Springer

KSME

Journal of Mechanical Science and Technology 33 (2) (2019) 791~800
www.springerlink.com/content/1738-494x(Print)/1976-3824(Online)
DOI 10.1007/s12206-019-0134-3

Optimized sequencing of CNC milling toolpath segments using metaheuristic algorithms [†]

B. Raja Chinna Karuppanan[1] and M. Saravanan[2,*]

[1]*Anna University, Chennai-600025, India*
[2]*Department of Mechanical Engineering, SSM Institute of Engineering & Technology, Dindigul - 624 002, Tamil Nadu, India*

(Manuscript Received March 19, 2018; Revised September 29, 2018; Accepted October 22, 2018)

Abstract

Intelligent selection of a short toolpath is made possible by reducing machining cycle time. Each metal cutting layer in a workpiece is composed of several entities, such as lines and arcs, which form the different cutting segments of a cutting plan. During machining, the cutter moves at controlled feed rates along various segments at a high speed in a single cutting pass. The end of a segment is bridged to the start point of the next segment by the non-cutting movement of the tool. Any two consecutive segments can be connected in eight different ways. Finding the shortest tool path at polynomial time is impossible because toolpaths are constructed in millions of ways by sequencing the segments. This paper presents an effective method that uses heuristic optimization techniques to solve this NP-hard problem, which is known as the traveling salesman problem, for segments. The proposed method adopts particle swarm optimization (PSO) and the genetic algorithm (GA) because of their capability to generate quality solutions for optimization problems. GA and PSO are implemented in the MATLABR2016b computing environment because of the platform's flexibility and simple coding method. The optimization procedure is validated by comparing its results with those of two industry standard CAM systems, namely, Autodesk Inventor HSM and Mastercam. Using the proposed optimization method saves up to 40 % of the tool's airtime during machining.

Keywords: CAM; CNC milling; Genetic algorithm; Metaheuristics; Particle swarm optimization; Toolpath optimization; Traveling salesman problem

1. Introduction

Milling is a versatile machining process commonly used to manufacture industrial products with various shapes and sizes. Rotary cutters remove a material through several small, separate cuts from a workpiece by feeding the workpiece to a path. The workpiece dimensions are defined using a computer-aided design program and transformed into machining commands by a computer-aided manufacturing system. When metal removal is performed with a typical toolpath, a large amount of unproductive time is spent on positioning the workpiece between the ends of one segment of the path to the starting coordinate of the subsequent segment. The non-cutting movement of the tool in air follows a straight line in many operations. Non-machining time can be minimized through efficient toolpath selection. Cycle time is the sum of tool engagement time, the time the tool is away from the workpiece, and the time required for other activities, such as tool changes and inspection. This non-engagement time increases with the number of segments planned and multiplies further with the number of passes for a required cutting depth to complete metal removal. With an increase in production volume, this non-productive time becomes considerable in the overall manufacturing time of the product. The standard procedure of creating toolpaths involves the use of CAM software, which has built-in functions to optimize the cycle time to a certain extent. Toolpaths must be optimized with intelligent techniques to achieve efficiency and increased productivity in a large product mix business.

In contour milling, the tool plunges into the workpiece at one end of the arc and leaves for the next segment after performing the cut. Kovacic and Balic [1] developed a method for optimized toolpath between cutting trajectories in a laser-cutting operation. They reported that the tool for a toolpath with a single segment could begin from either end. Thus, selecting between these two paths is an option. A 10-segment cutting plan has 3.71 billion possible routes. The possible number of toolpaths for a given number of "n" trajectories is $2^n n!$ (for bidirectional edges). Table 1 shows the complexity of selecting a toolpath. Determining the best solution via traditional calculation is difficult, even for a simple toolpath program with 10 segments. The best solution obtained after exhaustive calculation becomes minimally effective because of the considerable time spent, and this necessitates the use of heuristic approaches to determine the solution within a short

[*]Corresponding author. Tel.: +91 9443274296
E-mail address: sarandgl2k@gmail.com
[†]Recommended by Associate Editor Yongho Jeon

Table 1. Number of possible toolpaths.

Number of trajectories	No. of toolpaths
1	2
2	8
3	48
10	3.71 billion

time.

Many researchers have discussed the optimization of machining parameters, such as cycle time, surface finish, tool wear, production costs, and accuracy. Khan et al. [2] applied simulated annealing to path optimization between polygons. The polygons were represented as cells with possible entry and exit points. Kim [3] presented an optimized two-dimensional toolpath generation algorithm for roughing and finishing stages in direction-parallel milling. The method is based on an incomplete two-manifold mesh model, and the author claimed that the method maintains a constant material removal rate (MRR) for steady cutting forces and avoids chatter vibrations. Toolpath optimization for drilling holes is an extensively investigated topic because the number of holes is high, and the minimization time achieved is a considerable percentage of the cycle time. Qudeiri et al. [4] developed an optimization procedure to solve the sequence of operations for hole cutting in CNC machines. Kolahan and Liang [5] optimized hole-making operations by using the Tabu-search approach to minimize the total processing cost, which consists tooling, machining, non-productive tool traveling, and tool switching costs.

The sequencing of hole-drilling operations to minimize cutter airtime has been the subject of numerous studies. Kumar and Pachauri [6] adopted the genetic algorithm (GA) to optimize the drilling sequence. Lim et al. [7] used combinatorial cuckoo search to determine optimized toolpaths for drilling printed circuit boards. Al-Sahib and Abdulrazzaq [8] optimized the toolpath with GA, and the results were verified with ArtCAM software. Pezer [9] solved traveling salesman problem (TSP) using GA for a drill hole optimization problem and verified the results with the CATIA v5 CAM system. Raja Chınna Karuppanan and Saravanan [10] optimized the tool in the air path of drilling operations by using GA. Balic et al. [11] employed GA for the optimization of toolpath sequences in turning operation.

Xu et al. [12] generalized Euclidean TSP by adopting a given set of n-separate segments in two-dimensional space and finding a tour to traverse all segments, such that the total Euclidean distance of the tour is minimized. Castelino et al. [13] used a heuristic algorithm to minimize the non-productive time of a tool by connecting the toolpaths optimally. Qudeiri et al. [14] proposed an optimization method using GA to find the shortest toolpath between asymmetrically located operations at different levels in a workpiece. Oysu and Bingul [15] used GA and simulated annealing in contour parallel milling to minimize the tool airtime. Gupta et al. [16] used a hybrid GA, in which the initial seed solution is found by a special heuristic and combined with a randomly generated population of the GA algorithm. Kumar et al. [17] adopted GA to optimize the non-productive machining time in contour parallel machining. Qudeiri [18] optimized the sequence of operations in milling by considering tool change time and air travel time. Barclay et al. [19] created a simplified workpiece model by slicing to create layers, and each layer was discretized into a grid of squares. The cutting tool was also described as occupying a certain square or number of squares at any one time, and the optimized toolpath was found by using GA. Abdullah et al. [20] proposed a new optimization technique to generate a clear toolpath that removes the entire uncut region by contour parallel milling in minimum cutting time.

