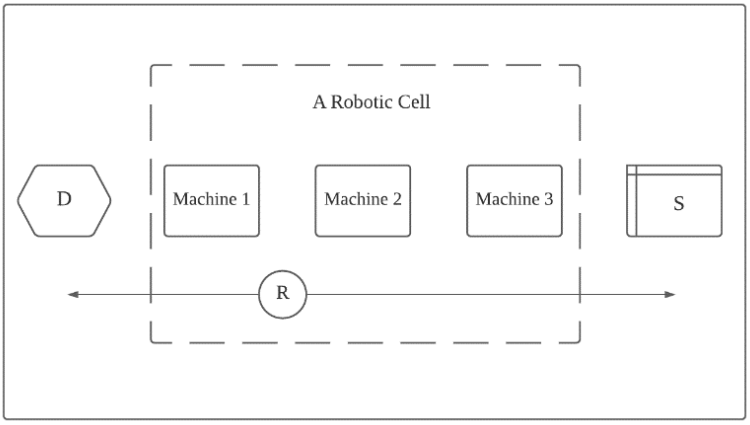

Figure 3. A typical robotic cell.

The assumptions of the studied RJSPDT are summarized as follows.

1) All jobs and machines are available at time zero.
2) The first job has been initialised at time zero.
3) The processing time of each operation is fixed and known in advance.
4) The processing time of initialization and stock activities is zero.
5) The activities of loading or unloading spend no time.
6) The operation immediately starts after the job is loaded onto the machine.
7) Each stationary machine can only process one operation at any given time.
8) Pre-emption is not allowed.
9) Both the robot and machines can handle one job at each time.
10) There is no other space for stocking within the robotic cell.

For ease of presentation, the notations used in the following sections are listed in Table 1. The execution process of any O_{ij} ($j \neq 1$) is introduced as follows. When the robot determines that the next step is to execute O_{ij}, two possible scenarios can be derived. First, if the robot stays at M_{ij-1} (the machine that performs O_{ij}'s same-job predecessor O_{ij-1}), then the robot can conduct a direct loaded movement that delivers the part from M_{ij-1} to M_{ij}. Second, if the robot is not at M_{ij-1}, then, an empty movement back to M_{ij-1} is necessary to pick up the part before transporting it to M_{ij}. Furthermore, in the second scenario, two sub-scenarios are needed to be considered: (i) O_{ij-1} has not been finished yet when the robot arrives. In this situation, the robot waits for the completion of O_{ij-1} and then pick up the job from the machine. (ii) O_{ij-1} has been finished in advance, so that the finished part will be blocked on that machine until the robot comes. The above described execution processes apply to the operations which are not the first operation in the corresponding job. Note that if O_{ij} is the first operation of job i (other than job 1), then an empty movement to the input depot should be conducted to obtain the product before transporting it to M_{i1}.

Table 1. Notations.

I	The set for jobs (i.e. job assignment).
$\|I\|$	The total number of jobs.
i, m, h	The indexes for jobs, where $i, m, h \in I = \{1,2, \dots, \|I\|\}$.
J_i	The set for the operations involved in job *i*.
$\|J_i\|$	The number of operations in job *i*.
$\|J_i\| + 1$	The dummy stock operation at the output depot for job *i*.
j, n, g	The index for operations in a job, where $j \in (1, \|J_i\| + 1), n \in (1, \|J_m\| + 1), g \in (1, \|J_h\| + 1)$.
O_{ij}, O_{mn}, O_{hg}	The index for the *j-th*, *n-th*, and *g-th* operation of job *i*, job *m*, and job *h*; $O_{ij} \in J_i$, $O_{mn} \in J_m, O_{hg} \in J_h$.
N_o	The set for all operations in the assignment, including the dummy stock operations.
$\|N_o\|$	The number of total operations in the assignment, including the dummy stock operations.
k, p	The *k-th* and *p-th* operations (the operations with priorities *k*, *p*) executed by the robot.
PT_{ij}	The processing time of O_{ij} on the designated stationary machine ($PT_{i,\|J_i\|+1} = 0$ as the processing time of the dummy stock operation is zero).
M	The set of machines.
$\|M\|$	The number of machines.
D	The input depot.
S	The output depot.
M_{ij}	The index for the stationary machine assigned to execute O_{ij}.
PM_{ij}	The position of the machine assigned to O_{ij} ($PM_{i\|J_i\|+1} = \|M\| + 1$).
PM_D	The position of the input depot, $PM_D = 0$.
tu_{ijmn}	The travelling time for an empty movement of the robot from M_{ij} to M_{mn}.
tl_{ij}	The travelling time for a loaded movement of the robot from M_{ij-1} to M_{ij}.
β	A large positive number.
F	A dummy sink node for the Network-RJSPDT.

It is noticeable that machine blocking is unavoidable in case of a deadlock situation. A deadlock appears if the single robot delivers a product *A* to a destination machine while that machine is occupied by another product *B*. Since only one robot is available, the robot needs to unload product *B* from the machine before loading *A* to that machine. However, since product *A* is occupying the only space of the robot and no other transporter is available, system conflicts are thus inevitable. Given that the occurrence of deadlock would disturb the normal operations of the automated system, a set of deadlock-avoidance constraints is demanded to avoid such a dilemma. Figure 4 depicts a feasible deadlock-free robot route with a simple example of two jobs (each with two operations).

Next, we will first present the proposed Position-RJSPDT in Section 4, which is followed by the Network-RJSPDT as formulated in Section 5.

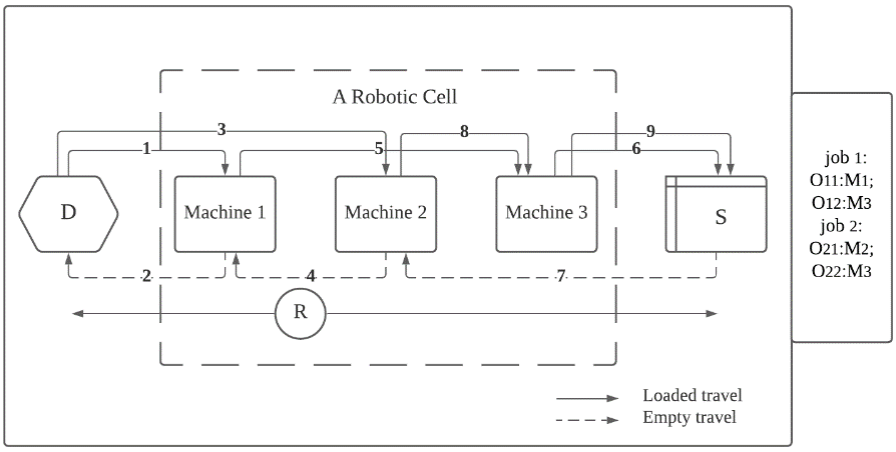

Figure 4. A deadlock-free robot route for two jobs and four operations.

4. Position-RJSPDT

The Position-RJSPDT is developed based on the position-based modelling idea widely used in the literature. This model is to establish the optimal execution sequence and the corresponding robot route with the objective of minimizing the makespan.

Table 2. Decision variables of the Position-RJSPDT.

Continuous Decision Variables	
C_{max}	Total time needed to finish all the operations and deliver all jobs to the output depot.
SM_{ij}	The start time of O_{ij} on the assigned machine.
Binary Decision Variables	
Z_{ijhg}	Binary decision variable. It equals 1 when both O_{ij} and O_{hg} are executed on the same machine, and O_{ij} precedes O_{hg} (not necessarily the immediate predecessor), and 0 otherwise.
X_{ijk}	Binary decision variable. It equals 1 when O_{ij} is scheduled as the *k-th* operation to be executed by the robot, and 0 otherwise.

The traditional position-based modelling approach aims to grant each operation an executive position on the designated machine. In the context of RJSPDT, even though the operations assigned to a machine are sequenced, the robot routing cannot be determined simply according to this sequence due to the equipment availability limitation. Actually, the undetermined sequences of operations on different machines complicate the movement process of the robot. Therefore, in our study, the Position-RJSPDT transforms the robot into a (dummy) machine. In this way, the problem is transformed to assign each operation with an execution priority on the robot. An operation is assigned with priority *k* means that this operation is the *k-th* operation being executed by the robot. Consequently, the output of the model should include the position of each operation, the start time of each operation, and the operation sequence for each machine. The activities related to initialization and stocking operations are regarded

as dummy operations with zero processing times. Based on the above descriptions, the (continuous & binary) decision variables are defined in Table 2, while the proposed Position-RJSPDT is presented in Table 3.

Table 3. The proposed Position-RJSPDT.

Obj. Min C_{max}		(1.0)
s.t.		
$C_{max} \geq SM_{ij}$,	$\forall i, \forall j \in (1, \lvert J_i \rvert + 1)$,	(1.1)
$SM_{11} = \lvert PM_{11} - PM_D \rvert \delta$, □		(1.2)
$X_{111} = 1$,		(1.3)
$SM_{ij} + PT_{ij} + tl_{i(j+1)} \leq SM_{i(j+1)}$,	$\forall i, \forall j \in (1, \lvert J_i \rvert)$,	(1.4)
$\sum_k X_{ijk} = 1$;	$\forall i, \forall j \in (1, \lvert J_i \rvert + 1), k \in (1, \lvert N_o \rvert)$,	(1.5)
$\sum_m \sum_p X_{imp} - (1 - X_{ijk}) * \beta \leq 0$;	$\forall i, \forall j \in (1, \lvert J_i \rvert + 1), \forall k \in (1, \lvert N_o \rvert), \forall m \in (\mathrm{j} + 1, \lvert J_i \rvert + 1), \forall \mathrm{p} \in (1, k)$,	(1.6)
$(SM_{ij} - SM_{mn}) * (k - p) + (2 - X_{ijk} - X_{mnp}) * \beta \geq 0$,	$\forall i, \forall m, i! = m, \forall j \in (1, \lvert J_i \rvert + 1), n \in (1, \lvert J_n \rvert + 1), \forall k, \mathrm{p} \in (1, \lvert N_o \rvert)$,	(1.7)
$SM_{ij} + (2 - X_{mn(k-1)} - X_{ijk}) * \beta \geq SM_{mn} + tu_{mni(j-1)} + tl_{ij}$,	$\forall i, \forall m, i! = m, \forall j \in (2, \lvert J_i \rvert + 1), \forall n \in (1, \lvert J_m \rvert + 1), \forall k \in (2, \lvert N_o \rvert)$,	(1.8)
$SM_{i1} + (2 - X_{i1k} - X_{mn(k-1)}) * \beta \geq SM_{mn} + tu_{mnD} + tl_{i1}$,	$\forall i, \forall m, i! = m, i! = 1, n \in (1, \lvert J_m \rvert + 1), \forall k \in (2, \lvert N_o \rvert)$,	(1.9)
$Z_{ijhg} + Z_{hgij} = 1$,	$\forall i, \forall h, i! = h, \forall j \in (1, \lvert J_i \rvert + 1), g \in (1, \lvert J_h \rvert + 1), M_{ij} = M_{hg}$,	(1.10)
$SM_{hg} \geq SM_{ij} + PT_{ij} - (1 - Z_{ijhg}) * \beta$,	$\forall i, \forall h, i! = h, \forall j \in (1, \lvert J_i \rvert + 1), g \in (1, \lvert J_h \rvert + 1), M_{ij} = M_{hg}$,	(1.11)
$SM_{ij} \geq SM_{hg} + PT_{hg} - Z_{ijhg} * \beta$,	$\forall i, \forall h, i! = h, \forall j \in (1, \lvert J_i \rvert + 1), g \in (1, \lvert J_h \rvert + 1), M_{ij} = M_{hg}$,	(1.12)
$SM_{hg} \geq SM_{i(j+1)} + tu_{i(j+1)h(g-1)} + tl_{hg} - (1 - Z_{ijhg}) * \beta$,	$\forall i, \forall h, i! = h, \forall j \in (1, \lvert J_i \rvert), g \in (2, \lvert J_h \rvert), M_{ij} = M_{hg}$,	(1.13)
$SM_{ij} \geq SM_{h(g+1)} + tu_{h(g+1)i(j-1)} + tl_{ij} - Z_{ijhg} * \beta$,	$\forall i, \forall h, i! = h, \forall j \in (2, \lvert J_i \rvert), g \in (1, \lvert J_h \rvert), M_{ij} = M_{hg}$,	(1.14)
$SM_{h1} \geq SM_{i(j+1)} + tu_{i(j+1)D} + tl_{h1} - (1 - Z_{ijh1}) * \beta$,	$\forall i, \forall h, i! = h, \forall j \in (1, \lvert J_i \rvert), M_{ij} = M_{h1}$,	(1.15)
$SM_{i1} \geq SM_{h(g+1)} + tu_{h(g+1)D} + tl_{i1} - Z_{i1hg} * \beta$,	$\forall i, \forall h, i! = h, \forall g \in (1, \lvert J_h \rvert), M_{i1} = M_{hg}$,	(1.16)
$X_{ijk} \in (0,1)$,	$\forall i, \forall j \in (1, \lvert J_i \rvert + 1), k \in (1, \lvert N_O \rvert)$,	(1.17)
$Z_{ijhg} \in (0,1)$,	$\forall i, \forall h, i! = h, \forall j \in (1, \lvert J_i \rvert + 1), g \in (1, \lvert J_h \rvert + 1)$,	(1.18)
$SM_{ij} > 0$,	$\forall i, j \in (1, \lvert J_i \rvert + 1)$,	(1.19)
$tu_{ijmn} = \lvert PM_{ij} - PM_{mn} \rvert \delta$,	$\forall i, m, j \in (2, \lvert J_i \rvert + 1), n \in (1, \lvert J_m \rvert + 1)$,	(1.20)
$tl_{ij} = \lvert PM_{ij} - PM_{i(j-1)} \rvert \delta$,	$\forall i, j \in (2, \lvert J_i \rvert + 1)$,	(1.21)
$tl_{i1} = \lvert PM_{i1} - PM_D \rvert \delta$,	$\forall i$,	(1.22)
$tu_{ijD} = \lvert PM_{ij} - PM_D \rvert \delta$	$\forall i, \forall j \in (1, \lvert J_i \rvert + 1)$.	(1.23)

