Minimisation of Maximum Lateness in Multiple Additive Manufacturing Machine Scheduling Problem: A MILP Model and GA Approach

Journal:	*Production Planning & Control*
Manuscript ID	TPPC-2018-0399
Manuscript Type:	Research paper for Regular Issue
Date Submitted by the Author:	19-Sep-2018
Complete List of Authors:	Kucukkoc, Ibrahim; Balikesir Universitesi, Department of Industrial Engineering
Keywords:	additive manufacturing, production planning, scheduling, MILP, genetic algorithm

Minimisation of Maximum Lateness in Multiple Additive Manufacturing Machine Scheduling Problem: A MILP Model and GA Approach

Ibrahim Kucukkoc

Department of Industrial Engineering, Balikesir University,
Cagis Campus, Balikesir, Turkey
ikucukkoc@balikesir.edu.tr
https://orcid.org/0000-0001-6042-6896

Abstract

Additive manufacturing is now increasingly utilised in industry especially when the batches are small but the customisation is high. The decisions on the allocation of orders to the machines used for additive manufacturing is important in terms of the completion time related measures (e.g. makespan, flow time) and delivery date related performance measures (e.g. total lateness, maximum lateness, number of tardy jobs), as well as the production cost. This research focuses on the production scheduling problem in a multiple additive manufacturing/3D printing machine environment. Parts have release dates and due dates. A genetic algorithm approach is developed to minimise the maximum lateness where there is more than one additive manufacturing machine with different capacity and speed specifications. The parts are allocated to these parallel additive manufacturing machines in batches considering their release dates, due dates as well as the resource constraints. The problem is defined and a mixed-integer linear programming model is developed. A genetic algorithm approach is also developed to solve the problem and its running mechanism is illustrated through a numerical example. Some improvements are also made to increase the performance of the proposed approach. The methodology developed in this paper can easily be adopted by practitioners to generate short term production schedules for additive manufacturing machines.

Keywords: Additive manufacturing, production planning, scheduling, MILP, genetic algorithm, 3D printing.

Minimisation of Maximum Lateness in Multiple Additive Manufacturing Machine Scheduling Problem: A MILP Model and GA Approach

Abstract

Additive manufacturing is now increasingly utilised in industry especially when the batches are small but the customisation is high. The decisions on the allocation of orders to the machines used for additive manufacturing is important in terms of the completion time related measures (e.g. makespan, flow time) and delivery date related performance measures (e.g. total lateness, maximum lateness, number of tardy jobs), as well as the production cost. This research focuses on the production scheduling problem in a multiple additive manufacturing/3D printing machine environment. Parts have release dates and due dates. A genetic algorithm approach is developed to minimise the maximum lateness where there is more than one additive manufacturing machine with different capacity and speed specifications. The parts are allocated to these parallel additive manufacturing machines in batches considering their release dates, due dates as well as the resource constraints. The problem is defined and a mixed-integer linear programming model is developed. A genetic algorithm approach is also developed to solve the problem and its running mechanism is illustrated through a numerical example. Some improvements are also made to increase the performance of the proposed approach. The methodology developed in this paper can easily be adopted by practitioners to generate short term production schedules for additive manufacturing machines.

Keywords: Additive manufacturing, production planning, scheduling, MILP, genetic algorithm, 3D printing.

1. Introduction

Additive manufacturing (AM), also known as 3D printing (3DP), is used to describe the technologies that build 3D objects by adding layer-upon-layer of material. It is now being increasingly utilised in industries requiring small-batch but high-customisation manufacturing. Among its variants, the metallic powder-bed fusion processes (such as Selective Laser Melting and Electron Beam Melting) have been widely adopted as an advanced direct rapid manufacturing method particularly in aerospace and orthopaedic implant sectors (Calignano et al., 2017; Schmidt et al., 2017).

The production with metallic powder-bed fusion process is job-based where the machine can handle one job at a time and the job can consist of a batch of parts which will be processed simultaneously (Kucukkoc et al., 2016). Both the production cost and the production time of an AM/3DP job dynamically depend on the combination of parts allocated to this job (Li et al., 2017). Therefore, the decision on the allocation of parts to the machines is crucial for the scheduling in AM/3DP production, while the uncertainties caused by variations in the production cost and time make it more challenging in decision-making. This is especially important when the delivery time related performance measures and cost-based objectives must be considered together.

The literature on the additive manufacturing technology is majorly focused on the process itself and the cost structures (Calignano et al., 2017), which are out of the scope of this paper. The study belongs to Li et al. (2017) aims to minimise the average cost per volume of material through a mathematical model and two basic heuristics, namely best-fit and adapted best-fit. The model proposed by Kucukkoc et al. 2016 aimed to maximise the area utilisation of AM/3DP machines. There is no work addressing the lateness of parts concerned to the completion times and due dates. Also, release dates of parts are also incorporated in that paper.

The considered production planning problem of AM/3DP machines in this paper is a kind of batch scheduling problem (Sule and Saxena, 2007) in such an environment with multiple machines. However, it is significantly important to note that the production time of jobs in AM/3DP production planning problem changes based on the total volume and maximum height of parts assigned to the job. The decision maker needs to make a decision based on the allocation of parts to jobs on machines with different specifications, e.g. speed, maximum area, maximum height, the time needed to set-up and cleaning etc. In that sense, this paper introduces the production planning of AM/3DP problem with the aim of minimising maximum lateness. The paper also contributes to the literature considering the part release dates. So that, a job containing a set of parts to be produced cannot start before the release date of any of those parts. Also, when a job consisting a set of parts is started, it is not possible to take out any of those parts until all parts finish. The first MILP model and the genetic algorithm (GA) approach, whose parameters have been tuned with preliminary tests, is proposed to solve the problem. Note that the detailed 2D or 3D nesting problem is not considered in this research.

This paper is organised as follows. The problem is defined and modelled mathematically via a mixed-integer linear programming (MILP) model in Section 2. The proposed GA is described illustratively through examples in Section 3. A medium sized numerical example consisting of three machines and eighteen parts are solved using the proposed approach in Section 4 and the paper is concluded in Section 5 with key research results and some future research directions.

