Operational Research (2022) 22:3063–3087
https://doi.org/10.1007/s12351-021-00649-y

ORIGINAL PAPER

Additive manufacturing scheduling problem considering assembly operations of parts

Oğuzhan Ahmet Arık [1]

Received: 4 April 2020 / Revised: 27 April 2021 / Accepted: 15 May 2021 /
Published online: 28 May 2021

Abstract
Additive manufacturing (AM) is a candidate to be one of the future general-purpose technologies. AM is called with 3D printing, Rapid Prototyping, Direct Digital Manufacturing, layered manufacturing, and/or additive manufacturing. Today, AM is a tool for producing customized small products with small lot sizes. Tomorrow, it will be possible for AM to enter every home and to be used for general purposes. The literature about AM has focused mainly on the technology to decrease the cost of AM, to increase the speed of AM machines, and to increase the common availability of those machines. There are so few papers investigating AM machines in view of scheduling problems. This paper considers a single AM machine that produces multiple parts in batches and then these parts are assembled to produce desired goods. Most AM machines have limitations because of the area of the machine tray and the height of the machine. Therefore, products are separated into small parts considering the area and height of the machine. Then, these separated small parts are assembled to produce the desired goods. In view of scheduling problems, the proposed problem includes single machine batch scheduling and assembly operations. In this paper, we propose a mixed-integer programming (MIP) model and a fast heuristic method with a simple local search mechanism for the problem. We investigate two cases for the same problem. In the first, we consider only the one-dimensional assignment of parts to baches and we just design our solution approaches to assign parts to the batch, if the total area of parts is less than or equal to the machine tray's area. In the second, we modify our solution approaches to consider parts' lengths and widths while assigning parts to batches in a 2D assignment. In the end, we compare the proposed heuristic with the proposed MIP by using some test problems within time limits for all cases. Experimental results show that the proposed heuristic provides promising results.

Keywords Additive manufacturing · 3D printing · Scheduling · MIP · Assembly · Production

Extended author information available on the last page of the article

Mathematics Subject Classification 90B35 · 90C59

1 Introduction

Additive manufacturing (AM) is an advanced technology where products are manufactured by building up thin layers of materials from digitized three-dimensional (3D) designs virtually constructed using advanced computer-aided design software (Achillas et al. 2015). As opposed to classical subtractive manufacturing methods that remove sections of material by machining or cutting that material AM adds layers of material to create a desired good with less material waste. Today, AM is a tool for producing customized small products with small lot sizes. Tomorrow, it will be possible for AM to enter every home and to be used for general purposes. Like all new technologies that are candidates to be future general purposed technology, AM technology is expensive. AM machines produce parts on their machine tray, so multiple parts are produced in the same tray. In contrast to classical batch processing in scheduling problems, the AM machine's batch processing time cannot be calculated by knowing the maximum processing time among all processing times of all parts in the tray or the sum of processing time of all parts. The total volume of parts and heights of parts are factors for determining the processing time of the batch in the AM machine. The machine tray has an area limitation so parts must be assigned to batches by considering this limitation. In that sense, parts should be separated into batches by considering their heights, volumes, and areas. Furthermore, batches must be created and scheduled considering the pre-determined performance criterion such as minimizing total completion time, the makespan, or total weighted earliness/tardiness duration of parts. Although there are lots of papers about how to increase the efficiency of AM machines or how to decrease the cost of AM machines or using new materials or technologies for AM machines, there are fewer studies about investigating AM machines within scheduling problems.

The study of Li et al. (2017) was a pioneer for introducing AM machine scheduling problems to the literature. They introduced a problem where there are different AM machines having different specifications and there are different parts having different properties such as height, volume, and area. In their problem, the performance criterion is to minimize the total cost of producing parts in AM machines. They proposed a novel mathematical model and two different heuristics called 'best-fit' and 'adapted best-fit' for the problem. Ransikarbum et al. (2017) proposed a decision aiding model based on multi-objective optimization for a batch of parts and multiple printers. Rudolph and Emmelmann (2017) proposed a cloud-based platform for automated order processing in AM where the order acceptance is determined according to the checking of manufacturing restrictions and designed guidelines. Zhou et al. (2018) investigated a problem with multi-task scheduling of distributed 3D printing services in cloud manufacturing.

Kucukkoc et al. (2018) developed a genetic algorithm approach to minimize the maximum lateness where there are more than two additive manufacturing machines with different capacity specifications. Chergui et al. (2018) studied a problem for production planning and scheduling of identical parallel AM machines to fulfill the different orders by due dates and to minimize the total tardiness. Dvorak et al. (2018) considered a scheduling problem for multiple AM machines and parts where the objective is to minimize time spent. Fera et al. (2018) proposed a modified genetic algorithm for time and cost optimization of an AM single-machine scheduling problem to balance the optimization of earliness/tardiness and production costs. Oh et al. (2018) designed a heuristic algorithm that was proposed for decision-making on build orientation, 2D packing, and scheduling on multiple AM machines based on the longest cycle time. Li et al. (2018) introduced an order acceptance and scheduling problem faced by metal AM companies for the first time in the literature and proposed a novel decision model inspired by Optimal Foraging Theory for solving this problem by maximizing the utilization of machine and optimizing the payoff from an order acceptance decision.

Li et al. (2019) introduced a dynamic order acceptance scheduling in on-demand production with powder bed fusion systems and aimed to provide an approach for manufacturers to make decisions simultaneously. Zhang et al. (2019) developed an improved evolutionary algorithm for application to AM, by combining a genetic algorithm with a heuristic placement strategy to take into account the allocation and placement of parts integrally. Luzon and Khmelnitsky (2019) investigated the problem of sequencing an AM process while referring to its relevant properties. Kucukkoc (2019) investigated scheduling problems of single and multiple AM machines and proposed mathematical optimization models. Mixed-integer linear programming models allocating parts into jobs to be produced on AM machines were proposed by Kucukkoc (2019) to minimize the makespan. Furthermore, Kucukkoc (2019) introduced how to calculate the batch processing time of the AM machine.

As different from the literature for AM machine scheduling problems, this paper addresses subsequent assembly operations after batch processing of AM machines. Taking account of assembly operations into the problem after AM batch scheduling increases the complexity of the problem but it is a real-life problem, for instance, let assume a healthcare 3D printing company that produces prosthetic arms or hands. There are so many parts for a 3D printed prosthetic arm to be merged or assembled to build a prosthesis after the AM machine's batch processing. As another example, 3D printed shoe producers need to utilize batch operations considering assembly operations done after batch processing. Turbine, engine parts, and other cabin interior components of aircraft are started to be produced with AM machines because of their complex geometries and aerodynamic properties. These parts of the aircraft are assembled after the AM machine production phase. Utilizations in both batch production and assembly operations of these parts are important for the aerospace industry in where being cost-effective and timing are so important. For the automobile industry, such an example of mass production systems, AM machine is important to build prototypes of new models or components before mass production. As another example, spare-part producers do not have to produce with high volumes, if they arrange their production batch operations considering assembly operations of parts produced in AM machines. Examples for batch processing in AM machines and assembly operations can be diversified. To consider the real-life situation, we propose an MIP model and a fast heuristic method for the problem. Furthermore, we combine a swap-based local search mechanism within our proposed heuristic to increase its solution quality. AM machine scheduling is a new area for researchers. Parts are processed in AM machines as batches but these parts are mainly assembled after batch processing of AM machine. In order to address a real-life problem, we investigate the problem and propose a mathematical model that considers both batch processing and assembly operations of parts. Furthermore, we propose a fast heuristic method that uses a novel criterion named the importance weight of parts and uses the first fit increased heuristic. Then we use the solution of our proposed heuristic as an initial solution within our proposed swap-based local search. Batch scheduling of AM machines and scheduling of assembly operations after batch processing have not been investigated together before. Therefore, this study provides an important contribution to the literature.