The objective function (1.0) is to minimize the makespan of the entire assignment. Constraints (1.1) ensure that the makespan covers all operations including the stocking operations of all jobs. Constraint (1.2) defines O_{11} as the entrance of the whole system. Therefore, the start time of O_{11} equals the travelling time from the input depot to M_{11}. Accordingly, constraint (1.3) specifies that O_{11} has the highest priority and will be the first to be executed.

Constraints (1.4) specify the operation sequence within a job. Job *i* has to be remained on M_{ij} for at least a period of PT_{ij} before being transported to the next machine $M_{i(j+1)}$. Constraints (1.5) and (1.6) mandate that each operation only has one priority, while each priority is related to only one operation.

As the operations with higher priorities should start earlier than the operations with lower priorities, constraints (1.7) link the priorities with the operation start times by mandating that if the priority of O_{ij} (k) is higher than the priority of O_{mn} (p) (i.e., $k<p$), then, the start time of O_{ij} should be earlier than O_{mn}.

Constraints (1.8) regulate the relationship between two operations with successive priorities. If the robot plans to handle O_{ij} after delivering O_{mn}, the robot should conduct an empty move to $M_{i(j-1)}$, pick up job i, and then transport it to M_{ij}. If $O_{i(j-1)}$ has not been completed when the robot arrives, the robot will wait until it is finished, which is restricted by constraints (1.4).

Constraints (1.9) specify the start time of the first operation in each job i (except the first job). In the robot route, the immediate predecessor of O_{i1} must be an operation from another job (including the final dummy stock operations). Thus, after that preceding job has been released to the designated machine, the robot should first go to the input depot to pick up the initialized job i before transporting it to M_{i1}.

Since the machine can only process one operation at a time and the operations cannot be interrupted after starting, the operations executed by a machine are also sequenced. Therefore, Constraints (1.10) - (1.12) are needed. Constraints (1.10) guarantee that two operations assigned to the same machine can only have one execution sequence (either O_{ij} precedes O_{hg}, or vice versa). To specify the sequential relationship on the machine, either Constraints (1.11) or Constraints (1.12) must hold. For example, if O_{ij} precedes O_{hg} as restricted by (1.11), the starting time of O_{hg} should be later than the starting time of O_{ij} plus the processing time of O_{ij}.

In case of a deadlock situation, Constraints (1.13) - (1.16) formulate the deadlock-avoidance constraints for the machines as well as the robot with the following four considerations: (i) whether the operation's same-machine predecessor is still in execution; (ii) whether the operation's same-machine predecessor has been finished while not been released from the machine; (iii) whether the robot is busy delivering the operation's same-machine predecessor to the next destination; (iv) whether the robot is conducting an empty movement or waiting at the machine that executes the operation's same-job predecessor. If any one of the above mentioned four scenarios happens, the delivery of the current operation will lead to a deadlock situation. Thus, for operations other than the first operation in each job, either Constraints (1.13) or Constraints (1.14) will hold. If O_{hg} is the successor of O_{ij} on the same machine, O_{ij} should be removed and delivered to $M_{i(j+1)}$ before O_{hg} can start. Thus, the start time of O_{hg} should be later than the start time of $O_{i(j+1)}$ plus the empty movement from $M_{i(j+1)}$ to $M_{h(g-1)}$ and a loaded movement that transports job h from $M_{h(g-1)}$ to M_{hg}. Constraints (1.15) and Constraints (1.16) are deadlock-avoidance constraints designed for the first operations in each job, as these operations need transportation from the input depot.

Constraints (1.17) - (1.23) define the decision variables and parameters.

5. Network-RJSPDT

Different from the Position-RJSPDT, the Network-RJSPDT characterizes the job-shop scheduling and robot routing through a connected network. Following the network theory, the RJSPDT could be transformed into routing the robot among all operations under resource restrictions and other operational requirements. In the network, operations perform as nodes. The connected network stresses the balanced flow in the network. There are two foundations to support this idea: (i) the inflow linkages and the outflow linkages of an operation share the same machine; (ii) the restriction of robot availability guarantees the connectivity and consistency of the movement map.

Table 4. Decision variables of the Network-RJSPDT.

Continuous Decision Variables	
C_{max}	The total time needed to finish all the operations and deliver all jobs to the output depot.
SM_{ij}	The start time of O_{ij} on the assigned machine. SM_F denotes the time that the algorithm reaches the dummy sink point, which equals the makespan.
Binary Decision Variables	
Z_{ijhg}	Binary decision variable. It equals 1 when both O_{ij} and O_{hg} are executed on the same machine, and O_{ij} precedes O_{hg}; 0 otherwise.
X_{ijmn}	Binary decision variable. It equals 1 when the robot leaves for O_{mn} after O_{ij} starts, where i and m are two different jobs; 0 otherwise.
$Y_{ij(j+1)}$	Binary decision variable. It equals 1 when the robot waits for the completion of the current operation O_{ij} and goes to $M_{i(j+1)}$; 0 otherwise.

It is easily identified that in this network, each node might have two categories of successor, i.e., the same-job successor and the robot-routing successor. Similarly, each node also has two categories of predecessor, i.e., the direct same-job predecessor and the robot-routing predecessor. These four types of linkages, however, can be simplified into two categories of arcs, namely the "waiting-delivering" arc and the "returning-delivering" arc, which represent the two options for the robot to carry out the next operation. Accordingly, the two types of arcs (i.e., *X* and *Y* types) are characterized by the binary decision variables X_{ijmn} and $Y_{ij(j+1)}$. X_{ijmn} indicates whether a robot-routing sequential relationship exists between O_{ij} and O_{mn} (two operations from different jobs). This type of arc corresponds to the "returning and delivering" situation and consists of two movements: an empty movement back to the same-job predecessor of O_{mn} (from M_{ij} to $M_{m(n-1)}$), and a loaded movement from $M_{m(n-1)}$ to M_{mn}. $Y_{ij(j+1)}$ records the "waiting and delivering" connection and consists of the waiting time at the current station (i.e., PT_{ij}) and the travel time to the same-job successor (from M_{ij} to $M_{i(j+1)}$). Each node should be connected in the network with an *X* or *Y* type of arc.

Besides, a dummy sink node (*F*) is established in the model, which is the exit for the entire assignment, preventing the model from getting into a loop. Since the final operation of the assignment

will be connected with the sink node, the time that the model reaches the sink node is equal to the makespan. In this way, the whole network is connected by the two types of arcs mentioned above and it is guaranteed that each node (operation) will be visited and no loops will appear due to the balanced one-degree inflow and outflow limitation. Definitions of decision variables are summarized in Table 4, based on which the proposed Network-RJSPDT is formulated in Table 5.

Table 5. The proposed Network-RJSPDT.

