Hindawi
Computational Intelligence and Neuroscience
Volume 2022, Article ID 4719266, 12 pages
https://doi.org/10.1155/2022/4719266

Research Article

A Divide-and-Conquer Bat Algorithm with Direction of Mean Best Position for Optimization of Cutting Parameters in CNC Turnings

Xingwang Huang,[1] Zongbao He,[1] Yong Chen,[1] and Shutong Xie [1,2]

[1]*School of Computer Engineering, Jimei University, Xiamen 361021, China*
[2]*Digital Fujian Big Data Modeling and Intelligent Computing Institute, Jimei University, Xiamen 361021, China*

Correspondence should be addressed to Shutong Xie; shutong@jmu.edu.cn

Received 2 November 2021; Revised 4 January 2022; Accepted 13 January 2022; Published 23 February 2022

Academic Editor: Diego Oliva

Optimization of machining parameters is an important problem in the modern manufacturing world due to production efficiency and economics. This problem is well known to be complex and is regarded as a strongly nondeterministic polynomial (NP)-hard problem. To reduce the production cost of work-pieces in computer numerical control (CNC) machining, a novel optimization algorithm based on a combination of the bat algorithm and a divide-and-conquer strategy is proposed. First, the basic bat algorithm (BA) is modified with the aim to avoid finding the local optimal solution. In addition, a Gaussian quantum bat algorithm with direction of mean best position is developed. Second, in order to reduce the complexity of the optimization problem, the whole optimization problem is divided into several subproblems by using a divide-and-conquer strategy according to the characteristic of multipass turning operations. Finally, under a large number of machining constraints, the cutting parameters of the two stages of roughing and finishing are simultaneously optimized. Simulation results show that the proposed algorithm can find better combinations of the machining parameters than other algorithms proposed previously to further reduce the production cost. In addition, the outcome of our work presents a novel way to solve the complex optimization problem of machining parameters with a combination of traditional mathematical methods and swarm intelligence algorithms.

1. Introduction

In the manufacturing field, computer numerical control (CNC) machining refers to the computerized digital control of automated machine tools used to process rough material into semifinished or finished parts; it is one of the most common technologies. The main purpose of CNC machining is to save machining costs and improve machining efficiency and machining quality. Machining costs can be saved by selecting reasonable machining parameters, which introduces an optimization problem, i.e., selecting the optimal machining parameters to achieve the goal of reducing machining costs under the given machining constraints. Earlier research on the optimization of machining parameters mainly used traditional mathematical processing methods such as dynamic programming, sequential unconstrained minimization technique (SUMT), and linear or nonlinear programming. However, in general, the optimization problems of machining parameters are nonlinear and complicated problems with multiple constraints. Therefore, it is difficult to obtain satisfactory optimization solutions using traditional methods [1, 2]. In recent years, many scholars have applied swarm intelligence algorithms to the optimization problems of machining parameters in the field of computer integrated manufacturing. By using swarm intelligence algorithms to search for approximate optimal solutions of the problem, some research results have been achieved [3–16].

However, most previous studies were devoted to combining swarm intelligence algorithms with various local improvement algorithms [3–11, 17] (e.g., population diversification, local greedy search, and the use of heuristics as local search) in the hope of obtaining better results. However, because they did not fully consider the characteristics

of the turning problem with multiple machining processes, the results obtained by the algorithms were similar, and it was difficult to significantly reduce the machining cost. To address this bottleneck, this paper proposes a novel optimization algorithm by fully considering the characteristics of the turning problem while effectively exploiting the global optimization performance of the swarm intelligence algorithm. By combining the improved bat algorithm with the divide-and-conquer strategy, the performance of the optimization algorithm is substantially improved. The final optimization algorithm is able to find better results.

The rest of this paper is organized as follows. Section 2 represents the related works, especially the intelligent algorithms for optimization problems in CNC turning. Section 3 describes the mathematical model for the optimization of machining parameters in CNC turnings. Section 4 first introduces the bat algorithm, then proposes the Gaussian quantum bat algorithm with direction of mean best position (GQMBA), and finally elaborates on the idea of the combination of GQMBA and the divide-and-conquer strategy for solving the machining parameter optimization problems. In Section 5, simulation experiments are conducted, and different algorithms are compared. Finally, the concluding comments and some future research directions are presented in the last section.

2. Related Works

Optimization of turning parameters is an important issue in the manufacturing field. Early studies used traditional mathematical methods to find optimized machining parameters. Metaheuristic algorithms are also used to solve optimization problems of machining parameters. Chen and Tsai first proposed a mathematical model for the optimization problem of machining parameters in turnings and then combined Hooke-Jeeves pattern search (PS) into the simulated annealing (SA) algorithm to form a hybrid optimization algorithm (SA/PS) to solve the optimization problem [3]. Onwubolu and Kumalo [4] proposed a genetic algorithm (GA) to optimize the machining parameters in turnings but did not consider the constraint that the number of rough passes must be an integer. Chen and Chen [5] pointed out this shortcoming in the research of Onwubolu and Kumalo [4]. However, the optimization results obtained by the GA corrected by Chen and Chen were not better than those obtained by SA/PS. Additionally, based on a GA, Sankar [6] used a modified genetic algorithm (MGA) to search for optimized cutting parameters in turnings. The improved MGA used a specific crossover operator and three different mutation operators to enhance the diversity of the population and prevent the algorithm from converging to a local optimal solution. In addition to SA and GA, some studies applied other intelligent algorithms to the optimization problem of machining parameters. Vijayakumar [7], Wang [8], and Xie and Guo [16] developed new heuristic algorithms to overcome optimization problems based on the ant colony optimization (ACO) algorithm. In addition, the particle swarm optimization algorithm (PSO) is also one of the most widely used swarm intelligence methods [18]. Srinivas et al. proposed a PSO algorithm where the inertia coefficient decreased linearly with every iteration to solve the cutting parameter optimization problem [9]. Yildiz [10] and Costa et al. [12] also contributed different solutions based on the PSO algorithm to the problem. After comprehensive analysis of the previous research methods and results, Raja and Baskar [11] applied three optimization algorithms (SA, GA, and PSO) to three different machining parameter optimization models (single-pass turning, multipass turning, and surface grinding) to conduct experiments and compare the results of various types of intelligent optimization algorithms to the machining parameter optimization problem. The results showed that the optimization effect and computational efficiency of PSO are better than those of SA and GA. Scatter search (SS) is one of the optimization algorithms developed in the field of metaheuristics. Chen [19] focused on the application of the scatter search method in solving the optimization problem in turnings. By comparing it with other algorithms, the experimental results showed that the SS obtained superior machining parameters than some of the metaheuristic methods.