Most of the previous studies cited focused on methods of minimizing the non-cutting time of hole drilling, in which the tool enters and leaves from the same location. The use of optimization procedures to sequence the cutting of segments has not been explicitly addressed. An optimization procedure for the toolpath that connects the end coordinates of machining trajectories in air is applied in many operations, such as engraving, laser cutting, 3D printing, and robotic machining.

An algorithm was developed in the current study to optimize the toolpath for cutting segments (trajectories). Specifically, an optimization procedure was developed to sequence the cutting segments of a machining operation on a two-dimensional plane to minimize the overall tool shift time between segments. The problem was mathematically defined, and a heuristic solution was obtained by GA and PSO with suitable operators. A procedure for assessing the performance of the proposed algorithm was also developed.

2. Problem description

2.1 Finding the optimum sequence of segments

The problem in the present study is to connect machining segments by efficiently using non-cutting tool movements to reduce the machining cycle time. Fig. 1 shows the possible ways of connecting two successive segments. The connecting path lengths of each of these eight paths are different, and if the number of segments is large, then finding the shortest distance becomes an NP-hard problem, which cannot be solved by a deterministic approach.

The machining plan of a workpiece (Fig. 2) comprises milling segments to be connected by non-cutting toolpaths.

The number of bridging paths (green lines) is represented as “n” for a closed tour of the tool because the number of contours (blue lines) is also denoted as “n”. Among the millions of possible toolpaths available, the path with the shortest length should be found. A sufficiently good solution can be found for this 10-segment problem by calculating the distance between each segment pair to determine the tool-in-air travel distance. Subsequently, the total length of all connect-

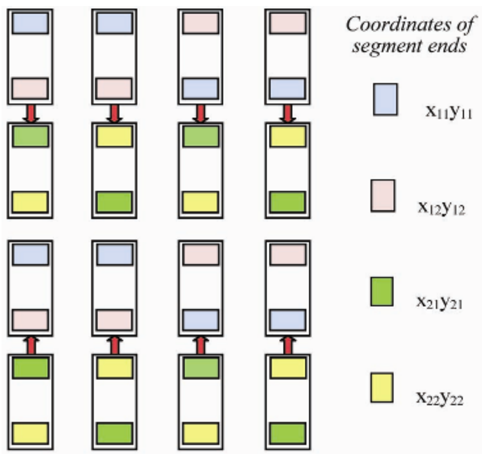


Fig. 1. Different methods for connecting two segments.

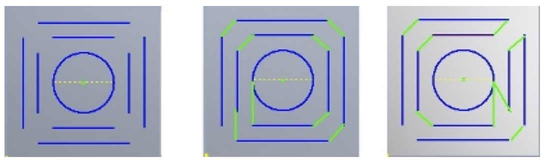


Fig. 2. Connection of segments by different routes.

ing segments on a two-dimensional plane can be calculated using Eq. (1).

$$\text{Total toolpath length } = \sum_{i=1}^{n} d(c_i, c_{i+1}), \tag{1}$$

where n is the number of connecting paths; c_i and c_{i+1} are the start and end points of a tool-in-air path, respectively; and "d" is the length of the connecting path.

Using mathematical algorithms to obtain the best solution among millions of possibilities in real time is practically impossible. The problem is a special case of TSP and referred to as the TSP of segments. An approximative algorithm can be used to obtain a solution within reasonable time. However, the solution may not be the best.

GA with the modified cycle crossover technique was used because the TSP for segments is a NP-hard problem. The application of metaheuristics algorithms was considered because of their capability to provide an answer from a very large search space, which can be improved further. The segments were arranged randomly in a sequence, in which half of them were rotated to achieve all possible connections between any two segment pairs.

2.2 TSP for segments

TSP is a classical combinatorial optimization problem with an exhaustive search space and is known as NP-hard. It is widely applied in robotics, drilling, welding, manufacturing, transportation, vehicle routing, and microchip manufacturing where the point-to-point control system is applied [21, 22]. If Euclidean TSP is for points, then segment TSP is for lines. For a given set of "n" segments on two-dimensional planes, finding the shortest tour path that traverses through all these segments is known as TSP for segments. Each segment must be traversed completely for a tour. Multiple trips to a segment are allowed when a segment is divided into sub-segments, and the entire segment is completely traversed. In a segment TSP, a set of n segments, $\{s_1, s_2, \cdot \cdot \cdot, s_n\}$, is connected at their end points by a set of "n-1" paths $\{c_1, c_2, \cdot \cdot \cdot, c_{n-1}\}$ for an open tour. The total tour distance is computed with Eq. (2).

Let $S = \{s_1, s_2, \cdots, s_n\}$ be the set of "n" contour segments to be machined. Then, total toolpath length (LT) = Sum of the segment lengths (LS) + Sum of connecting path lengths (LC).

$$LT = LS + LC \tag{2}$$

$$LS = \sum_{i=1}^{n} d(s_i, s_{i+1}) \tag{3}$$

$$LC = \sum_{i=1}^{n} d(c_i, c_{i+1}) . \tag{4}$$

Eq. (2) provides the total toolpath length, Eq. (3) determines the sum of the lengths of the machining segments, and Eq. (4) defines the total tool-in-air path length. The sum of the length of the bridging paths (LC) becomes the total toolpath length (TL) given by Eq. (5) because the tool engagement length (LS) cannot be reduced.

$$LT = LC. \tag{5}$$

Hence, the solution of the problem is obtained by minimizing LT, as expressed by Eq. (6).

$$LT = \sum_{i=1}^{n} d(c_i, c_{i+1}) , \tag{6}$$

where d is the distance between segments defined by Eq. (7).

$$d = \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2} , \tag{7}$$

where x_i and y_i and x_{i+1} and y_{i+1} are the end and start coordinates of successive segments, respectively. Eq. (6) should satisfy the following constraints.

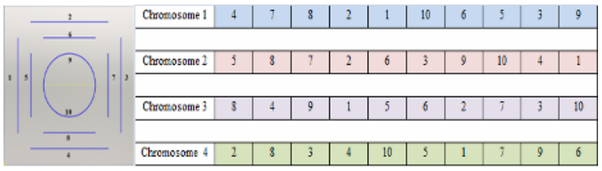
i) The tool moves through all contours.

ii) Each segment or sub-segment is visited only once by the tool in a single pass.

