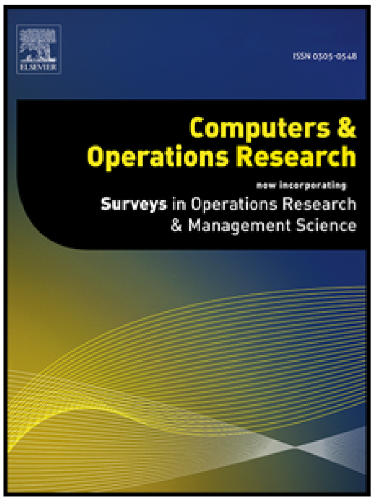

MILP models to minimise makespan in additive manufacturing machine scheduling problems

Ibrahim Kucukkoc

PII: S0305-0548(19)30015-2
DOI: https://doi.org/10.1016/j.cor.2019.01.006
Reference: CAOR 4628

To appear in: *Computers and Operations Research*

Received date: 23 January 2018
Revised date: 14 January 2019
Accepted date: 15 January 2019

Please cite this article as: Ibrahim Kucukkoc , MILP models to minimise makespan in additive manufacturing machine scheduling problems, *Computers and Operations Research* (2019), doi: https://doi.org/10.1016/j.cor.2019.01.006

Highlights

- The scheduling problem of multiple additive manufacturing machines is addressed.
- MILP models are developed to minimise the makespan on different machine types.
- Single, parallel identical and parallel non-identical machines are scheduled.
- Computational tests indicate the necessity of improved techniques.
- The models proposed can easily be adopted by additive manufacturing firms.

MILP models to minimise makespan in additive manufacturing machine scheduling problems

Ibrahim Kucukkoc

Department of Industrial Engineering, Balikesir University, Balikesir, Turkey

Email: ikucukkoc@balikesir.edu.tr , Tel: +902666121194 (Ext. 6402)

Abstract

Additive manufacturing (AM), also known as 3D printing, is gaining enormous importance in the production of highly customised quality and lightweight products in low quantities. In addition to AM's use in producing fully functional industrial components, it is also seen as a technology of the future that will enable civilisation in space. Although the cost structures for AM facilities have been sufficiently studied in the literature, no effort has been made to investigate the scheduling problem of AM machines with the aim of optimising processing time-related performance measures. This paper focuses on the scheduling problem of single and multiple AM machines and proposes mathematical models for optimisation. Mixed-integer linear programming models allocate parts into jobs to be produced on AM machines to minimise makespan. The problem was handled by considering different machine configurations (i.e. single machine, parallel identical machines, and parallel non-identical machines). The models were coded in IBM ILOG CPLEX Optimization Studio (v12.8.0) and solved through the CPLEX solver. This paper presents detailed solutions for numerical examples. A comprehensive computational study was also conducted, and the results are presented. The optimum solutions are reported for most problems. The best solutions obtained within the time limit (i.e. 1,800 and 2,400 seconds) are reported for the parallel identical and non-identical AM machine scheduling problems if optimum solution could not be verified.

Keywords: additive manufacturing, 3D printing, scheduling, mathematical modelling, MILP

1. Introduction

AM is a common domain for the manufacturing of parts layer by layer. AM is also called 3D printing, as the objects are made from 3D-model data through joining the materials, as opposed to subtractive traditional techniques [1]. Various AM technologies have been developed since the 1980s, and selective laser melting, laser engineered shaping, and electron beam melting have become the most famous techniques for rapid manufacturing. AM processes provide many significant advantages, such as design flexibility, high accuracy, resource efficiency, and material efficiency, over traditional techniques [2]. Among these, design flexibility is probably the most important, especially when AM technology has shifted from prototyping to direct part production (also known as direct digital manufacturing) with the rapid development of material science and manufacturing technologies.

Many companies in various industries, such as aeronautics, automotive, healthcare, defence, etc., can now digitally manufacture their end-use metallic parts from powder materials, leading to a new industrial revolution. The total AM market is predicted to rise nearly six-fold to $12 billion in 2025, driven by growth in various industries including prototypes, moulds and aerospace [3]. NASA has reported that 70 additively manufactured parts were used for the Mars Rover test vehicles. The fuel nozzles, each of which requires an assembly of 20 different parts, were directly manufactured by General Electric. NASA created the first ever *made-in-space* parts (including a wrench) using AM in 2014. Thus, along with AM's use in the automotive and aerospace industries for years (Boeing printed 22 thousand parts used on aircrafts by 2012), AM technology now helps astronauts build their own parts and tools when required. This is a major advancement in the aerospace industry, as it could take months or even years to get equipment to space, depending on the launch resupply schedule. Moreover, NASA additively manufactured a turbopump, used as a rocket engine fuel pump (made with hundreds of parts including a turbine that spins at over 90,000 rpm) [4]. It was reported that, in comparison with pumps made with traditional manufacturing, the additively manufactured pump has 45 per cent fewer parts.

AM manufacturing technologies have been extensively studied in the literature. However, the majority of these studies have focused on the process itself. For example, Cooper et al. [5] addressed the development of additively manufactured hydraulic components for Formula 1 racing cars to reduce weight and improve efficiency. Khajavi et al. [6] investigated the potential impact of AM technology on the configuration of spare parts supply chains. The study provided guidance for the deployment of AM machines in spare parts supply chains.

The AM process considered in this research was selective laser melting (SLM). Figure 1 presents the concept for the AM machine scheduling problem. Due to the high purchasing cost of SLM machines, it might be more practical for companies to outsource this service in terms of cost efficiency. Thus, the orders received from distributed customers are regrouped and allocated to machines as different sets of job batches. In this regrouping process, it is important to ensure that the machine's physical capacity (i.e. height and area) is sufficient to produce the allocated parts. This is a basic constraint that must be satisfied in terms of practicality. However, AM machines have various speed and cost characteristics that must be considered to get the parts produced with time and cost efficiency. The problem is how to regroup parts and allocate them into jobs to optimise performance measures such as total cost or makespan.

As seen in Figure 1, an AM machine can produce more than one part simultaneously on its platform, subject to its capacity and the production area needed by parts. A job in an AM machine scheduling problem is defined as the set of parts to be produced in the same batch. Therefore, any part produced in a job cannot be removed until the whole job (the production of all parts in the same job) is completed. To start a new job on an AM machine, a series of operations are performed to setup the machine, such as data preparation, powder material filling, machine adjustment, and protective atmosphere filling. Thus, the set-up time is shared by all parts allocated to the same job. In fact, in addition to the high purchasing cost of AM machines, one of the main limits to the wider application of the SLM process is the high cost caused by the layer-by-layer production due to the nature of the SLM process itself. If decentralised orders received from dispersed customers are grouped in such a way, the utilisation of AM facilities is increased. Hence, the common costs (e.g. setup and post processing operations) are shared by all parts and the average cost is decreased.