Assembly operations after batch processing or machine operations of jobs/parts have been considered for classical machine scheduling problems by researchers in the literature. Our investigated problem has two stages; the first one of these is to process jobs/parts in batches in AM batch processing and the second one of these is to assembly parts after AM batch processing. Therefore, there is a similarity between the classical two-or-more stage scheduling problem and our investigated problem. There is an obligation that the batch processing must be done before the assembly operations. This obligation makes our problem similar to the two-machine flow shop scheduling problem. Nevertheless, there are some similarities between our investigated problem, and other scheduling problems in the literature, the calculation methods of the batch processing time in classical batch processing problems are done with two methods. The first one sets the batch processing time to the sum of all processing times of jobs in the batch. The second method sets the batch processing time to the longest processing of jobs in the batch. In our problem, the batch processing time is a function that depends on parts' area, volume, and height values. With this aspect of the batch processing time calculation method, our investigated problem differs from the classical scheduling problems in the literature. There are lots of examples for classical two or more stages scheduling problems with assembly operations considering classical batch completion times. Lin and Cheng (2002) considered a two-machine flow shop where each job has a unique component and a common component to be processed on the first machine in batches. After batch processing in the first machine, the job is ready for its assembly operation in the second machine. The batch processing calculation method in their study is to set the batch processing time to the sum of all processing times of jobs in the batch. Kovalyov et al. (2004) investigated a two-stage assembly scheduling problem that includes the assembly of final or intermediate products from basic components. They used two different types of batching in their problem. In the first one, the batch processing time is equal to the maximum of the processing times of its operations. In the second one, the batch processing time is equal to the sum of the processing times of its operations. Lin et al. (2006) addressed a three-machine assembly-type flow shop scheduling problem. In the first stage of their problem, there are two machines in parallel for producing component parts individually. The last machine in the second stage is an assembly line for processing the component parts in batches. The batch processing time of a batch is the sum of all processing times of jobs and a constant setup time. Tajbakhsh et al. (2014) investigated a three-stage manufacturing system including machining, assembly, and batch processing stages. They proposed a hybrid metaheuristic algorithm for their problem. Loukil et al. (2007) investigated a production scheduling problem including the production of several sub-products followed by the assembly of the final product. They proposed a multi-objective simulated annealing approach for their problem. Liao et al. (2015) considered a two-stage assembly scheduling problem with setup times to minimize the makespan.

The remaining paper is organized as follows. In Sect. 2, we propose a mixed-integer programming model for the investigated problem. We present a lower bound of the makespan after task operations for the problem in Sect. 3. We propose a fast heuristic method that uses a novel criterion named the importance weight of parts and a swap-based local search mechanism for the problem in Sect. 4. We generate some test problems and use them in our experimental study in Sect. 5. Finally, we summarize our findings and give suggestions for future research in Sect. 6.

2 Mixed integer programming model for single AM machine scheduling with assembly operations

In this section, an MIP model is introduced for single AM machine scheduling with assembly operations. Let us assume there are n parts that will be produced by a single AM machine. Each part may be considered as a single batch or multiple parts can be produced within the same batch. Therefore, it is possible to have n alternative batches in a single AM machine. After producing parts in batches, these parts may be assembled to obtain desired m goods so the order of batches including different parts should let the operator assemble parts without waiting for needed parts for the assembly task. Considering assembly operations of desired goods while assigning parts to batches and scheduling batches have advantages for the completion time of the lastly produced goods. The first advantage is decreasing the waiting time of parts in assembly operations. Scheduling only batches without considering assembly operations may decrease the completion time of the last batch but it also may make the operator wait for necessary parts for assembly operations. The second advantage of considering both batch scheduling and assembly operations simultaneously is that the scheduler may utilize both batches and assembly operations for the performance measure of the company. As a performance measure, the latest task's completion time is the makespan for the problem. Kucukkoc (2019) suggested calculating each batch's processing time for AM machines with the powder-based system as follows:

$$PT_j = SET \times Z_j + VT \times \sum_{i=1}^{n} v_i \times X_{ij} + HT \times max\{h_i \cdot X_{ij}\} \tag{1}$$

where PT_j is the processing time of batch j $(j = 1, 2, \ldots, n)$, SET is setup time needed for initializing and cleaning, HT is the time spent for powder-layering, which is repeated for each layer based on the highest part produced in the job, VT is the time spend to form per unit volume of material, X_{ij} is a binary variable that equals 1 if part i $(i = 1, 2, \ldots, n)$ is assigned to batch j and Z_j is a binary variable that is equal to 1 if $\sum_{i=1}^{n} X_{ij} \geq 1\ \forall j$. In Eq. (1), each part's height h_i (area a_i) is equal to or less than the machine's maximum height (area). The used calculation method in Eq. (1)

proposed by Kucukkoc (2019) is applicable for metal powder bed fusion processes. In addition to powder bed fusion processes (Selective Laser Melting and Electronic Beam Melting), there are also several different processes such as Selective Laser Sintering and Binder Jetting. These processes can produce parts by letting them be overlapped in the layering phase. In these processes, the volume of the machine is a more important indicator of the machine's capability than the area of the machine tray. The proposed MIP for the problem as follows:

indexes
i, r: index of parts ($iandr = 1, 2, \dots, n$)
j: index of batches ($j = 1, 2, \dots, n$)
k: index of assembly tasks (operations) ($k = 1, 2, \dots, K$)

parameters

a_i: area of part i
v_i: volume of part i
h_i: height of part i
t_k: processing time of task k
$Prec_{kl}$: if task k is a processor of task l then $Prec_{kl}$=1. Otherwise, it is 0.
B_{ik}: if part i is required to complete task k then B_{ik}=1. Otherwise, it is 0.
MA: the area of the machine tray

decision variables

PT_j: processing time of batch j
S_j: start time of batch j in single AM machine
C_j: completion time of batch j in single AM machine
SI_i: start time of part i in single AM machine
CI_i: completion time of part i in single AM machine
ST_k: start time of task k
CT_k: completion time of task k
$X_{ij} : \begin{cases} 1, & \textit{if part i is assigned to batch j} \\ 0, & \textit{otherwise} \end{cases}$
$Z_j : \begin{cases} 1, & \textit{if there is a part assigned to batch j at least} \\ 0, & \textit{otherwise} \end{cases}$
C_{max}: the makespan of the schedule or the latest completion time of tasks

Objective function

$$Minf = C_{max} \tag{2}$$

Constraints

$$C_{max} \geq CT_k \ \forall k \tag{3}$$

$$CT_k = ST_k + t_k \ \forall k \tag{4}$$

$$ST_k \geq B_{ik} \times CI_i \ \forall k, i \tag{5}$$

$$ST_l \geq Prec_{kl} \times CT_k \ \forall k, l \quad where \quad k \neq l \quad and \quad Prec_{kl} = 1 \tag{6}$$

$$SI_i = \sum_{j=1}^{n} X_{ij} \times S_j \ \forall i \tag{7}$$

$$CI_i = \sum_{j=1}^{n} X_{ij} \times C_j \ \forall i \tag{8}$$

$$S_j \geq C_{j-1} \ \forall j \quad where\ j > 0 \tag{9}$$

$$C_j = S_j + PT_j \ \forall j \quad where\ j > 0 \tag{10}$$

$$S_0 = 0 \tag{11}$$

$$C_0 = 0 \tag{12}$$

$$PT_j = SET \times Z_j + VT \times \sum_{i=1}^{n} v_i \times X_{ij} + HT \times max\left\{h_i.X_{ij}\right\} \ \forall j > 0 \tag{13}$$

$$\sum_{i=1}^{n} X_{ij} \leq n \times Z_j \ \forall j \tag{14}$$

$$\sum_{j=1}^{n} X_{ij} = 1 \ \forall i \tag{15}$$

$$\sum_{i=1}^{n} X_{i,j+1} \leq n \times \sum_{i=1}^{n} X_{ij} \quad \forall j = 1, 2, \ldots, n-1 \tag{16}$$