In recent years, in addition to the abovementioned classical intelligent algorithms, some new swarm intelligent algorithms have been proposed by researchers. Xu et al. proposed an improved flower pollination algorithm (FPA) and compared the obtained results with those of related studies [20]. Mellal and Williams used the cuckoo optimization algorithm (COA), one of the advanced bioinspired optimization algorithms, to minimize the unit production cost [21]. The experimental results showed that the COA algorithm is very competitive compared with other algorithms. Due to the successful application of swarm intelligence algorithms for optimization problems, Sofuoğlu et al. used three heuristic algorithms, GA, PSO and COA, to solve three different problems, which were more efficient and effective than other algorithms [22]. Similarly, Yildiz developed a new hybrid optimization algorithm to minimize the production cost by adding the Taguchi method that actively acted on the differential evolution algorithm to form a hybrid Taguchi-differential evolution algorithm (HRDE) [23]. The results showed that the hybrid algorithm was more effective than evolutionary algorithms presented in many related studies. In another work by Yildiz, he proposed a similar hybrid optimization method to determine the optimal machining parameters [24]. This method combined the differential evolution algorithm and receptor editing algorithm (DERE). The goal of the mathematical model was to determine the optimal machining parameters to reduce the unit production cost. The method has been experimentally proven to be an effective technique for optimizing machining parameters. Furthermore, in 2013, Yildiz proposed a parameter optimization method based on the artificial bee colony (ABC) algorithm [25] and a hybrid robust teaching-learning-based optimization algorithm (HRTLBO) based on the combination of guided learning optimization and the Taguchi method [26]. Compared with other methods, these proposed algorithms perform well, and better solutions can be found with them. Belloufi et al. provided specific application examples to illustrate the

effectiveness of the proposed firefly algorithm (FA) for parameter optimization in multipass turnings [27].

3. Mathematical Model for Optimizing Machining Parameters in CNC Turnings

To optimize the machining parameters in multipass turnings, the mathematical model proposed in the literature [3, 20] takes a large number of actual machining constraints into account and is closer to real-world machining. Since the model has been cited in many studies, the optimization model is used in this paper. The cutting parameters to be optimized include rough cutting speed V_r, rough feeding rate f_r, rough depth of cut d_r, the number of rough cuts n, finish cutting speed V_s, finish feeding rate f_s, and finish depth of cut d_s. The unit production cost (UC) consists of the following four components:

(1) Machining cost during real cutting time C_M

(2) Machine idle cost for setup operations and tool idling motion C_I

(3) Cost of tool replacement C_R

(4) Tool cost C_T

Thus, the UC can be expressed as follows:

$$\begin{aligned} UC &= C_M + C_I + C_R + C_T, \\ &= \left[\frac{\pi DL}{1000V_r f_r}\left(\frac{d_t - d_s}{d_r}\right) + \frac{\pi DL}{1000V_s f_s}\right]k_0, \\ &\quad + \left[t_c + (h_1 L + h_2)\left(\frac{d_t - d_s}{d_r} + 1\right)\right]k_0, \\ &\quad + \left[\frac{\pi DL}{1000V_r f_r}\left(\frac{d_t - d_s}{d_r}\right) + \frac{\pi DL}{1000V_s f_s}\right]\frac{t_e}{T_p}k_0, \\ &\quad + \left[\frac{\pi DL}{1000V_r f_r}\left(\frac{d_t - d_s}{d_r}\right) + \frac{\pi DL}{1000V_s f_s}\right]\frac{k_t}{T_p}. \end{aligned} \tag{1}$$

where k_0 is the sum of worker cost and management cost per unit time (\$/min). D and L are the diameter and length of the work-piece (mm), respectively. d_t is the depth of material to be removed (mm). h_1, h_2 are the constants related to tool idle time and tool-in/out time, respectively. t_c, t_e are the preparation time for loading and unloading time (min) and time required to exchange a tool (min), respectively. T_p is the tool life (min). k_t is the cutting edge cost (\$/edge).

The number of the rough cut is as follows:

$$n = \frac{d_t - d_s}{d_r},\ n \in z^+. \tag{2}$$

The objective of the model is to minimize the UC (V_r, V_s, f_r, f_s, d_r, d_s, n) under many machining constraints on rough and finish turnings. The constraints are summarized as follows [3–8]:

(1) The upper and lower constraints of V_r, V_s, f_r, f_s, d_r, and d_s

(2) Tool life constraints

(3) Cutting force, cutting power, and surface roughness constraint

(4) Stable cutting region constraint; chip-tool interface temperature constraint

(5) Constraints on the interconnection between roughing and finishing parameters

4. Optimization Algorithm Based on the Bat Algorithm and the Divide-and-Conquer Strategy

4.1. Overview of the Bat Algorithm. The bat algorithm (BA) [28] is a swarm intelligence search algorithm proposed to simulate the echolocation mechanism of bats when foraging. It achieves the localization of search targets by continuously adjusting the frequency and loudness of sound waves. It uses frequency tuning to increase the population diversity and uses automatic scaling to maintain a balance between global and local searches. The frequency, velocity, and position of the i-th bat in the bat population are represented as follows:

$$\begin{aligned} f_i &= f_{\min} + (f_{\max} - f_{\min})\beta, \\ v_i^t &= v_i^{t-1} + \left(v_i^t - gb^t\right)f_i, \\ x_i^t &= x_i^{t-1} + v_i^t. \end{aligned} \tag{3}$$

Here, f_i denotes the ultrasound frequency emitted by the i-th bat. $f_{\max}$ and $f_{\min}$ denote the upper and lower bounds of ultrasound frequency, respectively. β denotes a random number generated by a uniform distribution within the range of [0,1]. v_t^t and v_t^{t-1} denote the velocity of the i-th bat during the t-th and (t−1)-th iterations, respectively. x_i^t and x_i^{t-1} denote the position value of the i-th bat during the t-th and (t−1)-th iterations, respectively. The above expressions ensure the global search capability of the algorithm.

In the local search phase, BA uses a random walk strategy to generate feasible solutions at candidate locations. This strategy can be given by the following equation:

$$x_{\text{new}} = x_{\text{old}} + \varepsilon \overline{A}^t. \tag{4}$$

where ε is the random number generated by the uniform distribution on the range [−1, 1] that determines the direction of the new candidate feasible solution and $\overline{A}^t$ denotes the average acoustic loudness of all bats in the t-th iteration.

During the foraging process, as the bat approaches the foraging target, the bat will gradually adjust the loudness A and emission rate r of the ultrasound, making the loudness gradually decrease and the emission rate gradually increase to achieve more accurate positioning. The process is shown in the following equations:

$$A_i^t = \alpha A_i^{t-1}. \tag{5}$$

$$r_i^t = r_i^0\left[1 - \exp(-\gamma t)\right]. \tag{6}$$

In formula (6), r_i^0 denotes the initial pulse emission rate of the i-th bat, and both α and γ are constants between (0, 1).

4.2. Overview of the Quantum-Behaved Bat Algorithm with Mean Best Position Directed. Due to the lack of population diversity in the original BA, there is the problem of falling into local optima during the search. By analyzing the flight trajectory of bats, Zhu et al. proposed the quantum-behaved bat algorithm with mean best position directed (QMBA) [29]. The quantum computing mutation operator introduced by the algorithm can enhance population diversity and avoid premature convergence. At the same time, its average optimal position introduced in the local search phase can improve the convergence speed in the later stage of the search. The QMBA still retains the main body of the BA, which controls the global search and local search based on the ultrasonic loudness and the sending rate. The difference lies in its improved position update formula and local search strategy.