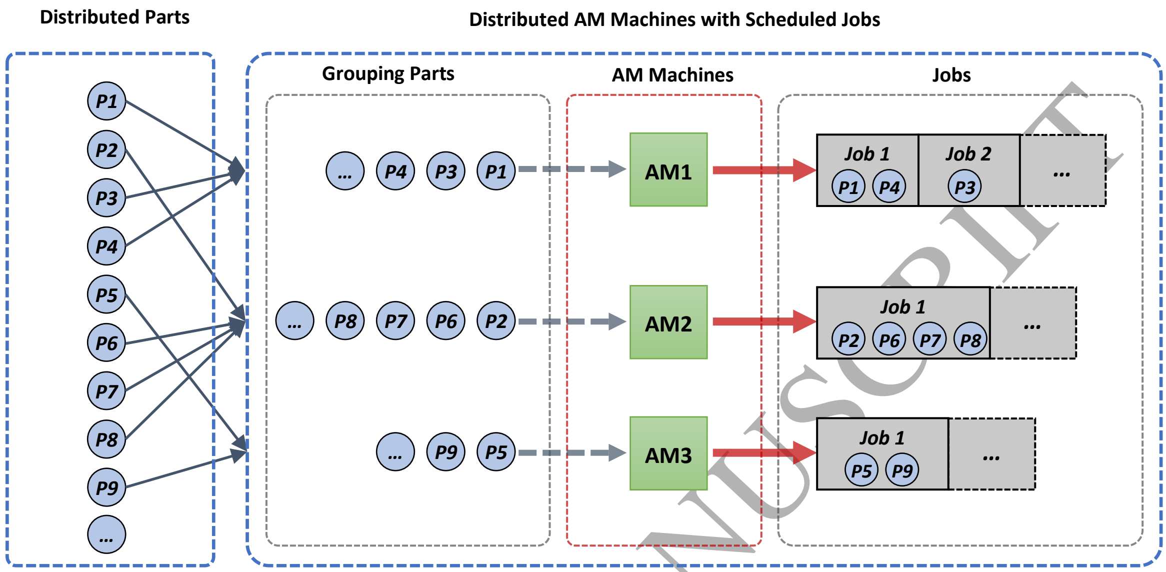

Figure 1. Concept model for AM production scheduling

In the sense that the parts are scheduled in batches on the machines, the problem may look similar to the well-known batch scheduling problem with arbitrary job sizes [7,8] in the literature. Note that the terms *part* and *job* are used within the AM machine scheduling concept, which correspond to *job* and *batch* used in the batch scheduling problem, respectively. In its simplest form, only one batch machine is considered and the single batch machine scheduling problem [9] arises when there are batches and setup times required between these batches [10]. The aim is to allocate jobs (referred to as *parts* in the AM machine scheduling problem) into batches. If there is a single batch machine to which a set of jobs (with the same processing times) needs to be allocated in batches, the makespan problem is equivalent to a well-known bin packing problem [11]. The bin packing problem is strongly NP-hard based on Garey and Jackson [12], so minimizing the makespan in the single batch machine scheduling problem with unequal processing times (or arbitrary job sizes) is strongly NP-hard [11]. Therefore, our single AM machine scheduling problem is also strongly NP-hard. Considering that the processing time of a job is a function (as will be explained in Section 2), it is more complex than the single batch machine scheduling problem. When there are parallel batch machines ($Pm|batch|C_{max}$), the problem of minimizing makespan (which is already strongly NP-hard in a single batch machine environment), becomes even harder to solve [13]. Hence, the parallel identical and parallel non-identical AM machine scheduling problems studied in this paper are also strongly NP-hard as contextualised above.

The single machine batch scheduling problem has been extensively studied in the literature. To cite a few, Cheng et al. [14] aimed to minimise the total flow time of items produced on a batch machine. They assumed the flow time of an item to be the completion time of the batch that contains it. Ghosh and Gupta [15] addressed the problem with the aim of minimising maximum lateness. Relatively recent studies have addressed the parallel batch machine scheduling problems with identical and non-identical machines and proposed heuristic/metaheuristic techniques due to the complexity of the problem (see, for example, Jia and Leung [16], and Zhou et al. [17] for algorithms for solving parallel identical batch machine scheduling problems; and Li e al. [8], and Shahidi-Zadeh et al. [18] for algorithms for parallel non-identical batch machine scheduling problems). Trindade et al. [19] discussed four different versions of batch scheduling problems, considering a single processing machine or parallel processing machines and considering jobs with or without release times. Li and Zhang [20] provided more details on exact and heuristic methods to minimise makespan in the single batch machine scheduling problem, and Mendez et al. [21] made a state-of-the-art review of optimisation methods for the short-term scheduling of batch processes. Potts and Kovalyov [22] presented a comprehensive survey on batch scheduling problems and Webster and Baker [23] presented an overview of algorithms and complexity results for scheduling batch processing machines.

While there are similarities between the batch scheduling problems and AM machine scheduling problems studied in this paper, the AM machine scheduling problem differs from the batch scheduling problem in several ways (regardless of considering single or parallel machines) [1]. First of all, the production time of the parts in the AM machine scheduling problem is not known in advance, because the processing time of a job on an AM machine is resource dependent and dynamically characterised by the total volume of the parts included in the job, as well as by the maximum heights of those parts. On the other hand, in the batch scheduling problems, the processing time of a batch is determined by the largest processing time of the jobs included in the batch. Therefore, in the AM machine scheduling problem, allocating parts with similar heights to the same job may help reduce the makespan. Also, as the parts produced in AM machines involve complex structures (not conveniently produced through traditional manufacturing processes), the processing times are dramatically long. Secondly, as the processing time of a job is calculated via a function, different sets or combinations of parts will lead to different costs in the AM machine scheduling problem [1]. Thirdly, in the AM machine scheduling problem, there is a trade-off between machine utilisation and job completion time due to the unique nature of this technology. While it is necessary to increase the utilisation of the AM machine to minimise the number of batches and reduce setup times, this causes an increase in the production time of the job batch. For this aim, the AM machine scheduling problem may include a very complex 2D nesting problem (or even a 3D packing problem) to further increase the utilisation of the machines. However, including such a complex problem into the strongly NP-hard AM machine scheduling problem may transform it to an unsolvable problem.

As seen from the summary given above, different combinations of parts in batches affect the feasibility of the solution and yield different performance measures in terms of completion time. Additionally, the production area capacity constraint needs to be satisfied to obtain a feasible solution. These issues make the problem even more complex to solve in comparison to the traditional single-batch machine scheduling problem. Therefore, sophisticated techniques are required.

The research on the planning and scheduling of production in AM, i.e. the decision for the allocation of parts to job batches on the AM machines, is limited. Kucukkoc et al. [2] introduced the AM machine scheduling problem to maximise the utilisation of AM machines in terms of the production areas used. They developed a mathematical model to maximise resource utilisation considering the delivery times of parts. However, no experiments were conducted, to either run the model or to test its performance. Li et al. [1] introduced the problem of planning AM machines. A mixed-integer linear programming (MILP) model was proposed and solved via CPLEX and they newly developed two heuristics, best-fit and adapted best-fit heuristics, to minimise the average production cost per material volume. Their computational study resulted in two important outcomes: (i) planning AM machines helps reduce processing costs considerably and (ii) the algorithms have promising solution capacity. Kucukkoc et al. [24] proposed a genetic algorithm approach to minimise maximum lateness in the multiple machine environment. However, as seen from this survey, no research has been conducted to address the production scheduling of AM machines to optimise a processing time-related performance measure, such as makespan or flow time. Chergui et al. [25] addressed the production scheduling and nesting problem in additive manufacturing and proposed a heuristic approach. The parallel identical AM machines were considered to minimise the maximum lateness. Fera et al. [26] presented a cost-based model for the scheduling problem of a single AM machine to minimise the weighted total of earliness and tardiness costs. Dvorak et al. [27] studied the AM machine scheduling problem with part due dates to minimise the number of tardy parts (referred to as tardy builds in the corresponding paper) and addressed to the main challenges of the problem. Fera et al. [28] addressed the single AM machine scheduling problem to minimise time and cost. Zhang et al. [29] analysed the multi-parts placement problem in AM and proposed a two-step strategy. The problem is a special case of well-known nesting problem, referred to as NP-hard [29]. Zhang et al. [30] focussed on the optimisation process of build orientation for multi-part production in additive manufacturing and proposed a two-stage approach as a solution method. There also are some conceptual works which aim at addressing both the build orientation and 2D packing and scheduling problem, see for example Oh et al. [31].