$$\sum_{i=1}^{n} a_i * X_{ij} \leq MA \ \forall j \tag{17}$$

$$\textstyle\sum_{i=1}^{n} X_{ij} \geq Z_j \ \forall j \tag{18}$$

$$PT_j, S_j \quad and \quad C_j \geq 0 \; \forall j \tag{19}$$

$$SI_i \quad and \quad CI_i \geq 0 \; \forall i \tag{20}$$

$$ST_k \quad and \quad CT_k \geq 0 \; \forall k \tag{21}$$

$$X_{ij} \in \{0, 1\} \; \forall i, j \tag{22}$$

$$Z_j \in \{0, 1\} \; \forall j \tag{23}$$

The objective function (2) is to minimize the makespan. Constraint (3) shows that the makespan is greater than the completion times of all tasks. Constraint (4) shows that a task's completion time is equal to the sum of its start time and task time. Constraint (5) assures that the start time of any task is greater than completion times of parts in a single AM machine if these tasks are required to complete that task. Constraint (6) assures that if a task is a predecessor of other tasks, then the starting time of these tasks is greater than its processors' completion times. Constraints (7–8) show to calculation way for determining each part's starting and completion times by considering in which batch it is assigned to. Constraint (9) shows that the starting time of a batch is greater than or equal to the previous batch's completion time. Constraint (10) assures that the completion time of a batch is equal to the sum of its starting time and processing time. Constraints (11–12) show that the AM machine is ready to process batches at the beginning. Constraint (13) calculates each batch's processing time. Constraint (14) assures that if a part is assigned to batch j at least, Z_j will be 1. Constraint (15) assures that each job is assigned to only one batch. Constraint (16) guarantees that batches are utilized in an incremental order starting from batch 1. Constraint (17) assures that the total area of parts assigned to any batch is less than or equal to the machine tray's area. Constraint (18) assures that no batch will be deployed if there is no assigned part for it. Constraints (19–23) define the necessary domains of decision variables. Constraints (7) and (8) make the proposed model a mixed-integer nonlinear programming model. Thus, there is a chance to decrease the complexity of the current model and to save computation times in solving the model by using disjunctive inequalities as follows:

$$SI_i \leq S_j + M \times \left(1 - X_{ij}\right) \; \forall ij \tag{24}$$

$$SI_i \geq S_j - M \times \left(1 - X_{ij}\right) \; \forall ij \tag{25}$$

$$CI_i \leq C_j + M \times \left(1 - X_{ij}\right) \; \forall ij \tag{26}$$

$$CI_i \geq C_j - M \times \left(1 - X_{ij}\right) \; \forall ij \tag{27}$$

When constraints (7–8) are replaced with constraints (24–27), the model is linearized with help of disjunctive inequalities.

Even constraint (17) assures that the total area of assigned parts to a batch is not more than the area of the tray, we need a 2D assignment consideration for batches and parts. Although the total area of parts that are assigned to the batch is less than the area of the machine tray, parts in that batch may not be fitted perfectly to the machine tray and they can be overlapped. Therefore, we need a 2D consideration for batches and parts. We assume that the machine tray and parts are rectangular. Each combination of parts for a batch must be well fitted without overlapping on the tray. Parts have two dimensions (length and width) and a part can be placed in a batch by rotating it. Parts are placed parallel to the edges of the machine tray. Kucukkoc (2019) just considered whether the total area of parts is less than the area of the machine tray. The above model does not consider the dimensional matching of part combinations to the batches. We call this model as MIP1. We use the additional parameters, decision variables, and constraints in the study of Li and Zhang (2018) to modify MIP1 to make it consider the 2D assignment of parts. We call this modification MIP2. The new adding to the model as follows:

additional parameters

dw_i : the width of the part i
dh_i : the length of the part i
WMA : the width of the machine tray
HMA : the length of the machine tray

additional decision variables

cx_i *and* cy_i : the horizontal and vertical coordinates of part i on the machine tray

$$O_i : \begin{cases} 1, & \textit{if job j is placed in such a way that its width is parallel to the width of the machine} \\ 0, & \textit{otherwise} \end{cases}$$

$$PL_{ir} : \begin{cases} 1, & \textit{if job i is placed to the left of job r} \\ 0, & \textit{otherwise} \end{cases}$$

$$PB_{ir} : \begin{cases} 1, & \textit{if job i is placed below to job r} \\ 0, & \textit{otherwise} \end{cases}$$

additional constraints

$$cx_i + dw_i * O_i + dh_i \times \left(1 - O_i\right) \le WMA \ \forall i \tag{28}$$

$$cy_i + dh_i * O_i + dw_i \times \left(1 - O_i\right) \le HMA \ \forall i \tag{29}$$

$$cx_i + dw_i * O_i + dh_i \times \left(1 - O_i\right) \le cx_r + WMA \times \left(1 - PL_{i,r}\right) \ \forall i, r \text{ where } i \ne r \tag{30}$$

$$cy_i + dh_i * O_i + dw_i \times \left(1 - O_i\right) \le cy_r + WMA \times \left(1 - PB_{i,r}\right) \ \forall i, r \text{ where } i \ne r \tag{31}$$

$$PL_{i,r} + PB_{i,r} + PL_{r,i} + PB_{r,i} \leq x_{i,j} + x_{r,j} - 1 \; \forall i, r \; and \; j \; where \; i < r \tag{32}$$

$$cx_i, cy_i \geq 0 \; \forall i \tag{33}$$

$$PL_{i,r}, PB_{i,r}, O_i \in \{0, 1\} \; \forall i, r \tag{34}$$

Above constraints (28–34) are added to MIP1 to make it consider the 2D assignment of parts to batches. Constraints (28) and (29) assure that the parts must not be placed outside of the machine tray. If two parts (i and r) are assigned to the same batch, then these parts cannot overlap in positions of the batch. This is assured by Constraints (30–32). If part i is placed to the left of part r ($PL_{ir} = 1$), then constraint (30) assures that the horizontal coordinate of part i is less than the horizontal coordinate of part r. If part i is placed below to part r ($PB_{ir} = 1$), then constraint (31) assures that the vertical coordinate of part i is less than the vertical coordinate of part r. If part i and part r are assigned to the same batch, then there are four possible values for PB_{ir} and PL_{ir} values. One of these values must be equal to one if part i and part r are assigned to the same batch. Constraints (33–34) define the necessary domains of additional decision variables.

As a small numerical example, we used the example of Kucukkoc (2019). There are 12 parts and a single AM machine in the example given in Table 1. There are two desired finished goods that need parts produced by AM machine and there are 9 tasks required to be done for producing desired finished goods. The precedence diagrams for assembly operations are given in Fig. 1. Task durations are given in Table 2. The machine parameters are taken from the example of Kucukkoc (2019) and they are VT=0.030864 h/cm^3, HT=0.7 h/cm, SET= 1 h and MA=900 cm^2.

In Fig. 1, each task may need some parts to be completed. For instance, parts 1 and 2 are required to complete task 1 and there is no part required for tasks 4 and 8. This numerical example is taken from the study of Kucukkoc (2019). Without

Table 1 The numerical example (Kucukkoc 2019)

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

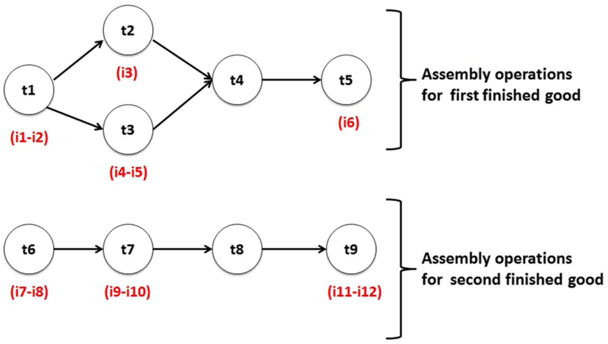

t1
(i1-i2)
t2
(i3)
t3
(i4-i5)
t4
t5
(i6)
Assembly operations for first finished good
t6
(i7-i8)
t7
(i9-i10)
t8
t9
(i11-i12)
Assembly operations for second finished good