2. Problem Statement

2.1. Definition of the Problem

AM/3DP problem aims to decide the allocation of part orders, where a part order (or part, shortly) is represented by p_i ($i = 1,2,\dots,ni$ and $i \epsilon I$), received from customers to job batches, represented by b_j ($j = 1,2,\dots,nj$ and $j \epsilon J$). Jobs are utilised on more than one AM/3DP machine, represented by k_m ($m = 1,2,\dots,nm$ and $m \epsilon M$) with different specifications. Parts have different specifications, i.e. height (h_i), area (a_i), volume (v_i), release date (r_i) and due date (d_i). Machines also have maximum supported production area (MA_m) and height (MH_m) specifications. Thus, it is required to ensure that the height of parts assigned to a machine must be smaller or equal to the maximum height supported by that machine. It is also necessary to ensure that the total area of parts assigned to a machine cannot exceed the production area supported by that machine. The aim is to minimise the maximum lateness. The lateness of a part (L_i) is the time difference between the completion time of a part (γ_{mji}) and its due date. As seen, a job is considered to be late if it is completed later than its due date.

γ_{mji} is characterised by the completion time of the job (CT_{mj}) in which the part is allocated. CT_{mj} is calculated accumulating the earliest start time of the job (σ_{mj}) and its production time (δ_{mj}) as seen in Equation (1).

$$CT_{mj} = \sigma_{mj} + \delta_{mj} \qquad (1)$$

where, $\sigma_{mj} = \max\{CT_{m(j-1)}, \max_{i \in I_{mj}}\{r_i\}\}$. The first expression ($CT_{m(j-1)}$) here corresponds to the completion time of the previous job on the same machine. If b_j is the first job ($j = 1$) on k_m, $CT_{m(j-1)}$ is considered to be zero as there is no job scheduled earlier than b_j. I_{mj} is the set of jobs allocated to b_j on k_m. This denotes that $\sigma_{mj} \geq r_i$ for all $i \in I_{mj}$. So that a job cannot start before the release date of any part to be produced in that job. δ_{mj} is calculated using Equation (2):

$$\delta_{mj} = SET_m + VT_m \cdot \sum_{i \in I_{mj}} v_i + HT_m \cdot \max_{i \in I_{mj}}\{h_i\} \tag{2}$$

where, SET_m is the set-up time for k_m, VT_m is the time needed to release per volume material for k_m, and HT_m is the accumulated interval time per unit height for k_m. Thus, the production time of a job basis on the total volume and maximum height of parts assigned to that job as well as the set-up time of the machine on which the job is scheduled.

One of the basic assumptions of the problem is that, a job cannot stop until it is fully completed. Therefore, even a small and short part is completed before the completion of the whole job, it cannot be taken out until the job completely finishes. Also, it is not possible to add a part into a job after it starts. Another assumption is that all parts must be assigned to exactly one job. It is not possible to split parts or jobs on to more than one AM machine. It is also necessary to indicate that all AM machines work in parallel, independently from other(s). They may have different speed and require different amount of time for set-up.

2.2. Mathematical Model

This section provides a MILP model for the multiple additive manufacturing machine scheduling problem. The aim of the model is to minimise the maximum lateness of the parts, i.e. the positive difference between the completion time and due date.

The notation used in the mathematical model is presented as follows:

ni : Maximum number of parts

I : Set of parts

i : Part index, where $i = 1,2, \dots, ni$ and $i \epsilon I$

p_i : Part i

nj : Maximum number of jobs

J : Set of jobs

j : Job index, where $j = 1,2, \dots, nj$ and $j \epsilon J$

b_j : Job j

nm : Maximum number of machines

M : Set of machines

m : Machine index, where $m = 1,2, \dots, nm$ and $m \epsilon M$

k_m : Machine m

h_i : The height of p_i

a_i : The area of p_i

v_i : The volume of p_i

r_i : The release date of p_i

d_i : The due date of p_i

MA_m : The maximum supported production area of k_m

MH_m : The maximum supported height of k_m

CT_i : The completion time of a p_i

d_i : The due date of a p_i

L_i : The lateness of p_i

σ_{mj} : The earliest start time of b_j on k_m

δ_{mj} : The production time of b_j on k_m

CT_{mj} : The completion time of b_j on k_m

γ_{mji} : The completion time of p_i assigned to b_j on k_m

SET_m : The set-up time for k_m

VT_m : The time needed to release per volume material for k_m

HT_m : The accumulated interval time per unit height for k_m

$maxh_{mj}$: The maximum height of the parts assigned to b_j on k_m

X_{mji}: 1, if p_i is assigned to b_j on k_m; 0, otherwise

Z_{mj}: 1, if at least one job is assigned to b_j on k_m; 0, otherwise

The objective function and constraints of the MILP model are presented as follows.

Objective Function

$$\text{Minimize} \quad max_{i\in I}\{L_i\} \tag{3}$$

Subject to

$$\sum_{m=1}^{nm}\sum_{j=1}^{nj} X_{mji} = 1, \quad \forall i \in I \tag{4}$$

$$\sum_{i=1}^{ni} X_{mji}\, a_i \leq MA_m, \quad \forall m \in M, j \in J \tag{5}$$

$$X_{mji}\, h_i \leq MH_m, \quad \forall m \in M, j \in J, i \in I \tag{6}$$

$$CT_{mj} \geq \sigma_{mj} + \delta_{mj}, \quad \forall m \in M, j \in J \tag{7}$$

$$\sigma_{mj} \geq CT_{m(j-1)}, \quad \forall m \in M, j \in J \tag{8}$$

$$\sigma_{mj} \geq r_i\, X_{mji}, \quad \forall m \in M, j \in J, i \in I \tag{9}$$

$$\delta_{mj} \geq SET_m Z_{mj} + VT_m \sum_{i=1}^{ni} X_{mji}\, v_i + HT_m maxh_{mj}, \quad \forall m \in M, j \in J, i \in I \tag{10}$$

$$Z_{mj} \geq X_{mji}, \quad \forall m \in M, j \in J, i \in I \tag{11}$$

$$maxh_{mj} \geq h_i X_{mji}, \quad \forall m \in M, j \in J, i \in I \tag{12}$$

$$\gamma_{mji} \geq CT_{mj}, \quad \forall m \in M, j \in J, i \in I \tag{13}$$

$$L_i \geq \gamma_{mji} - d_i, \quad \forall m \in M, j \in J, i \in I \tag{14}$$

