ARTIFICIAL BEE COLONY – DIFFERENTIAL EVOLUTION (ABC-DE) ALGORITHM FOR CNC TURNING PROCESS OPTIMIZATION

R.S.S. PRASANTH[1], K. HANS RAJ[2]
[1, 2] Faculty of Engineering, Dayalbagh Educational Institute, India

ABSTRACT

Optimization of any machining process is a highly challenging task per se, because it essentially involves prediction of optimal cutting parameters and operating constrains that are complex and very nonlinear in nature and affect the total production time which in turn affects production cost and quality of the work piece. Artificial Bee Colony (ABC) algorithm is a nature inspired algorithm (NIA) for process optimization, which mimics the intelligent foraging behavior of honey bees. This paper proposes to improve the rate of convergence of ABC algorithm by integrating Differential Evolution (DE) operators with ABC technique to develop ABCDE algorithm for the optimization of CNC turning process model to minimize the production time by predicting the optimal the process parameters such as cutting speed, feed rate for different depth of cuts, while satisfying the process constraints such as cutting force, cutting power, chip tool interface temperature, and surface roughness. The performance of ABCDE on process optimization is ascertained by comparing its results with Real Coded Genetic Algorithm integrated with Laplace Crossover and Power Mutation operator (RCGA-LXPM), Differential Evolution (DE), and ABC algorithms in terms of standard performance metrics. The results suggest that ABC-DE algorithm outperforms RCGA - LXPM, DE and ABC algorithms.

KEYWORDS - NATURE INSPIRED ALGORITHMS, SWARM INTELLIGENCE, ARTIFICIAL BEE COLONY, DIFFERENTIAL EVOLUTION, CNC TURNING.

1.0 INTRODUCTION

Machining is a manufacturing process that involves removal of unwanted metal from the work piece. Such manufacturing process is complex because while machining it is crucial to maintain the dimensional accuracy, surface integrity, surface roughness and metal removal rate while adhering to operating ranges of machine tool and not violating dynamic machining constraints. For the past few decades we have been witnessing a rapid up-gradation of conventional machine tools to computer controlled machine tools, in manufacturing sector. But the economics of computer controlled machine tools essentially depends upon minimizing the production cost by increasing the production rate by reducing the production time. Therefore, to derive best performance from a CNC turning machine it is imperative to optimize its process by determining the optimal cutting parameters. For precision manufacturing, traditional parameter selection which relies on handbooks and operator expertise is inappropriate, as such approach to estimate optimal process parameters is limited to a specific machining environment. And also, conventional techniques of process variable selection more often fails to produce repeatable results and neglect the economics of machining in dynamic manufacturing environment. And thus there is a definite need of robust optimization techniques that can give us optimal or near optimal ranges of process variables in dynamically changing machining environments. Mathematical optimization techniques although are useful to certain extent but they are not robust as they depend on initial solution, get stuck in sub optima, are problem specific and computationally difficult in case of multi variable and multi object problems and thus they are not suitable for dynamic process settings [1, 2].

Due to the fact that, there is a definite financial constraints involved in all the product manufacturing process activities, and thus there is an imperative need of optimization techniques. Since manufacturing processes are exceedingly complex and non linear in nature with number of process variables and dynamic constraints, their optimization is indeed a challenging task. And therefore there is a compelling need of intelligent optimization techniques that can determine optimal parameters to address the limitations of conventional optimization algorithms. Therefore, researchers are exploring, Nature Inspired Algorithms (NIAs) [3] that are flexible and easy to implement than their counterparts. Even in industrial environments NIAs are steadily gaining acceptance. Among NIAs, swarm intelligence inspired algorithms, such as Ant Colony Optimization (ACO), Particle Swarm Optimization (PSO), Bees Algorithm (BA) etc., have gained popularity because of their efficiency [4].

Swarm intelligence is as defined by Bonabeau, as " … any attempt to design algorithms or distributed problem solving devices inspired by the collective behavior of social insect colonies and other animal societies …" [5]. One among such recent swarm intelligence inspired algorithms is Artificial Bee Colony (ABC) algorithm [5] which is increasingly inviting the attention of many researchers due to its computational efficiency over Genetic Algorithms (GAs), Differential Evolution (DE) and PSO [6-8].