3. Experimental problems

3.1 Ten-segments problem

The selected prismatic workpiece is 100 mm × 100 mm × 16 mm in size and has 10 contour segments to be machined.



Chromosome 1	4	7	8	2	1	10	6	5	3	9
Chromosome 2	5	8	7	2	6	3	9	10	4	1
Chromosome 3	8	4	9	1	5	6	2	7	3	10
Chromosome 4	2	8	3	4	10	5	1	7	9	6

Fig. 3. Segments identified by numbers and solutions represented as chromosomes in Permutation encoding.

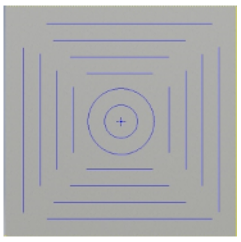


Fig. 4. Workpiece with twenty machining segments.

The segments were identified by their numbers in this problem (Fig. 3). The candidate solutions were genetically represented as chromosomes.

3.2 Twenty-segments problem

A rectangular aluminum workpiece of size 140 mm × 140 mm × 16 mm has to be machined with 20 contour segments, as shown in Fig. 4.

4. GA

GA is a population-based metaheuristic optimization algorithm in artificial intelligence. Batish et al. [23] optimized the different process parameters for rough and finish machining by using GA to determine the optimum combination of input parameters. Machined surfaces were subsequently analyzed through XRD, followed by an analysis of the grain and crystallite sizes of the machined samples and SEM analysis. Ko and Lee [24] applied GA to improve the surface finish of work pieces produced by FDM. They minimized the displacements of end effectors due to the cutting forces in the finish cutting operation.

GA uses mechanisms, such as reproduction, mutation, and selection, based on nature's evolution process. In GA, probable solutions in the search space, known as chromosomes, play the role of individuals in the population. The quality of the solutions is defined by the fitness function and improved by the repeated application of GA operators in successive generations. The algorithm produces solutions with desired qualities that satisfy the conditions laid down in the problem. GA, which was introduced by Holland [25], is simple but can solve complex problems. GAs differ by the way organisms are encoded and by the choice of genetic operators. Permutation encoding with modified cyclic crossover was adopted for the current problem. Fig. 5 provides the working details of the GA algorithm.

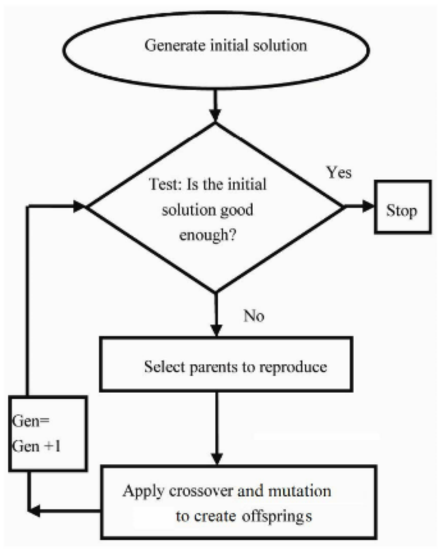
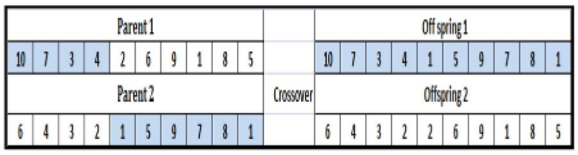

Generate initial solution
Test: Is the initial solution good enough?
Yes
Stop
No
Select parents to reproduce
Gen= Gen +1
Apply crossover and mutation to create offsprings

Fig. 5. Flowchart of GA.



Parent 1											Offspring 1									
10	7	3	4	2	6	9	1	8	5		10	7	3	4	1	5	9	7	8	1
Parent 2										Crossover	Offspring 2									
6	4	3	2	1	5	9	7	8	1		6	4	3	2	2	6	9	1	8	5

Fig. 6. Single-point crossover that produces defective offsprings.

4.1 Cycle crossover

The crossover operator produces new chromosomes from parent solutions analogous to biological reproduction. Normal crossover is unsuitable for this problem because permutation encoding is applied to create chromosomes from probable solutions. Thus, each segment needs to be visited only once, but a normal crossover creates chromosomes with some missing and duplicated genes (Fig. 6). Kwak and Lee [26] conducted a study that aimed to implement a novel crossover operation in a real-coded GA for a number of nonlinear/nonconvex functions and engineering optimization problems. Their crossover method was implemented by measuring the probabilistic distance between individuals.

The production of defective offspring is avoided by using cycle crossover (Table 2). The procedure is carried out for Parents 1 and 2 through the following steps.

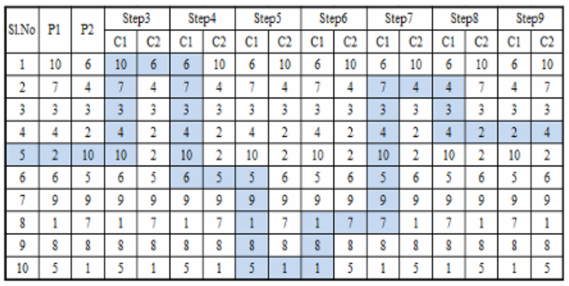
Step1: A location in the parent chromosome is randomly selected (Position 5).

Step2: At this position, the genes are swapped (2 →10) and (10 → 2).

Step3: In the first child, 10 is repeated and 2 is missing. In the second child, 2 is repeated and 3 is missing. Genes 10 and 6 are swapped.

The process is repeated until no duplicates and missing

Table 2. Cycle crossover technique used in permutation encoding.



Sl.No	P1	P2	Step3		Step4		Step5		Step6		Step7		Step8		Step9	
			C1	C2	C1	C2	C1	C2	C1	C2	C1	C2	C1	C2	C1	C2
1	10	6	10	6	6	10	6	10	6	10	6	10	6	10	6	10
2	7	4	7	4	7	4	7	4	7	4	7	4	4	7	4	7
3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
4	4	2	4	2	4	2	4	2	4	2	4	2	4	2	2	4
5	2	10	10	2	10	2	10	2	10	2	10	2	10	2	10	2
6	6	5	6	5	6	5	5	6	5	6	5	6	5	6	5	6
7	9	9	9	9	9	9	9	9	9	9	9	9	9	9	9	9
8	1	7	1	7	1	7	1	7	1	7	7	1	7	1	7	1
9	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8
10	5	1	5	1	5	1	5	1	1	5	1	5	1	5	1	5

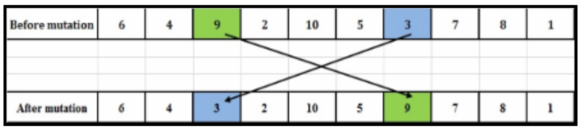


Fig. 7. Swap mutation interchanges two randomly selected genes in a chromosome.

genes are observed in the offspring.