In contrast to the studies summarised above, this paper is original and contributes to the literature by introducing and mathematically modelling the scheduling problem of single and multiple AM machines to optimise a processing time-related performance measure with various machine type considerations. First, this study is the first in the literature to define the problem as a single AM machine scheduling problem to optimise the makespan, represented as C_{max}. To be consistent with the scheduling literature, this problem is denoted as $1|batch\{AM\}|C_{max}$. Second, the problem is extended to a parallel AM machine scheduling problem with identical machines (denoted with $Pm|batch\{AM\}|C_{max}$), which is also the first time this has been done. Thus, more than one machine with the same specifications (i.e. speed, set-up time, height, and area capacity) can be scheduled simultaneously. Furthermore, non-identical machines are considered and the problem of parallel non-identical AM machine scheduling problem ($Rm|batch\{AM\}|C_{max}$) is defined for the first time. In this version, the machines may have different specifications, as given above. Each problem type was modelled mathematically as a MILP model and the numerical examples were solved using the models coded in IBM ILOG CPLEX Optimization Studio (v12.8.0).

The paper is organised as follows. Section 2, Section 3, and Section 4 present the MILP models developed for single, parallel identical, and parallel non-identical machine scheduling problems, respectively. Numerical examples and their optimum solutions are also provided in the corresponding sections. A comprehensive computational study was conducted and the results are reported for each problem type in Section 5. The paper concludes in Section 6, together with some insights for future research directions.

2. Single machine environment ($1|batch\{AM\}|C_{max}$)

The problem of $1|batch\{AM\}|C_{max}$ consists of one AM machine and a total of i_n parts ($i \in I$ and $i = 1,2,\ldots,i_n$) to be allocated to a total of j_n jobs. Note that the value of j_n may be lower than or equal to i_n ($j_n \leq i_n$), as there can be at most a total of i_n jobs in case each part is allocated to a job individually. The parts may have different specifications, i.e. height (h_i), area (a_i), and volume (v_i), which form the basic parameters of the model. The other resource related parameters of the problem are as follows:

- VT : Time spent to form per unit volume of material
- HT : Time spent for powder-layering, which is repeated for each layer based on the highest part produced in the job
- SET : Set-up time needed for initialising and cleaning
- MA : The production area of the machine's tray

The shapes of the parts are not considered in the model as the major aim of this paper is to focus on the scheduling problem of AM machines. Instead, it is ensured that the total area of the parts assigned to a job does not exceed MA. This is because considering a two-dimensional placement of shapes on a building platform (or a tray) may make the problem, which is already NP-hard in a strong sense (as explained in the previous section), very complicated and hard to solve. There are some papers focusing only the problem of determining building orientations or placements of multi-parts on a tray [29], rather than scheduling the AM machines. Another reason contributing to this outcome is that a_i is calculated considering the rectangular shapes of the projection area of parts on the building platform. Thus, some security tolerances exist between a_i and real or exact shapes of the parts to be scheduled, which does not cause infeasibility when allocating exact shapes on the tray. It is also worthy to note that the parts are usually designed with a pre-determined build orientation to maximise the quality. So, it is not practical to change the building orientation of the parts to gain more space on the tray.

It is assumed that the heights (areas) of all parts are smaller than or equal to the maximum height (area) supported by the machine. Different from many conventional batch scheduling problems, the production time of a job increases depending on the content of the job, i.e. the total volume of the allocated parts and especially the maximum height included in the job. The job processing time is a function, such that it is calculated using Equation (1).

$$PT_{j\in J} = SET \cdot Z_j + VT \cdot \sum_{i\in I} v_i \cdot X_{ji} + HT \cdot \max_{i\in I}\{h_i \cdot X_{ji}\}, \tag{1}$$

where X_{ji} is a binary variable that equals 1 if part i is assigned to job j. Similarly, Z_j is a binary variable that equals 1 if $\sum_{i=1}^{i_n} X_{ji} \geq 1$ for $\forall j \in J$. That means Z_j is equal to 1 if there is at least one part assigned to job j (if job j is utilised, in other words). Note that the setup time is included in the job processing time, as shown in Equation (1). As seen in the equation, PT_j dynamically changes with the allocation of parts to jobs in different ways. Thus, with some minor changes in their allocations to jobs, it is possible to get the same parts produced with different durations.

The jobs on the same machine are processed in the order of $1,2,\ldots,j_n$ as they are utilised sequentially and only one job can be processed at a time. Hence, the completion time of job j, represented by C_j, is determined using PT_j as in Equation (2).

$$C_j \geq C_{j-1} + PT_j \qquad \forall j \in J. \tag{2}$$

As a basic assumption of the scheduling problems, only one job can be processed on the machine at a time. Therefore, a job can start upon completing the previous job on the same machine. C_0 is assumed to be zero, so that the machine is available at the beginning of the planning period.

The objective is to minimise the makespan, represented by $C_{max} = \max_{j\in J}\{C_j\}$ as formulated in Equation (3).

$$min\ Z = \max_{j\in J}\{C_j\}. \tag{3}$$

The constraints of the problem are part occurrence constraint (Equation 4), area capacity constraint (Equation 5), job utilisation constraint (Equation 6), completion time constraints (Equations 7 and 8), production time calculation constraint (Equation 9), and sign constraints (Equation 10). Note that ψ is a large positive number.

$$\sum_{j\in J} X_{ji} = 1; \quad \forall i \in I. \tag{4}$$

$$\sum_{i \in I} a_i \cdot X_{ji} \leq MA\,; \quad \forall j \in J. \tag{5}$$

$$\sum_{i \in I} X_{(j+1)i} \leq \psi \cdot \sum_{i \in I} X_{ji}\,; \quad \forall j = 1,2, \dots, j_n - 1. \tag{6}$$

$$C_{j-1} + PT_j \leq C_j; \quad \forall j \in J. \tag{7}$$

$$C_0 = 0. \tag{8}$$

$$PT_j = SET \cdot Z_j + VT \cdot \sum_{i \in I} v_i \cdot X_{ji} + HT \cdot \max_{i \in I}\{h_i \cdot X_{ji}\}; \quad \forall j \in J. \tag{9}$$

$$X_{ji}, Z_j \in (0,1); \quad \forall j \in J,\ i \in I. \tag{10}$$

The part occurrence constraint (4) ensures that each part is assigned to exactly one job, while the area capacity constraint (5) ensures the total area of parts assigned to a job does not exceed the machine's capacity. The job utilisation constraint (6) guarantees that jobs are utilised in an incremental order starting from job 1. For example, job 3 would not be utilised if there were no part assigned to job 2. The first completion time constraint (7) ensures that the completion time of each job is greater than or equal to the summation of its start time (i.e. the completion time of the previous job, $j - 1$, on the machine) and production time. The other completion time constraint (8) sets the starting time of the first job to zero (the machine is made available at the beginning). The production time constraint (9) calculates the job production time. Finally, the sign constraints define the binary variables X_{ji} and Z_j. Note that $Z_j = 1$ if $\sum_{i \in I} X_{ji} \geq 1$ for $\forall j \in J$.