Fig. 1 Precedence diagrams of tasks for finished goods for the numerical example

Table 2 Task durations for numerical example

Tasks (*k*)	1	2	3	4	5	6	7	8	9
Duration (hr)	0.50	0.25	0.60	1.00	2.00	1.25	0.90	0.80	1.50

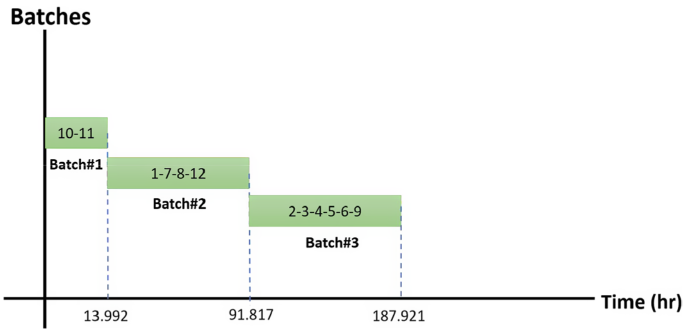


Fig. 2 Gantt chart for the optimum solution of the single AM machine scheduling problem (Kucukkoc 2019)

assembly operations and 2D assignment of parts to batches, the optimum makespan was found as 187.921 h by Kucukkoc (2019). We use Gantt charts to illustrate batches and parts produced in each batch. Gantt charts are mainly used in scheduling problems for illustrating which job is done with which machine in a timeline.

Firstly, we solve the problem without 2D assignment of parts in batches (MIP1 model) then we solve the same problem with 2D assignment of parts in batches (MIP2 model). As seen from Fig. 2, there are 3 batches in the optimum schedule. Figure 2 shows that the first batch includes part 10 and part 11. The completion time of the first batch in Fig. 2 is 13.992. If we solve the same problem with assembly operations (without 2D assignment of parts) given in Table 2 and Fig. 1, the completion time of the latest batch is found as 188.867 h and the

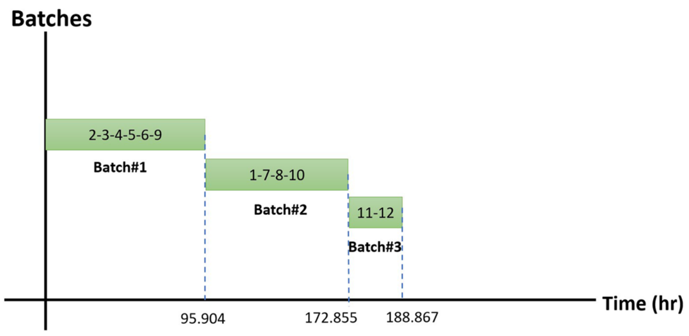


Fig. 3 Gannt chart of the optimum solution of single AM machine scheduling problem with assembly operations without 2D assignment of parts to batches

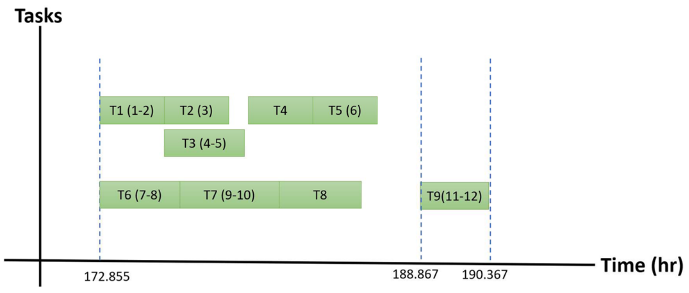


Fig. 4 Gantt chart of tasks after batch processing in single AM machine without 2D assignment of parts to batches

optimum schedule has 3 batches as seen in Fig. 3. The small numerical example is solved with the Cplex solver in GAMS 28.2 software on a standard desktop having an Intel i5 10th generation CPU with 4.2 GHz and 8 GB RAM. The Gantt chart for assembly operations after batch processing in a single AM is given in Fig. 4.

As seen from Fig. 3, the last batch completion time is 187.921 h but the start time of the assembly tasks T1 and T6 are 172.855 h because their needed jobs are completed in the second batch so assembly operations of products may start before the last batch's completion. Only task 9 waits for the last batch completion because parts 11 and 12 are processed in the last batch. The makespan of the schedule is 190.367 h. Assembly task operations after batch processing of AM machine may affect the schedule and batches in the machine as seen when we compare the schedules in Figs. 2 and 3.

Considering parts' 2D placement to baches, we solve the same problem with our MIP2. Since the data of the small example doesn't include lengths and widths of parts, we assume that $dh_i = dw_i = \sqrt{a_i} \forall i$ and the width and length of the machine tray are assumed as $WMA = HMA = \sqrt{MA}$. By using these, we solve the same problem with MIP2. As expected, the 2D placement of parts increases the

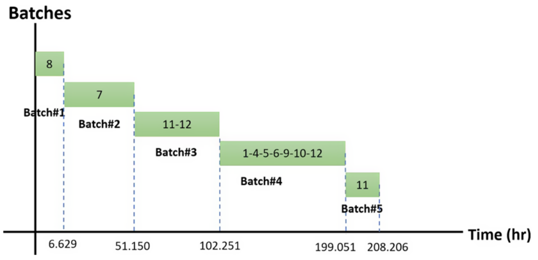


Fig. 5 Gannt chart of the optimum solution of single AM machine scheduling problem with assembly operations and the 2D assignment of parts to batches

number of batches so the makespan is increased. As seen from Fig. 5, there are 5 batches in the optimum schedule for the problem with the 2D assignment of parts. Figure 5 shows that the first batch includes only part 8. The completion time of the first batch in Fig. 5 is 6.629. The completion time of the latest (the fifth) batch is found as 208.206 h and the optimum schedule has 5 batches as seen in Fig. 5. The Gantt chart for assembly operations after batch processing with the 2D assignment of parts in a single AM is given in Fig. 6.

As seen from Fig. 5, the last batch completion time is 208.206 h but the start times of the assembly tasks T6 and T1 are 51.150 h and 199.051 h. Task T6 starts earlier than task T1 because part 7 is produced in the second batch on 51.150. Task T1 has to wait until the fourth batch is completed on 199.051. Even the most of the parts are produced until the fifth batch, the last task T9 needs part 11 to be done so the start time of task T9 is the completion time of the fifth batch that includes only part 11. As seen from Fig. 6, the makespan of the schedule is 209.709 h and it is more than 20 h than the makespan of the problem that does not consider the 2D assignment of parts. Considering 2D assignment of parts to batches increases the required number of batches and completion time for the schedule.

The problem without assembly operation is similar to single machine batch processing problems. With the notation of Kucukkoc (2019), the problem without assembly operations can be illustrated as $1|batch\{AM\}|Cmax$ and this problem has differences from classical single machine batch problems because each part produced in a batch has effects on processing time considering its three dimensions (height, width, and depth). The problem $1|batch\{AM\}|Cmax$ without assembly operation is strongly NP-Hard. We can define the problem with the notation of $1|batch\{AM\}, ASSEM|Cmax$.

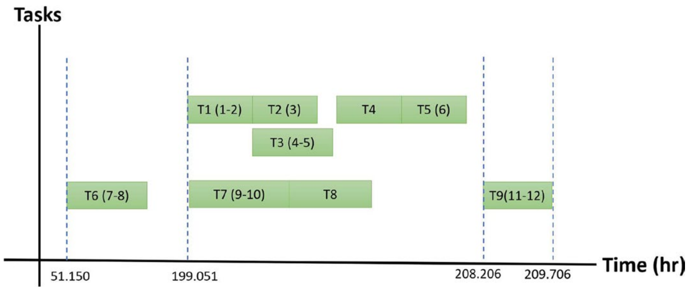


Fig. 6 Gantt chart of tasks after batch processing in single AM machine with the 2D assignment of parts to batches

Taking account of assembly operations into the problem after AM batch scheduling increases the complexity of the problem but it is a real-life problem. For instance, let assume a healthcare 3D printing company that produces prosthetic arms or hands. There are so many parts for a 3D printed prosthetic arm to be merged or assembled to build a prosthesis after the AM machine's batch processing. As another example, 3D printed shoe producers need to utilize batch operations considering assembly operations done after batch processing. Examples for AM machines can be diversified.