Obj. Min C_{max}		(2.0)
s.t.		
$C_{max} \geq SM_F$,		(2.1)
$SM_F \geq SM_{ij}$	$\forall i, \forall j \in (1, \lvert J_i \rvert + 1)$,	(2.2)
$SM_{11} = \lvert PM_{11} - PM_D \rvert \delta$,		(2.3)
$SM_{ij} + PT_{ij} + tl_{i(j+1)} \leq SM_{i(j+1)}$,	$\forall i, \forall j \in (1, \lvert J_i \rvert)$,	(2.4)
$\sum_m \sum_n X_{ijmn} + Y_{ij(j+1)} = 1$,	$\forall i, \forall m, i! = m, j \in (1, \lvert J_i \rvert), n \in (1, \lvert J_m \rvert + 1)$,	(2.5)
$\sum_m \sum_n X_{mnij} + Y_{i(j-1)j} = 1$,	$\forall i, \forall m, i! = m, j \in (2, \lvert J_i \rvert + 1), n \in (1, \lvert J_m \rvert + 1)$,	(2.6)
$\sum_m \sum_n X_{i(\lvert J_i \rvert+1)mn} + X_{i(\lvert J_i \rvert+1)F} = 1$,	$\forall i, \forall m, i! = m, n \in (1, \lvert J_m \rvert + 1)$,	(2.7)
$\sum_m \sum_n X_{mni1} = 1$,	$\forall i, \forall m, i! = m, i\,! = 1, n \in (1, \lvert J_m \rvert + 1)$,	(2.8)
$SM_{ij} + PT_{ij} + tl_{i(j+1)} \leq SM_{i(j+1)} + (1 - Y_{ij(j+1)}) * \beta$,	$\forall i, \forall j \in (1, \lvert J_i \rvert)$,	(2.9)
$SM_{ij} \geq SM_{mn} + tu_{mni(j-1)} + tl_{ij} - (1 - X_{mnij}) * \beta$,	$\forall i, \forall m, i! = m, j \in (2, \lvert J_i \rvert + 1), n \in (1, \lvert J_m \rvert + 1)$,	(2.10)
$SM_{m(n+1)} \geq SM_{ij} + tu_{ijmn} + tl_{m(n+1)} - (1 - X_{mnij}) * \beta$,	$\forall i, \forall m, i! = m, j \in (1, \lvert J_i \rvert + 1), n \in (1, \lvert J_m \rvert)$,	(2.11)
$SM_{i1} \geq SM_{mn} + tu_{mnD} + tl_{i1} - (1 - X_{mni1}) * \beta$,	$\forall i, \forall m, i! = m, n \in (1, \lvert J_m \rvert + 1)$,	(2.12)
$Z_{ijhg} + Z_{hgij} = 1$,	$\forall i, \forall h, \forall j \in (1, \lvert J_i \rvert), g \in (1, \lvert J_h \rvert), M_{ij} = M_{hg}$,	(2.13)
$SM_{hg} \geq SM_{ij} + PT_{ij} - (1 - Z_{ijhg}) * \beta$,	$\forall i, \forall h, \forall j \in (1, \lvert J_i \rvert), g \in (1, \lvert J_h \rvert), M_{ij} = M_{hg}$,	(2.14)
$SM_{ij} \geq SM_{hg} + PT_{hg} - Z_{ijhg} * \beta$,	$\forall i, \forall h, \forall j \in (1, \lvert J_i \rvert), g \in (1, \lvert J_h \rvert), M_{ij} = M_{hg}$,	(2.15)
$SM_{hg} \geq SM_{i(j+1)} + tu_{i(j+1)h(g-1)} + tl_{hg} - (1 - Z_{ijhg}) * \beta$,	$\forall i, \forall h, \forall j \in (1, \lvert J_i \rvert), g \in (2, \lvert J_h \rvert), M_{ij} = M_{hg},$,	(2.16)
$SM_{ij} \geq SM_{h(g+1)} + tu_{h(g+1)i(j-1)} + tl_{ij} - Z_{ijhg} * \beta$,	$\forall i, \forall h, \forall j \in (2, \lvert J_i \rvert), g \in (1, \lvert J_h \rvert), M_{ij} = M_{hg},$,	(2.17)
$SM_{h1} \geq SM_{i(j+1)} + tu_{i(j+1)D} + tl_{h1} - (1 - Z_{ijh1}) * \beta$,	$\forall i, \forall h, \forall j \in (1, \lvert J_i \rvert), M_{ij} = M_{h1}$,	(2.18)
$SM_{i1} \geq SM_{h(g+1)} + tu_{h(g+1)D} + tl_{i1} - Z_{i1hg} * \beta$,	$\forall i, \forall h, \forall g \in (1, \lvert J_h \rvert), M_{i1} = M_{hg}$,	(2.19)
$X_{ijmn} \in (0,1)$,	$\forall i, \forall m \in (1, \lvert I \rvert + 1), i! = m, j \in (1, \lvert J_i \rvert + 1), \mathrm{n} \in (1, \lvert J_m \rvert + 1)$,	(2.20)
$Y_{ij(j+1)} \in (0,1)$,	$\forall i, \forall j \in (1, \lvert J_i \rvert)$,	(2.21)
$Z_{ijhg} \in (0,1)$,	$\forall i, \forall h, i! = h, \forall j \in (1, \lvert J_i \rvert), g \in (1, \lvert J_h \rvert)$,	(2.22)
$SM_{ij} > 0$,	$\forall i, \forall j \in (1, \lvert J_i \rvert + 1)$,	(2.23)
$tu_{ijmn} = \lvert PM_{ij} - PM_{mn} \rvert \delta$,	$\forall i, \forall m, j \in (2, \lvert J_i \rvert + 1), n \in (1, \lvert J_m \rvert + 1)$,	(2.24)
$tl_{ij} = \lvert PM_{ij} - PM_{i(j-1)} \rvert \delta$,	$\forall i, \forall j \in (2, \lvert J_i \rvert + 1)$,	(2.25)
$tl_{i1} = \lvert PM_{i1} - PM_D \rvert \delta$,	$\forall i$,	(2.26)
$tu_{ijD} = \lvert PM_{ij} - PM_D \rvert \delta$	$\forall i, \forall j \in (1, \lvert J_i \rvert + 1)$.	(2.27)

The objective function (2.0) is to minimize the makespan of the entire assignment which is defined in Constraint (2.1). Constraints (2.2) regulate that the completion time of the last operation in the assignment equals the time that the algorithm reaches the dummy sink node (*F*). Constraints (2.3) and

(2.4) are equivalent to Constraints (1.2) and (1.4). Constraints (2.5) - (2.8) are the balanced flow restrictions designed to guide the robot's movement throughout the network. Constraints (2.5) and (2.6) guarantee an exact coverage for each operation by one of the two types of arcs in the network. In particular, the robot has two options at each node: going to the same-job successor directly or turning to another job. Therefore, the value of either $Y_{ij(j+1)}$ or a certain X_{ijmn} will be 1. If $Y_{ij(j+1)}$=1, then the sum of X_{ijmn} equals to zero, which means that the robot will wait at the current station for transporting the same job for its next operation. Otherwise, if there is a certain X_{ijmn} equal to 1, then $Y_{ij(j+1)}$ must be zero, which means that the robot will turn to handling another operation of a different job.

Constraints (2.7) and (2.8) model the boundary situations for each job (i.e., the first and last operations). $O_{i|Ji|+1}$ is the last dummy stock operation in job i and this operation must be connected to an operation in a different job, or the sink node F as regulated by Constraints (2.7). Actually, only one dummy stock operation can be linked with F. Correspondingly, as the first operation O_{i1} of job i has no same-job predecessor, Constraints (2.8) indicate that the preceding operation of O_{i1} should be an operation in a different job. O_{11} is excluded from this constraint since it has been defined as the system entrance point by Constraints (2.2), which is same as the Position-RJSPDT.

Constraints (2.9) - (2.11) formulate the relationship between the starting times of two connected operations in the robot route. First, as specified in Constraints (2.9), if the robot chooses to implement the same-job successor O_{ij+1} after loading job i onto M_{ij}, a Y arc is used to connect O_{ij} and O_{ij+1}. In such circumstances, the robot needs to wait for a period of PT_{ij} before picking up job i and conducting a loaded movement to M_{ij+1}. Second, Constraints (2.10) ensure that if O_{mn} is the predecessor of O_{ij} in the robot route, O_{mn} is linked with O_{ij} through an X arc. In this circumstance, the robot will first conduct an empty movement to M_{ij-1}, pick up job i when O_{ij-1} is finished (the robot will wait if O_{ij-1} has not been completed as restricted by Constraints (2.4)), and then deliver it to M_{ij}. It is noticeable that Constraints (2.11) are critical in terms of the completeness of the robot movement map, which formulates the relationship between the starting times of O_{mn+1} and O_{ij}, when O_{ij} is the successor of O_{mn} in the robot route. In this circumstance, O_{ij} must precede O_{mn+1}. Thus, if O_{mn+1} is to be handled, then at least the time spent on an empty movement from M_{ij} to M_{mn} should be considered before the loaded movement to O_{mn+1}. This situation appears when O_{mn+1} is immediately after O_{ij}, which could be controlled by Constraints (2.11).

Constraints (2.12) distinguish the starting time of O_{i1} from other operations. Supposing that O_{i1} is connected with O_{mn} through an X arc, then the robot should firstly conduct an empty movement from M_{mn} to the input depot before delivering job i to M_{i1}.

Constraints (2.13) - (2.15) show that the operations should satify the machine-sequence restrictions. Besides, Constraints (2.16) - (2.19) are the deadlock-avoidance constraints which have been discussed for the Position-RJSPDT. Last, Constraints (2.20) - (2.27) define the decision variables and parameters.

To further illustrate the characteristics of the Position-RJSPDT and Network-RJSPDT, Table 6 summarizes the differences between the two models.

Table 6. Differences between the two models.

	Position-RJSPDT	**Network-RJSPDT**
Modelling idea	Each operation should be assigned with an execution position on the robot.	Each node has one inflow arc and one outflow arc which share the same machine.
Binary decision variable	X_{ijk}= 1 if O_{ij} is scheduled as the *k-th* operation to be executed by the robot.	X_{ijmn}= 1 if the robot leaves for O_{mn} after O_{ij} starts, where *i* and *m* are two different jobs. Y_{ijj+1}= 1 if the robot waits for the completion of the current operation O_{ij} and goes to M_{ij+1}.
Constraints	Successive executions by the robot.	Network balance and connection.

6. The Efficiency of the Deadlock-avoidance Constraints

In this section, a numerical example that contains three jobs and three operations in each job (please see Table 7) is used to demonstrate the solutions of the Position-RJSPDT and Network-RJSPDT. Then, we generalize the significance of the deadlock-avoidance constraints by comparing the models with and without the deadlock-avoidance constraints.

Table 7. An example with 3 jobs × 3 operations.

	job 1	**job 2**	**job 3**
Operations	O_{11}, O_{12}, O_{13}	O_{21}, O_{22}, O_{23}	O_{31}, O_{32}, O_{33}
Specified Operating Machine	Machine 1, Machine 3, Machine 1	Machine 1, Machine 2, Machine 3	Machine 3, Machine 2, Machine 1
Processing Time (mins)	2, 3, 4	3, 4, 5	2, 4, 3

The optimal solution to this instance is given in Table 8. Both models obtain the same optimal makespan of 40. The Position-RJSPDT directly generates the execution priority for each operation, while the Network-RJSPDT derives the robot-routing connections between each pair of predecessor and successor within the network. Most predecessors and successors are connected by *X* arcs except that three *Y* arcs are used to connect O_{21} & O_{22}, O_{31} & O_{32}, and O_{33} & O_{34}. The movement of the robot is depicted in Figure 5, where *Y* arcs are denoted by solid lines, while *X* arcs consisting of an empty movement and a loaded movement are depicted by dash lines. The components of the route for the robot are indexed with an increasing number in Figure 5.

Table 8. The operation starting times (min) for the two proposed models in the example.

Position-RJSPDT					**Network-RJSPDT**				
	operation 1	operation 2	operation 3	stock		operation 1	operation 2	operation 3	stock
job 1	1	11	31	38	job 1	1	11	31	38
job 2	24	28	33	40	job 2	24	28	33	40

job 3	5	8	13	19	job 3	5	8	13	19

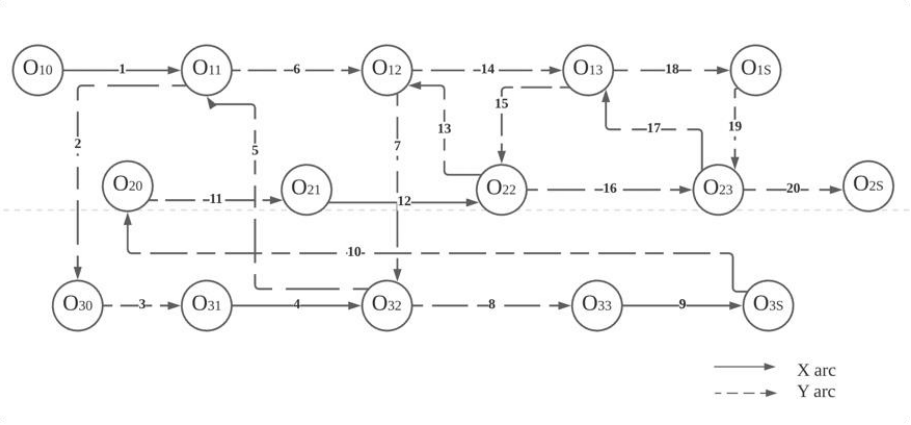

Figure 5. The robot route in the example.