$$X_{mji}, Z_{mj} \in \{0,1\} \tag{15}$$

The objective function represented with Equation (3) aims to minimize the maximum lateness of the parts. Equation (4) ensures that every part is assigned to a job on a machine exactly once. Equations (5) and (6) satisfy the capacity constraints (i.e. total area and maximum height supported by the machines). Equation (7) helps to calculate the completion times of jobs on machines. Specifically, the completion time of a job on a machine must be higher than or equal to the sum of the earliest start time of that job and its production time. Equation (8) ensures that the jobs can only start after the completion of the previous job on the same machine while Equation (9) guaranties that the jobs can start after the release date of parts assigned to that job. Equation (10) calculates the production time of the jobs (which is used in Equation (7) as well). Equation (11) confirms that a job is utilised only if at least one job is assigned to that job. The maximum height of the parts in a job is calculated using Equation (12) and the completion times of the jobs are determined using Equation (13). The lateness of a part is calculated via Equation (14) and finally the sign constraints are represented in Equation (15).

3. The Proposed GA

GA is a well-known nature-based optimisation technique inspired by the natural selection process in the nature (Goldberg and Holland, 1988). GA was selected in this research due to its successful implementations on many combinatorial optimisation problems from manufacturing to transportation and design. It is a powerful method for solving sophisticated problems, and has been applied to many scheduling problems, see for example Cheng et al. (1996) for a survey of job-shop scheduling problems using genetic algorithms; Gonçalves et al. (2005) and Pezzella (2008) for GAs developed for job-shop and flexible job-shop scheduling problems; Kucukkoc and Zhang (2015) and Kucukkoc and Zhang (2016) for GA and hybrid GA algorithms for assembly line balancing problems; and Woo et al. (2017) for GA developed to solve parallel machine scheduling problem.

The general outline of the proposed GA is depicted in Figure 1. The algorithm starts with generating $popSize$ number of chromosomes to form the initial population. Each chromosome is made up with a randomly permuted string of numbers corresponding to part numbers. Therefore, the length of the chromosome is equal to the number of parts. All individuals in the population are evaluated one-by-one using the procedure which will be given in Section 3.1. Genetic operators are applied to the chromosomes selected through tournament selection, with a tournament size of $popSize/6$. After the application of the genetic operators, the new generation is formed and genetic operators are applied again. This cycle continues until the maximum number of iterations ($maxIter$) is exceeded with no improvement.

3.1. Decoding Procedure

The procedure used when evaluating the chromosomes is shown in Figure 2. As seen in the figure, chromosomes are decoded allocating parts to the machines in the order of their appearance on the chromosome. The decoding procedure starts with the first job on the first machine and assigns as many parts as possible starting from the first gene on the chromosome while the machine is high enough to produce the part and the remaining production area (RA_{mj}) is large enough. If any part is not assignable to a job on a specific machine, it is skipped and considered for the next job for the next machine. This procedure is continued until all parts have been assigned to a job.

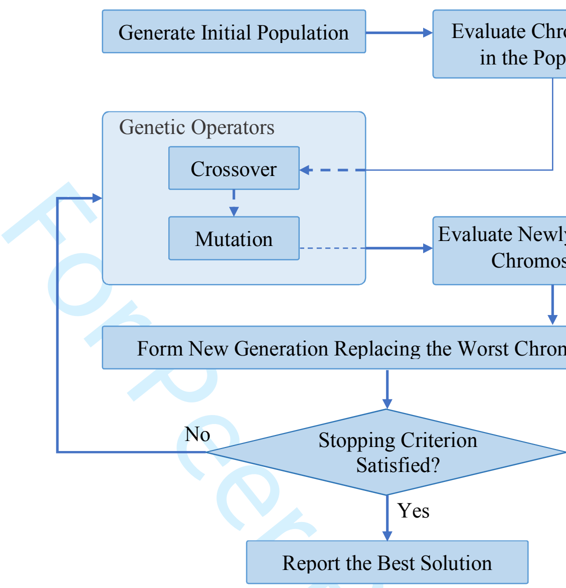


Figure 1. The general outline of the proposed algorithm

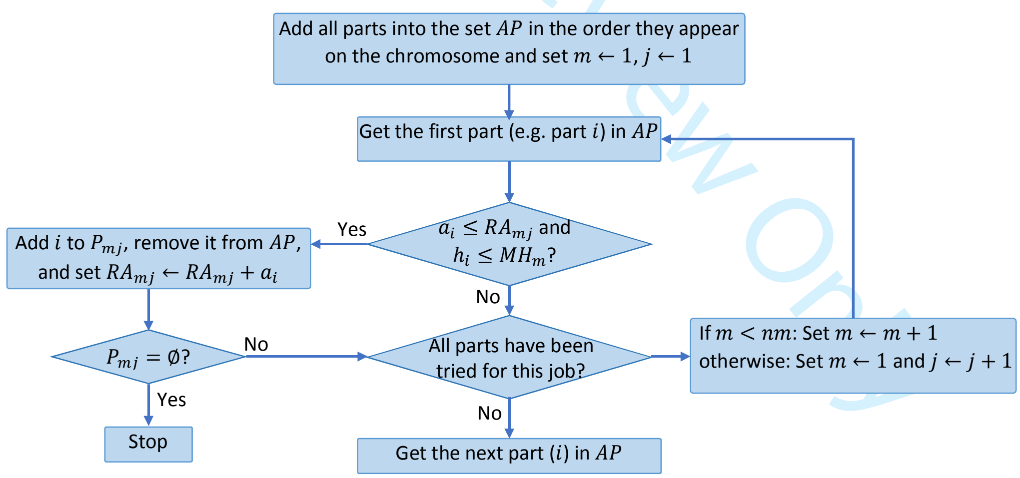


Figure 2. The flow chart of the decoding procedure

Let us assume an AM/3DP problem with two machines and ten parts, for which the detailed specifications are given in Table 1 and Table 2. Figure 3 presents a chromosome sample and illustrates its decomposition based on the decoding procedure considering the data for the numerical example given in Table 1 and Table 2. As seen in the figure, after the allocation of p_3, p_6 and p_1 to job 1 on machine 1 (i.e. $P_{11} = \{3,6,1\}$), none of the other parts can be assigned due to the insufficient remaining production area. Therefore, the machine is changed and p_4, p_2 and p_5 are assigned to the first job on machine 2. As the remaining production area (326.9 cm^2) is not large enough to assign p_8, it is skipped and p_7, p_{10} and p_9 are also assigned to the same job. Upon filling up the first two jobs on both of the machines, a new job is opened on the first machine again and p_8 is assigned. Please see Figure 4 for the steps of the decoding procedure in details. The σ_{mj}, δ_{mj} and CT_{mj} values of the jobs are also calculated and presented in the same figure.