The position update formula in QMBA introduces a mechanism for adaptively adjusting the step size according to the distance. Its strategy for updating the position is expressed as follows:

$$x_{i\,d}^{t} = \begin{cases} x_{i\,d}^{t-1} + \left(gb_d - x_{i\,d}^{t-1}\right)\eta, & \delta_d > TH \\ x_{i\,d}^{t-1} + \epsilon, & \delta_d \le TH \end{cases}, \tag{7}$$

where η is the random number generated through the uniform distribution probability function between [0,1]. δ_d represents the distance between the d-th dimensional value of the current global optimum position and the d-th dimensional value of the i-th bat, which is mathematically expressed as follows:

$$\delta_d = \left|gb_d - x_{i\,d}^{t-1}\right|. \tag{8}$$

When δ_d is less than a given threshold TH, the i-th bat can fly for food at will; when the value of δ_d is greater than a given threshold TH, the i-th bat flies toward the current global optimal position. This strategy ensures the global search ability of the bat population to fly toward food.

In the local search process, QMBA no longer uses the random walk search strategy but decides the selection of the mutation strategy based on the mutation probability p_{m}.

The first mutation strategy is to use the quantum computing mutation operator, which is expressed as follows:

$$x_{i\,d}^{t} = \begin{cases} gb_d^t + \mu \times \left|m\mathrm{best}_d - x_{i\,d}^{t}\right| \times \ln(1/U), r\mathrm{and} < 0.5 \\ gb_d^t - \mu \times \left|m\mathrm{best}_d - x_{i\,d}^{t}\right| \times \ln(1/U), r\mathrm{and} \ge 0.5 \end{cases}, \tag{9}$$

where both U and *rand* are the random numbers generated by the uniform distribution probability function between [0,1]. μ is an adaptive linearly decreasing weighting factor that can be expressed as follows:

$$\mu = \mu_{\max} - \frac{t\left(\mu_{\max} - \mu_{\min}\right)}{t_{\max}}. \tag{10}$$

where $\mu_{\max}$ and $\mu_{\min}$ denote the initial and final values of μ, respectively.

*m*best represents the average of the current optimal position of all bats during the t-th iteration, i.e., the average optimal position, which can be obtained from the following equation:

$$\mathrm{mbest} = \frac{1}{M}\left(\sum_{i=1}^{M} P_{i1}^{t}, \sum_{i=1}^{M} P_{i2}^{t}, \cdots, \sum_{i=1}^{M} P_{i\,D}^{t}\right), \tag{11}$$

where P_i^t denotes the current optimal position of the i-th bat, M denotes the population size, and D denotes the dimension of the problem.

The second mutation strategy, which also introduces the average optimal position into the mutation operator, is expressed as follows:

$$x_i^t = x_i^{t-1} + \left(\mathrm{mbest} - x_i^{t-1}\right)\phi, \tag{12}$$

where ϕ denotes the random number generated by a uniform distribution between [0,1].

Both mutation strategies introduce the average optimal position to guide the local search, which can improve the accuracy of the search and speed up the convergence of the algorithm due to the use of statistical information of bat positions.

4.3. Gaussian Quantum Bat Algorithm with Direction of Mean Best Position. The QMBA algorithm improves the global search capability and accuracy of the algorithm by introducing the mechanism of the distance adaptive adjustment step, quantum computing mutation operator, and average optimal position-oriented mechanism on the basis of BA. However, the probability density functions used to generate the random numbers in QMBA are all uniformly distributed. Several works [30, 31] have shown that long-tailed distributions such as Gaussian distributions are able to perform more accurate searches in the region near the previous generation of individuals, improving the local search capability while providing larger search steps and random walk distances. Expanding the search space can improve the ability of the algorithm to jump out of the local optimum. Based on the above findings, this paper proposes the Gaussian quantum bat algorithm with direction of mean best position (GQMBA) for quantum behavior bats using a Gaussian distribution [32].

In GQMBA, random numbers are no longer generated by the uniformly distributed probability density function. To meet the requirements of the quantum computing mutation operator for random numbers in QMBA, we use the absolute value of the Gaussian distribution probability density function, in which the mean is zero and the variance is one instead (i.e., normal distribution). The one-dimensional probability density function of abs(N(0, 1)) is expressed as follows:

$$q(x) = \frac{2}{\sqrt{2\pi}} exp\left(-\frac{x^2}{2}\right), \; x \ge 0, \tag{13}$$

GQMBA modifies the three formulas in QMBA accordingly. First, the parameter η in (14) is changed to be generated with a Gaussian distribution, and (7) is modified in GQMBA as follows:

$$x_{id}^{t} = \begin{cases} x_{id}^{t-1} + \left(gb_d - x_{id}^{t-1}\right)G, & \delta_d > TH \\ x_{id}^{t-1} + \epsilon, & \delta_d \le TH \end{cases}, \tag{14}$$

where $G = abs(N(0,1))$.

Similarly, substituting for the parameter U in (9), the modified quantum computing mutation operator is expressed as follows:

$$x_{id}^{t} = \begin{cases} gb_d^t + \mu \times \left|m\text{best}_d - x_{id}^t\right| \times \ln(1/G), & r\text{and} < 0.5 \\ gb_d^t - \mu \times \left|m\text{best}_d - x_{id}^t\right| \times \ln(1/G), & r\text{and} \ge 0.5 \end{cases} \tag{15}$$

Since $q(0) = 0$, $G = abs(N(0,1))$ satisfies the domain of definition of the function ln().

Finally, the random number ϕ in (12) is replaced with (16) as follows:

$$x_i^t = x_i^{t-1} + \left(\text{mbest} - x_i^{t-1}\right)G. \tag{16}$$

The pseudocode of GQMBA is given by Figure 1, in which Np denotes the total number of bats.

Our first significant contribution is that the Gaussian distribution is introduced in QMBA to generate random numbers. The theoretical analysis above shows that the strategy can enhance the ability of the algorithm to jump out of the local optimum and avoid premature convergence. Therefore, it is applied to the optimization problem of the cutting parameter in this paper.

4.4. Divide-and-Conquer Strategy for the Optimization Problem in Multipass Turnings. To improve the performance of the algorithm, the idea of the divide-and-conquer strategy is used to decompose the original problem into several subproblems, which can reduce the complexity of the original optimization problem. For each subproblem, the number of rough cuts is a fixed value. By conquering the subproblems one by one, the whole optimization problem can be solved. In addition, we calculate the theoretical lower bound on UC for each subproblem. Modified BA is first used to search for the optimal solution in the case of the minimum theoretical lower bound on UC, thus hopefully reducing the enumeration of the subproblems. The divide-and-conquer strategy is depicted in Figure 2 and described as follows:

(1) Divide the optimization problem into m subproblems based on the number of possible combinations of rough cuts.