Figure 6 presents the execution flow of machines and the robot according to the elapse time. The metrics for the elapsed time in Figure 6-8 are minutes (mins). The horizontal axis represents the time elapsed, while the vertical axis stands for the three machines and the robot. Three jobs and their related activities on the robot are distinguished by using colors: job 1 (green), job 2 (orange), and job 3 (blue). Specifically, the lighter color is for jobs, while the corresponding deeper color is for the robot. Besides, loaded movements and empty movements are distinguished by L_{ij} and E_{ij}, respectively. Moreover, ES_i is the empty movement for the stock operation of job i, while LS_i is the loaded movement for the stock operation of job i. The blocking activity is denoted by B_i. For example, at the beginning, the robot delivers the first operation of the first job from the input depot to its operating machine (i.e., Machine 1). Then, the robot moves to the input depot without carrying anything to pick up the third job, and then transport it to the operating machine for its first operation (i.e., Machine 3). After O_{11} is finished by Machine 1, as the robot is not available, job 1 has to be blocked in Machine 1 to wait for the robot. Similarly, job 1 would also wait at Machine 3 for the robot after its second operation is completed.

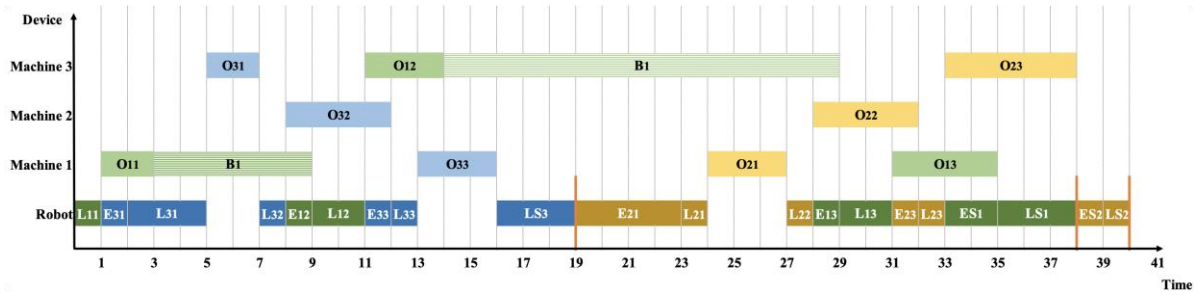

Figure 6. The optimal flow of machines and robot for the example.

From the results of the example, it is seen that there is no conflict for the system (i.e., no deadlock). This is because that the deadlock-avoidance constraints are incorporated in both proposed models. To verify the efficacy of these constraints, we examine the performances of the two models without the

deadlock-avoidance constraints (named as the Position-RJSPDTND and Network-RJSPDTND) and the results are summarized in Table 9. The optimal makespan obtained now becomes 32. However, this reduced makespan is actually infeasible due to deadlocks as indicated by the infeasible outcomes, with which the robot will definitely meet a transportation failure in real operations.

Table 9. The operation starting times (min) without the deadlock-avoidance constraints.

	Position-RJSPDTND					**Network-RJSPDTND**			
	operation 1	operation 2	operation 3	stocking		operation 1	operation 2	operation 3	stocking
job 1	1	13	19	30	job 1	1	13	19	30
job 2	3	16	25	32	job 2	11	16	25	32
job 3	7	10	15	22	job 3	5	8	15	22

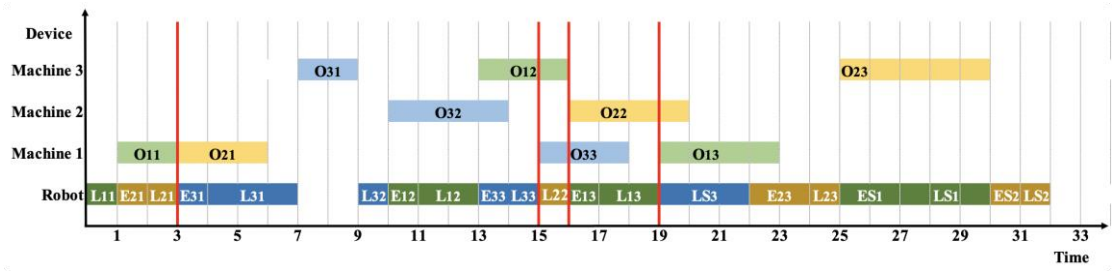

Figure 7. System conflicts generated by the Position-RJSPDTND.

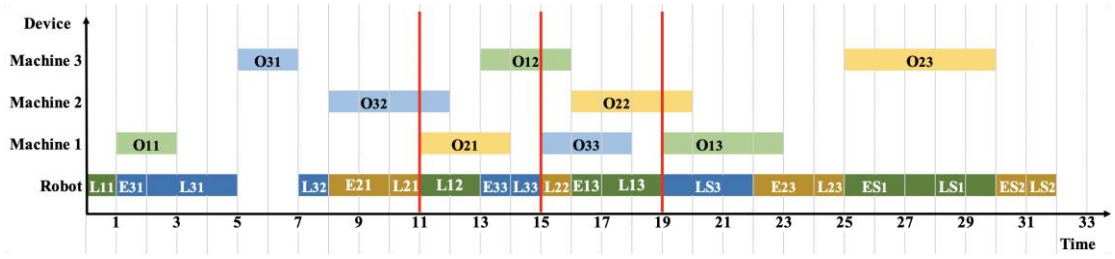

Figure 8. System conflicts generated by the Network-RJSPDTND.

Take a closer look at Figure 7, the execution of O_{11} on Machine 1 is scheduled from 1s to 3s. The Position-RJSPDTND, therefore, supposes that O_{21} can start from 3s based on two reasons. Firstly, O_{11} on the same machine has been finished. Second, there is enough time for the robot to return to the input depot and bring job 2 to Machine 1. However, such a schedule violates the no-buffer restriction so that a deadlock appears, since job 1 is still remained on Machine 1 while the single robot is occupied by job 2. Accordingly, the subsequent plans are also infeasible. Similar dilemma also occurs in the solution provided by the Network-RJSPDTND. In the result (as depicted in Figure 8), the robot starts the delivery of job 2 after job 3 is placed on Machine 2 for the execution of O_{32}. However, at the time when job 2 is delivered to Machine 1 (11s), job 1 has not been removed from Machine 1 yet. Accordingly, a deadlock situation occurs. The conflict points in both schedules are highlighted with vertical red lines in Figure 7 and Figure 8.

In conclusion, through the performance comparisons between the models with and without the deadlock-avoidance constraints, the significance of the proposed deadlock-avoidance constraints in avoiding system conflicts is demonstrated.

7. Computational Experiments

In this section, computational experiments are carried out to test the performances of the proposed models. Both the Position-RJSPDT and the Network-RJSPDT are coded in OPL. The IBM commercial solver CPLEX Studio IDE 12.10 is used to solve these problems on a desktop MacBook Pro with 1.4 GHz Intel Core i5 processor and 8 GB of RAM memory. The upper limit for the running time of each instance is set as 5200s. Twenty instances are applied for the experiments, and the features are summarized in Table 10. Note that each instance is featured by three dimensions: the number of jobs, the number of operations in a job, and the number of machines, as shown in the second and the fourth columns of Table 10. In the following, we first compare the model sizes and computational times of the two models in Section 7.1. Then, sensitivity analysis is carried out to test the impacts of the number of jobs and number of operations in a job on the model performances in Section 7.2. Last, Section 7.3 examines how the entrance strategies would affect the solutions.

Table 10. Instance characteristics.

Instance	Scale	Instance	Scale
1	3×3×3	11	6×5×4
2	3×5×3	12	7×4×3
3	3×5×4	13	7×4×4
4	3×7×4	14	7×5×4
5	4×4×4	15	8×4×3
6	4×5×4	16	8×5×3
7	5×4×3	17	9×4×3
8	5×4×4	18	9×5×3
9	5×5×4	19	10×3×3
10	6×5×3	20	10×4×3

7.1 Comparisons of Model Sizes and Computational Times

The performances of the Position-RJSPDT and the Network-RJSPDT are compared from two perspectives, namely the model size and computational time.

First of all, the features of the two proposed models for the twenty instances are summarized in Table 11. To be specific, two indicators are used to measure the model sizes: the number of constraints (NCs) and the number of decision variables (NBVs) (the binary ones). The last two columns of Table 11 shows the comparisons between the two models (i.e., (Position-RJSPDT - Network-RJSPDT)/ Position-RJSPDT). As shown from this table, the Network-RJSPDT generates 99.6% less constraints and 15.9% less decision variables compared with the Position-RJSPDT. From Figure 9 and Figure 10,

we can also clearly observe the larger sizes of the Position-RJSPDT over the Network-RJSPDT for all instances. When the sizes of instances become larger, the number of constraints derived by the Position-RJSPDT grows dramatically, while that by the Network-RJSPDT can remain in a relatively low level. Accordingly, the Network-RJSPDT shows higher modelling stability than the Position-RJSPDT.

Table 11. Comparisons of model sizes between the two models.

Instance	Position-RJSPDT		Network-RJSPDT		Comparisons	
	NCs	NBVs	NCs	NBVs	NCs	NBVs
1	15131	170	269	128	98.2%	24.7%
2	74191	382	607	286	99.2%	25.1%
3	74156	368	572	272	99.2%	26.1%
4	230872	652	1006	478	99.6%	26.7%
5	126312	458	743	366	99.4%	20.1%
6	259642	662	1085	530	99.6%	19.9%
7	325508	747	1265	627	99.6%	16.1%
8	325448	723	1205	603	99.6%	16.6%
9	670238	1044	1756	874	99.7%	16.3%
10	1439592	1578	2772	1368	99.8%	13.3%
11	1439427	1512	2607	1302	99.8%	13.9%
12	1323944	1495	2614	1313	99.8%	12.2%
13	1323784	1431	2454	1249	99.8%	12.7%
14	2731806	2068	3616	1816	99.9%	12.2%
15	2297174	1950	3431	1734	99.9%	11.1%
16	4743346	2822	5067	2526	99.9%	10.5%
17	3727457	2477	4388	2225	99.9%	10.2%
18	7700372	3584	6486	3242	99.9%	9.5%
19	2362723	1960	3365	1750	99.9%	10.7%
20	5739263	3064	5455	2774	99.9%	9.5%
				Average	**99.6%**	**15.9%**

NCs: the number of constraints; NBVs: the number of decision variables.

The reasons of the discrepancies in model sizes between the Network-RJSPDT and the Position-RJSPDT are discussed as follows. For the Position-RJSPDT, it contains a series of functions that would create a huge number of constraints in multiple loops. For instance, assuming that there are ρ jobs and μ operations in each job, Functions (1.8) formulating the relationship between the two operations which are delivered by the robot successively, generate $[\rho(\rho\text{-}1)\times\mu(\mu\text{+}1)\times(\rho(\mu\text{+}1)\text{-}1)]$ pieces of constraints. Moreover, the number of constraints generated by Functions (1.7) will reach up to ρ^8 when ρ equals μ. Accordingly, when the problem sizes increase, the growth in the number of constraints for the Position-RJSPDT is drastic. In contrast, the Network-RJSPDT focuses on local relationships based on each individual node, instead of the whole network. Assuming that there are ρ jobs and μ operations in each job, the number of constraints generated by each function of the Network-RJSPDT is no more than

$\rho^2\mu^2$. Accordingly, the modelling idea of the Network-RJSPDT is shown to be advantageous over that of the Position-RJSPDT by generating much fewer constraints. In conclusion, it is demonstrated that the Position-RJSPDT encounters larger model sizes than the Network-RJSPDT, which implies a higher problem complexity and a lower solution efficiency.