Table 1. Specifications of the AM machines

Parameters	k_1	k_2
VT_m, the time consumption to form per unit volume	0.030864 (hour/cm^3)	0.030864 (hour/cm^3)
HT_m, the accumulated time per unit height	1 (hour/cm)	1 (hour/cm)
SET_m, the time consumption for setting up a new job	2 (hour)	1 (hour)
MH_m, the maximum height supported	32.5 (cm)	32.5 (cm)
MA_m, the maximum production area supported	625 (cm^2)	625 (cm^2)

Table 2. Part specifications

Part Index (i)	Height-*cm* (h_i)	Area-*cm^2* (a_i)	Volume-*cm^3* (v_i)	Release Date-*hr* (r_i)	Due Date-*hr* (d_i)
1	16.7	300.8	1573.8	6.3	305.9
2	8.8	152.8	421.5	7.3	282.2
3	20.3	19.5	147.8	9.8	378.3
4	7.4	84.2	285.2	20.9	214.7
5	27.3	61.1	583.3	36.5	149.0
6	25.8	299.3	3282.5	51.5	211.6
7	14.5	148.7	1265.5	56.3	240.4
8	3.5	376.4	723.3	69.9	576.2
9	20.4	20.5	278.5	75.0	330.9
10	23.3	91.1	1051.8	86.0	388.0

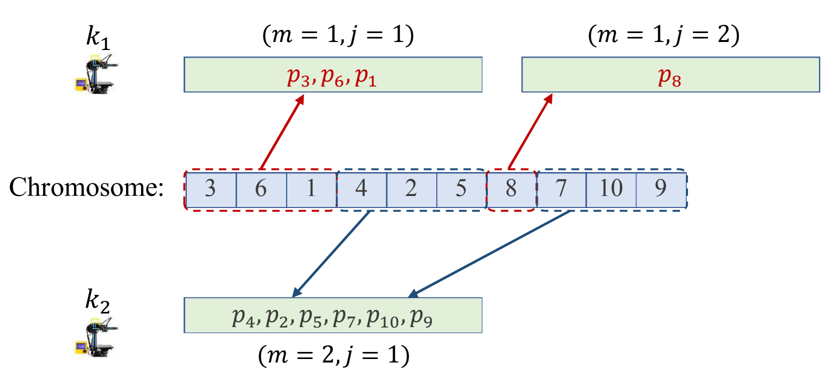


Figure 3. A chromosome sample and its decoding procedure

(m,j)	$CT_{m(j-1)}$	Remaining Area (RA_{mj})	Considered Part	h_i cm	a_i cm^2	Fits	Assigned Parts	Total Volume cm^3
(1,1)	0	625.0	3	20.3	19.5	**Yes**	{3}	147.8
		605.5	6	25.8	299.3	**Yes**	{3,6}	3430.3
		306.2	1	16.7	300.8	**Yes**	{3,6,1}	5004.1
		5.4	4	7.4	84.2	No	{3,6,1}	5004.1
		5.4	2	8.8	152.8	No	{3,6,1}	5004.1
		5.4	5	27.3	61.1	No	{3,6,1}	5004.1
		5.4	8	3.5	376.4	No	{3,6,1}	5004.1
		5.4	7	14.5	148.7	No	{3,6,1}	5004.1
		5.4	10	23.3	91.1	No	{3,6,1}	5004.1
		5.4	9	20.4	20.5	No	**{3,6,1}**	**5004.1**

$P_{11} = \{3,6,1\}$, $\delta_{11} = 2 + 0.030864 \cdot (5004.1) + 1 \cdot max\{20.3, 25.8, 16.7\} = 182.2465$ hr $\cong 182.2$ hr
$\sigma_{11} = \max\{CT_{10}, \max\{9.8, 51.5, 6.3\}\} = 51.5$ hr, $CT_{11} = \sigma_{11} + \delta_{11} = 51.5 + 182.2 = 233.7$ hr

(m,j)	$CT_{m(j-1)}$	Remaining Area (RA_{mj})	Considered Part	h_i cm	a_i cm^2	Fits	Assigned Parts	Total Volume cm^3
(2,1)	0	625.0	4	7.4	84.2	**Yes**	{4}	285.2
		540.8	2	8.8	152.8	**Yes**	{4,2}	706.7
		388.0	5	27.3	61.1	**Yes**	{4,2,5}	1290.0
		326.9	8	3.5	376.4	No	{4,2,5}	1290.0
		326.9	7	14.5	148.7	**Yes**	{4,2,5,7}	2555.5
		178.2	10	23.3	91.1	**Yes**	{4,2,5,7,10}	3607.3
		87.1	9	20.4	20.5	**Yes**	**{4,2,5,7,10,9}**	**3885.8**

$P_{21} = \{4,2,5,7,10,9\}$, $\delta_{21} = 1 + 0.030864 \cdot (3885.8) + 1 \cdot max\{7.4, 8.8, 27.3, 14.5, 23.3, 20.4\} = 148.2313$ hr $\cong 148.2$ hr
$\sigma_{21} = \max\{CT_{20}, \max\{20.9, 7.3, 36.5, 56.3, 86.0, 75.0\}\} = 86.0$, $CT_{21} = 86.0 + 148.2 = 234.2$ hr

(m,j)	$CT_{m(j-1)}$	Remaining Area (RA_{mj})	Considered Part	h_i cm	a_i cm^2	Fits	Assigned Parts	Total Volume cm^3
(1,2)	233.7	625.0	8	3.5	376.4	**Yes**	**{8}**	**723.3**

$P_{12} = \{8\}$, $\delta_{12} = 2 + 0.030864 \cdot (723.3) + 1 \cdot max\{3.5\} = 27.8239$ hr $\cong 27.8$ hr
$\sigma_{12} = \max\{CT_{11}, \max\{69.9\}\} = 233.7$ hr, $CT_{12} = \sigma_{12} + \delta_{12} = 233.7 + 27.8 = 261.5$ hr

Figure 4. The detailed decoding procedure of the sample chromosome

The calculations on the lateness values of parts are provided in Table 3. As the GA aims to minimise the maximum lateness, which can be formulated as $Min\ OBJ = \max_{i \in I}\{L_i\}$, the objective function value of the sample chromosome can simple be found as 85.2 hr.