4.2 Mutation

Mutation is used to introduce diversity in the search space. Mutation is essential for the convergence of GA and is applied with low probability. The mutation operator randomly selects and swaps two genes in a chromosome from the population (Fig. 7).

4.3 Implementation of GA

The proposed optimization procedure has the following steps.

1. Define the geometrical values of the contour segments and code them in the format required by MATLAB.
2. Create a script file for the GA algorithm and an "m" file for the objective function invoked from the main program.
3. Code each segment, represented by its x_i ,y_i and x_{i+1},y_{i+1} value, using a unique number via permutation encoding.
4. Create the initial population randomly.
5. Select GA parameters by several trial runs of the algorithm and by varying the population size and number of iterations.
6. Determine the total toolpath distance for every solution and arrange it in an order based on the fitness value.
7. Create the next population generation by selection and mutation operators.
8. Run the algorithm until the termination criteria are met.
9. Print and plot the results for visualization.

The distance between any two successive segments was calculated based on the connection method shown by the solid arrow in Fig. 8. All eight connection possibilities between two successive segments (Fig. 1) were achieved by randomly sequencing them, in which half of them were flipped.

The proposed optimization method was applied to the problems shown in Figs. 3 and 4. The first problem has 10 segments, and the second one is similar to the first one in the arrangement of segments, except that the number of segments is 20. MATLAB R2016b's scientific environment was used for programming, and a PC with Intel® core ™ i-3 2350M CPU at 2.30 GHz was used for this purpose.

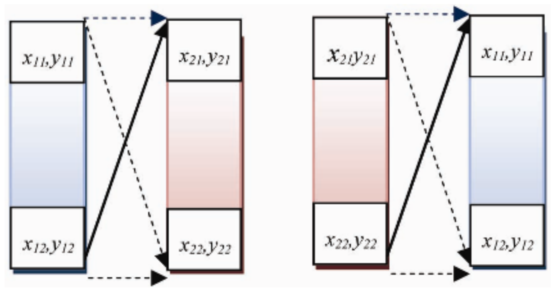


Fig. 8. Dark line shows the connection choice made to connect the successive segments: (a) Segment 2 follows segment 1; (b) segment 1 follows segment 2.

4.4 Selection of GA parameters

GA parameters, namely, population size, number of iterations, selection probability, and mutation probability, exert a significant influence on the optimization process. A randomly arranged set of chromosomes make up the initial population. Population size is decided before the execution of the algorithm with trial runs by varying the population size. Population size affects the toolpath distance. A population size of 1000 was selected from the graph because all iteration values have the minimum tool travel distance. The number of generations was achieved, and the toolpath length was plotted against iteration values. The iteration values of different population sizes showed that the toolpath length decreased with the increase in the number of iterations. The value of 8000 was selected because no significant improvement and waste of computation time were observed beyond this point. The evolution process terminated when the selected number of generations was reached. The following parameter values were selected for the optimization algorithm: population size = 1000, number of iterations = 8000, crossover fraction = 0.5, and mutation fraction = 0.08. In the second experiment, 20 segments were encoded by using a permutation method, and the initial population of the solution was created randomly. GA was implemented with various population sizes and numbers of iterations to arrive at the parameter values of GA. The selected parameters were population size = 1600, number of iterations = 14000, crossover fraction = 0.5, and mutation fraction = 0.08.

5. Particle swarm optimization (PSO) algorithm

PSO is a population-based metaheuristic technique used to solve optimization problems. PSO is a collaborative search

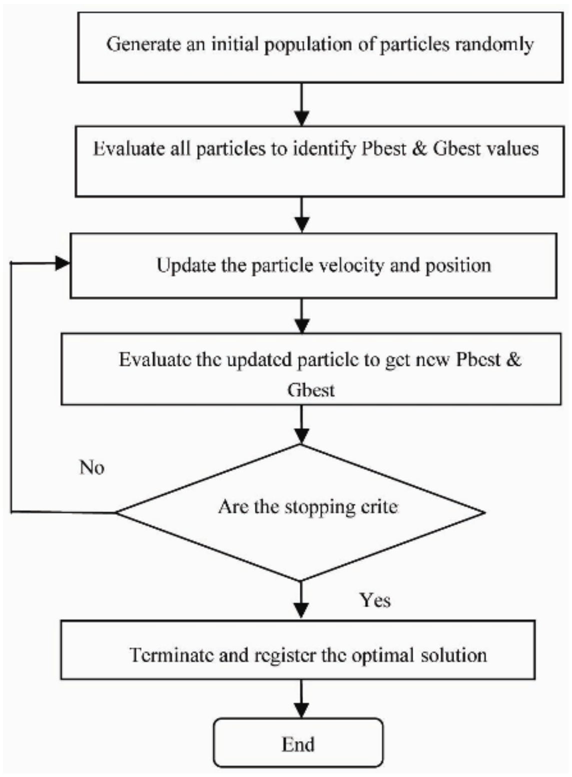

Generate an initial population of particles randomly
Evaluate all particles to identify Pbest & Gbest values
Update the particle velocity and position
Evaluate the updated particle to get new Pbest & Gbest
No
Are the stopping crite
Yes
Terminate and register the optimal solution
End

Fig. 9. Flowchart of the PSO algorithm.

technique inspired by the social and cognitive behavior of a flock of birds or a school of fish. Majumder described the application of PSO in optimizing process parameters in electric discharge machining (EDM) of AISI 316LN stainless steel [27]. The author performed experiments under different machining conditions, and machining performance, such as MRR and EWR, was evaluated.

In PSO, a population of random solutions is initially created, and the best fitness value of the population is improved toward a satisfactory result in successive iterations. The feasible solutions, which are known as particles, search through the problem space for the best solution by adapting to the current best values. Unlike in other evolutionary algorithms, the population is not subjected to the selection of parents for reproduction because all particles are updated and moved to a new generation. Updating a particle's value is achieved by using a memory system that stores the previously reached best values of all particles. PSO adopts a different information-sharing method, in which the particles with the best values alone are allowed to share the information with the rest of the population. Thus, the population converges to the optimum results quickly. An illustration of PSO is shown in Fig. 9. The PSO algorithm generally has the following steps.

1. Generate an initial population of "n' particles with randomly assigned positions and velocity values.
2. Find the fitness of each particle.
3. Select the best objective fitness value of each particle and set it as P_{best}.
4. Identify the particle with the best fitness value among the entire swarm and keep it as G_{best}.
5. Evaluate the particle velocity and new position of each particle according to Eqs. (8) and (9), respectively.

$$V_n^{i+1} = w \cdot v_n^i + c_1 \cdot r_1 (P_{best} - x_n^i) + c_2 \cdot r_2 (G_{best} - x_n^i), \tag{8}$$

$$X_n^{i+1} = X_n^i + V_n^{i+1}, \tag{9}$$

where V is particle velocity, X is the particle, r1 and r2 are random numbers with a value between 0 and 1, and C1 and C2 are learning factors with values between 1 and 4.