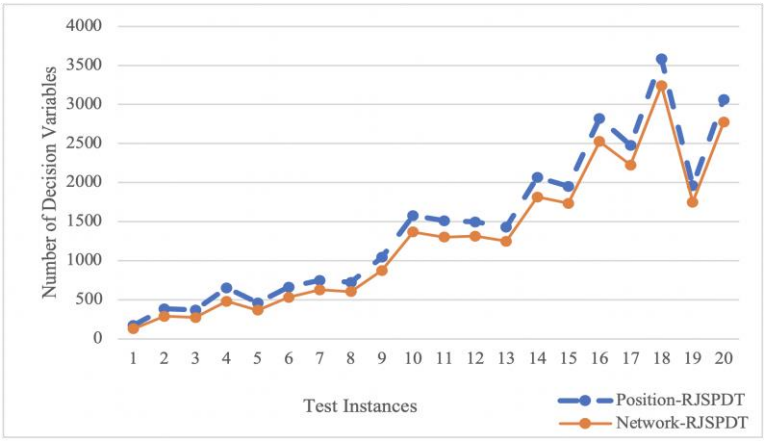

Figure 9. Comparisons of the number of variables for the tested instances.

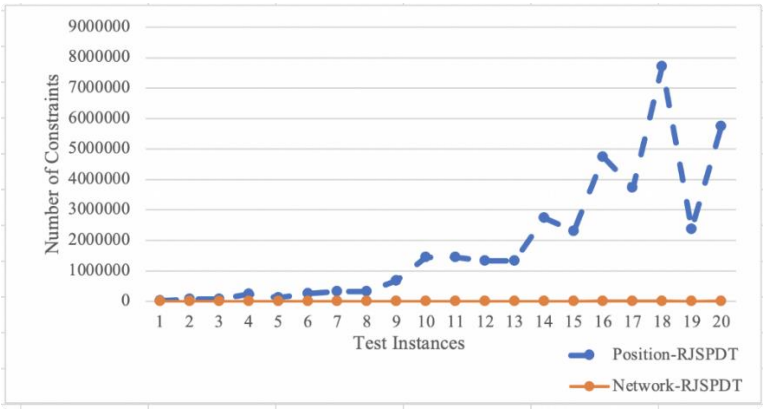

Figure 10. Comparisons of the number of constraints for the tested instances.

Accordingly, in terms of the computational times (as listed in Table 12), it is reasonable to observe the superior performances of the Network-RJSPDT over the Position-RJSPDT. On one hand, for Instances 1 to 7 (named as "*small-scale instances*" hereafter), both models can obtain the optimal solutions with the same makespan within the upper time limit. However, as shown in the last column of Table 12, the Network-RJSPDT consumes much less computational time than the Position-RJSPDT for these small-scale instances (i.e., an average of 96%). Besides, it is seen that along with the increase in the instance size, the computational time reduction achieved by the Network-RJSPDT over the Position-RJSPDT becomes increasingly remarkable, which shows the shortcomings of the Position-RJSPDT when the problem sizes grow. On the other hand, for Instances 8 to 20 (named as "*large-scale instances*" hereafter), the Position-RJSPDT fails to identify a solution within 5200s, while the Network-

RJSPDT can obtain the shortest makespan for most of these instances. Note that Instances 13 & 14 involve more machines than Instances 15-18. Therefore, Instances 13 & 14 are more computationally difficult in finding a solution than Instances 15-18 by the Network-RJSPDT. Besides, for Instances 19 & 20, the overall problem complexity is the highest among all instances due to the integrated effect of the three dimensions. Accordingly, Instances 19 & 20 are also unsolvable in the given time limit.

In conclusion, the modelling advantages of the network-based approach over the traditional position-based approach are verified through the comparisons regarding model sizes and computational times. From the modelling process and the computational results, we can obtain the following important insights: (i) the large number of binary decision variables and multi-loop constraints produced by the position-based modelling idea lead to the high computational difficulty; and (ii) the network-based approach can effectively reduce the model size and computational time because the constraints in the Network-RJSPDT concentrate on modelling the neighboring relationships for each node (or operation).

Table 12. Comparisons of the computational times between the two models.

<table>
<tr><th rowspan="2">Instance</th><th colspan="3">Position-RJSPDT</th><th colspan="3">Network-RJSPDT</th><th rowspan="2">Comparison of computational time</th></tr>
<tr><td>Computational time (s)</td><td>Iterations</td><td>Makespan (mins)</td><td>Computational time (s)</td><td>Iterations</td><td>Makespan (mins)</td></tr>
<tr><td>1</td><td>1.7</td><td>8263</td><td>45</td><td>0.36</td><td>56</td><td>45</td><td>78.80%</td></tr>
<tr><td>2</td><td>19</td><td>69096</td><td>60</td><td>0.24</td><td>83</td><td>60</td><td>98.70%</td></tr>
<tr><td>3</td><td>11</td><td>50492</td><td>69</td><td>0.3</td><td>261</td><td>69</td><td>97.30%</td></tr>
<tr><td>4</td><td>90</td><td>121451</td><td>79</td><td>0.6</td><td>9288</td><td>79</td><td>99.30%</td></tr>
<tr><td>5</td><td>76</td><td>169390</td><td>73</td><td>1.3</td><td>26523</td><td>73</td><td>98.30%</td></tr>
<tr><td>6</td><td>384</td><td>494540</td><td>86</td><td>2.3</td><td>77448</td><td>86</td><td>99.40%</td></tr>
<tr><td>7</td><td>326</td><td>271364</td><td>76</td><td>1.8</td><td>26555</td><td>76</td><td>99.40%</td></tr>
<tr><td>8</td><td rowspan="12">Unsolvable within the given time limit</td><td></td><td></td><td>12</td><td>476289</td><td>91</td><td rowspan="12">NA</td></tr>
<tr><td>9</td><td></td><td></td><td>25</td><td>728398</td><td>104</td></tr>
<tr><td>10</td><td></td><td></td><td>8</td><td>83944</td><td>113</td></tr>
<tr><td>11</td><td></td><td></td><td>746</td><td>11954872</td><td>130</td></tr>
<tr><td>12</td><td></td><td></td><td>48</td><td>1945193</td><td>111</td></tr>
<tr><td>13</td><td colspan="3" rowspan="2">Unsolvable within the given time limit</td></tr>
<tr><td>14</td></tr>
<tr><td>15</td><td></td><td></td><td>200</td><td>3698493</td><td>123</td></tr>
<tr><td>16</td><td></td><td></td><td>682</td><td>3984601</td><td>142</td></tr>
<tr><td>17</td><td></td><td></td><td>1930</td><td>26227306</td><td>135</td></tr>
<tr><td>18</td><td></td><td></td><td>5155</td><td>48832905</td><td>160</td></tr>
<tr><td>19</td><td colspan="3" rowspan="2">Unsolvable within the given time limit</td></tr>
<tr><td>20</td></tr>
</table>

7.2 Sensitivity Analysis

In this section, we conduct sensitivity analysis to investigate the impacts of crucial parameters (i.e., the number of jobs (NJ) and the number of operations in a job (NOJ)) on the performance of models.

We carry out two groups of experiments for comparisons. In the first group (i.e., G1), the NOJ remains unchanged as 3, while the NJ is increased from 2 to 10 by a step of 1. In the second group (i.e., G2), the NJ remains unchanged as 3, while the NOJ is increased from 2 to 10 by a step of 1. The Position-RJSPDT and the Network-RJSPDT are then examined for these derived instances. Figure 11 depict the increase in the makespan (metric: minutes) along with the rise in instance scales for the two groups of experiments. It is seen that the Position-RJSPDT becomes intractable within the given time limit along with the growth in the instance size (i.e., from (7×3×3) in Group 1 and (3×8×3) in Group 2), while the Network-RJSPDT can solve all the problems.

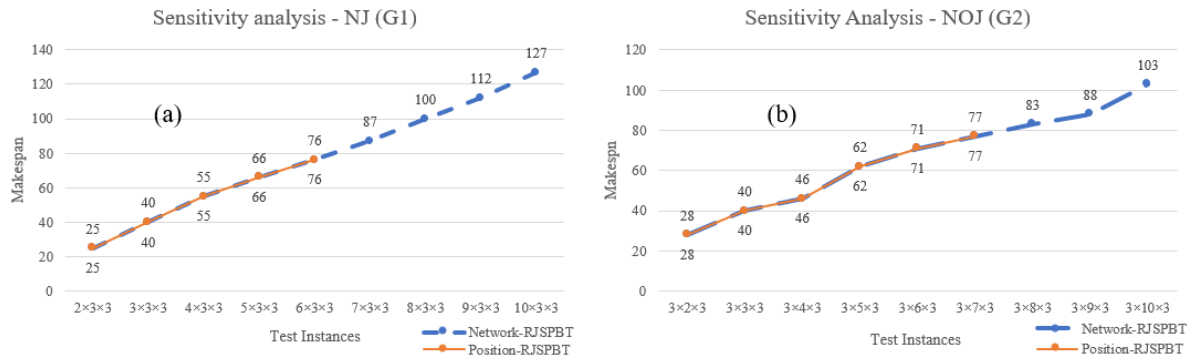

Figure 11. Makespan growths of Position-RJSPDT and Network-RJSPDT by increasing NJ (G1) and NOJ (G2).

Table 13. Sensitivity analysis for Position-RJSPDT.

Instance	Number of Jobs (NJ) - Group 1				Instance	Number of operations in a job (NOJ) - Group 2			
	Makespan (mins)	CT (s)	NBVs	NCs		Makespan (mins)	CT (s)	NBVs	NCs
2×3×3	25	0.3	72	2387	3×2×3	28	0.24	93	4947
3×3×3	40	1.1	168	15126	3×3×3	40	1.1	168	15126
4×3×3	55	16	304	52459	3×4×3	46	6.5	263	36224
5×3×3	66	62	480	134738	3×5×3	62	12.4	380	74186
6×3×3	76	270	696	288459	3×6×3	71	31	519	136248
7×3×3	Unsolvable within the given time limit				3×7×3	77	69	678	230937
8×3×3					3×8×3	Unsolvable within the given time limit			
9×3×3					3×9×3				
10×3×3					3×10×3				

CT (s): Computational time in seconds.

7.2.1 Impacts of NJ and NOJ on Model Sizes and Computational Times - Position-RJSPDT

Table 13 presents the model performance changes caused by the increase of the NJ and the NOJ for the Position-RJSPDT. It is obvious to identify that the increase in the NJ leads to much longer computational times than that in the NOJ (as depicted in Figure 14(a) and Figure 14(b)). For instance, when the NJ grows from 2 to 6, the Position-RJSPDT consumes 269.7 (i.e., 270-0.3) more seconds for solution derivation. However, this figure becomes 30.76 (i.e., 31-0.24) when the NOJ grows from 2 to 6. Moreover, the NJ also shows a greater impact on the model size than the NOJ, especially in terms of the number of constraints (as shown in Figure 12 and Figure 13). For example, the number of constraints generated by Instance (6×3×3) is 111.7% more than that by Instance (3×6×3).