(2) Calculate the theoretical lower bound on UC for each subproblem UC_{iL}.

(3) Sort theoretical lower bound UC_{iL} for all subproblems in ascending order. $UC_{1L} \le UC_{2L}, \ldots, \le UC_{mL}$ are called the first theoretical lower bound, the second theoretical lower bound ..., the m-th theoretical lower bound, and the corresponding numbers of rough cuts $N_1, N_2, \ldots, N_m$ (N_i is the number of rough cuts corresponding to UC_{iL}) [16].

(4) Starting from subproblem i, BA is used to solve subproblem i, and the optimal solution, UC_{iO}, is obtained.

(5) If all subproblems are enumerated or the UC_{iO} found is less than the theoretical lower bound of subsequent subproblems $UC_{(i+1)\mathrm{L}}$, the method terminates and the optimal solution is output.

4.5. The Framework of the Proposed Algorithm Based on GQMBA and the Divide-and-Conquer Strategy. By dividing the complicated multipass turning optimization problem into simple subproblems, the optimization problem can be solved by solving these subproblems one by one. The framework of the optimization algorithm based on GQMBA and the divide-and-conquer strategy (referred to as the GQMBA-DC algorithm) is shown in Figure 3, and the main steps are as follows:

(1) Divide the optimization problem into n subproblems based on the number of possible combinations of rough cuts.

(2) Let $i = i + 1$; set the number of rough cuts $n = N_i$; and start the search in the i-th subproblem.

(3) Initialize the population, develop appropriate encoding and decoding strategies for each subproblem for GQMBA, and set the current iteration number $t = 1$.

(4) Initialize the parameters and set the ultrasonic frequency f_i, ultrasonic emission rate r_i, and ultrasonic loudness A_i.

(5) The global search and local search are controlled by continuously adjusting the acoustic frequency and loudness to update the speed and position to generate new solutions. For the GQMBA-DC, the main body of the BA is retained, but the position update formula and the local search strategy are different. The position update formula introduces a mechanism for adaptively adjusting the step size according to the distance, while the local search strategy also introduces the average optimal position to guide the local search.

(6) If A_i is greater than the random value *rand* and the current solution is the optimal solution, perform the next step; otherwise, return to step (5).

(7) Accepting the new solution increases r_i and decreases A_i (as a bat gets closer to the target, the two values change to achieve more accurate localization).

(8) Repeat the above steps until the maximum number of iterations is reached, and the GQMBA stops.

(9) At present, the optimal solution of the i-th subproblem is obtained.

(10) If $Min(UC_{1O}..., UC_{iO}) \le U_{(i+1)L}$ or $i = m$, then execute the next step. Otherwise, return to step (2) to start the process of solving the next subproblem.

Algorithm 1: Pseudocode of GQMBA.

Initialize the position x_i and velocity v_i of bats ($i = 1,2,\cdots,Np$);

Setup the pulse frequency f_i, pulse rate r_i and the loudness A_i;

while ($t < t_{max}$) **do**

for $i = 1\ to\ Np$ **do**

Calculate the distance between the position of i-th bat (x_i) and the current global best position, and then update the position by using formula (10) and formula (16);

if (rand $>$ r_i) **then**

if (rand $>$ p_m) **then**

This bat flies with quantum behavior and its position is updated using formulas (12), (13) and (17);

else

Guided by the mean best position, the position of i-th bat is updated using formulas (13) and (18);

end if

end if

if (rand$<$ A_i && $f(x_i) < f(gb)$) **then**

Accept the new solution to update x_i;

Increate r_i and reduce A_i using formulas (5) and (7), respectively;

end if

Rank bats to find the current global best location gb;

end for

$t = t + 1$;

end while

Figure 1: Pseudocode of GQMBA.

(11) Select the minimum solution, which is the best optimal solution, from the obtained optimized solutions. Finally, output the global optimal solution $UC_o = Min(UC_{1O}, \ldots, UC_{iO})$, and terminate the algorithm.

4.6. Handling of Constraints. The processing of constraints is very important for the swarm intelligence optimization algorithm; constraint processing by adding a penalty function is one of the common methods in optimization algorithms. The penalty function is a kind of constraint function. In the process of finding the optimal solution of the algorithm, the objective function is calculated by combining the penalty function, which can gradually eliminate solutions that do not satisfy the constraints and retain solutions that satisfy the constraints.

For the handling of constraints in the optimization algorithm, the bats (individuals) that violate the constraints are penalized using a penalty function to reduce the value of the objective. Different levels of penalties are imposed for different constraint violations. The more constraints that are violated, the heavier the penalty will be. Thus, by using a reasonable penalty function, the objective function value can converge to the direction of the optimal solution. The penalty function is expressed as

$$\text{penalty}(X) = \sum_{i=1}^{k} a_i + h_i,\ \text{where}\ a_i = \begin{cases} 0, & \text{satisfy constraints} \\ 1, & \text{violate constraints} \end{cases}, \tag{17}$$

where k is the number of constraints and h_i is a nondimensional constraint violation.

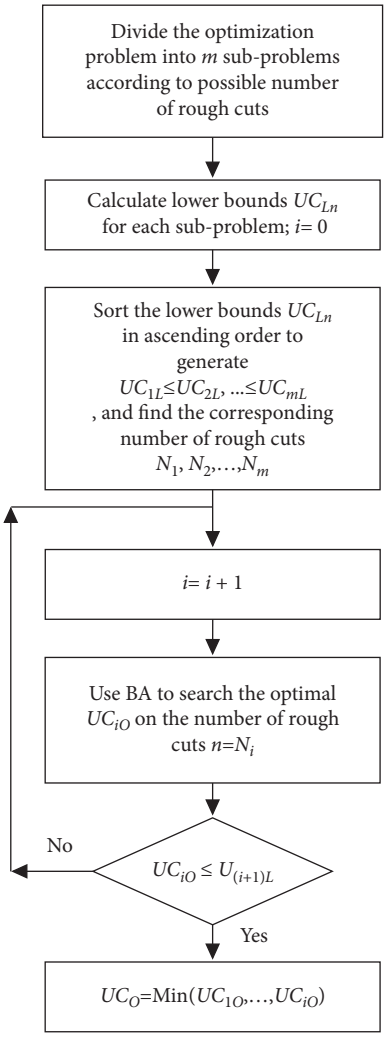

Figure 2: Flow chart of the divide-and-conquer strategy.