3 Lower bound of the proposed problem

As the introduced problem is a new scheduling problem and it is complex, we need a lower bound for the problem before introducing the proposed heuristic method. Since the problem is like the bin packing problem. For a bin packing problem, the lower bound for the total number of batches is simply obtained by dividing the total sum of part's weights (lengths or processing times) by the batch capacity (Martello and Toth 1990). This lower bound presents an averagely required number of batches. We use a similar approach for obtaining the lower bound of the proposed problem. Dealing with a batch scheduling problem, the first improvement should be made for the number of batches because there is setup time needed to make the machine ready for processing. Considering the similarity of the problem with the bin packing problem, if we divide the total area of the parts by the area of the machine tray, then we have the least number of batches for the problem as follow:

$$NB = \left\lfloor \frac{\sum_{i=1}^{n} a_i}{MA} \right\rfloor \tag{35}$$

where *NB* is the number of batches for the lower bound for the proposed problem. Since the machine tray is the major limitation for batch processing, we round down $\frac{\sum_{i=1}^{n} a_i}{MA}$ value to get its nearest and smallest integer value as the lower bound of the total number of batches for the problem. The remaining calculations in the lower bound for volumes, heights, batch processing times, and completion times of finished goods are considered with average or minimum values. After determining the number of batches for the lower bound, the average volume value *AV* for each batch can be calculated as follows:

$$AV = \frac{\sum_{i=1}^{n} v_i}{NB} \tag{36}$$

The maximum height is in the batch is a factor for batch processing time calculation in Eq. (1), so we can use the minimum height value *AH* of parts for the lower bound as follows:

$$AH = \min\{h_i\} \tag{37}$$

Then, the minimum batch processing time *AB* can be calculated as follows:

$$AB = SET + VT \times AV + HT \times AH \tag{38}$$

The average time for completing the desired finished goods can be calculated as follows:

$$AT = \frac{\sum_{k=1}^{K} t_k}{m} \tag{39}$$

The lower bound C_{max}^{LB} of the proposed problem is calculated as follows:

$$C_{max}^{LB} = NB \times AB + AT \tag{40}$$

For numerical example in previous section; NB=2, $= AV 2486.82$ cm^3, $= AH$ 2.18 cm^2, AB= 79.27921 h, AT= 4.4 h and C_{max}^{LB}=158.5584 h. The optimum C_{max}^{*} values were found as 190.367 h and 209.706 h for the numerical example without 2D assignment of parts and with the 2D assignment of parts, respectively. We can deduce that the lower bound calculation method may produce good results for comparison when not considering the 2D assignment of parts. In the case of considering the 2D assignment of parts to batches, the proposed lower bound method cannot produce a realistic lower bound for the problem because the 2D placement of parts certainly requires more batches.

4 Heuristics for single AM machine scheduling with assembly operations

Even problem $1|batch\{AM\}|Cmax$ without assembly operation is strongly NP-Hard, considering assembly operations in the problem increase the complexity of the problem. Decision-makers or real-life schedulers need a time-efficient and simple approach to deal with the problem. Therefore, we propose a simple heuristic method that is a modified first fit increased (FFI) heuristic for bin packaging problems. Assembly operations after batch processing have effects on the schedule and batches in the single AM machine. To determine which task will be produced firstly, we suggest a way to calculate the importance weights of parts considering the latest tasks after batch processing as follows:

$$I_i = \left(1 + \sum_{k=1}^{K} \left[\sum_{l=1}^{K} Prec_{kl} \times \left(K - \sum_{l=1}^{K} Prec_{lk}\right)\right] \times B_{ik}\right) \times a_i \ \forall i \tag{41}$$

where I_i is the importance weight of part i for all tasks after the single AM machine's batch processing. If I_i value of part i is less than others then part i should be processed before other tasks whose importance degrees are greater than part i. The formula in Eq. (41) considers each part's (i) area a_i, the assembly task k where part i is required to do, precedence relations of task k with other tasks (l). If parts i and j belong to the task that is the predecessor of other tasks and this task has most of the precedence relations, these parts' importance weights will be equal with an assumption$=a_i a_j$. If $a_i < a_j$ then the importance degree of part i will be less than part j. In the case of $a_i = a_j$, part i belongs to task k and part j belongs to task l; if task k has more precedence relations than task l, then the importance weight of part i is less than the importance weight of part j. With this indicator, the proposed heuristic algorithm will decide which parts will be assigned firstly by considering their importance weights. The smaller importance weight is an indicator of tasks having smaller area values and part-task relations that are critical to complete the final product. For the numerical example introduced before, I_i values of parts are given in Table 3.

By using I_i values, we propose an approximation algorithm for the problem in Algorithm 1. We call this algorithm FFI-1.

Algorithm 1: FFI-1

```
Step 1. Order parts in increasing order of their importance weights and assign all parts to set
        $\varphi$
Step 2. Assign parts into batches using the following rule. Create the first empty batch $b_1$ and
        put the first part from set $\varphi$ in it and remove the assigned part from set $\varphi$. Then continue
        to assign first possible part to the batch $b_1$ until there is no possible part to assign to
        batch $b_1$. If there is no possible part remaining in $\varphi$, create the second empty batch $b_2$
        and assign the first possible part to the batch $b_2$ until there is no possible part to assign
        to batch $b_2$. Repeat this step until there is no unassigned part left.
Step 3. Calculate each batch's processing time and completion times of batches.
Step 4. Calculate start times of tasks considering completion times of them in single AM
        machine and task precedence relations.
Step 5. Calculate completion times of tasks considering task durations and task precedence
relations.
Step 6. Calculate the makespan.
```

In Step 2 of the proposed heuristic in Algorithm 1, parts are tried to assign to batches if the machine tray's capacity (batch capacity) is not exceeded when the decision-maker does not consider the 2D assignment of parts to batches. To check whether all parts in a batch can be placed on a single machine tray, our FFI-1 algorithm uses an MIP model that minimizes the total number of batches (min $\sum_j Z_j$). We call this MIP model MIP3 and it requires 2D assignment constraints. Constraints (14–18), constraints (22–23), and constraints (28–34) are subjected to the objective function of min $\sum_j Z_j$. If the objective function value is more than 1, then the parts in the batch cannot be placed in the machine tray. In this case, the FFI-1 algorithm will pass the current iteration and check other batch-part pairs.

For an illustration of the proposed heuristic, the numerical example without considering 2D assignment is solved, and results in Table 4 are obtained. Table 4 shows the comparison of solutions for the proposed FFI-1 and the MIP model. As seen in Table 4, each solution has three batches. There are 2.50 h between makespan values of solutions of the proposed FFI-1 and MIP model and batch schedules of solutions are different. The MIP model coded in the Gams software is executed until its elapsed time is reached 1800 s. The proposed FFI-1 just solves the problem in less than one second. The 2.50 h makespan difference between two solution approaches can be easily eliminated, if we apply a local search to the solution obtained from the

Table 3 Importance weights of numerical example

Part (i)	1	2	3	4	5	6	7	8	9	10	11	12
I_i	209.06	550.11	212.67	895.77	511.65	500.2	435.66	84.97	434.43	1103.58	1783.4	1340.8

Table 4 Comparison of solutions of the proposed heuristic and MIP for the numerical example without considering the 2D assignment of parts

Method	# of batches	Completion time of the latest batch	The makespan
MIP	3	188.867 h	190.367 h
FFI-1	3	192.7664 h	194.2664 h
FFI-1+LS	3	188.867 h	190.367 h