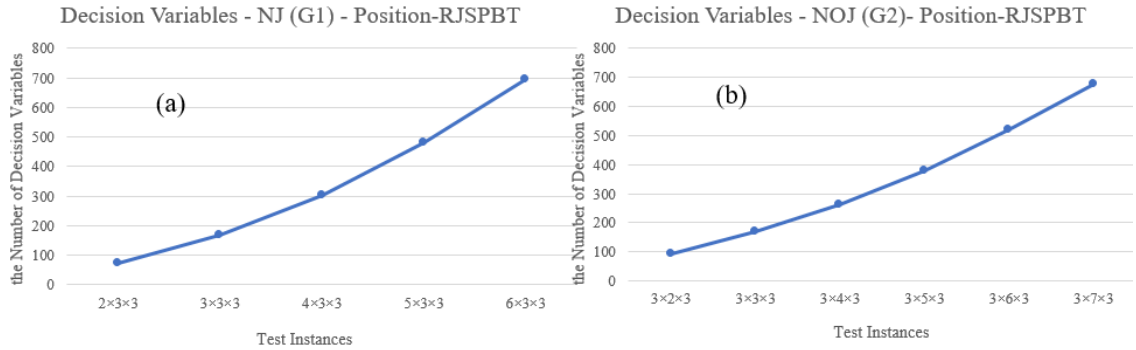

Figure 12. Decision variable growths of Position-RJSPDT by increasing NJ (G1) and NOJ (G2).

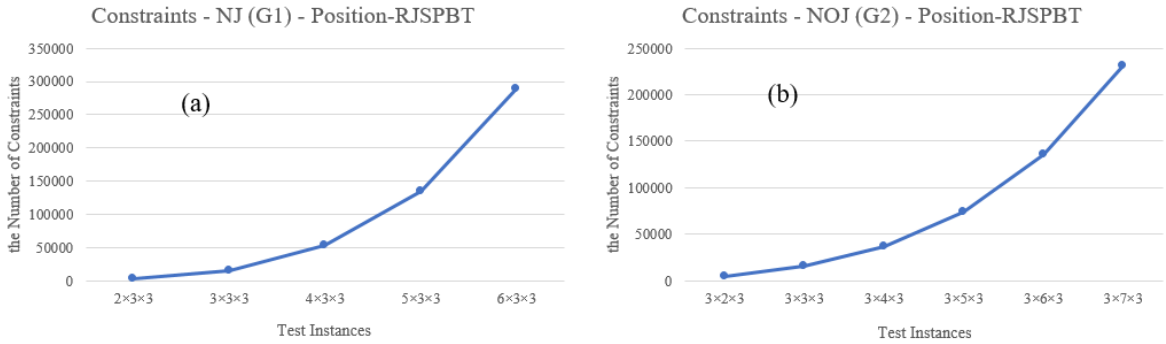

Figure 13. Constraint growths of Position-RJSPBT by increasing NJ (G1) and NOJ (G2).

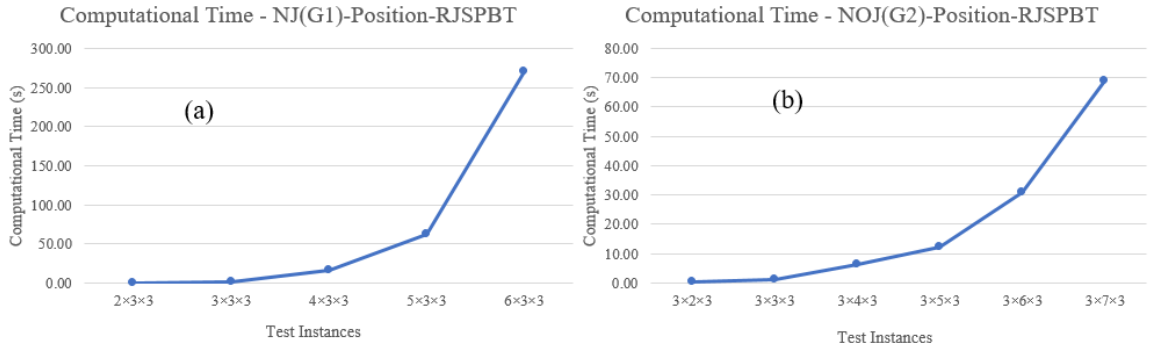

Figure 14. Computational time growths of Position-RJSPDT by increasing NJ (G1) and NOJ (G2).

7.2.2 Impacts of NJ and NOJ on Model Sizes and Computational Times - Network-RJSPDT

With the increase in the NJ and NOJ, the number of decision variables and the number of constraints in the Network-RJSPDT both show stable growths (see Figure 15 and Figure 16). Similar to the Position-RJSPDT, the impact of the NJ is greater than that of the NOJ. In terms of the computational time, a much sharper increasing trend can be observed along with the increase in the NJ. To be specific, when the NJ grows from 7 to 8, the computational time increases sharply from 7s to 182s, while the figure rises from 619s to the surprising 3620s when the NJ grows from 9 to 10. On the other hand, all instances can be solved within 1s when the NOJ increases from 2 to 10 (as demonstrated in Table 14 and Figure 17).

Table 14. Sensitivity analysis for Network-RJSPDT.

Number of Jobs (NJ) - Group 1					**Number of operations in a job (NOJ) - Group 2**				
Instance	Makespan (mins)	CT (s)	NBVs	NCs	Instance	Makespan (mins)	CT (s)	NBVs	NCs
2×3×3	25	0.12	46	107	3×2×3	28	0.12	69	144
3×3×3	40	0.12	126	267	3×3×3	40	0.13	126	267
4×3×3	55	0.2	244	498	3×4×3	46	0.16	197	419
5×3×3	66	0.55	421	800	3×5×3	62	0.17	284	605
6×3×3	76	2.3	594	1173	3×6×3	71	0.22	387	825
7×3×3	87	7.0	826	1617	3×7×3	77	0.33	504	1074
8×3×3	100	182	1096	2123	3×8×3	83	0.46	641	1367
9×3×3	112	619	1404	2718	3×9×3	88	0.65	792	1689
10×3×3	127	3620	1750	3365	3×10×3	103	1.0	957	2040

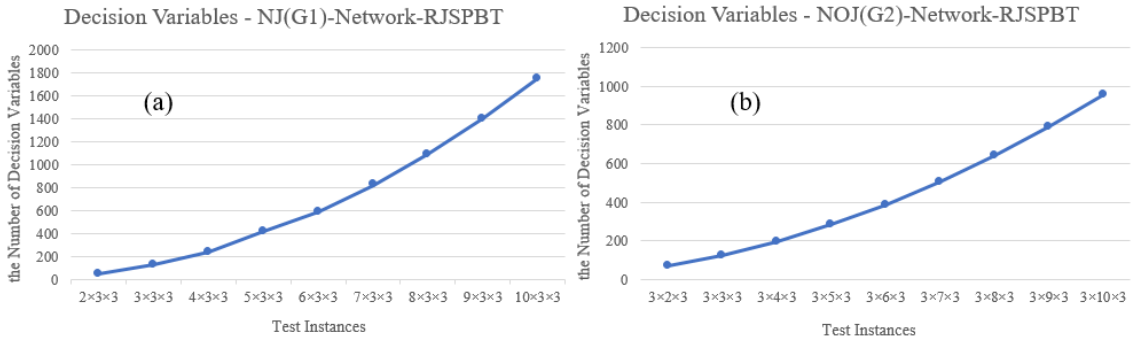

Figure 15. Decision variable growths of Network-RJSPDT by increasing NJ (G1) and NOJ (G2).

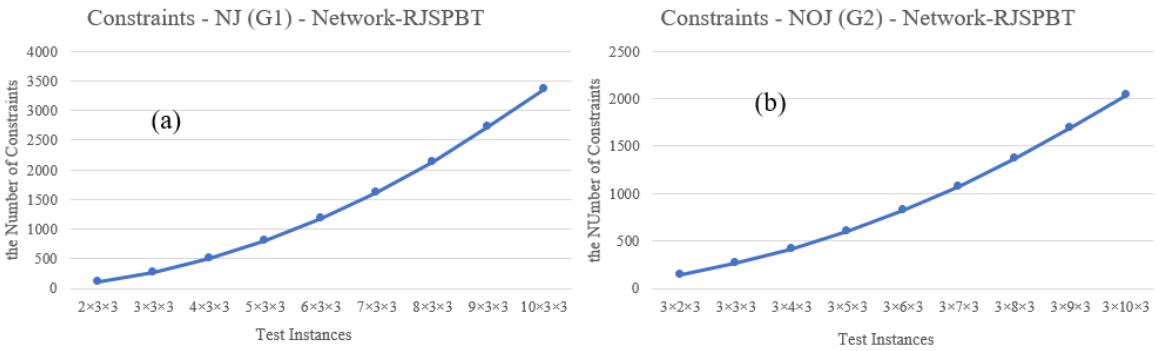

Figure 16. Constraint growths of Network-RJSPBT by increasing NJ (G1) and NOJ (G2).

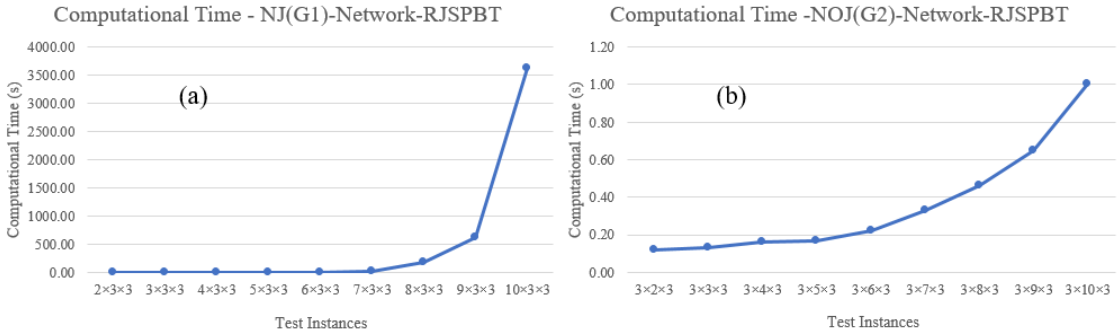

Figure 17. Computational time growth of Network-RJSPDT by increasing NJ (G1) and NOJ (G2).

In conclusion, both the Position-RJSPDT and Network-RJSPDT are more sensitive to the changes in the number of jobs compared with the number of operations in a job.

Table 15. The impact of different entrance strategies.