6. Evaluate the fitness of the particles.
7. Update the P_{best} value of each particle by comparing it with its own fitness value.
8. The G_{best} value is updated with the current best fitness value of the entire swarm.
9. Repeat steps 2-8 until the termination condition of the loop is met.

5.1 Implementation of PSO

The machining segments of the component were represented by numbers as in permutation encoding, and sequences were identified as particles. The initial population of particles was randomly generated, and the objective function value (OFV) of each particle was calculated. The following are the present, P_{best}, and G_{best} sequences of a particle.

Present	1. $Y_1 \uparrow Y_2$	2. $X_1 \rightarrow X_2$	3. Y_2-$\downarrow$-Y_1
	4. $X_2 \leftarrow X_1$	5. Y_1-$\uparrow Y_2$	6. $X_1 \rightarrow X_2$
	9. $\curvearrowleft$ 10. $\curvearrowleft$	7. $Y_2 \downarrow Y_1$	8. $X_2 \leftarrow X_1$
P_{best}	3. Y_2-$\downarrow$-Y_1	2. $X_1 \rightarrow X_2$	1. $Y_1 \uparrow Y_2$
	4. $X_2 \leftarrow X_1$	5. Y_1-$\uparrow Y_2$	9. $\curvearrowleft$ 10. $\curvearrowleft$
	6. $X_1 \rightarrow X_2$	7. $Y_2 \downarrow Y_1$	8. $X_2 \leftarrow X_1$
G_{best}	8. $X_2 \leftarrow X_1$	5. Y_1-$\uparrow Y_2$	3. Y_2-$\downarrow$-Y_1
	4. $X_2 \leftarrow X_1$	2. $X1 \rightarrow X2$	6. $X1 \rightarrow X2$
	9. $\curvearrowleft$ 10. $\curvearrowleft$	7. $Y2 \downarrow Y1$	1. $Y1 \uparrow Y2$.

The difference in the sequences was calculated, and the individuals of the present sequence were rearranged to obtain the P_{best} sequence.

1. $Y_1 \uparrow Y_2$ 2. $X_1 \rightarrow X_2$ 3. Y_2-$\downarrow$-Y_1 4. $X_2 \leftarrow X_1$ 5. Y_1-$\uparrow Y_2$
6. $X_1 \rightarrow X_2$ 9. $\curvearrowleft$ 10. $\curvearrowleft$ 7. $Y_2 \downarrow Y_1$ 8. $X_2 \leftarrow X_1$

$\downarrow$ (1,3)

3. Y_2-$\downarrow$-Y_1 2. $X_1 \rightarrow X_2$ **1. $Y_1 \uparrow Y_2$** 4. $X_2 \leftarrow X_1$ 5. Y_1-$\uparrow Y_2$
6. $X_1 \rightarrow X_2$ 9. $\curvearrowleft$ 10. $\curvearrowleft$ 7. $Y_2 \downarrow Y_1$ 8. $X_2 \leftarrow X_1$

$\downarrow$ (6,9)

3. Y_2-$\downarrow$-Y_1 2. $X_1 \rightarrow X_2$ 1. $Y_1 \uparrow Y_2$ 4. $X_2 \leftarrow X_1$ 5. Y_1-$\uparrow Y_2$
9. $\curvearrowleft$ **6. $X_1 \rightarrow X_2$** 10. $\curvearrowleft$ 7. $Y_2 \downarrow Y_1$ 8. $X_2 \leftarrow X_1$.

Therefore, (P_{best} - Present) = {(1,3) (6,9)}.

Similarly, individuals were swapped to obtain G_{best} from the present sequence.

1. $Y_1 \uparrow Y_2$ 2. $X_1 \rightarrow X_2$ 3. $Y_2\text{-}\downarrow\text{-}Y_1$ 4. $X_2 \leftarrow X_1$ 5. $Y_1\text{-}\uparrow Y_2$
6. $X_1 \rightarrow X_2$ 9. $\curvearrowleft$ 10. $\curvearrowleft$ 7. $Y_2 \downarrow Y_1$ 8. $X_2 \leftarrow X_1$

$\downarrow$ (1, 8)

8. $X_2 \leftarrow X_1$ 2. $X_1 \rightarrow X_2$ 3. $Y_2\text{-}\downarrow\text{-}Y_1$ 4. $X_2 \leftarrow X_1$ 5. $Y_1\text{-}\uparrow Y_2$
6. $X_1 \rightarrow X_2$ 9. $\curvearrowleft$ 10. $\curvearrowleft$ 7. $Y_2 \downarrow Y_1$ **1. $Y_1 \uparrow Y_2$**

$\downarrow$ (5, 2)

8. $X_2 \leftarrow X_1$ 5. $Y_1\text{-}\uparrow Y_2$ 3. $Y_2\text{-}\downarrow\text{-}Y_1$ 4. $X_2 \leftarrow X_1$ 2. $X_1 \rightarrow X_2$
6. $X_1 \rightarrow X_2$ 9. $\curvearrowleft$ 10. $\curvearrowleft$ 7. $Y_2 \downarrow Y_1$ 1. $Y_1 \uparrow Y_2$.

Therefore, (G_{best} - Present) = {(1,8) (5,2)}.

Particle velocity was determined using Eq. (8). Many researchers have used benchmark problems to determine the parameters of the algorithm. In this study, the parameters were selected after several trial runs of the algorithm.

Velocity = 1 * 0.70 {(1,3) (6,9)} +1 * 0.30{(1,8)(5,2)}.

New sequence = Present + ((1,3), (6,9), (1,8)).