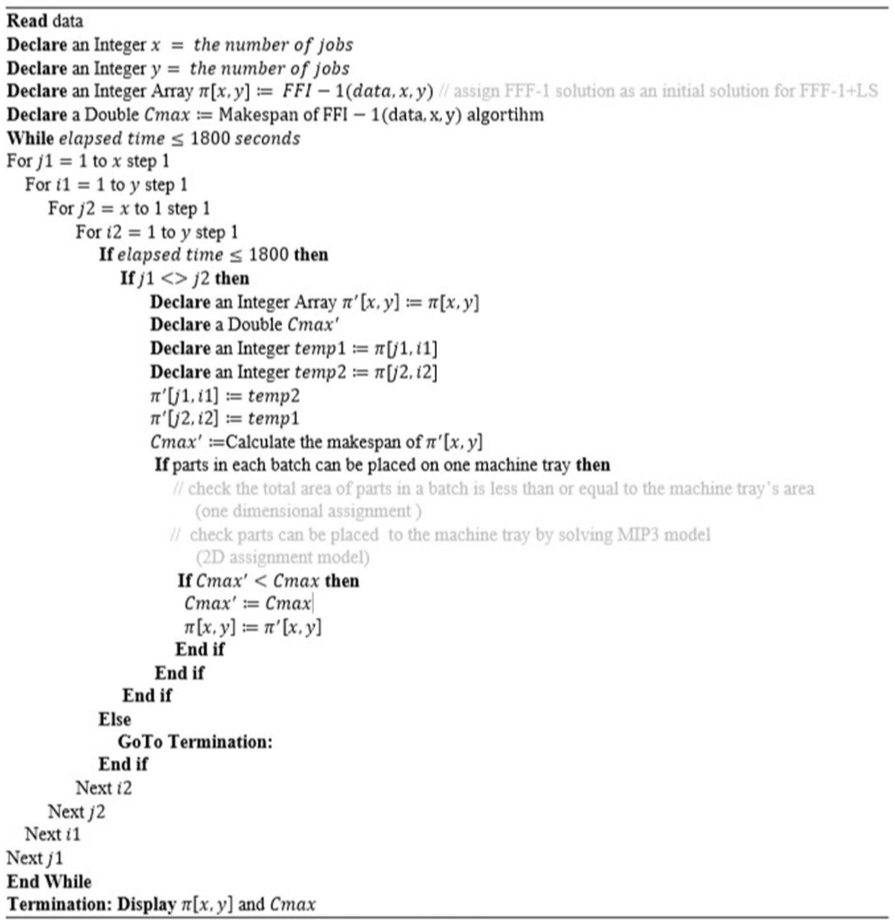
proposed FFI-1. In Algorithm 2, we express the initial solution of algorithm FFI-1 with an integer array ($\pi[x, y]$) where row index ($j \in \{1, 2, .., x\}$) expresses batches and column index ($i \in \{1, 2, .., y\}$) expresses positions available in the batch. If part 2 is assigned to position i of the batch j, then $\pi[j, i] = 2$. If no job is assigned to the position i of the batch j, then $\pi[j, i] = 0$. By using this array structure, we propose a local search mechanism that swaps jobs in different two positions of different batches. If one position of two selected positions for swapping has a job at least, then the swap operation takes place. After swapping, the new solution's makespan considering the assembly operations after batch processing is recalculated. If the new makespan value is better than the previous best makespan, the best solution in the memory is replaced with the new solution. We adopt this local search mechanism to our proposed FFI-1 algorithm and call this algorithm FFI-1+LS. In each swapping, the algorithm checks whether all new parts in each batch are fitted to the machine tray. If the decision-maker considers the 2D assignment of parts to batches, then algorithm FFI-1+LS uses the MIP3 model to check whether the parts are fitted to the single batch. If the decision-maker does not consider the 2D assignment of parts to batches, then algorithm FFI-1+LS just checks the total area of the parts is less than or equal to the area of the machine tray. The pseudo-code of algorithm FFI-1+LS is given in Algorithm 2. In order to observe the impact of the proposed local search mechanism on the FFI-1 algorithm, we solve the example problem with our proposed algorithm FFI-1+LS. The results are seen in Table 4 for the problem without considering the 2D assignment of parts. The 2.50 h gap between solutions of the proposed FFI-1 and the MIP model is eliminated with the local search mechanism in the new algorithm. Thus, the solution is found by algorithm FFI-1+LS is equal to the optimal solution that is found by the MIP model coded in the Gams software using the Cplex solver. The same numerical example is also solved with consideration of the 2D assignment of parts to the batches. The results are given in Table 5 for the numerical example with consideration of the 2D assignment of parts to the batches. The MIP model coded in the Gams software is executed until its elapsed

Table 5 Comparison of solutions of the proposed heuristic and MIP for the numerical example with considering the 2D assignment of parts

Method	# of batches	Completion time of the latest batch	The makespan
MIP	5	208.206 h	208.706 h
FFI-1	4	208.095 h	208.595 h
FFI-1+LS	4	208.095 h	208.595 h

time is reached 1800 s. The proposed FFI-1 just solves the problem in less than five seconds. Also, algorithm FFI-1+LS with MIP3 model is executed until its elapsed time is reached 1800 s. FFI-1 found a better solution than the MIP model by using less time but its solution is not improved by algorithm FFI-I+LS with MIP3 because it may be the optimal solution.

Algorithm 2: FFI-1+LS



```
Read data
Declare an Integer $x$ = $the\ number\ of\ jobs$
Declare an Integer $y$ = $the\ number\ of\ jobs$
Declare an Integer Array $\pi[x,y] := FFI-1(data,x,y)$ // assign FFF-1 solution as an initial solution for FFF-1+LS
Declare a Double $Cmax$ := Makespan of FFI − 1(data, x, y) algortihm
While $elapsed\ time \leq 1800\ seconds$
For $j1 = 1$ to $x$ step 1
  For $i1 = 1$ to $y$ step 1
    For $j2 = x$ to 1 step 1
      For $i2 = 1$ to $y$ step 1
        If $elapsed\ time \leq 1800$ then
          If $j1 <> j2$ then
            Declare an Integer Array $\pi'[x,y] := \pi[x,y]$
            Declare a Double $Cmax'$
            Declare an Integer $temp1 := \pi[j1,i1]$
            Declare an Integer $temp2 := \pi[j2,i2]$
            $\pi'[j1,i1] := temp2$
            $\pi'[j2,i2] := temp1$
            $Cmax'$ :=Calculate the makespan of $\pi'[x,y]$
            If parts in each batch can be placed on one machine tray then
              // check the total area of parts in a batch is less than or equal to the machine tray's area
                 (one dimensional assignment )
              // check parts can be placed to the machine tray by solving MIP3 model
                 (2D assignment model)
              If $Cmax' < Cmax$ then
                $Cmax' := Cmax$
                $\pi[x,y] := \pi'[x,y]$
              End if
            End if
          End if
        Else
          GoTo Termination:
        End if
      Next $i2$
    Next $j2$
  Next $i1$
Next $j1$
End While
Termination: Display $\pi[x,y]$ and $Cmax$
```

5 Experimental results

In order to illustrate numerical examples for the problem, the dataset created by Li et al. (2015) is used. In Table 5, there are five different problems, and each problem's data such as areas, heights, and volumes are taken from the dataset of Li et al. (2015). Problem P1 in Table 6 includes the first 12 parts of the dataset and problem

Table 6 Numerical examples

Problem	#of parts	#of products	#of tasks
P1	12	2	9
P2	25	3	12
P3	50	4	15
P4	100	5	20
P5	150	8	30

P2 includes the first 25 parts of the dataset and this goes on for all problems. The numbers of desired finished products and the numbers of required tasks for assembling these products for all problems are also given in Table 6. Task durations and precedence diagrams are arbitrarily generated for problems. The heuristic coded with C++ in MS Visual Studio solves problems in reasonable short times. To solve the MIP3 model, we integrate the Cplex solver via GAMS 28.2 library into our proposed MIP3 model for checking whether all parts in a batch can be placed in a single batch considering their widths and lengths. The MIP models coded with GAMS 28.2 by using the Cplex solver are limited to 1800 s for all problems. Furthermore, the proposed FFI-1 algorithm and the local search mechanism are combined and executed until 1800 s for test problems. All solution approaches are executed in the same standard desktop having Intel i5 10th generation CPU with 4.2 GHz and 8 GB RAM.