Instance	Network-FE-RJSPDT				Network-RJSPDT				Comparison	
	Makespan (mins)	CT (s)	NCs	NBVs	Makespan (mins)	CT (s)	NCs	NBVs	Makespan (mins)	CT (s)
1	45	0.59	295	131	45	0.36	269	128	0.00%	-39.00%
2	60	0.5	645	289	60	0.24	607	286	0.00%	-52.00%
3	68	0.47	620	275	69	0.3	572	272	1.40%	-36.20%
4	77	1.7	1056	481	79	0.6	1006	478	2.50%	-64.70%
5	69	2.3	806	370	73	1.3	743	366	5.50%	-43.50%
6	83	6.8	1159	534	86	2.3	1085	530	3.50%	-66.20%
7	72	6	1367	632	76	1.8	1265	627	5.30%	-70.00%
8	89	50.8	1307	608	91	12	1205	603	2.20%	-76.40%
9	102	45	1887	879	104	25	1756	874	1.90%	-44.40%
10	113	102	2954	1374	113	8	2772	1368	0.00%	-92.20%
12	110	145	2826	1320	111	48	2614	1313	0.90%	-66.90%
15	119	749	3713	1742	123	198	3431	1734	3.30%	-73.60%
								Average	2.21%	-60.43%

7.3 The Impact of Entrance Strategies

In this section, we examine the impact of different entrance strategies (i.e., fixed or flexible entrance strategies) on the model performances. In our proposed models, we apply the fixed entrance strategy. That is, both the Position-RJSPDT and Network-RJSPDT designate O_{11} (i.e., the first operation in the first job) as the production line entrance by ranking O_{11} with the highest priority or defining O_{11} as the first operation to execute. It is worthwhile to note that the fixed entrance strategy may lead to sub-optimality compared with the flexible entrance strategy. Therefore, we modify the job operation entrance strategy to allow the algorithm to determine the production process to start from which specific job in the assignment (i.e., the flexible entrance strategy). From our previous discussions, it is shown that the Position-RJSPDT suffers from larger model sizes and longer computational times than the Network-RJSPDT. Besides, the flexible entrance strategy complicates the problem significantly than the fixed entrance strategy which is applied in our problem setting. Therefore, in this section, a set of experiments is carried out to examine the impacts of the entrance strategies based on the Network-RJSPDT for the tractable instances. Accordingly, the Network-RJSPDT is modified into the *Network-based Flexible Entrance RJSPDT* (Network-FE-RJSPDT). To build the Network-FE-RJSPDT, a dummy starting point is established to connect with a certain O_{i1} ($i \in I$) which serves as the starting operation. The impacts of this modification on the model sizes, makespan, and computational times are demonstrated in Table 15.

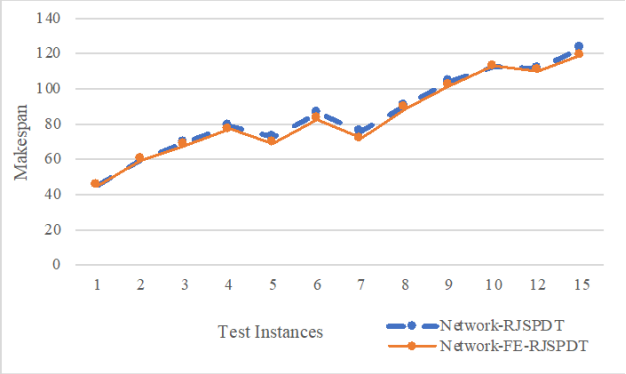

Figure 18. Comparisons of makespan of the Network-RJSPDT and the Network-FE-RJSPDT.

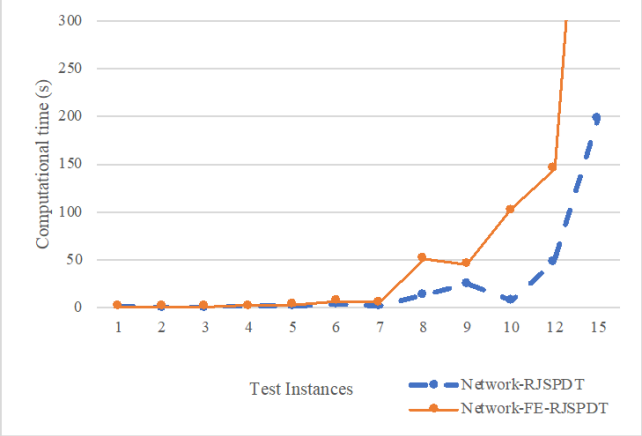

Figure 19. Comparisons of computational times of the Network-RJSPDT and the Network-OERJSPDT.

The last two columns in Table 15 reflect the comparisons in terms of the makespan and computational times of the Network-RJSPDT and the Network-FE-RJSPDT (i.e., (*Network-RJSPDT-Network-FE-RJSPDT*)/*Network-FE-RJSPDT*). It is shown that by allowing flexible entrace, the makespan (metric: minutes) can be slighted reduced by an average of 2.21% (i.e., Figure 18). This is reasonable as a higher level of system flexibility is guaranteed in the flexible entrace strategy. However, the reduction in makespan is at a great cost of the computational times. Taking Instance 10 as an example, the makespan is not improved by the Network-FE-RJSPDT, while the computational time dramatically increases by 92.2% compared with the Network-RJSPDT. From the last column of Table 15 and Figure 19, we can see that the Network-RJSPDT averagely consumes 60% less computational time than the Network-FE-RJSPDT due to the much smaller solution spaces. Actually, through our analysis, it is found that when the NJ exceeds 8, the Network-FE-RJSPDT becomes unsolvable within the given time limit. Accordingly, it is concluded that the flexible entrance strategy can only bring slight makespan reduction with a great sacrifice in computational efforts. As a result, it is reasonable to apply the fixed entrance strategy in our proposed models.

8. Conclusion

Due to the wide application of robots for material transportation in modern logistics and supply chain systems, the job-scheduling problem (JSP) in robot-driven production lines is becoming increasingly important. The traditional JSP studies majorly suffer from two assumptions: (i) the material transportation time between two machines is negligible or could be incorporated into the processing time of an operation, and (ii) machines have infinite buffers. These assumptions make the robotic job-scheduling problem (RJSP) impractical as the robot movement procedures as well as the capacity and availability of robots and machines should be considered in the scheduling process. Besides, the deadlock dilemma caused by the limited buffer of machines and robots, together with the corresponding machine blocking strategy are important considerations for the RJSP, but have received little attention

in the literature. In this work, we propose two novel robotic job-shop scheduling models with deadlock and robot movement considerations (RJSPDT), namely the Position-RJSPDT and the Network-RJSPDT, in order to fulfil the important research gaps. To be specific, the proposed Network-RJSPDT is the first JSP model applying the network-based modelling approach which is widely used in aviation scheduling problems with the considerations of deadlock and robot movement. Besides, a set of novel tight deadlock-avoidance constraints is proposed to deal with the deadlock dilemma. Through numerical examples and computational experiments, several important insights can be derived. First of all, the proposed models are shown to completely avoid the conflicts in the production line through the tight deadlock-avoidance constraints. Second, the Network-RJSPDT demonstrates higher solution efficiency over the Position-RJSPDT due to the smaller model size. In small-scale problems, the Network-RJSPDT even realizes an average reduction of 96% in the computational time. Besides, it is identified that the number of jobs greatly affects the performances of the two models, while the fixed entrance strategy can help reduce the computational time by 60% averagely.

Managerial Implications & Future Studies

The analyses derived from this study demonstrate the differences between the traditional JSP and the robotic JSP in robot-driven production lines which are becoming increasingly important nowadays. Therefore, in order to enhance productivity and profitability, the practitioners in the robot-driven logistics and supply chain systems should carefully consider the schedules of operations in jobs and the movement procedures of the robot at the same time, in order to avoid the potential deadlock which significantly impairs the system efficiency. The industry will be benefited from this study by applying the proposed novel robotic job-shop scheduling models with deadlock and robot movement considerations. By utilizing our proposed models, industrialists can make production schedules with fewer system failures and better adjust the production line according to the dynamically changing business environment.

For future research, it will be interesting to integrate robot movement with other JSP variants, like the flexible JSP. Besides, in our current study, we explore a single-robot problem. Therefore, a promising future research direction is to consider a multi-robot production line. Moreover, machines are operated without buffer in our current model. Different buffer rules can thus be evaluated in future studies. Furthermore, it is valuable to study the application of other modelling ideas (like the sequence-based approach) on the considered problem. Last but not the least, intelligent algorithms which can shorten the computational time can be developed for practical application of the proposed models.

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. *Computers & Operations Research* 113**,** 104793.

Arroyo, J.E.C., Armentano, V.A., 2005. Genetic local search for multi-objective flowshop scheduling problems. *European Journal of Operational Research* 167 (3)**,** 717-738.

Basdere, M., Bilge, Ü., 2014. Operational aircraft maintenance routing problem with remaining time consideration. *European Journal of Operational Research* 235 (1), 315-328.

Baker, B.M., Ayechew, M.A., 2003. A genetic algorithm for the vehicle routing problem. *Computers & Operations Research* 30(5), 787-800.

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. *Computers & Operations Research* 103**,** 46-63.

Bowman, E.H., 1959. The schedule-sequencing problem. *Operations Research* 7 (5)**,** 621-624.

Brucker, P., Burke, E.K., Groenemeyer, S., 2012a. A mixed integer programming model for the cyclic job-shop problem with transportation. *Discrete Applied Mathematics* 160 (13-14)**,** 1924-1935.

Brucker, P., Burke, E.K., Groenemeyer, S., 2012b. A branch and bound algorithm for the cyclic job-shop problem with transportation. *Computers & Operations Research* 39 (12)**,** 3200-3214.

Caumond, A., Lacomme, P., Moukrim, A., Tchernev, N., 2009. An MILP for scheduling problems in an FMS with one vehicle. *European Journal of Operational Research* 199 (3)**,** 706-722.

Choi, T.-M., Wen, X., Sun, X., Chung, S.-H., 2019. The mean-variance approach for global supply chain risk analysis with air logistics in the blockchain technology era. *Transportation Research Part E: Logistics and Transportation Review* 127, 178-191.

Choi, T.-M., Yeung. W., Cheng, T.C.E., Yue, X., 2018. Optimal scheduling, coordination, and the value of RFID technology in garment manufacturing supply chains. *IEEE Transactions on Engineering Management* 65(1), 72-84.

Chung, S.-H., Ma, H.-L., Hansen, M., Choi, T.-M., 2020. Data science and analytics in aviation. *Transportation Research Part E: Logistics and Transportation Review 134*, 101837.

Crama, Y., Kats, V., Van de klundert, J., Levner, E., 2000. Cyclic scheduling in robotic flowshops. *Annals of Operations Research* 96 (1-4)**,** 97-124.

Dai, M., Tang, D., Giret, A., Salido, M.A., 2019. Multi-objective optimization for energy-efficient flexible job shop scheduling problem with transportation constraints. *Robotics and Computer-Integrated Manufacturing* 59**,** 143-157.

Dawande, M., Geismar, H.N., Sethi, S.P., Sriskandarajah, C., 2005. Sequencing and scheduling in robotic cells: Recent developments. *Journal of Scheduling* 8 (5)**,** 387-426.

Demir, Y., İsleyen, S.K., 2013. Evaluation of mathematical models for flexible job-shop scheduling problems. *Applied Mathematical Modelling* 37 (3)**,** 977-988.

Deroussi, L., Gourgand, M., Tchernev, N., 2008. A simple metaheuristic approach to the simultaneous scheduling of machines and automated guided vehicles. *International Journal of Production Research* 46 (8)**,** 2143-2164.

Drobouchevitch, I.G., Geismar, H.N., Sriskandarajah, C., 2010. Throughput optimization in robotic

cells with input and output machine buffers: A comparative study of two key models. *European Journal of Operational Research* 206 (3), 623-633.

Egbelu, P.J., Tanchoco, J.M., 1984. Characterization of automatic guided vehicle dispatching rules. *International Journal of Production Research* 22(3), 359-374.

Gharehgozli, A., Zaerpour, N., 2020. Robot scheduling for pod retrieval in a robotic mobile fulfillment system. *Transportation Research Part E: Logistics and Transportation Review 142*, 102087.