Table 3. The lateness calculations for parts

(m,j)	CT_i (hr)	Part Index (i)	d_i (hr)	L_i (hr)
(1,1)	233.7	3	378.3	-144.6
		6	211.6	22.1
		1	305.9	-72.2
(2,1)	234.2	4	214.7	19.5
		2	282.2	-48.0
		5	149.0	85.2
		7	240.4	-6.2
		10	388.0	-153.8
		9	330.9	-96.7
(1,2)	261.5	8	576.2	-314.7

3.2. Genetic Operators

Crossover and mutation are two basic operators of GA used to search the solution space through differentiating the chromosomes chosen stochastically from the population. The parents are chosen using the tournament selection (Miller and Goldberg, 1995), *i.e.* a total of $tourSize$ (where $tourSize = popSize/6$) candidates are selected from the population and the one which has the lowest objective value win the tournament. The determination of the tournament size is important to give a chance to all chromosomes in the population while favouring the better ones. This is important to keep the diversity in the population while sustaining the convergence capacity of the GA.

Crossover is applied using two chromosomes selected from the population with tournament selection. A random cutting point between $[1, ni - 1]$, including boundaries, is determined and two children are built as shown in when the randomly determined cutting point is assumed to be four.

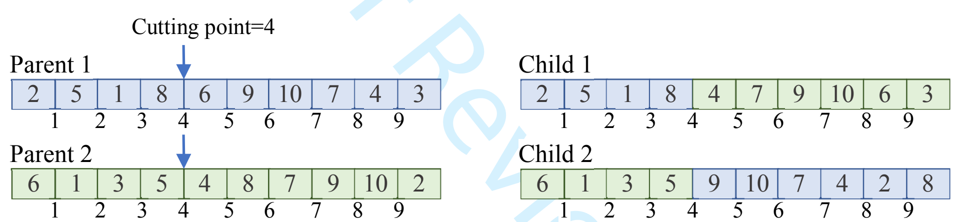


Figure 5. The application of the crossover procedure

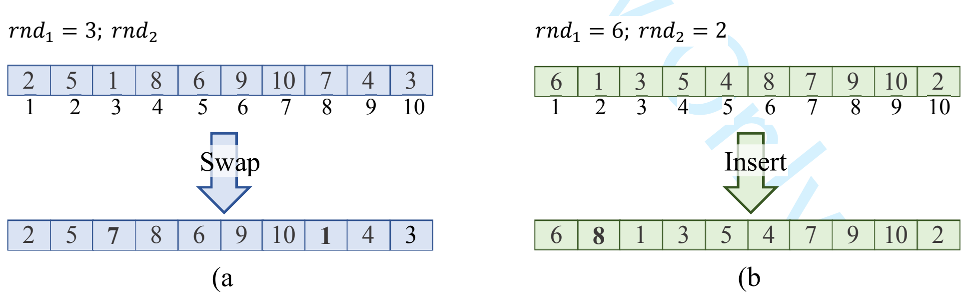


Figure 6. The application of the swap and insert mutations

Mutation helps algorithm avoid getting stuck in local optima. It plays crucial role to explore the different regions of the search space. Therefore, mutation is applied in two ways in this research, i.e. swap and insert. A random number (rnd_0) is determined between [0,1). If $rnd_0 < 0.5$, swap mutation is applied; otherwise, insert mutation is applied. Figure 6 illustrates the application of the mutation operator.

To apply the swap mutation, two random numbers (rnd_1 and rnd_2) are determined between $[1,ni]$ and the genes corresponding to these numbers are exchanged (see Figure 6a). To apply the insert mutation, the gene located at the location rnd_1 is removed and located at the location rnd_2, where rnd_1 and rnd_2 are generated randomly between $[1,ni]$, again (see Figure 6b).

3.3. Forming the New Generation

The new generation is formed replacing the worst in the population by a cyclic manner. For this aim, the fitness values of all newly generated solutions are calculated after the application of the genetic operators. The worst chromosome in the population is replaced with the best chromosome among the new individuals. This procedure is continued until there is no newly generated chromosome better than any individual in the population. It should be noted here that the duplication of the chromosomes is not allowed during the replacement process. This will be exemplified in the following subsection.

4. Detailed GA Solution of a Numerical Example

A numerical example consisting of three AM machines and eighteen parts is solved using the GA proposed in this research. Table 4 and Table 5 present the data related to the machine specifications and part details.