The solutions obtained MIP model coded in GAMS 28.2 are also given in Table 7. C_{max}^{M} and NB^{M} are the makespan and number of batches of the problem solved by GAMS 28.2 by using Cplex within a time limitation. C_{max}^{H} and NB^{H} are the makespan and number of batches of the problem solved by the proposed heuristic coded with C++ in MS Visual Studio 2019. The machine for all problems in Table 6 is EOS M 400 that is a 3D printing machine of metal parts on an industrial scale. The parameters of that machine are taken from the dataset of Li et al. (2015) as $= VT$0.0740740740740741 h/cm^3, HT=1.4 h/cm, SET= 2 h and MA=1600 cm^2.

Without considering the 2D assignment of parts to batches, the proposed heuristic FFI-1 solves all test problems in less than 1 s. Furthermore, if we compare the lower bounds of the problems with solutions of the proposed heuristic, the results

Table 7 Results of the proposed methods within the time limit for numerical examples without considering the 2D assignment of parts to batches

	Lower Bound		MIP		Heuristic FFI-1		Heuristic FFI-1+LS	
Problem	NB	C_{max}^{LB}	NB^{M}	C_{max}^{M}	NB^{H}	C_{max}^{H}	NB^{HLS}	C_{max}^{HLS}
P1	3	2566.730	4	**2661.485**	4	2693.349	4	**2661.485**
P2	7	7837.744	8	**8075.982**	9	8163.466	8	8076.482
P3	14	12,710.063	17	13,164.881	15	13,325.617	16	**13,215.407**
P4	28	27,122.823	30	28,333.277	30	28,399.157	31	**28,254.765**
P5	40	38,611.230	42	40,410.344	42	40,422.636	42	**40,267.110**

The best solutions are marked with bold font of both methods for problems are close. For test problem P1, lower bounds for the number of batches and makespan are 3 and 2566.730 h. For the same problem, the proposed heuristic FFI-1 presents 4 batches and 2693.349 h makespan. The solution elapsed time for the proposed MIP model coded in Gams with Cplex solver is limited to 1800 s. The best MIP solutions obtained from Cplex solver within 1800 s for problems are given in Table 6. Without the local search mechanism illustrated in the previous section, the FFI-1 algorithm gives promising solutions until 1 s that is a reasonable time duration comparing with 1800 s. We combine the FFI-1 algorithm and the local search mechanism in a new algorithm FFI-1+LS and execute this new algorithm until 1800 s. The makespan values for problems obtained from FFI-1+LS are given in the last column of Table 7. The best makespan values are marked with bold font in Table 7 and it is easily seen from Table 7, four of the best makespan values are found by algorithm FFI-1+LS. Except for problem P2, algorithm FFI-1+LS presents better solutions for test problems. The local search mechanism increases the computational time of algorithm FFI-1 but it also improves the solution quality of the algorithm comparing with the MIP model that is an exact solution technique. Even algorithm FFI-1 without LS doesn't present the best results, its solution quality is quite close to other algorithms' solution quality. If we focus just on computational times of solution approaches, algorithm FFI-1 just consumes less than one second per problem. The optimal results for the P1 problem are found as 7 batches and 2661.485 h makespan. As understood, there are small differences among the makespan values of the proposed FFI-1 and the best makespan values marked with bold font.

If we only compare the results of the proposed FFI-1, FFI-1+LS and MIP model with lower bounds of problems, Eqs. (42) and (43) help us as follows:

$$RPD_{Cmax} = \frac{C_{max}^{M} - C_{max}^{LB}}{C_{max}^{LB}} \tag{42}$$

$$RPD_{NB} = \frac{NB^{M} - NB}{NB} \tag{43}$$

where RPD_{Cmax} is the relative percentage derivation from well-known makespan value and RPD_{NB} is the relative percentage derivation from the well-known number of batches. For all problems, the average RPD_{Cmax} value of the MIP model is 3.89% and the average RPD_{NB} value of the MIP model is 16.24%. For all problems, the average RPD_{Cmax} value of the proposed algorithm FFI-1 is 4.67% and the average RPD_{NB} value of the proposed algorithm FFI-1 is 16.24%. For all problems, the average RPD_{Cmax} value of the proposed algorithm FFI-1+LS is 3.84% and the average RPD_{NB} value of the proposed algorithm FFI-1+LS is 15.52%. Considering both performance criteria for the makespan and the total number of batches, the algorithm FFI-1+LS is the best alternative in case of non-considering 2D assignment of parts to batches.

With considering the 2D assignment of parts to batches, the same five problems were solved by the proposed MIP model and heuristics. Since Li et al. (2015) did not state lengths and widths of parts, we assume that $dh_i = dw_i = \sqrt{a_i} \forall i$ and the width and length of the machine tray are assumed as $WMA = HMA = \sqrt{MA}$ for problems. As expected, considering 2D assignments of parts to batches increases the number of required batches. As termination criterion, we used the same predetermined elapsed time (1800 s) for all solution approaches. As seen in Table 8, algorithm FFI-1+LS presents the best solutions for all problems. The best solutions are marked with bold font in Table 8. Since the lower bound calculation depends on the one-dimensional assignment of parts to batches, the calculated number of batches of each solution approach is quite more than NB. We used the same comparison methods in Eqs. (42) and (43). Since the proposed MIP model cannot solve test problem P5 until 1800 s, we calculated RPD_{Cmax} and RPD_{NB} values for the first four problems. The average RPD_{Cmax} value of the MIP model is 6.64% and the average RPD_{NB} value of the MIP model is 109.52%. The average RPD_{Cmax} value of the proposed algorithm FFI-1 is 6.23% and the average RPD_{NB} value of the proposed algorithm FFI-1 is 85.42%. The average RPD_{Cmax} value of the proposed algorithm FFI-1+LS is 6.03% and the average RPD_{NB} value of the proposed algorithm FFI-1+LS is 81.85%. RPD_{NB} values of solution approaches in case of considering 2D assignment of parts to batches are much more than RPD_{NB} values in Table 7 because taking only area constraint into account ignores two-dimensional placement of parts to baches and this increases the required number of batches. Considering both performance criteria for the makespan and the total number of batches, the algorithm FFI-1+LS is the best alternative in the case of considering the 2D assignment of parts to batches.

The lower bound of the problems helps us to compare the performance of any method for the solution of the problem. Using these lower bounds and solutions obtained from the proposed heuristic, we can say that the proposed FFI-1 presents good solutions at extremely reasonable times for all test problems. If we combine a local search mechanism with the proposed algorithm, then its performance outperforms the MIP model when the computational time is limited until 1800 s for solution approaches. The proposed heuristic is a modified version of an existing method called the FFI heuristic. We only use importance weights for finding the most fitted part among unassigned parts. Therefore, we may state that we propose an MIP (it is

Table 8 Results of the proposed methods within the time limit for numerical examples with considering the 2D assignment of parts to batches

	Lower Bound		MIP		Heuristic FFI-1		Heuristic FFI-1+LS	
Problem	NB	C_{max}^{LB}	NB^{M}	C_{max}^{M}	NB^{H}	C_{max}^{H}	NB^{HLS}	C_{max}^{HLS}
P1	3	2566.730	**5**	**2707.665**	5	2712.891	**5**	**2707.665**
P2	7	7837.744	15	8261.996	14	8293.208	**13**	**8249.537**
P3	14	12,710.063	30	13,670.429	25	13,539.524	**25**	**13,536.000**
P4	28	27,122.823	68	29,319.245	**55**	**28,989.787**	**55**	**28,989.787**
P5	40	38,611.230	*	*	82	41,412.104	81	**41,368.692**

*Cplex Solver in GAMS 28.2 could not find a solution until 1800 s

The best solutions are marked with bold font an exact method but it requires an extremely high time requirement for a solution) for the problem and compared it with a modified state-of-art method. The MIP is an exact method for the problem but the required time for finding an optimal is so high. The MIP was modeled for problems with different input sizes and solved with a strong commercial solver. Although the commercial solver for MIP could find a solution within 1800 s, the proposed algorithm FFI-1 solved all test problems in a reasonable time second and presented fair enough solution quality for all cases. As understood from this experiment, the complexity of the problem is so high for the commercial solver but algorithm FFI-1 algorithm produces very promising solutions in a reasonable time. When algorithm FFI-1 is combined with the proposed local search, its performance outperforms the MIP model.