Ham, A., 2020. Transfer-robot task scheduling in job shop. *International Journal of Production Research* 1-11.

Hasan, S.K., Sarker, R., Essam, D., 2011. Genetic algorithm for job-shop scheduling with machine unavailability and breakdowns. *International Journal of Production Research* 49 (16), 4999-5015.

Hsu, C.H., Yang, H.C., 2016. Real-time near-optimal scheduling with rolling horizon for automatic manufacturing cell. *IEEE Access* 5, 3369-3375.

Hurink, J., Knust, S., 2001. Makespan minimization for flow-shop problems with transportation times and a single robot. *Discrete Applied Mathematics* 112(1-3), 199-216.

Hurink, J., Knust, S., 2005. Tabu search algorithms for job-shop problems with a single transport robot. *European Journal of Operational Research* 162(1), 99-111.

International Federation of Robotics, 2019. Executive summary world robotics 2019 industrial robots. Retrieved from https://ifr.org/downloads/press2018/Executive%20Summary%20WR%202019%20Industrial%20Robots.pdf

Karimi, S., Ardalan, Z., Naderi, B., Mohammadi, M. 2017., Scheduling flexible job-shops with transportation times: Mathematical models and a hybrid imperialist competitive algorithm. *Applied mathematical modelling* 41, 667-682.

Khan, W.A., Chung, S.H., Awan, M.U., Wen, X., 2019a. Machine learning facilitated business intelligence (Part I): Neural networks learning algorithms and applications. *Industrial Management & Data Systems* 120 (1) 164-195.

Khan, W.A., Chung, S.H., Awan, M.U., Wen, X., 2019b. Machine learning facilitated business intelligence (Part II): Neural networks optimization techniques and applications. *Industrial Management & Data Systems* 120 (1) 128-163.

Khan, W.A., Chung, S.H., Ma, H.-L., Liu, S.Q., Chan, C.Y., 2019c. A novel self-organizing constructive neural network for estimating aircraft trip fuel consumption. *Transportation Research Part E: Logistics and Transportation Review* 132, 72-96.

Koulamas, C., Panwalkar, S., 2019. The two-stage no-wait/blocking proportionate super shop scheduling problem. *International Journal of Production Research* 57 (10), 2956-2965.

Li, X., Zhao, Y., Yang, X., Dong, Y., 2019. A metaheuristic to solve a robotic cell job-shop scheduling problem with time window constraints. Proceedings of the 2019 4th *International Conference*

on Mathematics and Artificial Intelligence 128-132.

Liang, Z., Chaovalitwongse, W.A., Huang, H.C., Johnson, E.L., 2011. On a new rotation tour network model for aircraft maintenance routing problem. *Transportation Science* 45 (1)**,** 109-120.

Liu, S.Q., Kozan, E., 2017. A hybrid metaheuristic algorithm to optimise a real-world robotic cell. *Computers & Operations Research* 84**,** 188-194.

Liu, S.Q., Kozan, E., Masoud, M., Zhang, Y., Chan, F.T., 2018. Job shop scheduling with a combination of four buffering constraints. *International Journal of Production Research* 56 (9)**,** 3274-3293.

Ma, H., Chung, S.H., Chan, H.K., Cui, L., 2019. An integrated model for berth and yard planning in container terminals with multi-continuous berth layout. *Annals of Operations Research* 273(1-2), 409-431.

Manne, A.S., 1960. On the job-shop scheduling problem. *Operations Research* 8 (2)**,** 219-223.

Maoudj, A., Bouzouia, B., Hentout, A., Kouider, A., Toumi, R., 2019. Distributed multi-agent scheduling and control system for robotic flexible assembly cells. *Journal of Intelligent Manufacturing* 30 (4)**,** 1629-1644.

Mascis, A., Paccoarelli, D., 2002. Job-shop scheduling with blocking and no-wait constraints. *European Journal of Operational Research* 143 (3)**,** 498-517.

Mati, Y., Lahlou, C., Dauzereperes, S., 2011. Modelling and solving a practical flexible job-shop scheduling problem with blocking constraints. *International Journal of Production Research* 49 (8)**,** 2169-2182.

Meng, L., Zhang, C., Shao, X., Zhang, B., Ren, Y., Lin, W., 2020. More MILP models for hybrid flow shop scheduling problem and its extended problems. *International Journal of Production Research* 58 (13)**,** 3905-3930.

Naderi, B., Azab, A., 2014. Modeling and heuristics for scheduling of distributed job shops. *Expert Systems with Applications* 41(17), 7754-7763.

Ng, K.K.H., Lee, C.K., Zhang, S.Z., Wu, K., Ho, W., 2017. A multiple colonies artificial bee colony algorithm for a capacitated vehicle routing problem and re-routing strategies under time-dependent traffic congestion. *Computers & Industrial Engineering* 109, 151-168.

Nouri, H.E., Driss, O.B., Ghédira, K., 2016. Hybrid metaheuristics for scheduling of machines and transport robots in job shop environment. *Applied Intelligence* 45(3), 808-828.

Özgüven, C., Özbakir, L., Yavuz, Y., 2010. Mathematical models for job-shop scheduling problems with routing and process plan flexibility. *Applied Mathematical Modelling* 34 (6), 1539-1548.

Özgüven, C., Yavuz, Y., Özbakir, L., 2012. Mixed integer goal programming models for the flexible job-shop scheduling problems with separable and non-separable sequence dependent setup times. *Applied Mathematical Modelling* 36 (2)**,** 846-858.

Pillac, V., Gendreau, M., Guéret, C., Medaglia, A.L., 2013. A review of dynamic vehicle routing problems. *European Journal of Operational Research* 225(1), 1-11.

Pisinger, D., Ropke, S., 2007. A general heuristic for vehicle routing problems. *Computers & Operations Research* 34(8), 2403-2435.

Qin, Y., Chan, F.T., Chung, S.H., Qu, T., & Niu, B., 2018. Aircraft parking stand allocation problem with safety consideration for independent hangar maintenance service providers. *Computers & Operations Research* 91, 225-236.

Qin, Y., Ma, H.L., Chan, F.T.S., Khan, W.A., 2020a. A scenario-based stochastic programming approach for aircraft expendable and rotable spare parts planning in MRO provider. *Industrial Management & Data Systems* 120 (9), 1635-1657.

Qin, Y., Zhang, J.H., Chan, F.T.S., Chung, S.H., Niu, B, Qu, T., 2020b. A two-stage optimization approach for aircraft hangar maintenance planning and staff assignment problems under MRO outsourcing mode. *Computers & Industrial Engineering* 146, 106607.

Quinton, F., Hamaz, I., Houssin, L., 2020. A mixed integer linear programming modelling for the flexible cyclic jobshop problem. *Annals of Operations Research* 285 (1)**,** 335-352.

Rahman, H.F., Nielsen, I., 2019. Scheduling automated transport vehicles for material distribution systems. *Applied Soft Computing* 82**,** 105552.

Ren, S., Choi, T.-M., Lee, K.-M., Lin, L., 2020. Intelligent service capacity allocation for cross-border-E-commerce related third-party-forwarding logistics operations: A deep learning approach. *Transportation Research Part E: Logistics and Transportation Review* 134, 101834.

Roshanaei, V., Azab, A., Eimaraghy, H., 2013. Mathematical modelling and a meta-heuristic for flexible job shop scheduling. *International Journal of Production Research* 51(20), 6247-6274.

Roy, D., Nigam, S., de Koster, R., Adan, I., Resing, J., 2019. Robot-storage zone assignment strategies in mobile fulfillment systems. *Transportation Research Part E: Logistics and Transportation Review* 122, 119-142.

Shahgholi zadeh, M., Katebi, Y., Doniavi, A., 2019. A heuristic model for dynamic flexible job shop scheduling problem considering variable processing times. *International Journal of Production Research* 57 (10)**,** 3020-3035.

Simoni, M.D., Kutanoglu, E., Claudel, C.G., 2020. Optimization and analysis of a robot-assisted last mile delivery system. *Transportation Research Part E: Logistics and Transportation Review 142*, 102049.

Soukhal, A., Martineau, P., 2005. Resolution of a scheduling problem in a flowshop robotic cell. *European Journal of Operational Research* 161 (1)**,** 62-72.

Sun, X., Chung, S.-H., Choi, T.-M., Sheu, J.-B., Ma, H.L., 2020a. Combating lead-time uncertainty in global supply chain's shipment-assignment: Is it wise to be risk-averse? *Transportation Research Part B: Methodological* 138, 406-434.

Sun, X., Chung, S.H., Ma, H.L., 2020b. Operational risk in airline crew scheduling: Do features of

flight delays matter? *Decision Sciences* published online.

Sun, X.T., Chung, S.H., Chan, F.T.S., Wang, Z., 2018. The impact of liner shipping unreliability on the production–distribution scheduling of a decentralized manufacturing system. *Transportation Research Part E: Logistics and Transportation Review* 114, 242-269.

Vital-soto, A., Azab, A., Baki, M.F., 2020. Mathematical modelling and a hybridized bacterial foraging optimization algorithm for the flexible job-shop scheduling problem with sequencing flexibility. *Journal of Manufacturing Systems* 54**,** 74-93.

Wagner, H.M., 1959. An integer linear‐programming model for machine scheduling. *Naval Research Logistics Quarterly* 6 (2)**,** 131-140.

Wang, S., Wu, R., Chu, F., Yu, J., Liu, X., 2020a. An improved formulation and efficient heuristics for the discrete parallel-machine makespan ScheLoc problem. *Computers & Industrial Engineering* 140, 106238.

Wang, Y., Li, X., Ruiz, R., Sui, S., 2017. An iterated greedy heuristic for mixed no-wait flowshop problems. *IEEE Transactions on Cybernetics* 48 (5)**,** 1553-1566.

Wang, Z., Khan, W.A., Ma, H.-L., Wen, X., 2020b. Cascade neural network algorithm with analytical connection weights determination for modelling operations and energy applications. *International Journal of Production Research* published online.

Wen, X., Ma, H.L., Chung, S.H., Khan W.A. 2020. Robust airline crew scheduling with flight flying time variability. *Transportation Research Part E: Logistics and Transportation Review* 144, 102132.

Xi, X., Changchun, L., Yuan, W., Loo Hay, L., 2020. Two-stage conflict robust optimization models for cross-dock truck scheduling problem under uncertainty. *Transportation Research Part E: Logistics and Transportation Review* 144, 102123.

Yan, P., Liu, S.Q., Sun, T., Ma, K., 2018. A dynamic scheduling approach for optimizing the material handling operations in a robotic cell. *Computers & Operations Research* 99**,** 166-177.

Zhang, R., Chiong, R. 2016. Solving the energy-efficient job shop scheduling problem: A multi-objective genetic algorithm with enhanced local search for minimizing the total weighted tardiness and total energy consumption. *Journal of Cleaner Production* 112**,** 3361-3375.

Zheng, Y., Xiao, Y., Seo, Y., 2014. A tabu search algorithm for simultaneous machine/AGV scheduling problem. *International Journal of Production Research* 52 (19)**,** 5748-5763.

Zhou, C., Lee, B.K., Li, H., 2020. Integrated optimization on yard crane scheduling and vehicle positioning at container yards. *Transportation Research Part E: Logistics and Transportation Review* 138, 101966.

Zhou, Z., Che, A., Yan, P., 2012. A mixed integer programming approach for multi-cyclic robotic flowshop scheduling with time window constraints. *Applied Mathematical Modelling* 36 (8), 3621-3629.