And, thus efforts are underway to integrate such NIAs into precision manufacturing equipped with computer controlled machine tools to strike a balance between quality and cost. As regard to the optimization of turning process, researchers [9, 10] developed a mathematical model of average surface roughness in turning process on steel alloy work pieces, and predicted optimal cutting parameters using a Real Coded Genetic Algorithm (RCGA). In order to achieve improved the surface finish, on aforesaid turning process, DE and ABC algorithms were applied [11-12] and found that DE is better than RCGA and ABC achieves improved surface roughness over DE. In their study [13] authors applied quantum inspired evolutionary algorithm for solving plane turning problem. In an another study [14] a hybrid algorithm i.e., Particle Swarm and Receptor Editing-PSRE is applied on product design and manufacturing and compared its performance against Hybrid Genetic Algorithm (HGA), Scatter Search Algorithm (SSA), Genetic Algorithm (GA), and Simulated Annealing (SA) integrated with Hooke-Jeeves Pattern Search (HJPS). Researchers attempted [15] to concurrently optimize the manufacturing cost of piston and cylinder components by optimizing the operating parameters of the machining processes by using GA without violating any constraint. PSO is reportedly hybridized with SA [16] to optimize a multi pass turning problem and its performance with PSO, SA, and Ant Colony Optimization (ACO). On similar lines, researchers [17] developed a GA for intelligent prediction of process parameters in multi pass turning and compared its performance with PSO, SA, and SSA. A new variant of RCGA equipped with Laplace Crossover operator and Power Mutation operator (LXPM) [18] was developed and applied on CNC turning optimization problem and estimated optimal cutting variables. The performance of RCGA-LXPM algorithm was further compared with several other optimization algorithms like DE, PSO, GA, SA, Nelder Mead Simplex (NMS) algorithm, and Binary Space Partitioning (BSP) algorithm. It is reported that that PSO, RCGA-LXPM and DE performed better than GA, NMS, SA, and BSP. However, if

LXPM and DE are compared to each other, then it is observed that LXPM performed better than DE.

According to a recent review, although ABC and its improved versions are finding wider applicability, ABC still suffers from inherent limitation such as slow convergence in local minimum [19]. Some studies reported [20, 21] cascading Differential Evolution (DE) algorithm and replacement of neighborhood operator of ABC with mutation operator of DE for improved results, but such attempts chip away the utility of crossover operator of DE which facilitates uniform distribution of population. Therefore, this paper attempts to integrate mutation and crossover operators of DE into ABC algorithm so as to improve the quality of the solution and rate of convergence of ABC algorithm for such constrained CNC turning process optimization problem in order to determine the optimal cutting parameters such as cutting speed, and feed rate for different depth of cuts while obeying the constraints of cutting force, cutting power, temperature and surface roughness within the specified boundaries of process variables. This is paper is further organized as follows. In section two, the empirical model for optimization of CNC turning process is presented followed by description of ABC and DE algorithms. The integration of DE operators into ABC algorithm to form Artificial Bee Colony – Differential Evolution (ABC-DE) algorithm is presented in section three. The results are compared and analyzed on standard performance metrics such as best, standard deviation, number of fitness evaluations, in section four. In section five, conclusions are drawn accordingly which are followed by references.

2.0 EMPIRICAL MODEL OF CNC TURNING PROCESS

The empirical model of CNC turning process is to minimize the production time which is the sum of machining time, tool changing time, quick return time and work piece handling time. The model is adopted form [22, 23], that considers cutting parameters such as cutting speed, feed rate and depth of cut besides four constraints i.e., cutting force, cutting power, surface roughness, and temperature.

$$T_u = t_m + t_{cs}\,(t_m /T) + t_r + t_h \qquad \text{…. (1)}$$

Where cutting time (per pass)

$$t_m = \Pi DL / 1000V f \qquad \text{…. (2)}$$

Taylor's series tool life equation may be given as

$$V f^{a1} doc^{a2} T^{a3} = K \quad \text{.....(3)}$$

Subject to:

(i) Boundary conditions:

$$V_{min} \leq V \geq V_{max} \quad \text{.....(4)}$$

$$f_{min} \leq f \geq f_{max} \quad \text{.....(5)}$$

$$doc_{min} \leq doc \geq doc_{max} \quad \text{.....(6)}$$

(ii) Constraints:

Four practical constraints such as cutting force, cutting power, chip-tool temperature and surface roughness are considered in this model.

$$F = 844\, V^{-0.010133} f^{0.725} doc^{0.75} \leq F_{max} \quad \text{.... (7)}$$

$$P = 0.0373\, V^{0.91} f^{0.78} doc^{0.75} \leq P_{max} \quad \text{.... (8)}$$

$$\theta = 74.96\, V^{0.4} f^{0.2} doc^{0.105} \leq \theta_{max} \quad \text{.... (9)}$$

$$R = 14.785\, V^{-1.52} f^{1.004} doc^{0.25} \leq R_{max} \quad \text{.... (10)}$$

Nomenclature and values

Symbol	**Description**	**Value**
D	Diameter of the work piece (mm)	152
L	Length of the work piece (mm)	203
V_{min}	Minimum cutting speed (m/min)	30
V_{max}	Maximum cutting speed (m/min)	200
f_{min}	Minimum feed rate (mm/rev)	0.254
f_{max}	Maximum feed rate (mm/rev)	0.762
R	Surface roughness (μm)	
R_{max}	Maximum surface roughness of rough and finish cut (μm)	50
P_{max}	Maximum power of the machine (KW)	5
F_{max}	Maximum cutting force (N)	900