6 Conclusion and future researches

As different from reported studies from the literature, both scheduling problems are investigated simultaneously in this study. The reported studies about AM machine scheduling have mainly focused on AM machines with different capacities and minimization on a predetermined objective such as total production cost, makespan, and total tardiness. Furthermore, solution approaches in the literature are mainly evolutionary algorithms such as a genetic algorithm for AM machine scheduling problems without considering assembly operations of parts. For AM machine scheduling problems reported in the literature, there are some limitations such as the machine tray's capacity and batch processing time determined by using the total volume and heights of assigned parts to that batch. This study investigates a new problem that includes batch scheduling of parts produced in the single additive manufacturing machine and then parts are assembled for desired finished goods for costumers. In view of scheduling problems, the proposed problem includes single machine batch scheduling and assembling operation scheduling. By combining these two different scheduling problems, we propose a mixed-integer nonlinear programming model for the problem. Furthermore, a lower bound calculation method for makespan and the number of batches is introduced for the problem. Then a fast heuristic FFI-1 combining the first fit increasing method and the importance weights of parts is introduced for the problem. Furthermore, a simple local search mechanism is proposed to increase the solution quality of the proposed fast heuristic. We examined two cases; one-dimensional assignment of parts (considering only the total area of parts in each batch) and 2D assignment of parts to batches. Experimental results of some generated test problems indicate that the proposed heuristic FFI-1 present efficient solutions in an extremely reasonable time. Even the mathematical model coded in a commercial solver presents slightly better solutions, it needs more computational time. Thus, we combined a local search mechanism to algorithm FFI-1+LS and executed it until the same predetermined termination criterion that the MIP model has. For all cases, experimental results with the same problems show that the proposed FFI-1+LS outperforms the MIP model coded in a commercial solver. For future research, the problem can be extended into parallel additive manufacturing machine environments where produced parts should be assembled after batch scheduling. Other performance criteria such as earliness/tardiness, lateness, and the sum of completion times can be investigated for the problem. Furthermore, precedence relations among jobs in AM machines can be considered as a constraint for the problem. Metaheuristic algorithms considering parallel possible assembly patterns can also be developed for the introduced problem and the proposed heuristic can be compared with them in solution quality and execution time requirement.

Declarations

Conflict of interest The author declared that there is no conflict of interest.

References

Achillas C, Aidonis D, Iakovou E et al (2015) A methodological framework for the inclusion of modern additive manufacturing into the production portfolio of a focused factory. J Manuf Syst 37:328–339. https://doi.org/10.1016/j.jmsy.2014.07.014

Chergui A, Hadj-Hamou K, Vignat F (2018) Production scheduling and nesting in additive manufacturing. Comput Ind Eng 126:292–301. https://doi.org/10.1016/j.cie.2018.09.048

Dvorak F, Micali M, Mathieu M (2018) Planning and scheduling in additive manufacturing. Intel Artif 21:40–52. https://doi.org/10.4114/intartif.vol21iss62pp40-52

Fera M, Fruggiero F, Lambiase A et al (2018) A modified genetic algorithm for time and cost optimization of an additive manufacturing single-machine scheduling. Int J Ind Eng Comput 9:423–438. https://doi.org/10.5267/j.ijiec.2018.1.001

Kovalyov MY, Potts CN, Strusevich VA (2004) Batching decisions for assembly production systems. Eur J Oper Res 157:620–642. https://doi.org/10.1016/S0377-2217(03)00250-9

Kucukkoc I (2019) MILP models to minimise makespan in additive manufacturing machine scheduling problems. Comput Oper Res 105:58–67. https://doi.org/10.1016/j.cor.2019.01.006

Kucukkoc I, Li Q, He N, Zhang DZ (2018) Scheduling of multiple additive manufacturing and 3D printing machines to minimise maximum lateness. Twent Int Work Semin Prod Econ 1:237–247

Li Q, Kucukkoc I, Zhang DZ (2015) No title. Prod Plan Addit Manuf 3D Print (Dataset), ORE-Repository

Li Q, Kucukkoc I, Zhang DZ (2017) Production planning in additive manufacturing and 3D printing. Comput Oper Res 83:157–172. https://doi.org/10.1016/j.cor.2017.01.013

Li Q, Kucukkoc I, He N et al (2018) Order acceptance and scheduling in metal additive manufacturing: an optimal foraging approach. Twent Int Work Semin Prod Econ 1:1–11

Li Q, Zhang D, Wang S, Kucukkoc I (2019) A dynamic order acceptance and scheduling approach for additive manufacturing on-demand production. Int J Adv Manuf Technol. https://doi.org/10.1007/s00170-019-03796-x

Li X, Zhang K (2018) Single batch processing machine scheduling with two-dimensional bin packing constraints. Int J Prod Econ 196:113–121. https://doi.org/10.1016/j.ijpe.2017.11.015

Liao C-J, Lee C-H, Lee H-C (2015) An efficient heuristic for a two-stage assembly scheduling problem with batch setup times to minimize makespan. Comput Ind Eng 88:317–325. https://doi.org/10.1016/j.cie.2015.07.018

Lin BMT, Cheng TCE (2002) Fabrication and assembly scheduling in a two-machine flowshop. IIE Trans 34:1015–1020. https://doi.org/10.1023/A:1016190815843

Lin BMT, Cheng TCE, Chou ASC (2006) Scheduling in an assembly-type production chain with batch transfer. Omega 35:143–151. https://doi.org/10.1016/j.omega.2005.04.004

Loukil T, Teghem J, Fortemps P (2007) A multi-objective production scheduling case study solved by simulated annealing. Eur J Oper Res 179:709–722. https://doi.org/10.1016/j.ejor.2005.03.073

Luzon Y, Khmelnitsky E (2019) Job sizing and sequencing in additive manufacturing to control process deterioration. IISE Trans 51:181–191. https://doi.org/10.1080/24725854.2018.1460518

Martello S, Toth P (1990) Lower bounds and reduction procedures for the bin packing problem. Discret Appl Math 28:59–70. https://doi.org/10.1016/0166-218X(90)90094-S

Oh Y, Zhou C, Behdad S (2018) Production planning for mass customization in additive manufacturing: Build orientation determination, 2D packing and scheduling. In: Proceedings of the ASME design engineering technical conference

Ransikarbum K, Ha S, Ma J, Kim N (2017) Multi-objective optimization analysis for part-to-Printer assignment in a network of 3D fused deposition modeling. J Manuf Syst 43:35–46. https://doi.org/10.1016/j.jmsy.2017.02.012

Rudolph J-P, Emmelmann C (2017) A cloud-based platform for automated order processing in additive manufacturing. In: Procedia CIRP, pp 412–417

Tajbakhsh Z, Fattahi P, Behnamian J (2014) Multi-objective assembly permutation flow shop scheduling problem: a mathematical model and a meta-heuristic algorithm. J Oper Res Soc 65:1580–1592. https://doi.org/10.1057/jors.2013.105

Zhang J, Yao X, Li Y (2019) Improved evolutionary algorithm for parallel batch processing machine scheduling in additive manufacturing. Int J Prod Res. https://doi.org/10.1080/00207543.2019.1617447

Zhou L, Zhang L, Laili Y et al (2018) Multi-task scheduling of distributed 3D printing services in cloud manufacturing. Int J Adv Manuf Technol 96:3003–3017. https://doi.org/10.1007/s00170-017-1543-z

Authors and Affiliations

Oğuzhan Ahmet Arık [1]

✉ Oğuzhan Ahmet Arık
oaarik@nny.edu.tr

[1] Industrial Engineering Department, Nuh Naci Yazgan University, Kayseri 38170, Turkey