1. $Y_1 \uparrow Y_2$ 2. $X_1 \rightarrow X_2$ 3. $Y_2\text{-}\downarrow\text{-}Y_1$ 4. $X_2 \leftarrow X_1$ 5. $Y_1\text{-}\uparrow Y_2$
6. $X_1 \rightarrow X_2$ 9. $\curvearrowleft$ 10. $\curvearrowleft$ 7. $Y_2 \downarrow Y_1$ 8. $X_2 \leftarrow X_1$

$\downarrow$ (1, 3)

3. $Y_2\text{-}\downarrow\text{-}Y_1$ 2. $X_1 \rightarrow X_2$ **1. $Y_1 \uparrow Y_2$** 4. $X_2 \leftarrow X_1$ 5. $Y_1\text{-}\uparrow Y_2$
6. $X_1 \rightarrow X_2$ 9. $\curvearrowleft$ 10. $\curvearrowleft$ 7. $Y_2 \downarrow Y_1$ 8. $X_2 \leftarrow X_1$

$\downarrow$ (6, 9)

3. $Y_2\text{-}\downarrow\text{-}Y_1$ 2. $X_1 \rightarrow X_2$ 1. $Y_1 \uparrow Y_2$ 4. $X_2 \leftarrow X_1$ 5. $Y_1\text{-}\uparrow Y_2$
9. $\curvearrowleft$ **6. $X_1 \rightarrow X_2$** 10. $\curvearrowleft$ 7. $Y_2 \downarrow Y_1$ 8. $X_2 \leftarrow X_1$

$\downarrow$ (1, 8)

3. $Y_2\text{-}\downarrow\text{-}Y_1$ 2. $X_1 \rightarrow X_2$ **8. $X_2 \leftarrow X_1$** 4. $X_2 \leftarrow X_1$ 5. $Y_1\text{-}\uparrow Y_2$
9. $\curvearrowleft$ 6. $X_1 \rightarrow X_2$ 10. $\curvearrowleft$ 7. $Y_2 \downarrow Y_1$ **1. $Y_1 \uparrow Y_2$**.

Through this procedure, all "Present" particles were updated, and OFV was evaluated. The P_{best} and G_{best} particles were identified in every iteration, and the optimal solution was determined from the required number of iterations.

6. Computational results

The optimized toolpath sequence achieved by the proposed PSO and the GA method is as follows:

**1. $Y_1 \uparrow Y_2$ 2. $X_1 \rightarrow X_2$ 3. $Y_2\text{-}\downarrow\text{-}Y_1$ 4. $X_2 \leftarrow X_1$ 5. $Y_1\text{-}\uparrow Y_2$
6. $X_1 \rightarrow X_2$ 9. $\curvearrowleft$ 10. $\curvearrowleft$ 7. $Y_2 \downarrow Y_1$ 8. $X_2 \leftarrow X_1$**.

The arrow mark points to the direction of tool travel from bottom to top, right to left, clockwise, etc. The minimum toolpath achieved is 163.07 mm, as shown by the MATLAB plot in Fig. 10. The optimized toolpath for this problem is shown in Fig. 11.

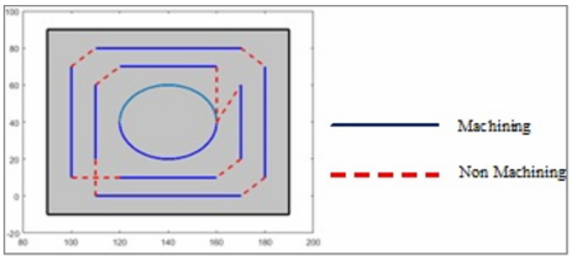


Fig. 10. Optimized toolpath obtained using MATLAB.

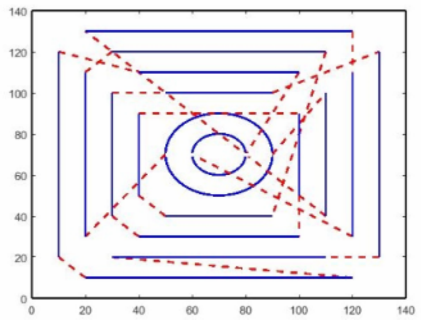


Fig. 11. Optimized toolpath of the 20-segment problem. The dotted lines indicate the segment connection paths.

The optimized sequence of the 20-segment problem is given below. The time taken by MATLAB to run GA was 2059.806867 seconds.

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

6.1 Discussions about the optimization procedure

TSP for segments was solved for a closed tour. Figs. 10 and 11 show the MATLAB plots of the optimized toolpaths with closed ends of the selected problems. While creating the NC program, one of the links in the closed path chain had to be removed to make it open and thus allow the programmer to make a choice regarding the tool's starting position. In addition, the length of the toolpath was shortened by this removal. This method is valid only in the case of a single pass, and the toolpath has to be a closed one for a deeper cut depth. The savings by the optimization method increased with the number of passes.

The performance of a toolpath optimization algorithm can be verified and validated using CAM systems. NC part programming is necessary to perform machining in a CNC machine by using either a manual or CAM-assisted method. Thus, the usage of CAM systems is sufficient for validating the optimization algorithm. In the present study, Autodesk Inventor integrated with HSM and Mastercam were used to generate and simulate the toolpaths of the experimental problems. The toolpaths found by the optimization technique were simulated using Inventor HSM and Mastercam systems. For all simulations, cutting parameters, such as relief and clearance values, were kept constant. The following machining information was

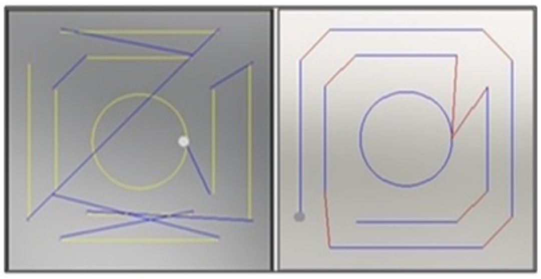


Fig. 12. (a) Toolpath generated by inventor; (b) simulated optimized toolpath.

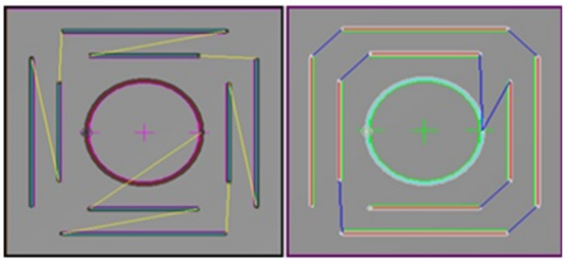


Fig. 13. (a) Toolpath generated by Mastercam; (b) optimized toolpath simulated in Mastercam.

provided by the CAM systems.

1. Cycle time 2. Cutting time 3. Rapid time
4. Rapid toolpath length 5. Cutting toolpath length.

The Inventor-generated and optimized toolpaths used for the experimental 10-segment problem are shown in Fig. 12. A comparison of the Inventor-generated and optimized toolpaths are given in Tables 3 and 4.

A similar comparison was made between Mastercam-generated and optimized toolpaths for the same problem. Fig. 13 displays the Mastercam-simulated and optimized toolpaths (simulated with Mastercam).

Table 3 compares the two toolpaths created by the two approaches. Table 4 shows the rapid toolpath lengths and savings achieved by the optimization procedure.