Let us assume a small example with 12 jobs and one machine. Table 1 provides the part specifications, and the machine parameters are assumed to be $VT = 0.030864\ hr/cm^3$, $HT = 0.7\ hr/cm$, $SET = 1\ hr$, and $MA = 900\ cm^2$.

Table 1. Details of the 12 parts for the example problem

Part (i)	Height (h_i) - cm	Area (a_i) - cm^2	Volume (v_i) - cm^3
1	6.90	209.06	826.08
2	26.04	550.11	952.60
3	15.97	23.63	71.91
4	17.04	99.53	703.08
5	27.94	56.85	272.92
6	17.38	50.02	125.70
7	11.81	435.66	1142.25
8	2.67	84.97	121.82
9	17.13	48.27	315.00
10	4.27	122.62	102.83
11	2.18	178.34	214.79
12	6.48	134.08	124.66

The optimum solution was obtained by solving the proposed model coded in IBM ILOG CPLEX Optimization Studio (v12.8.0) through the CPLEX solver. The code was run on an Intel® Core™ i7-6700HQ CPU @2.60 GHz with 16 GB RAM. The objective value of 187.92 was obtained in less than a second. Table 2 presents the allocation of parts to jobs. The table shows that all parts are allocated to a total of three jobs ($Z_1 = Z_2 = Z_3 = 1$).

Table 2. The allocation of parts and the details of jobs for the optimum solution

Job (j)	Allocated parts	Total Area ($\sum_{i\in I} a_i \cdot X_{ji}$)	Total Volume ($\sum_{i\in I} v_i \cdot X_{ji}$)	$\max\{h_i\}$	PT_j	C_j
1	10, 11	300.96	317.62	4.27	13.992	13.992
2	1, 7, 8, 12	863.77	2214.81	11.81	77.825	91.817
3	2, 3, 4, 5, 6, 9	828.41	2441.21	27.94	96.104	187.921

To obtain the solution given above, the value of j_n (the upper bound for the number of possible jobs) was set to $j_n = 4$ (one possible job is empty as no parts allocated). The same optimum solution would be obtained when $j_n = 5$ or even $j_n = 6$ (with more empty possible jobs), but the program would consume larger computation times to find the same optimum solution and prove its optimality. On the one hand, some slackness is needed to obtain the optimum solution, which requires a higher j_n value. On the other hand, the preliminary tests showed that setting j_n to a higher value (such as $j_n = i_n$) is quite time consuming for the models proposed. The difference in computation time may be ignored for the small size problems, but it considerably increases when the problem size gets bigger.

Based on the optimum solution presented above, the Gantt chart of the solution can be drawn as in Figure 2. The numbers given in the bars denote part numbers while the length of the bars corresponds to the production time. The parts in the same job start and finish processing at the same time, as it is not possible to interrupt a job or take the completed parts out while a job is continuing. Note that the setup time needed for jobs is included in the production time.

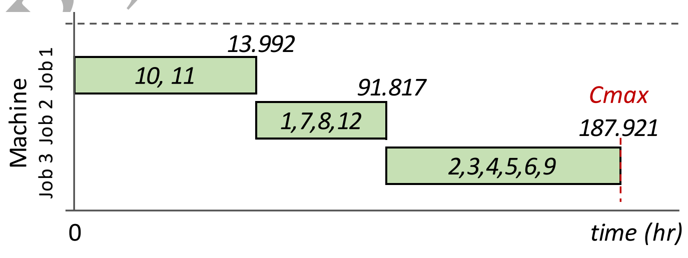

Figure 2. Gantt chart for the optimum solution of the single AM machine scheduling problem

3. Parallel identical machines ($Pm|batch\{AM\}|C_{max}$)

The parallel identical AM machine scheduling problem with the aim of minimising makespan is symbolised with $Pm|batch\{AM\}|C_{max}$. Apparently, more than one AM machine is operating in parallel. As the machine specifications are the same, any part can be processed on any machine, assuming that the machines are tall and large enough to produce any part.

There is a total of i_n parts ($i \in I$ and $i = 1,2,\dots,i_n$) to be grouped in job batches ($j \in J$ and $j = 1,2,\dots,j_n$) on each AM machine ($m \epsilon M$ and $m = 1,2,\dots,m_n$). The binary decision variable X_{mji} is 1 if part i is assigned to job j on machine m; it is zero, otherwise. Another binary decision variable Z_{mj} is 1 if any part is assigned to job j on machine m (if the job is utilised); it is zero, otherwise. The notation in the previous subsection also defines the part specifications, i.e. h_i for height, a_i for area, and v_i for volume. The same notation is also used for the machine specifications, i.e. VT, HT, SET, and MA, to keep consistency. The production time (PT_{mj}) and the completion time (C_{mj}) for job j on machine m are calculated with Equations (11) and (12), respectively.

$$PT_{mj} = SET \cdot Z_{mj} + VT \cdot \sum_{i \in I} v_i \cdot X_{mji} + HT \cdot \max_{i \in I}\{h_i \cdot X_{mji}\}; \quad \forall m \in M, j \in J, \tag{11}$$

$$C_{mj} \geq C_{m(j-1)} + PT_{mj}; \quad \forall m \in M, j \in J \tag{12}$$

With the addition of the machine index, m, the production and completion times of jobs become machine specific. So, a new job on an AM machine can start after the completion of the last job on the same machine, $C_{mj} \geq C_{m(j-1)} + PT_{mj}$ for $\forall m \in M, j \in J$. C_{m0} is set to zero, so all machines are available at the beginning of the planning period.

The objective is to minimise the makespan, see Equation (13).

$$min\ Z = \max_{m \in M, j \in J}\{C_{mj}\}. \tag{13}$$

The constraints were slightly modified as follows due to the newly included machine index.

$$\sum_{m \in M} \sum_{j \in J} X_{mji} = 1; \quad \forall i \in I. \tag{14}$$

$$\sum_{i \in I} a_i \cdot X_{mji} \leq MA; \quad \forall m \in M;\ \forall j \in J. \tag{15}$$

$$\sum_{i \in I} X_{m(j+1)i} \leq \psi \cdot \sum_{i \in I} X_{mji}; \quad \forall m \in M;\ \forall j \in J. \tag{16}$$

$$C_{m(j-1)} + PT_{mj} \leq C_{mj}; \quad \forall m \in M;\ \forall j \in J. \tag{17}$$

$$C_{m0} = 0; \quad \forall m \in M. \tag{18}$$

$$PT_{mj} = SET \cdot Z_{mj} + VT \cdot \sum_{i \in I} v_i \cdot X_{mji} + HT \cdot \max_{i \in I}\{h_i \cdot X_{mji}\}; \quad \forall M \in M;\ \forall j \in J. \tag{19}$$

$$X_{mji}, Z_{mj} \in (0,1); \quad \forall M \in M; \forall j \in J; \forall i \in I. \tag{20}$$

Equations (14) and (15) satisfy the part occurrence constraint and area capacity constraint, respectively. The jobs are utilised incrementally (starting from job 1) on each machine with the help of Equation (16). For example, job 2 cannot be utilised if job 1 is empty on the same machine. Equation (17) guarantees that the completion time of a job on a specific machine is greater than or equal to the sum of its production time and the completion time of the previous job on the same machine. Thus, a job can start on a machine after completion of either the previous or an ongoing job on the same machine. The start times of first jobs on each machine are set to zero to ensure that the machines are available at the beginning of the planning period with the help of Equation (18). The production times of jobs are calculated using Equation (19). Equation (20) denotes the sign constraints of binary decision variables where $Z_{mj} = 1$ if $\sum_{i \in I} X_{mji} \geq 1$ ($\forall m \in M$ and $\forall j \in J$).