5. Simulation Experiments

During the machining processes, a cut tool is used for both roughing cuts and finishing cuts. Due to different machining conditions, the tool wear rates for rough and finish turnings are usually different. The tool life equation can be expressed as follows: $T_p = \theta T_r + (1 - \theta) T_s$. Some studies use another tool life calculation formula: $T_p = T_r + T_s$. Therefore, this paper separately compares the performance of the algorithms under different tool life formulations. The algorithm in this paper is implemented in the MATLAB programming language. The parameters of the BA are set as follows:

Population size: 200

Maximum number of iterations: 400

Initial loudness: $A = u(0, 1)$

Initial pulse emission rate: $r_0 = 0.001$

Loudness update: $\alpha = 0.9$

Emission rate update: $\gamma = 0.9$

Threshold: $TH = 0.005$

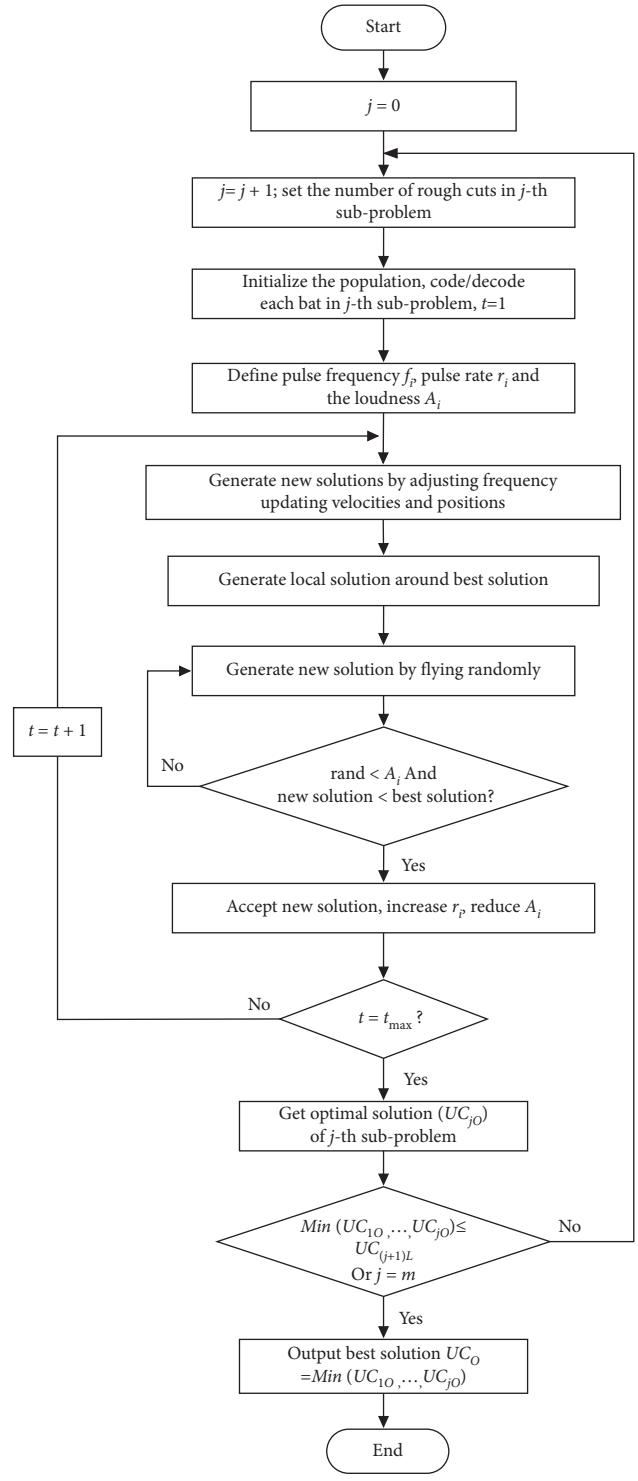

Figure 3: The framework of the proposed algorithm (GQMBA-DC) based on GQMBA with a divide-and-conquer strategy.

Mutation probability: $P_m = 0.01$

Machining examples from the literature [3–8] were used to test the performance of the optimization algorithm with the specific parameters shown in Table 1. Additionally, two different machining optimization problems with cutting depths of 6 mm and 8 mm were tested.

Table 1: Condition parameters for turning examples.

D = 50 mm	L = 300 mm	d_t = 6 mm	V_{rL} = 50 m/min
f_{rL} = 0.1 mm/rev	d_{rL} = 1 mm	V_{rU} = 500 m/min	f_{rU} = 0.9 mm/rev
d_{rU} = 3 mm	V_{sL} = 50 m/min	f_{sL} = 0.1 mm/rev	d_{sL} = 1 mm
V_{sU} = 500 m/min	f_{sU} = 0.9 mm/rev	d_{sU} = 3 mm	p = 5
q = 1.75	r = 0.75	μ = 0.75	ν = 0.95
η = 0.85	λ = 2	υ = −1	τ = 0.4
ϕ = 0.2	δ = 0.105	R = 1.2 mm	C_0 = 6×1011
T_L = 25 min	T_U = 45 min	F_U = 200 kgf	P_U = 5 kW
SC = 140	Q_U = 1000 °C	SR_U = 10 μm	h_1 = 7×10−4
h_2 = 0.3	t_e = 1.5 min/edge	t_c = 0.75 min/piece	k_t = 2.5 $/edge
k_0 = 0.5 $/min	k_1 = 108	k_2 = 132	k_3 = 1
k_4 = 2.5	k_5 = 1		

The algorithm was run 100 times independently on a Windows platform (CPU E3 3.5 GHz and 16 GB memory). The average value of UC was given and compared with the results obtained by previous algorithms, such as SA/PS [3], FE-GA [5], MGA [6], ACO [6], and PSO [9]. The average UC, standard deviation, number of search points, and running time for each algorithm are shown in Tables 2-5. The best results have been underlined and bolded in Tables 2-5.

Tables 2–4 show that the average UCs obtained by the proposed GQMBA-DC are smaller than those given by other algorithms. The standard deviations of the results are small, which in turn indicates that the algorithm is stable. The proposed algorithm can find the optimization results within 30 seconds for different tool life formulas and cutting depths, which shows that the proposed algorithm is an efficient algorithm. Specifically, as shown in Tables 2 and 3, the results of both cases of GQMBA-DC outperform PSO [9] when the tool life equation is $T_p = T_r + T_s$. As shown in Table 4, when the tool life equation is $T_p = \theta T_r + (1-\theta)T_s$, the proposed GQMBA-DC can save 10% compared with the result given by the MGA [6]. Compared with other algorithms, such as SA/PS [3], FE-GA [5], HC [6], NM [6], ACO [6], and DP-FS [33], GQMBA-DC can further save production costs. Because the case of $d_t = 8$ cm is not covered in previous literature, only the results of our algorithm are given in Table 5. Thus, the above experimental results show that the GQMBA-DC algorithm can effectively solve the optimization problem of cutting parameters to find optimal machining parameters, which, in turn, can further reduce the production cost.