Table 4 provides the comparison details of CAM-generated and optimized toolpaths. The optimized rapid toolpath length was shorter than the rapid toolpath length generated in Inventor by 178.7113 mm, and the savings was 43.44 %. For a machining plan, having a 10 mm depth cut and completion in 10 passes saved 1787.113 mm of rapid tool movement. The rapid toolpath length generated in Mastercam was 398.433 mm, which is 166.251 mm longer than the optimized toolpath. The optimization technique saved 41.72 % of rapid tool movements, which would be multiplied by the increase in the number of passes and segments.

The results showed that the optimization method using PSO was more effective than GA in creating near-optimum solutions for rapid tool movements in machining segments. The optimization procedure is applicable to computer-controlled milling, laser cutting, welding, and 3D printing operations. Actual machining of the experimental problem is not required because NC path generation and post processing are only conducted on a computer software.

Table 3. Comparison of CAM toolpaths with GA and PSO toolpaths.

Sequence of segments					
Inventor HSM toolpath			Mastercam toolpath		
Generated	GA optimized	PSO	Generated	GA optimized	PSO
9↶	1.$Y_1 \uparrow Y_2$	1.$Y_1 \uparrow Y_2$	1.$Y_1 \uparrow Y_2$	1.$Y_1 - \uparrow Y_2$	1.$Y_1 \uparrow Y_2$
10↶	2.$X_1 \rightarrow X_2$	2 $X_1 \rightarrow X_2$	5.$Y_1 - \uparrow Y_2$	2.$X_1 \rightarrow X_2$	2.$X_1 \rightarrow X_2$
7.$Y_1 \uparrow Y_2$	9↶	3.$Y_2 - \downarrow Y_1$	2.$X_1 \rightarrow X_2$	9↶	3. $Y_2 - \downarrow Y_1$
3.$Y_2 \downarrow Y_1$	10↶	4.$X_2 \leftarrow X_1$	6.$X_1 \rightarrow X_2$	10↶	4.$X_2 \leftarrow X_1$
8.$X_1 \rightarrow X_2$	3.$Y_2 \downarrow Y_1$	5.$Y_1 - \uparrow Y_2$	3.$Y_2 \downarrow Y_1$	3.$Y_2 \downarrow Y_1$	5.$Y_1 - \uparrow Y_2$
4.$X_1 \rightarrow X_2$	4.$X_1 \rightarrow X_2$	6. $X_1 \rightarrow X_2$	7.$Y_2 - Y_1$	4.$X_1 \rightarrow X_2$	6.$X_1 \rightarrow X_2$
5.$Y_1 \uparrow Y_2$	5.$Y_1 \uparrow Y_2$	9. ↶	4.$X_2 \leftarrow X_1$	5.$Y_1 \uparrow Y_2$	9. ↶
6.$X_1 \rightarrow X_2$	6.$X_1 \rightarrow X_2$	10. ↶	8.$X_2 \leftarrow X_1$	6.$X_1 \rightarrow X_2$	10. ↶
2.$X_1 \rightarrow X_2$	7.$Y_1 \uparrow Y_2$	7.$Y_2 \downarrow Y_1$	9. ↶	7.$Y_1 \uparrow Y_2$	7.$Y_2 \downarrow Y_1$
1.$Y_1 - \uparrow Y_2$	8.$X_1 \rightarrow X_2$	8. $X_2 \leftarrow X_1$	10. ↶	8.$X_1 \rightarrow X_2$	8.$X_2 \leftarrow X_1$

Table 4. Comparison of rapid toolpath lengths.

Rapid toolpath length in mm			Savings in distance
Inventor HSM toolpath	Generated	411.3304	178.7113 mm (43.44 %)
	GA	258.2429	
	PSO	232.6191	
Mastercam toolpath	Generated	398.433	166.251 mm (41.72 %)
	GA	245.345	
	PSO	232.182	

6.2 Toolpath optimization for sustainable manufacturing

The ever-increasing cost of energy and the release of greenhouse gases by fossil fuel necessitate researchers to minimize the use of energy by CNC machines [28]. Reducing energy consumption has become a critical issue in manufacturing. Industries are confronted with environmental regulations on pollution associated with electric energy consumptions.

Lee et al. [29] evaluated the environmental impacts of chemical mechanical polishing. The electric energy consumption associated with the process, greenhouse gas emissions, and their impacts on global warming were evaluated from an environmental standpoint.

Many strategies are practiced to reduce the energy consumption of CNC machines, reduce manufacturing expenditures, and minimize the carbon footprint. Electric power is consumed by CNC control with the main spindle and feed axis motors as well as by numerous auxiliary components. A dominant share of the total power is spent for lighting, ventilation, air conditioning, compressed air generation, pallet changer, cooling, hydraulics, automation, and lubricant processing. The energy requirement of the auxiliary components varies very slightly during different conditions of readiness, roughing, and finishing. This phenomenon means that the machine consumes great energy even during non-cutting tool movements because of the fixed energy requirements.

The travel distance and power consumption of feed axes can be reduced by using the optimization method presented in this work. The fixed energy required for that the machining period is saved because the machining cycle time is reduced. Table 4 shows that nearly 40 % of the non-cutting toolpath length can be reduced. If the machining is performed in a Takisawa Mac-V3 milling machine, then 2 seconds of savings can be obtained in tool airtime. This time increases with the number of passes and components produced. The energy saved in fixed energy alone will be 83.33 watts for machining a component for 10 passes with 10 minutes of cycle time in a Takisawa Mac-V3 milling machine. Raja Chinna Karuppanan and Saravanan [30] presented a method that uses GA to improve the energy efficiency of the drilling process. The authors used the difference in the mass of various feed drives in their approach.

7. Conclusion

A new method to sequence cutting segments efficiently in CNC milling was proposed in this work. The application of PSO was satisfactory, and the procedure was verified by comparing it with popular GA and NC code-generating CAM tools. Reasonable time saving of up to 40 % can be achieved, and this saving increases with the number of segments. The selection of the parameters of PSO is important for achieving satisfactory results. All of the different possible connections between any two segments were included in the random selection of the solution. To implement the optimization procedure in an NC program, the CNC programmer has to create the code manually by using the sequence of operations provided by the optimization method. Programs have to be developed to include the method as an add-on utility to generate NC codes of optimized toolpaths automatically in CAM systems.

- The presented method has to be enhanced further to create optimized toolpaths for components that may have obstacles hindering tool movements.
- An optimization algorithm for three-dimensional toolpaths may be developed in future work based on the scheme provided by the present work.

References

[1] M. Kovacic and J. Balic, Evolutionary programming of a CNC cutting machine, *International Journal of Advanced Manufacturing Technology,* 22 (2003) 118-124.

[2] W. A. Khan, D. R. Hayhurst and C. Cannings, Determination of optimal path under approach and exit constraints, *Europe an Journal of Operational Research,* 117 (2) (1999) 310-325.