Let us assume a simple numerical example with twenty parts (for which the details are given in Table 3), assuming two AM machines executing jobs in parallel with slightly different parameters. Due to the nature of the parallel machine scheduling problem, the parameters are the same for both machines (i.e. $VT = 0.030864\ hr/cm^3$, $HT = 0.7\ hr/cm$, $SET = 1\ hr$, and $MA = 900\ cm^2$).

Table 3. Details of the parts

Part (i)	Height (h_i) - cm	Area (a_i) - cm^2	Volume (v_i) - cm^3
1	6.90	209.06	826.08
2	26.04	550.11	952.60
3	15.97	23.63	71.91
4	17.04	99.53	703.08
5	27.94	56.85	272.92
6	36.50	742.97	1583.98
7	17.38	50.02	125.70
8	18.46	300.66	3144.39
9	11.81	435.66	1142.25
10	21.79	131.88	1840.39
11	12.59	349.83	2204.41
12	2.67	84.97	121.82
13	17.13	48.27	315.00
14	12.53	269.66	1786.36
15	18.09	175.77	1885.00
16	4.27	122.62	102.83
17	2.18	178.34	214.79
18	25.10	569.53	2867.59
19	37.25	464.89	2378.05
20	6.48	134.08	124.66

Parameter j_n was set to three and the mathematical model was solved with CPLEX running on the same computer whose specifications are given in the previous section. Table 4 shows the optimum solution, which was obtained with the objective value of 403.30 within four seconds. As seen from the results, the difference between the completion times of all jobs on machine 1 and machine 2 (403.30 *hr* and 403.22 *hr*, respectively) is very small.

Table 4 shows a total of six jobs being utilised on two machines. The gaps between the determined value of j_n and the number of jobs utilised on each machine ($j_n - \sum_{j=1}^{j_n} Z_{mj}$, $\forall m \in M$) are zero. If the value of j_n is set to four (rather than three), the same solution is found (so, the gap, $j_n - \sum_{j=1}^{j_n} Z_{mj}$, becomes 1 for $\forall m \in M$), but the solution time increases dramatically from four seconds to 166 seconds. That means the solution obtained is optimum and does not change if the value of j_n is set to a value even larger than four. Note that the total area of jobs is close to the capacity of the machines, which indicates high area utilisation.

Table 4. The allocation of parts and the details of jobs for the optimum solution

m	j	Allocated parts	Total Area ($\sum_{i\in I} a_i \cdot X_{mji}$)	Total Volume ($\sum_{i\in I} v_i \cdot X_{mji}$)	$\max\{h_i\}$	PT_{mj}	C_{mj}
1	1	4, 5, 6	899.35	2559.98	36.50	105.56	105.56
	2	2, 11	899.94	3157.01	26.04	116.67	222.23
	3	3, 7, 13, 15, 18	867.22	5265.20	25.10	181.08	**403.30**
2	1	9, 14	705.32	2928.61	12.53	100.16	100.16
	2	1, 12, 16, 17, 20	729.07	1390.18	6.90	48.74	148.90
	3	8, 10, 19	897.43	7362.83	37.25	254.32	403.22

* *Bold indicates* C_{max}

Monma and Potts [32] showed that the makespan problem is NP-hard for two identical parallel batch machines. Moving from that point, the parallel identical AM machine scheduling problem can be considered NP-hard as contextualised for the single AM machine scheduling problem in Section 1. Furthermore, the problem with non-identical (or unrelated) parallel machines, presented in the next section, is also NP-hard, as it is even more complicated than the identical machines form. In non-identical parallel machines, the machines support different heights, which greatly affects the combination of parts assigned to the machines. This is because the production time of a batch is characterised by the part with maximum height in that batch. Moreover, any part cannot be produced on any machine. The non-identical machines also have building platforms of different sizes that correspond to the production area constraint to limit the allocation of parts into batches. Last, but not least, the AM machines have different speed specifications, so the part processing times are resource depended.

4. Parallel non-identical AM machines ($Rm|batch\{AM\}|C_{max}$)

This section defines the parallel AM machine scheduling problem with different machine specifications, referred to as non-identical. Therefore, considering machine specific parameters for VT, HT, SET, and MA is required. Thus, these parameters have been identified with the index m (i.e. VT_m, HT_m, SET_m, and MA_m, respectively). There is also a new parameter to represent the height of each machine, MH_m, to differentiate the machines in terms of the maximum height supported. When machines have different MH_m values, some parts may be produced on an AM machine, while some others may not. This is another capacity constraint that needs to be satisfied in addition to the area capacity constraint. To formulate, Equation (21) must be satisfied to ensure the height capacity constraint.

$$h_i \cdot X_{mji} \le MH_m; \quad \forall m \in M;\ \forall j \in J;\ \forall i \in I. \tag{21}$$

In the parallel identical machine scheduling problem (in the previous section), it was assumed that the heights of parts must be lower than or equal to the heights of the machines. However, in the non-identical machine scheduling problem, this assumption can be changed to *the maximum height of parts must be lower than or equal to the maximum height supported by at least one machine* ($max_{i\in I}\{h_i\} \leq max_{m\in M}\{MH_m\}$). Similarly, there should be at least one machine that can produce the part with the maximum area, $max_{i\in I}\{a_i\} \leq max_{m\in M}\{MA_m\}$.

In contrast to the parallel identical AM machine scheduling problem, the production time calculation for job j on machine m can be made using Equation (22).

$$PT_{mj} = SET_m \cdot Z_{mj} + VT_m \cdot \sum_{i\in I} v_i \cdot X_{mji} + HT_m \cdot \max_{i\in I}\{h_i \cdot X_{mji}\}; \quad \forall m \in M, j \in J. \tag{22}$$

There is no difference in the calculation of job completion time ($C_{mj} \geq C_{m(j-1)} + PT_{mj}$ for $\forall m \in M, j \in J$) and the availability of the machines at the beginning of the planning period ($C_{m0} = 0$).

With the above modifications, the complete MILP model for the $Rm|batch\{AM\}|C_{max}$ problem is presented as follows.

$$min\ Z = \max_{m\in M, j\in J}\{C_{mj}\}. \tag{23}$$

Subject to:

$$\sum_{m\in M}\sum_{j\in J} X_{mji} = 1; \quad \forall i \in I. \tag{24}$$

$$\sum_{i\in I} a_i \cdot X_{mji} \leq MA_m; \quad \forall m \in M; \forall j \in J. \tag{25}$$

$$h_i \cdot X_{mji} \leq MH_m; \quad \forall m \in M; \forall j \in J; \forall i \in I. \tag{26}$$

$$\sum_{i\in I} X_{m(j+1)i} \leq \psi \cdot \sum_{i\in I} X_{mji}; \quad \forall m \in M; \forall j \in J. \tag{27}$$

$$C_{m(j-1)} + PT_{mj} \leq C_{mj}; \quad \forall m \in M; \forall j \in J. \tag{28}$$

$$C_{m0} = 0; \quad \forall m \in M. \tag{29}$$

$$PT_{mj} = SET_m \cdot Z_{mj} + VT_m \cdot \sum_{i\in I} v_i \cdot X_{mji} + HT_m \cdot \max_{i\in I}\{h_i \cdot X_{mji}\}; \quad \forall m \in M; \forall j \in J. \tag{30}$$

$$X_{mji}, Z_{mj} \in (0,1); \quad \forall m \in M; \forall j \in J; \forall i \in I. \tag{31}$$

The objective function given in Equation (23) aims to minimise the makespan, ensuring the constraints given above. The part occurrence constraint (24), job utilisation constraint (27), job completion time constraints (28-29), and sign constraints (31) are the same as in the parallel identical machine scheduling problem. However, the area capacity constraint (25) is modified and the height capacity constraint (26) is newly added. Also, the job production time calculation is made using the machine specific parameters given in Equation (30).