From the perspective of the optimal UC value, comparisons between the proposed GQMBA-DC and other algorithms were also conducted. The comparison of the optimal UC values in Tables 6–9 shows that the optimal UC results obtained by the GQMBA-DC are almost always smaller than the optimal results of UC obtained by other algorithms without constraint violation. The best results have been underlined and bolded in Tables 6–9. Specifically, the results found by the proposed algorithm are comparable to those achieved by HPSO [12], FPA [20], and COA [21] for the tool life equation of $T_p = T_r + T_s$ and $d_t = 6$ mm, which is only one ten-thousandth of the difference, as shown in Table 6. The GQMBA-DC can further reduce the production cost compared with the other algorithms (i.e., HRDE [23], DERE [24], DE [24], and HRTLBO [26]), as shown in Table 6. A similar situation can also be found in Tables 7–9 in different test examples. In addition, the optimal combination of cutting parameters (V_r, V_s, f_r, f_s, d_r, d_s) for different cases is also given in Tables 6–9. The GQMBA-DC algorithm can find better results than the previously proposed algorithms in terms of both the average UC and the best UC. Thus, it is clear that the proposed GQMBA-DC can perform significantly better than other algorithms on solution quality in CNC turnings. Therefore, the algorithm combining the modified BA with the divide-and-conquer strategy is effective.

Table 2: Comparison of average UC among different algorithms (when $T_p = T_r + T_s$, $d_t = 6$ mm).

Algorithm	Average UC ($)	Standard deviation	Search points (pcs)	Running time (sec)
PSO [9]	>2.2721	N/A	2,000	N/A
GQMBA-DC	**1.9592**	0.00005	80,000	28.4

Table 3: Comparison of average UC among different algorithms (when $T_p = T_r + T_s$, $d_t = 8$ mm).

Algorithm	Average UC ($)	Standard deviation	Search points (pcs)	Running time (sec)
PSO [9]	>3.306	N/A	2,000	N/A
GQMBA-DC	**2.4393**	0.00084	80,000	30.4

To overcome the different complex optimization problems in various fields, we need to carefully consider the characteristics of the specific problem and use the specific characteristics (domain knowledge) to design the optimization algorithm. In our work, for the optimization problem of machining parameters, since the machining process can be divided into different numbers of roughing cuts, we decompose the whole optimization problem of machining parameters into several simple subproblems according to the different numbers of roughing cuts. Each subproblem can be conquered individually, which greatly reduces the space of the problem solution. At the same time, to avoid enumerating all subproblems and save calculation time, we derived the theoretical lower bound on UC for each subproblem by

Table 4: Comparison of average UC among different algorithms (when $T_p = \theta T_r + (1\text{-}\theta)T_s$, $d_t = 6$ mm).

Algorithm	Average UC ($)	Standard deviation	Search points (pcs)	Running time (sec)
SA/PS [3]	2.2959	0.01624	$18{,}571 \times 5$	27.4
FE-GA [5]	2.3091	N/A	60,000	N/A
HC [6]	2.3017	N/A	100,000	N/A
NM [6]	2.2713	N/A	100,000	N/A
ACO [6]	2.2705	N/A	100,000	N/A
MGA [6]	2.2538	N/A	100,000	N/A
DP-FS [33]	2.2974	7.6×10^{-4}	16074×9	19.3
GQMBA-DC	**2.0280**	0.00015	80,000	27.9

Table 5: Comparison of average UC among different algorithms (when $T_p = \theta T_r + (1\text{-}\theta)T_s$, $d_t = 8$ mm).

Algorithm	Average UC ($)	Standard deviation	Search points (pcs)	Running time (sec)
GQMBA-DC	**2.5499**	0.00043	80,000	29.2

Table 6: Comparison of different algorithms (when $T_p = T_r + T_s$, $d_t = 6$ mm).

Algorithm	Cutting speed (m/min)		Feed rate (mm/rev)		Depth of cut (mm)		UC ($/piece)	Constraint violation
	V_r	V_s	F_r	f_s	d_r	d_s		
GQMBA-DC	123.3360	169.9697	0.5655	0.2262	3	3	**1.9592**	0
HPSO [12]	123.3424	169.9783	0.5655	0.2262	3	3	**1.9591**	0
FPA [20]	123.3431	169.9785	0.5655	0.2262	3	3	**1.9591**	0
COA [21]	123.1462	169.9876	0.5655	0.2262	3	3	**1.959**	0
GA [4]	1114.22	164.369	0.7	0.2978	2.9745	2.9863	1.7842	0.5148
ACO [7]	103.05	162.02	0.9	0.24	3	3	1.8450	0.5396
PSO [9]	106.69	155.89	0.897	0.28	2	2	2.2721	0
HRDE [23]	–	–	–	–	–	–	2.0461	–
AIA [23]	–	–	–	–	–	–	2.12	–
DERE [24]	–	–	–	–	–	–	2.046	–
ABC [24]	–	–	–	–	–	–	2.118	–
DE [24]	–	–	–	–	–	–	2.136	–
HABC [25]	–	–	–	–	–	–	2.046	–
HRTLBO [26]	–	–	–	–	–	–	2.046	–
FA [27]	98.4102	162.2882	0.82	0.2582	3	3	1.824	(24)

Table 7: Comparison of different algorithms (when $T_p = T_r + T_s$, $d_t = 8$ mm).

Algorithm	Cutting speed (m/min)		Feed rate (mm/rev)		Depth of cut (mm)		UC ($/piece)	Constraint violation
	V_r	V_s	f_r	f_s	d_r	d_s		
GQMBA-DC	119.1460	164.2166	0.6564	0.2625	2.6673	2.6613	**2.4384**	0
HRDE [23]	–	–	–	–	–	–	2.4791	–
AIA [23]	–	–	–	–	–	–	2.51	–
DERE [24]	–	–	–	–	–	–	2.4793	–
HABC [25]	–	–	–	–	–	–	2.4790	–
ABC [24]	–	–	–	–	–	–	2.503	–
DE [24]	–	–	–	–	–	–	2.512	–

using the characteristics of the subproblems. Then, the algorithm first searches the solution space from the subproblem with a smaller theoretical lower bound on UC. By following these steps, the algorithm can quickly find the optimal solution to the problem.

On the other hand, the performance of the combination of traditional divide-and-conquer strategy and swarm intelligence algorithm is better than the algorithms that only use traditional mathematical methods or swarm intelligence algorithms, as proven by the simulation experiments.

Table 8: Comparison of different algorithms (when $T_p = \theta T_r + (1\text{-}\theta)T_s$, $d_t = 6$ mm).

Algorithm	Cutting speed (m/min)		Feed rate (mm/rev)		Depth of cut (mm)		UC ($/piece)	Constraint violation
	V_r	V_s	f_r	f_s	d_r	d_s		
GQMBA-DC	109.6727	169.9756	0.5655	0.2262	3	3	**2.0278**	0
SA-PS [3]	–	–	–	–	–	–	2.3135	0.0667
HPSO [9]	109.6655	169.9796	0.5655	0.2262	3	3	2.0351	0
FPA [20]	109.6631	169.9785	0.5655	0.2262	3	3	2.0351	0
COA [21]	117.9322	123.1993	0.5655	0.2262	3	3	2.2390	0

Table 9: Comparison of different algorithms (when $T_p = \theta T_r + (1\text{-}\theta)T_s$, $d_t = 8$ mm).