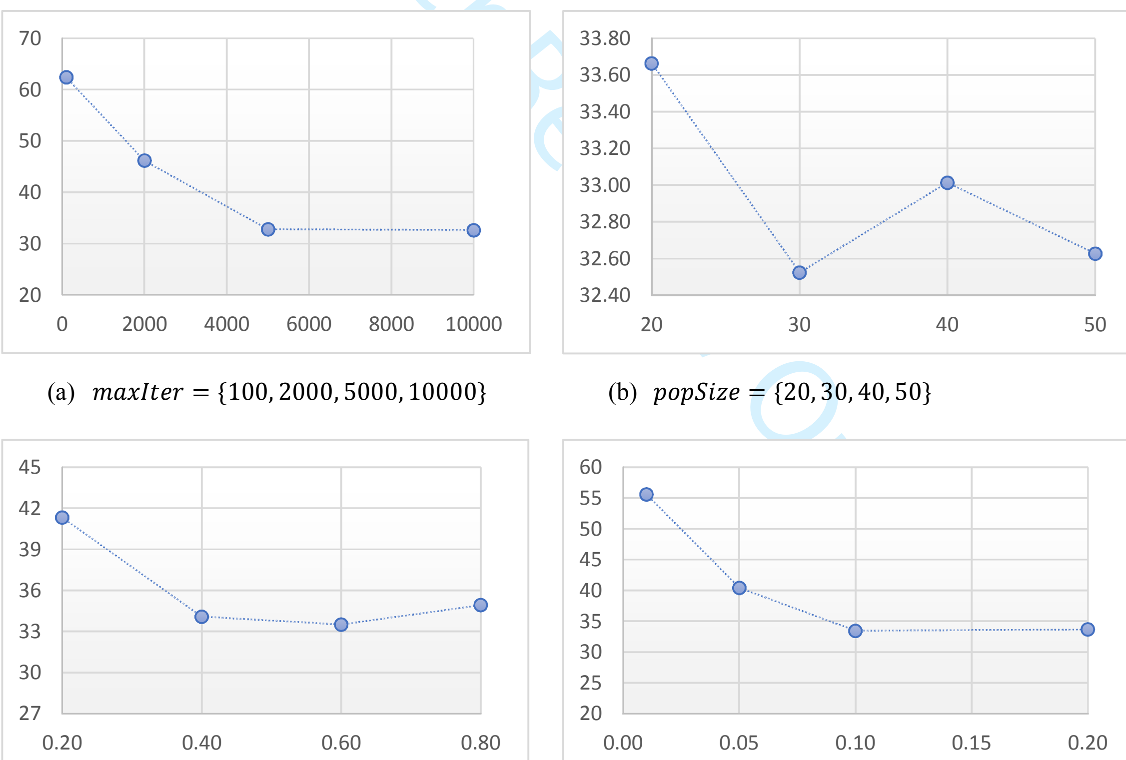
Table 4. Specifications of the AM machines

Parameters	k_1	k_2	k_3
VT_m (hr/ cm^3)	0.030864	0.030864	0.030864
HT_m (hr/cm)	0.7	0.7	0.7
SET_m (hr)	1.2	1.0	1.2
MH_m (cm)	32.5	32.5	32.5
MA_m (cm^2)	625	625	625

The parameters of the algorithm have been determined as $maxIter = 5000$, $popSize = 30$, $crossover\ rate\ (cr) = 0.6$, and $mutation\ rate\ (mr) = 0.1$ based on the common tendency in the literature and some preliminary tests as shown in Figure 7. The tournament size has been set to $popSize/6$.

Table 5. Parts data

i	$h_i\ (cm)$	$a_i\ (cm^2)$	$v_i\ (cm^3)$	$r_i\ (hr)$	$d_i\ (hr)$
1	20.3	19.5	147.8	9.8	378.3
2	7.4	84.2	285.2	21.0	214.7
3	27.3	61.1	583.3	36.6	149.0
4	25.8	299.3	3282.5	51.5	576.2
5	14.5	148.7	1265.5	56.4	240.4
6	3.5	376.4	723.3	69.9	211.6
7	20.4	20.5	278.5	75.1	330.9
8	23.3	91.1	1051.8	86.0	388.0
9	26.3	20.0	201.9	93.0	447.1
10	12.8	123.9	866.1	93.1	445.1
11	28.4	333.5	7347.6	97.2	634.6
12	10.9	74.2	333.6	97.3	177.6
13	14.1	268.5	2956.0	101.8	355.3
14	3.9	11.2	32.5	107.0	258.8
15	3.2	138.9	265.6	107.6	295.6
16	24.6	92.0	1387.4	115.6	509.6
17	7.1	424.0	1086.0	128.4	294.3
18	25.2	181.4	3559.2	132.3	575.7



(a) $maxIter = \{100, 2000, 5000, 10000\}$ (b) $popSize = \{20, 30, 40, 50\}$

(c) $cr = \{0.2, 0.4, 0.6, 0.8\}$ (d) $mr = \{0.01, 0.05, 0.1, 0.2\}$

Figure 7. Parameter tuning

The algorithm was coded in Java and run on Intel Core ® i7 6700HQ CPU@2.6GHz with 16GB of RAM using the parameters determined above. Table 6 shows (a) the chromosomes in the initial population, (b) new individuals obtained from the genetic operators, and (c) the new population after forming the new generation with fitness values. The sign "(-)" indicates that the chromosome has been replaced with that one indicated with (+) in the next generation. As seen from the table, while the best fitness value has not changed, the average fitness value was reduced from 282.30 to 231.69. This shows the convergence of the overall population in only one iteration.

Table 6. (a) The initial population, (b) new individuals obtained from the genetic operators, and (c) the new population after forming the new generation

(a) Initial population	*Fitness Value*
[14, 15, 4, 11, 3, 13, 16, 2, 17, 8, 9, 1, 18, 10, 6, 5, 12, 7]	340.65(-)
[18, 5, 1, 7, 16, 15, 10, 8, 4, 14, 9, 13, 11, 17, 3, 6, 2, 12]	249.07
[10, 15, 11, 3, 8, 6, 14, 16, 13, 7, 12, 18, 5, 17, 9, 1, 2, 4]	234.91
[4, 6, 14, 13, 5, 8, 9, 3, 2, 7, 18, 10, 1, 17, 15, 11, 12, 16]	262.44
[11, 8, 3, 2, 4, 17, 1, 18, 13, 5, 12, 10, 7, 15, 16, 6, 14, 9]	320.08(-)
[11, 16, 18, 1, 4, 7, 17, 14, 12, 3, 2, 15, 8, 5, 9, 10, 6, 13]	409.19(-)
[12, 3, 11, 10, 16, 4, 2, 7, 6, 13, 5, 9, 14, 18, 8, 15, 17, 1]	273.28
[16, 15, 14, 17, 12, 8, 9, 1, 13, 5, 4, 11, 2, 18, 6, 7, 3, 10]	110.83
[1, 13, 3, 15, 12, 8, 6, 18, 5, 14, 10, 4, 2, 16, 7, 9, 11, 17]	360.81(-)
[17, 1, 5, 3, 16, 10, 6, 15, 12, 11, 9, 2, 13, 8, 18, 7, 4, 14]	216.17
...	...
[9, 16, 13, 12, 17, 18, 1, 15, 4, 14, 3, 8, 5, 2, 10, 7, 11, 6]	368.24(-)
[10, 7, 3, 18, 1, 17, 9, 13, 11, 14, 16, 8, 5, 12, 6, 4, 15, 2]	274.91
[16, 2, 3, 13, 17, 8, 7, 4, 18, 12, 10, 11, 5, 14, 9, 15, 1, 6]	392.54(-)
[15, 4, 7, 5, 18, 12, 8, 9, 6, 17, 16, 14, 2, 13, 1, 10, 11, 3]	369.02(-)
[15, 10, 7, 13, 12, 17, 11, 5, 14, 2, 16, 3, 9, 18, 6, 1, 8, 4]	221.08