Let us consider the numerical example already given in the previous section (Section 3), assuming there is no change in the parts data. Table 5 gives the parameters of the non-identical machines, which shows that the machines require different setup times and support different heights and areas for production.

Table 5. The machine specific parameters for the numerical example

m	VT_m (hr/cm^3)	$HT_m (hr/cm)$	SET_m (hr)	MH_m (cm)	MA_m (cm^2)
1	0.030864	0.7	1.0	32	800
2	0.030864	0.7	1.2	40	1200

The optimum solution obtained through CPLEX within 16 seconds (when $j_n = 4$) is presented in Table 6, together with the details of jobs, including PT_{mj} and C_{mj}. As seen in the table, two and three jobs are utilised on machine 1 and machine 2, respectively. The gap between the value of j_n and the total number of utilised jobs on machine 1 and machine 2 are two and one, respectively. This shows that the value of j_n provides enough flexibility to guarantee the optimum solution, i.e. the optimum solution does not change when it is set to $j_n = 5$ or even larger.

Table 6. The allocation of parts and the details of jobs for the optimum solution of the parallel non-identical machine scheduling problem

m	j	Allocated parts	Total Area $(\sum_{i\in I} a_i \cdot X_{mji})$	Total Volume $(\sum_{i\in I} v_i \cdot X_{mji})$	$\max\{h_i\}$	PT_{mj}	C_{mj}
1	1	8, 13, 14, 15	794.36	7130.75	18.46	234.01	234.01
	2	10, 18	701.41	4707.98	25.10	163.88	**397.88**
2	1	5, 6, 7, 11	1199.67	4187.01	27.94	155.98	155.98
	2	2, 3, 4, 19	1138.16	4105.64	37.25	153.99	309.97
	3	1, 9, 12, 16, 17, 20	1164.73	2532.43	11.81	87.628	397.60

* *Bold indicates* C_{max}

The makespan is 397.88, determined by the completion time of the second job on machine 1. However, the difference between the makespan and the completion time of the last job on the second machine is very small. This minimises idle time, which also shows the power of the model. Investigating the parts included in the jobs reveals that parts 6 and 19, which exceed the height capacity of machine 1, are allocated to jobs 1 and 2 on machine 2, respectively. Also, the area capacities of machine 1 and machine 2 (800 cm^2 and 1200 cm^2) are not exceeded based on the values reported in the *Total Area* column.

Comparing the solutions obtained for identical and non-identical machines reveals that the total number of jobs decreased from six to five due to the increase in the area capacity of the second machine. This also helped decrease the value of C_{max}.

5. Computational tests

This section presents a comprehensive set of computational tests for the single machine, parallel identical machine, and parallel non-identical machine scheduling problems described in Sections 2, 3, and 4, respectively. The test data were mostly derived from the work of Li et al. [1] and organised in such a way to represent the problem characteristics considered in this research. Detailed problem data about the parts and machine characteristics are provided as supplementary material.

The models presented in this study were coded in IBM ILOG CPLEX Optimization Studio (v12.8.0) and test problems were solved through the CPLEX solver. Computations were conducted on an Intel® Core™ i7-6700HQ CPU @2.60 GHz with 16 GB RAM.

Table 7 presents a summary of the input data and the optimum results obtained (detailed results are available upon request). Columns i_n and j_n represent the number of parts and upper bounds for the number of jobs. Column C_{max} represents the objective value (makespan) of the optimum solution obtained. The time consumed to get the optimum solution is reported in column *Computation Time*. The number of jobs required for the optimum solution is also reported in column "$\sum_{j=1}^{j_n} Z_j$". When determining the value of j_n, consider that j_n does not cause any infeasibility, which gives enough slackness to obtain the optimum solution (as discussed in Sections 2, 3, and 4). As seen in the table, the results were obtained very quickly, as none required more than eight seconds.

Table 7. The problem data and the results for the single machine scheduling problem

Test Problem	Input		Output		
	i_n	j_n	C_{max}	$\sum_{j=1}^{j_n} Z_j$	Computation Time (s)
P1	6	3	201.36	2	4.80
P2	6	3	198.83	2	4.90
P3	7	4	181.23	3	5.20
P4	7	3	173.83	2	5.20
P5	8	4	190.96	3	5.00
P6	8	3	183.55	2	5.00
P7	9	5	266.10	4	5.50
P8	9	4	254.00	3	5.30
P9	10	5	283.03	4	5.30
P10	10	4	275.62	3	5.10
P11	11	5	374.22	4	5.20
P12	11	4	364.85	3	5.20
P13	12	7	538.09	5	5.00
P14	12	6	528.12	4	7.70

Table 8 presents the test data for the parallel identical machine scheduling problem and the results obtained. Columns i_n, m_n, and j_n represent the number of parts, the number of machines, and the upper bound for the number of jobs to be utilised on each machine, respectively. Column C_{max} reports the objective value (makespan) of the obtained solution and column *Optimality* exhibits the optimality condition. The letter "*Y*" indicates that the solution has been proven to be optimum within the time period reported in the column *Computation Time* given the value of j_n. The total number of jobs utilised on the machines is given in the column "$\sum_{m=1}^{m_n} \sum_{j=1}^{j_n} Z_{mj}$". The number of jobs utilised on each machine is also presented in the next column, "$[\sum_{j=1}^{j_n} Z_{mj}]$". For example, the optimum solution for test problem P15 was obtained within 11 seconds. A total of five jobs were utilised on two machines, two of which were on machine 1 and the remaining three on machine 2. The term "*Lim+*" indicates that the code was run under the time limit constraint given in the column *Computation Time*, but the optimum solution was not found. So, the table reports the best solution obtained within the period. That is the case for the large-sized instances due to the grown search space with increasing numbers of parts, machines, and jobs. For example, consider P33. The optimum solution could not be verified within the time limit of 1800 seconds. The best solution (not known whether optimum) requires three, one, and two jobs on the first, second, and third machines, respectively.