Algorithm	Cutting speed (m/min)		Feed rate (mm/rev)		Depth of cut (mm)		UC ($/piece)	Constraint violation
	Vr	Vs	Fr	fs	Dr	ds		
GQMBA-DC	106.0251	164.2238	0.6563	0.2624	2.6670	2.6660	**2.5495**	0
SA-PS [3]	–	–	–	–	–	–	2.7411	0

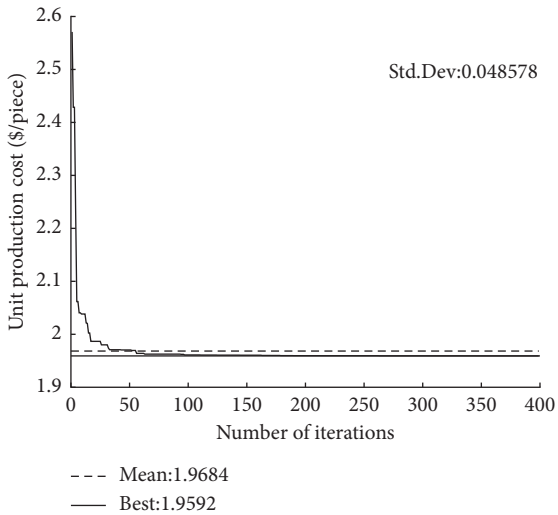

Figure 4: Convergence curve of the proposed GQMBA-DC (when $T_p = T_r + T_s$, $d_t = 6$ mm).

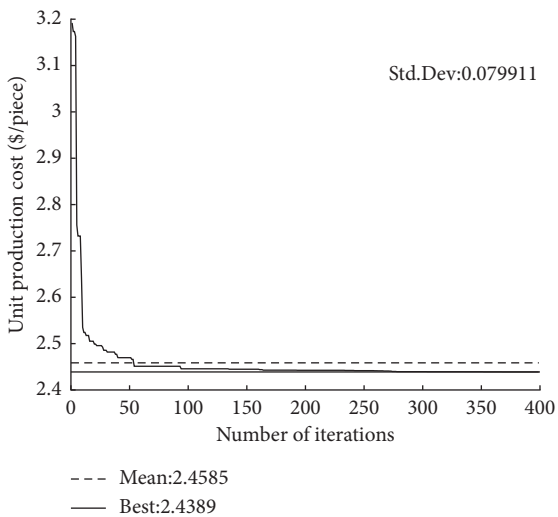

Figure 5: Convergence curve of the proposed GQMBA-DC ($T_p = T_r + T_s$, $d_t = 8$ mm).

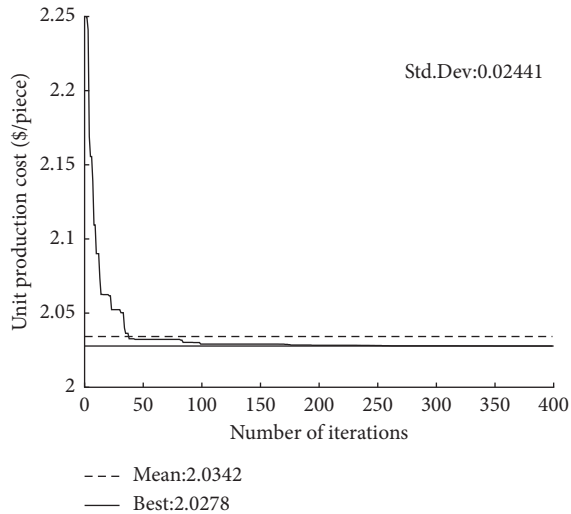

Figure 6: Convergence curve of the proposed GQMBA-DC (when $T_p = \theta T_r + (1\text{-}\theta)T_s$, $d_t = 6$ mm).

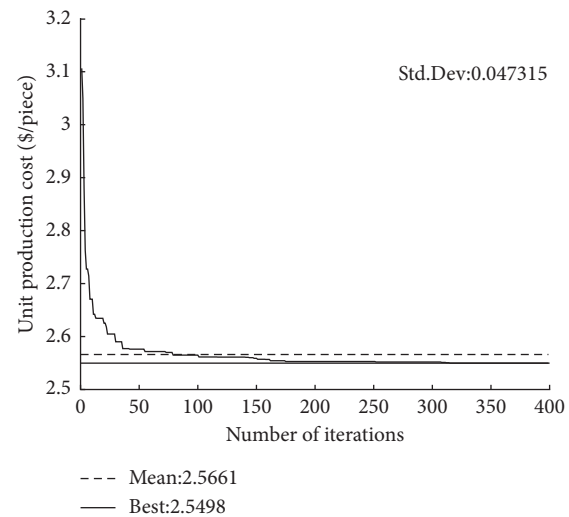

Figure 7: Convergence curve of the proposed GQMBA-DC (when $T_p = \theta T_r + (1\text{-}\theta)T_s$, $d_t = 8$ mm).

The convergence curves of GQMBA-DC for different mathematical models (tool life equation) and test cases are shown in Figures 4–7. The proposed algorithm converges to the final solution after approximately 150 generations, which indicates that the algorithm converges quickly to find satisfactory results.

6. Conclusions and Future Work

To solve the nonlinear optimization problem of machining parameters in CNC turnings, this paper proposes an optimization algorithm combining the bat algorithm and the divide-and-conquer strategy. First, based on the classical BA, the Gaussian quantum bat algorithm with direction of mean best position (GQMBA) is proposed by using a Gaussian distribution to generate random numbers. Second, the divide-and-conquer strategy is used to divide the complicated optimization problem into several subproblems and conquer them one by one. The simulation results show that the GQMBA-DC algorithm proposed in this paper has a stronger search capability than previous algorithms. Specifically, the proposed algorithm can find a better cutting parameter set and further reduce the production cost.

Future research can be considered from two aspects. From the algorithmic point of view, the emerging swarm intelligence algorithm can also be applied to the optimization problem, which may be able to find a better combination of machining parameters, thus reducing costs. In recent years, deep learning methods have been widely applied in various studies; deep learning methods may be considered to reconstruct mathematical models in the optimization of turning parameters [34]. On the other hand, from the perspective of new machining types, to improve the machining efficiency and quality, there are multiple tools to realize machining operations simultaneously in modern CNC turnings. Therefore, research on this type of machining optimization problem is also of great concern.

Data Availability

The data used to support the findings of this study are included in the paper.

Conflicts of Interest

The authors declare that they have no conflicts of interest to report regarding the present study.

Acknowledgments

This work was supported in part by the Natural Science Foundation of Fujian Province of China (Nos. 2020J01699 and 2020J01697).

References

[1] M. Chandrasekaran, "Application of soft computing techniques in machining performance prediction and optimization: a literature review," *International Journal of Advanced Manufacturing Technology*, vol. 46, no. 5, pp. 445–464, 2010.