Best Fitness: 110.83, Average Fitness: 282.30

Iteration #1

(b) New chcromosomes after crossover and mutation	*Fitness Value*
[16, 15, 14, 17, 12, 8, 9, 1, 13, 5, 4, 11, 2, 18, 6, 7, 3, 10]	110.83
[16, 15, 14, 17, 12, 8, 9, 1, 13, 5, 4, 11, 2, 18, 6, 7, 3, 10]	110.83
[15, 10, 7, 13, 12, 17, 14, 16, 18, 5, 9, 1, 2, 4, 11, 3, 8, 6]	211.78
...	...
[2, 16, 12, 18, 3, 11, 13, 8, 6, 5, 9, 14, 15, 17, 1, 10, 4, 7]	233.08
[4, 6, 14, 13, 5, 8, 9, 3, 2, 7, 18, 10, 1, 17, 16, 15, 12, 11]	277.50
[16, 15, 14, 17, 12, 8, 9, 1, 13, 5, 4, 11, 2, 18, 6, 3, 7, 10]	110.83
[15, 10, 7, 13, 12, 17, 11, 5, 14, 2, 16, 3, 9, 18, 6, 1, 4, 8]	221.08
[14, 9, 4, 16, 15, 1, 2, 5, 11, 17, 13, 3, 12, 6, 7, 10, 18, 8]	253.21

(c) The new generation after iteration #1	*Fitness Value*
[16, 15, 14, 17, 12, 8, 9, 1, 13, 5, 4, 11, 2, 18, 6, 3, 7, 10]	110.83(+)
[18, 5, 1, 7, 16, 15, 10, 8, 4, 14, 9, 13, 11, 17, 3, 6, 2, 12]	249.07
[10, 15, 11, 3, 8, 6, 14, 16, 13, 7, 12, 18, 5, 17, 9, 1, 2, 4]	234.91
[4, 6, 14, 13, 5, 8, 9, 3, 2, 7, 18, 10, 1, 17, 15, 11, 12, 16]	262.44
[15, 10, 7, 13, 12, 17, 14, 16, 18, 5, 9, 1, 2, 4, 11, 3, 8, 6]	211.78(+)
[15, 10, 7, 13, 12, 17, 11, 5, 14, 2, 16, 3, 9, 18, 6, 1, 4, 8]	221.08(+)
[12, 3, 11, 10, 16, 4, 2, 7, 6, 13, 5, 9, 14, 18, 8, 15, 17, 1]	273.28
[16, 15, 14, 17, 12, 8, 9, 1, 13, 5, 4, 11, 2, 18, 6, 7, 3, 10]	110.83
[2, 16, 12, 18, 3, 11, 13, 8, 6, 5, 9, 14, 15, 17, 1, 10, 4, 7]	233.08(+)
[17, 1, 5, 3, 16, 10, 6, 15, 12, 11, 9, 2, 13, 8, 18, 7, 4, 14]	216.17
...	...

[10, 15, 11, 3, 8, 6, 5, 14, 2, 16, 9, 18, 1, 4, 7, 13, 12, 17]	247.89(+)
[10, 7, 3, 18, 1, 17, 9, 13, 11, 14, 16, 8, 5, 12, 6, 4, 15, 2]	274.91
[14, 9, 4, 16, 15, 1, 2, 5, 11, 17, 13, 3, 12, 6, 7, 10, 18, 8]	253.21(+)
[12, 3, 11, 10, 16, 4, 2, 7, 14, 6, 9, 15, 17, 1, 5, 18, 13, 8]	270.48(+)
[15, 10, 7, 13, 12, 17, 11, 5, 14, 2, 16, 3, 9, 18, 6, 1, 8, 4]	221.08

Best Fitness: 110.83, Average Fitness: 231.69

The best solution was found in the 15th iteration with the fitness value of 28.03 and the algorithm was terminated after 5000 iterations as no improvement was observed in the fitness value. The best solution obtained is shown in Table 7.

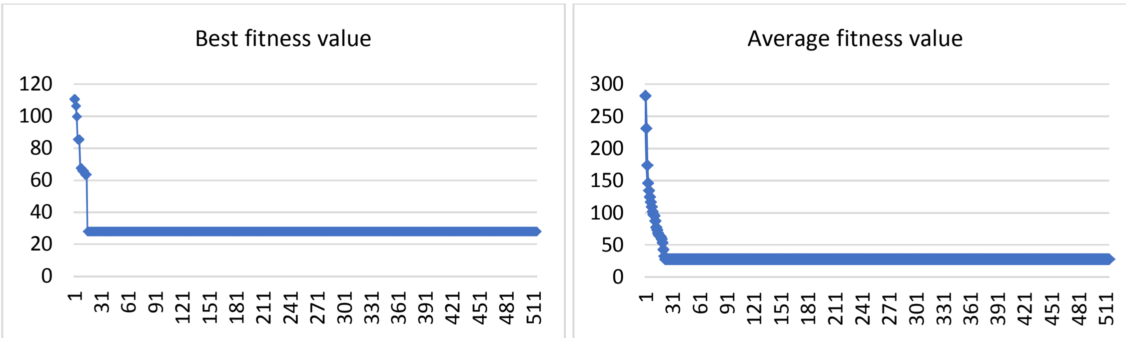
Table 7. The best solution obtained

(m, j)	P_{mj} (hr)	δ_i (hr)	CT_i (hr)
(1,1)	17, 5, 14, 7, 9	108.01	236.41
(2,1)	8, 13, 16, 15, 1	197.49	313.09
(3,1)	6, 2, 3, 12	79.73	177.03
(1,2)	11, 18	357.70	594.12
(2,2)	10, 4	147.10	460.19
Chromosome: [17, 5, 8, 14, 13, 6, 7, 16, 9, 15, 2, 11, 3, 12, 1, 18, 10, 4]			