Table 8. The problem data and the results for the parallel identical machine scheduling problem

Test Problem	Input			Output				
	i_n	m_n	j_n	C_{max}	Optimality	$\sum_{m=1}^{m_n}\sum_{j=1}^{j_n} Z_{mj}$	$[\sum_{j=1}^{j_n} Z_{mj}]$	Computation Time (s)
P15	15	2	3	197.51	Y	5	[2], [3]	8.20
P16	15	2	3	203.89	Y	4	[2], [2]	7.80
P17	17	2	4	389.97	Y	6	[2], [4]	45.00
P18	17	2	3	397.78	Y	5	[3], [2]	9.00
P19	18	2	3	381.17	Y	6	[3], [3]	9.90
P20	18	2	3	385.08	Y	5	[2], [3]	18.20
P21	21	2	3	280.61	Y	5	[2], [3]	30.40
P22	21	2	3	294.95	Y	6	[3], [3]	21.50
P23	22	2	3	414.32	Y	5	[2], [3]	18.50
P24	22	2	3	433.10	Y	6	[3], [3]	10.10
P25	23	2	3	435.43	Y	6	[3], [3]	40.50
P26	23	2	3	454.85	Y	6	[3], [3]	35.00
P27	25	2	3	438.41	Y	6	[3], [3]	294.30
P28	25	2	3	462.36	Y	6	[3], [3]	533.00
P29	28	3	3	348.54	Lim+	7	[3], [1], [3]	1800.00
P30	28	3	3	358.80	Lim+	7	[1], [3], [3]	1800.00
P31	30	3	2	341.51	Y	6	[2], [2], [2]	12.2
P32	30	3	3	349.25	Lim+	6	[3], [1], [3]	1800.00
P33	36	3	3	368.68	Lim+	6	[1], [3], [3]	1800.00
P34	36	3	3	378.79	Lim+	7	[3], [3], [1]	1800.00
P35	38	3	3	361.05	Lim+	7	[3], [3], [1]	1800.00
P36	38	3	3	371.74	Lim+	7	[1], [3], [3]	1800.00
P37	46	3	3	435.71	Lim+	8	[2], [3], [3]	1800.00
P38	46	3	3	447.10	Lim+	8	[3], [2], [3]	1800.00

Similar to the parallel identical machine scheduling problem, Table 9 presents the problem data and the results for the parallel non-identical machine scheduling problem. Note that the problems contain the same number of parts and machines as in the parallel identical machine scheduling problems. However, in contrast with the parallel identical machine scheduling problem, the machine specifications may vary. Therefore, while all parts can be processed on any machine in the parallel identical machine scheduling problems, some parts cannot be produced on some machines due to height capacity constraints. The optimum solutions are reported in column C_{max} if it is denoted with the letter "*Y*" in the column *Optimality*. When it was not possible to obtain the optimum solution (those identified with "*Lim+*"), the best solution obtained within 2400 seconds is reported.

Table 9. The problem data and the results for the parallel non-identical machine scheduling problem

Test Problem	Input			Output				
	i_n	m_n	j_n	C_{max}	Optimality	$\sum_{m=1}^{m_n}\sum_{j=1}^{j_n} Z_{mj}$	$[\sum_{j=1}^{j_n} Z_{mj}]$	Computation Time (s)
P39	15	2	3	195.44	Y	5	[3], [2]	5.80
P40	15	2	3	199.45	Y	5	[3], [2]	5.80
P41	17	2	3	385.59	Y	5	[2], [3]	8.40
P42	17	2	3	396.93	Y	6	[3], [3]	5.80
P43	18	2	3	372.58	Y	6	[3], [3]	7.40
P44	18	2	3	380.22	Y	6	[3], [3]	7.60
P45	21	2	3	286.53	Y	6	[3], [3]	21.40
P46	21	2	3	293.09	Y	6	[3], [3]	28.20
P47	22	2	3	425.93	Y	6	[3], [3]	26.20
P48	22	2	3	435.51	Y	6	[3], [3]	38.90
P49	23	2	3	447.48	Y	6	[3], [3]	67.00
P50	23	2	3	456.31	Y	6	[3], [3]	84.00
P51	25	3	3	296.05	Y	7	[1], [3], [3]	48.90
P52	25	3	3	299.71	Y	6	[1], [3], [2]	141.10
P53	28	3	3	351.67	Lim+	8	[2], [3], [3]	2400
P54	28	3	3	357.76	Lim+	7	[3], [2], [3]	2400
P55	30	3	3	342.30	Lim+	7	[2], [2], [3]	2400
P56	30	3	3	345.04	Lim+	8	[1], [3], [3]	2400
P57	36	3	3	374.05	Lim+	8	[2], [3], [3]	2400
P58	36	3	3	377.12	Lim+	9	[3], [3], [3]	2400
P59	38	3	3	364.62	Lim+	8	[2], [3], [3]	2400
P60	38	3	3	368.94	Lim+	8	[2], [3], [3]	2400
P61	46	3	4	443.71	Lim+	9	[2], [3], [4]	2400
P62	46	3	3	445.38	Lim+	9	[3], [3], [3]	2400

The optimum solutions have been proven to be optimum under the constraint that the upper bound for the number of jobs on each machine is set to j_n, given in the table. In some cases, relaxing the value of j_n may yield better solutions for the parallel non-identical machine scheduling problems. This is because the machines may have different characteristics. So, allocating more parts and jobs on those with lower values of VT_m and HT_m parameters can lead to better solutions. For example, the optimum solution for P42 was 396.93 when j_n was set to three; three jobs were utilised on each of the AM machines. However, if j_n were set to four (instead of three), the optimum solution would be 394.35 with two jobs utilised on the first machine and four jobs utilised on the second machine. Obviously, the objective value reduces very slightly. However, the computation time increases to 35 seconds, which corresponds to a remarkable increase over the 5.8 seconds when j_n was set to three. However, this is not the case for P43, where the optimum solution ($C_{max} = 372.58$) does not change even if the value of j_n is set to four. On the contrary, the computation time increases from 7.40 seconds to 42 seconds. Therefore, enough gap should be provided for the j_n parameter, but increases in computation time should be kept in mind when doing so. There is a trade-off between the computation time and possible improvement in the objective value.

6. Conclusions and future work

This paper introduced the production scheduling problem of AM machines with the aim of minimising makespan. In terms of the configuration of the machines, three types of problems have been considered: single machine, parallel identical machines, and parallel non-identical machines. The problems have been defined, modelled as MILPs, and explored through numerical examples. Computational tests were conducted through test data derived from the literature and adapted to the characteristics of the problems. The optimum solutions have been reported where the computational capacity allows, e.g. all single model problems, up to 30 parts-3 parallel identical machines and up to 25 parts-3 parallel non-identical machines. The best solutions obtained within the time limits (1800 and 2400 seconds for the parallel identical and parallel non-identical machine scheduling problems, respectively) are reported for problems for which computational capacity does not allow an optimum solution. The results of this study clearly show the difficulty of solving the problems when problem size increases. This indicates the necessity of heuristic and metaheuristics to solve large-scale problem instances.

The methods proposed in this study can easily be adopted by any firm using AM technology for developing efficient schedules. Even AM firms that provide on-demand supply for additively manufactured products through the internet can use the methods proposed in this study to simulate their production environment and give accurate quotes to their customers.

In addition to the proposal given above that indicates the necessity of heuristic/metaheuristic approaches for solving large-scale instances, the models developed here can be even extended in several ways. First, a methodology to determine a well-defined upper bound can be developed to balance the trade-off between computation time and optimality for the parallel identical/non-identical machine scheduling problem. Second, the due dates of the parts/products can be considered and the objectives of minimising lateness, maximum lateness, total number of late jobs, etc. can be optimised individually or concurrently. Finally, a comprehensive nesting procedure can be integrated into the model to allocate parts on the machines based on their exact dimensions on the vertical and horizontal axes, rather than the production area.