[2] I. Mukherjee and P. K. Ray, "A review of optimization techniques in metal cutting processes," *Computers & Industrial Engineering*, vol. 50, no. 1-2, pp. 15–34, 2006.

[3] M.-C. Chen and D.-M. Tsai, "A simulated annealing approach for optimization of multi-pass turning operations," *International Journal of Production Research*, vol. 34, no. 10, pp. 2803–2825, 1996.

[4] G. C. Onwubolu and T. Kumalo, "Optimization of multipass turning operations with genetic algorithms," *International Journal of Production Research*, vol. 39, no. 16, pp. 3727–3745, 2001.

[5] M.-C. Chen and K.-Y. Chen, "Optimization of multipass turning operations with genetic algorithms: a note," *International Journal of Production Research*, vol. 41, no. 14, pp. 3385–3388, 2003.

[6] R. S. Sankar, "Selection of machining parameters for constrained machining problem using evolutionary computation," *International Journal of Advanced Manufacturing Technology*, vol. 32, no. 9-10, pp. 892–901, 2007.

[7] K. Vijayakumar, "Optimization of multi-pass turning operations using ant colony system," *International Journal of Machine Tools and Manufacture*, vol. 43, no. 15, pp. 1633–1639, 2003.

[8] Y.-C. Wang, "A note on optimization of multi-pass turning operations using ant colony system," *International Journal of Machine Tools and Manufacture*, vol. 47, no. 12-13, pp. 2057–2059, 2007.

[9] J. Srinivas, R. Giri, and S.-H. Yang, "Optimization of multi-pass turning using particle swarm intelligence," *International Journal of Advanced Manufacturing Technology*, vol. 40, no. 1, pp. 56–66, 2009.

[10] A. R. Yıldız, "A novel particle swarm optimization approach for product design and manufacturing," *International Journal of Advanced Manufacturing Technology*, vol. 40, no. 5, pp. 617–628, 2009.

[11] S. B. Raja and N. Baskar, "Optimization techniques for machining operations: a retrospective research based on various mathematical models," *International Journal of Advanced Manufacturing Technology*, vol. 48, no. 9, pp. 1075–1090, 2010.

[12] A. Costa, G. Celano, and S. Fichera, "Optimization of multi-pass turning economies through a hybrid particle swarm optimization technique," *International Journal of Advanced Manufacturing Technology*, vol. 53, no. 5, pp. 421–433, 2011.

[13] A. Jeang, H.-C. Li, and Y.-C. Wang, "A computational simulation approach for optimising process parameters in cutting operations," *International Journal of Computer Integrated Manufacturing*, vol. 23, no. 4, pp. 325–340, 2010.

[14] G. Singh, "An evolutionary approach for multi-pass turning operations," *Proceedings of the Institution of Mechanical Engineers - Part B: Journal of Engineering Manufacture*, vol. 220, no. 2, pp. 145–162, 2006.

[15] L. Tang, R. G. Landers, and S. N. Balakrishnan, "Parallel turning process parameter optimization based on a novel heuristic approach," *Journal of Manufacturing Science and Engineering-Transactions of the Asme*, vol. 130, no. 3, pp. 031002–0310012, 2008.

[16] S. Xie and Y. Guo, "Optimisation of machining parameters in multi-pass turnings using ant colony optimisations," *International Journal of Machining and Machinability of Materials*, vol. 11, no. 2, pp. 204–220, 2012.

[17] W. H. Bangyal, "Comparative analysis of low discrepancy sequence-based initialization approaches using population-based algorithms for solving the global optimization problems," *Applied Sciences*, vol. 11, no. 16, p. 7591, 2021.

[18] S. Pervaiz, "A systematic literature review on particle swarm optimization techniques for medical diseases detection," *Computational and Mathematical Methods in Medicine*, 2021.
[19] M. C. Chen, "Optimizing machining economics models of turning operations using the scatter search approach," *International Journal of Production Research*, vol. 42, no. 13, pp. 2611–2625, 2004.
[20] S. Xu, Y. Wang, and F. Huang, "Optimization of multi-pass turning parameters through an improved flower pollination algorithm," *International Journal of Advanced Manufacturing Technology*, vol. 89, no. 1-4, pp. 503–514, 2017.
[21] M. A. Mellal and E. J. Williams, "Cuckoo optimization algorithm for unit production cost in multi-pass turning operations," *International Journal of Advanced Manufacturing Technology*, vol. 76, no. 1, pp. 647–656, 2015.
[22] M. A. Sofuoğlu, F. H. Çakır, and S. Gürgen, "An efficient approach by adjusting bounds for heuristic optimization algorithms," *Soft Computing*, vol. 23, no. 13, pp. 5199–5212, 2019.
[23] A. R. Yildiz, "Hybrid Taguchi-differential evolution algorithm for optimization of multi-pass turning operations," *Applied Soft Computing*, vol. 13, no. 3, pp. 1433–1439, 2013.
[24] A. R. Yildiz, "A comparative study of population-based optimization algorithms for turning operations," *Information Sciences*, vol. 210, pp. 81–88, 2012.
[25] A. R. Yildiz, "Optimization of cutting parameters in multi-pass turning using artificial bee colony-based approach," *Information Sciences*, vol. 220, pp. 399–407, 2013.
[26] A. R. Yildiz, "Optimization of multi-pass turning operations using hybrid teaching learning-based approach," *International Journal of Advanced Manufacturing Technology*, vol. 66, no. 9-12, pp. 1319–1326, 2013.
[27] A. Belloufi, M. Assas, and I. Rezgui, "Intelligent Selection Of Machining Parameters In Multipass Turnings Using Firefly Algorithm," *Modelling and Simulation in Engineering*, vol. 2014, Article ID 592627, , 2014.
[28] X.-S. Yang, "A new metaheuristic bat-inspired algorithm," in *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)*, J. R. González, Ed., Springer, Berlin, Heidelberg, 2010.
[29] B. Zhu, "A Novel Quantum-Behaved Bat Algorithm with Mean Best Position Directed for Numerical Optimization," *Computational Intelligence and Neuroscience*, Article ID 6097484, 2016.
[30] X. Cai, "Bat algorithm with Gaussian walk," *International Journal of Bio-Inspired Computation*, vol. 6, no. 3, pp. 166–174, 2014.
[31] L. dos Santos Coelho, "Gaussian quantum-behaved particle swarm optimization approaches for constrained engineering design problems," *Expert Systems with Applications*, vol. 37, no. 2, pp. 1676–1683, 2010.
[32] X. Huang, "Gaussian Quantum Bat Algorithm with Direction of Mean Best Position for Numerical Function Optimization," *Computational intelligence and neuroscience*, vol. 2019, 2019.
[33] Y. C. Shin and Y. S. Joo, "Optimization of machining conditions with practical constraints," *International Journal of Production Research*, vol. 30, no. 12, pp. 2907–2919, 1992.
[34] W. H. Bangyal, "Detection of fake news text classification on COVID-19 using deep learning approaches," *Computational and mathematical methods in medicine*, vol. 2021, Article ID 5514220, , 2021.