[3] H.-C. Kim, Optimum toolpath generation for 2.5D direction-parallel milling with incomplete mesh model, *Journal of Mechanical Science and Technology*, 24 (5) (2010) 1019-1027.

[4] J. E. A. Qudeiri, A.-M. Raid, M. A. Jamali and H. Yamamoto, Optimization hole-cutting operations sequence in CNC machine tools using GA, *International Conference on Service Systems and Service Management,* Troyes, France (2006) 501-506.

[5] F. Kolahan and M. Liang, Optimization of hole-making operations: A tabu-search approach, *International Journal of Machine Tools and Manufacture*, 40 (12) (2000) 1735-1753.

[6] A. Kumar and P. Pachauri, Optimization drilling sequence by genetic algorithm, *International Journal of Scientific and Research Publications,* 2 (9) (2012) 1-7.

[7] W. C. E. Lim, G. Kanagaraj and S. G. Ponnambalam, PCB drill path optimization by combinatorial cuckoo search algorithm, *The Scientific World Journal,* Article ID 264518 (2014) 11.

[8] N. K. A. Al-Sahib and H. F. Abdulrazzaq, Toolpath optimization of drilling sequence in CNC machine using genetic algorithm, *Innovative Systems Design and Engineering,* 5 (2014) 15-26.

[9] D. Pezer, Efficiency of toolpath optimization using genetic algorithm in relation to the optimization achieved with the CAM software, *Procedia Engineering,* 149 (2016) 374-379.

[10] B. R. C. Karuppanan and M. Saravanan, Genetic algorithm for TSP in optimizing CNC toolpath, *International Journal of Engineering Technology, Management and Applied Science*, 5 (2) (2017) 139-146.

[11] J. Balic, F. Mirko, S. A. Hajdin and G. Afrim, Optimization of cutting toolpath generation using genetic algorithm, *Annals of DAAAM for 2011 & Proceedings of the 22nd International DAAAM Symposium*, 22 (1) (2011) 569-570.

[12] J. Xu, Y. Yang and Z. Lin, Traveling salesman problem of segments, *International Computing and Combinatorics Conference*, *Cocoon, LNCS*, 2697 (2003) 40-49.

[13] K. Castelino, R. D'Souza and P. K. Wright, Toolpath optimization for minimizing airtime during machining, *Journal of Manufacturing Systems,* 22 (3) (2003) 173-180.

[14] J. E. A. Qudeiri, H. Yamamoto and R. Ramli, Optimization of operation sequence in CNC machine tools using genetic algorithm, *Journal of Advanced Mechanical Design, Systems, and Manufacturing,* 1 (2) (2007) 272-282.

[15] C. Oysu and Z. Bingul, Application of heuristic and hybrid-GASA algorithms to tool-path optimization problem for

minimizing airtime during machining, *Engineering Applications of Artificial Intelligence,* 22 (3) (2009) 389-396.

[16] A. K. Gupta, P. Chandna and P. Tandon, Hybrid genetic algorithm for minimizing non productive machining time during 2.5 D milling, *International Journal of Engineering, Science and Technology*, 3 (1) (2011) 183-190.

[17] S. Kumar, A. K. Gupta and P. Chandna, Minimization of nonproductive time during 2.5D milling, *World Academy of Science, Engineering and Technology: International Journal of Mechanical, Aerospace, Industrial and Mechatronics Engineering,* 8 (6) (2014) 1147-1152.

[18] J. E. A. Qudeiri, Optimization and program generation of a toolpath for multi-cutting tool operations in CNC machines, *International Journal of Emerging Technology and Advanced Engineering,* 4 (Special Issue 5) (2014) 15-23.

[19] J. Barclay, V. Dhokia and A. Nassehi, Generating milling toolpaths for prismatic parts using genetic programming, *Procedia CIRP,* 33 (2015) 490-495.

[20] H. Abdullah, R. Ramli and D. A. Wahab, Toolpath length optimisation of contour parallel milling based on modified ant colony optimization, *International Journal of Advanced Manufacturing Technology,* 10 March (2017) 1-14.

[21] R. Matai, S. P. Singh and M. L. Mittal, Traveling salesman problem: An overview of applications, formulations and solution approaches, *Traveling Salesman Problem, Theory and Applications,* D. Davendra (Ed.), InTech (2010) 1-24.

[22] G. Gutin and A. P. Punnen, *The Travelling Salesman Problem and its Variations (Combinatorial Optimization),* Springer, New York, USA, 12 (2007).

[23] A. Batish et al., Hard turning: Parametric optimization using genetic algorithm for rough/finish machining and study of surface morphology, *Journal of Mechanical Science and Technology*, 28 (5) (2014) 1629-1640.

[24] S. Ko and D. Lee, Stiffness optimization of 5-axis machine tool for improving surface roughness of 3D printed products, *Journal of Mechanical Science and Technology,* 31 (7) (2017) 3355-3369.

[25] J. H. Holland, *Adaptation in Natural and Artificial Systems,* MIT Press Cambridge, MA, USA (1992).

[26] N. S. Kwak and J. Lee, An enhancement of selection and crossover operations in real-coded genetic algorithm for large-dimensionality optimization, *Journal of Mechanical Science and Technology*, 30 (1) (2016) 237-247.

[27] A. Majumder, Process parameter optimization during EDM of AISI 316 LN stainless steel by using fuzzy based multi-objective PSO, *Journal of Mechanical Science and Technology,* 27 (7) (2013) 2143-2151.

[28] J. Jeswiet and S. Kara, Carbon emissions and CES^{TM} in manufacturing, *CIRP Annals - Manufacturing Technology*, 57 (1) (2008) 17-20.

[29] H. Lee, S. Park and H. Jeong, Evaluation of environmental impacts during chemical mechanical polishing (CMP) for sustainable manufacturing Technology, *Journal of Mechanical Science and Technology,* 27 (2) (2013) 511-518.

[30] B. R. C. Karuppanan and M. Saravanan, Toolpath optimization by genetic algorithm for energy efficient machining, *Taga Journal of Graphic Technology*, 14 (2018) 1670-1679.

Raja Chinna Karuppanan B. received his M.Tech. degree in mechanical engineering from IIT Madras. His research interests include CAD/CAM, optimization using evolutionary techniques, and robot programming. He has 28 years of teaching experience. At present, he is pursuing his doctoral research degree at Chennai Anna University, India.

Saravanan M. is working as the Principal of SSM Institute of Engineering and Technology, Dindigul, India. He has more than 26 years of teaching experience. He received his Ph.D. in scatter search algorithm for scheduling various manufacturing systems from Anna University, Chennai. His research interests include scheduling for manufacturing systems, robotics, production planning, composites, optimization techniques, and agile manufacturing. He has published more than 100 technical papers in refereed international journals and more than 140 papers in national and international conferences.