Acknowledgement

The authors would like to thank the Area Editor, Dr. Mikhail Kovalyov, and anonymous reviewers for their valuable comments which have helped improve the quality and presentation of the manuscript.

References

[1] Li Q, Kucukkoc I, Zhang DZ. Production planning in additive manufacturing and 3D printing. Comput Oper Res 2017;83:157–72.

[2] Kucukkoc I, Li Q, Zhang DZ. Increasing the utilisation of additive manufacturing and 3D printing machines considering order delivery times. 19th Int. Work. Semin. Prod. Econ., Innsbruck, Austria: 2016, p. 195–201.

[3] Vicari A. Advanced Applications of 3D Printing: From Prototypes and Parts, Additive Manufacturing for Defence and Aerospace Summit, London, https://additivemanufacturing.iqpc.co.uk/downloads/advanced-applications-of-3d-printing-from-prototypes-and-parts; 2015.

[4] NASA Marshall. Additive Manufacturing: Pioneering Affordable Aerospace Manufacturing. Georg C Marshall Sp Flight Cent 2017. https://www.nasa.gov/sites/default/files/atoms/files/additive_mfg.pdf (accessed January 5, 2018).

[5] Cooper DE, Stanford M, Kibble KA, Gibbons GJ. Additive Manufacturing for product improvement at Red Bull Technology. Mater Des 2012;41:226–30. doi:10.1016/J.MATDES.2012.05.017.

[6] Khajavi SH, Partanen J, Holmström J. Additive manufacturing in the spare parts supply chain. Comput Ind 2014;65:50–63. doi:10.1016/J.COMPIND.2013.07.008.

[7] Brucker P, Gladky A, Hoogeveen H, Kovalyov MY, Potts CN, Tautenhahn T, et al. Scheduling a batching machine. J Sched 1998;1:31–54. doi:10.1002/(SICI)1099-1425(199806)1:1<31::AID-JOS4>3.0.CO;2-R.

[8] Li X, Huang Y, Tan Q, Chen H. Scheduling unrelated parallel batch processing machines with non-identical job sizes. Comput Oper Res 2013;40:2983–90. doi:10.1016/J.COR.2013.06.016.

[9] Brucker P, Kovalyov MY. Single machine batch scheduling to minimize the weighted number of late jobs. Math Methods Oper Res 1996;43:1–8. doi:10.1007/BF01303431.

[10] Albers S, Brucker P. The complexity of one-machine batching problems. Discret Appl Math 1993;47:87–107. doi:10.1016/0166-218X(93)90085-3.

[11] Koh S-G, Koo P-H, Ha J-W, Lee W-S. Scheduling parallel batch processing machines with arbitrary job sizes and incompatible job families. Int J Prod Res 2004;42:4091–107. doi:10.1080/00207540410001704041.

[12] Garey, R. M. Computers and Intractability : A Guide to the Theory of NP-completeness, Freeman. Fundamental 1997.

[13] Damodaran P, Vélez-Gallego MC, Maya J. A GRASP approach for makespan minimization on parallel batch processing machines. J Intell Manuf 2011;22:767–77. doi:10.1007/s10845-009-0272-z.

[14] Cheng TCE, Chen Z-L, Oguz C. One-machine batching and sequencing of multiple-type items. Comput Oper Res 1994;21:717–21. doi:10.1016/0305-0548(94)90001-9.

[15] Ghosh JB, Gupta JND. Batch scheduling to minimize maximum lateness. Oper Res Lett 1997;21:77–80. doi:10.1016/S0167-6377(97)00028-X.

[16] Jia Z, Leung JY-T. A meta-heuristic to minimize makespan for parallel batch machines with arbitrary job sizes. Eur J Oper Res 2015;240:649–65. doi:10.1016/J.EJOR.2014.07.039.

[17] Zhou S, Chen H, Li X. Distance matrix based heuristics to minimize makespan of parallel batch processing machines with arbitrary job sizes and release times. Appl Soft Comput 2017;52:630–41. doi:10.1016/J.ASOC.2016.10.008.

[18] Shahidi-Zadeh B, Tavakkoli-Moghaddam R, Taheri-Moghadam A, Rastgar I. Solving a bi-objective unrelated parallel batch processing machines scheduling problem: A comparison study. Comput Oper Res 2017;88:71–90. doi:10.1016/J.COR.2017.06.019.

[19] Trindade RS, de Araújo OCB, Fampa MHC, Müller FM. Modelling and symmetry breaking in scheduling problems on batch processing machines. Int J Prod Res 2018:1–18. doi:10.1080/00207543.2018.1424371.

[20] Li X, Zhang K. Single batch processing machine scheduling with two-dimensional bin packing

constraints. Int J Prod Econ 2018;196:113–21. doi:10.1016/J.IJPE.2017.11.015.

[21] Méndez CA, Cerdá J, Grossmann IE, Harjunkoski I, Fahl M. State-of-the-art review of optimization methods for short-term scheduling of batch processes. Comput Chem Eng 2006;30:913–46. doi:10.1016/J.COMPCHEMENG.2006.02.008.

[22] Potts CN, Kovalyov MY. Scheduling with batching: A review. Eur J Oper Res 2000;120:228–49. doi:10.1016/S0377-2217(99)00153-8.

[23] Webster S, Baker KR. Scheduling Groups of Jobs on a Single Machine. Oper Res 1995;43:692–703.

[24] Kucukkoc I, Qiang L, He N, Zhang D. Scheduling of Multiple Additive Manufacturing and 3D Printing Machines to Minimise Maximum Lateness. Twent. Int. Work. Semin. Prod. Econ., 2018, p. 237–47.

[25] Chergui A, Hadj-Hamou K, Vignat F. Production scheduling and nesting in additive manufacturing. Comput Ind Eng 2018;126:292–301. doi:10.1016/J.CIE.2018.09.048.

[26] Fera M, Macchiaroli R, Fruggiero F, Lambiase A. Production Management Fundamentals for Additive Manufacturing. 3D Print., InTech; 2018. doi:10.5772/intechopen.78680.

[27] Dvorak F, Micali M, Mathieug M. Planning and Scheduling in Additive Manufacturing. Intel Artif 2018;21:40. doi:10.4114/intartif.vol21iss62pp40-52.

[28] Fera M, Fruggiero F, Lambiase A, Macchiaroli R, Todisco V. A modified genetic algorithm for time and cost optimization of an additive manufacturing single-machine scheduling. Int J Ind Eng Comput 2018:423–38. doi:10.5267/j.ijiec.2018.1.001.

[29] Zhang Y, Gupta RK, Bernard A. Two-dimensional placement optimization for multi-parts production in additive manufacturing. Robot Comput Integr Manuf 2016;38:102–17. doi:10.1016/J.RCIM.2015.11.003.

[30] Zhang Y, Bernard A, Harik R, Karunakaran KP. Build orientation optimization for multi-part production in additive manufacturing. J Intell Manuf 2017;28:1393–407. doi:10.1007/s10845-015-1057-1.

[31] Oh Y, Zhou C, Behdad S. Production Planning for Mass Customization in Additive Manufacturing: Build Orientation Determination, 2D Packing and Scheduling. Vol. 2A 44th Des. Autom. Conf., ASME; 2018, p. V02AT03A033. doi:10.1115/DETC2018-85639.

[32] Monma CL, Potts CN. On the Complexity of Scheduling with Batch Setup Times. Oper Res 1989;37:798–804. doi:10.1287/opre.37.5.798.

Graphical Abstract