Table 8. The calculation of lateness values of tasks based on the best solution obtained

(m, j)	CT_i (hr)	i	d_i (hr)	L_i (hr)
(1,1)	236.41	17	294.3	-57.89
		5	240.4	-3.99
		14	258.8	-22.39
		7	330.9	-94.49
		9	447.1	-210.69
(2,1)	313.09	8	388.0	-74.91
		13	355.3	-42.21
		16	509.6	-196.51
		15	295.6	17.49
		1	378.3	-65.21
(3,1)	177.03	6	211.6	-34.57
		2	214.7	-37.67
		3	149.0	28.03
		12	177.6	-0.57
(1,2)	594.12	11	634.6	-40.48
		18	575.7	18.42
(2,2)	460.19	10	445.1	15.09
		4	576.2	-116.01

As seen from the table, the late parts are p_{15} in job b_1 on k_2, p_3 in job b_1 on k_3, p_{18} in job b_2 on k_1 and p_{10} in job b_2 on k_2. Among those, the maximum lateness belongs to p_3 with 28.03 hr, which determines the objective value. The convergence of the best fitness and average fitness values are depicted in Figure 8 for the first 511 iterations.



(a) *(b)*

Figure 8. The convergence of the best fitness and average fitness values

5. Conclusions

This paper introduced the problem of scheduling additive manufacturing machines considering part release dates and due dates in a multiple machine environment with the aim of minimising maximum lateness, i.e. the maximum positive gap between the completion time and due date of jobs. The machines may have different specifications, e.g. dimensions, set up time, speed, etc. The problem has been defined and a numerical example is presented to show the production time and lateness calculations based on an example allocation of parts to machines. A GA approach, which employs a machine oriented decoding mechanism, was developed for solving the problem efficiently. Some preliminary tests have been conducted to determine the parameters of the algorithm. The steps of the algorithm have been illustrated through examples and a numerical example consisting of three machines and eighteen parts was solved using the proposed GA. The convergence of the best fitness and average fitness values were plotted and the best solution was reported. The preliminary results show that the proposed GA has a promising performance.

The methodology proposed in this research can be implemented by practitioners to schedule their production on additive manufacturing machines, especially in SLM machines. The results can also be extended to other additive manufacturing technologies. The major limitation of this work is that the performance of the proposed algorithm was not compared to any other algorithm or absolute minimum thorough a set of comprehensive computational tests. The authors' ongoing work aims to enhance the decoding mechanism and compare the performance of the algorithm to other heuristics and/or metaheuristics through a comprehensive computational study.

6. References

Calignano, F., Manfredi, D., Ambrosio, E., Biamino, S., Lombbardi, M., Atzeni, E., Salmi, A., Minetola, P., Iuliano, L., Fino, P., 2017. Overview on Additive Manufacturing Technologies. Proc. IEEE 105, 593–612. https://doi.org/10.1109/JPROC.2016.2625098

Cheng, R., Gen, M., Tsujimura, Y., 1996. A tutorial survey of job-shop scheduling problems using genetic algorithms—I. representation. Comput. Ind. Eng. 30, 983–997. https://doi.org/10.1016/0360-8352(96)00047-2

Goldberg, D.E., Holland, J.H., 1988. Genetic Algorithms and Machine Learning. Mach. Learn. 3, 95–99. https://doi.org/10.1023/A:1022602019183

Gonçalves, J.F., de Magalhães Mendes, J.J., Resende, M.G.C., 2005. A hybrid genetic algorithm for the job shop scheduling problem. Eur. J. Oper. Res. 167, 77–95. https://doi.org/10.1016/J.EJOR.2004.03.012

Kucukkoc, I., Li, Q., Zhang, D.Z., 2016. Increasing the utilisation of additive manufacturing and 3D printing machines considering order delivery times, in: 19th International Working Seminar on Production Economics. Innsbruck, Austria, pp. 195–201.

Kucukkoc, I., Zhang, D.Z., 2016. Integrating ant colony and genetic algorithms in the balancing and scheduling of complex assembly lines. Int. J. Adv. Manuf. Technol. 82, 265–285. https://doi.org/10.1007/s00170-015-7320-y

Kucukkoc, I., Zhang, D.Z., 2015. A mathematical model and genetic algorithm-based approach for parallel two-sided assembly line balancing problem. Prod. Plan. Control 26, 874–894. https://doi.org/10.1080/09537287.2014.994685

Li, Q., Kucukkoc, I., Zhang, D.Z., 2017. Production planning in additive manufacturing and 3D printing. Comput. Oper. Res. 83, 157–172.

Miller, B.L., Goldberg, D.E., 1995. Genetic Algorithms, Tournament Selection, and the Effects of Noise. Complex Syst. 9, 193–212. https://doi.org/10.1.1.30.6625

Pezzella, F., Morganti, G., Ciaschetti, G., 2008. A genetic algorithm for the Flexible Job-shop Scheduling Problem. Comput. Oper. Res. 35, 3202–3212. https://doi.org/10.1016/J.COR.2007.02.014

Schmidt, M., Merklein, M., Bourell, D., Dimitrov, D., Hausotte, T., Wegener, K., Overmeyer, L., Vollertsen, F., Levy, G.N., 2017. Laser based additive manufacturing in industry and academia. CIRP Ann. 66, 561–583. https://doi.org/10.1016/j.cirp.2017.05.011

Sule, D. R., Saxena, S. Batch scheduling for a limited capacity fixed period process problem, Prod. Plan. Control, 3:1, 61-69, DOI: 10.1080/09537289208919374

Woo, Y.-B., Jung, S., Kim, B.S., 2017. A rule-based genetic algorithm with an improvement heuristic for unrelated parallel machine scheduling problem with time-dependent deterioration and multiple rate-modifying activities. Comput. Ind. Eng. 109, 179–190. https://doi.org/10.1016/J.CIE.2017.05.007