Θ_{max}	Maximum temperature of tool work piece interface (^{0}C)	550
doc_{min}	Minimum depth of cut (mm)	2.0
doc_{max}	Maximum of depth of cut (mm)	5.0
T	Tool life (min)	
t_m	Machining time (min)	
t_{cs}	Tool change time (min/edge)	0.5
t_h	Loading and unloading time (min/pass)	1.5
t_r	Quick return time (min/pass)	0.13
T_u	Total production time (min)	
C_0	Operating cost (RS/piece)	3.5
C_t	Tool cost per cutting edge (RS/edge)	17.5
C_T	Total production cost (RS/edge)	
a_1,a_2, a_3, K	Constants used in tool life equation	0.29; 0.35; 0.25; 193.3

3.0 ARTIFICIAL BEE COLONY (ABC) ALGORITHM

Artificial Bee Colony (ABC) algorithm is a nature inspired algorithm that mimics the intelligent foraging behavior of honey bees. ABC algorithm essentially simulates the interactions of three artificial agents namely employee bees, onlooker bees and scout bees. These bees perform a specific role in finding rich food sources so as to maximize the quality and quantity of nectar deposited at the hive. Working of ABC algorithm can be better explained in its three phases. In the initialization phase, the basic control parameters (Number of Food sources, Number of Iterations, Modification Rate and the Limit or Scout Production Period) are initialized and the artificial scouts are allowed to randomly generate the population of food sources and assign one employee bee to a randomly generated food source. In the employee bee phase, the employee bee searches for the better neighborhood food sources. Once the neighborhood food source is located, the employee bee applies the greedy selection strategy to find the best of its known two food sources and shares acquired information with the onlooker bees waiting at the hive, by dancing on the dancing area. The onlooker bee gathers the information from the dancing pattern, tempo and the duration of the employee bee and adopts the employee bee food source probabilistically. Once the onlooker bee chooses its food source, the onlooker bee agents further searches for its neighborhood solution and applies the greedy selection strategy to find the best its two food sources. The food sources, that were not improved, even after a predefined number of trials, will be rendered to scouts for random search. All the three phases are repeated until a termination condition (number of cycles or the CPU time) is satisfied.

The pseudo-code of artificial bee colony algorithm [15] is presented below:

```
1. Initialize the Colony Size (CS), Number of Food Sources/Solutions (SN), Number of dimensions to each solution (D), Modification Rate(MR), SPP(Scout Production Period-limit)
2. Initialize the population of solutions $x_{i,j}$    where i= 1 .. SN and j= 1 .. D
3. Evaluate the population
4. cycle = 1
5. **REPEAT**
6. Produce a new solution $v_i$ for each employed bee by using (1) $V_{ij} = X_{ij} + \varnothing_{ij} (X_{ij} - X_{kj})$,
   if $R_j < MR$, otherwise $X_{ij}$    …… (11)
   [$\varnothing_{ij}$ - is a random number in the range $[-1, 1]$. $k \in \{1, 2 \dots SN\}$ (SN: Number of solutions in a colony) is randomly chosen index. Although k is determined randomly, it has to be different from *i*. $R_j$ is a randomly chosen real number in the range [0, 1] and $j \in \{1, 2 \dots D\}$ (D: Number of dimensions in a problem). MR, modification rate, is a control parameter.]
7. Apply greedy selection process for the employed bees between the $v_i$ and $x_i$
8. Calculate the probability values Pi using (2) for the solutions $x_i$
   $P_i = Fitness_i / \sum_{N=1}^{SN} (Fitness_N)$    …… (12)
```

9. For each onlooker bee, produce a new solution v_i by using (1) in the neighborhood of the solution selected depending on P_i and evaluate it.
10. Apply greedy selection process for the onlooker bees between the v_i and x_i
11. If Scout Production Period (SPP) is completed, determine the abandoned solutions by using "limit" parameter for the scout, if it exists, replace it with a new randomly produced solution using (3)

$$X^j_i = X^j_{min} + \text{rand}\,(X^j_{max} - X^j_{min}) \quad \ldots\ldots(13)$$

12. Memorize the best solution achieved so far
13. cycle = cycle +1
14. **UNTIL** (Max Cycle Number or Max CPU time)

3.10 Differential Evolution Algorithm

Differential Evolution (DE) algorithm [24] is another popular numerical optimization algorithm that is similar to GA, as it uses mutation, crossover, and selection operators. However DE is different from GA, as it largely depends on its mutation operator, while GA is based on crossover operator. In DE, mutation operator provides good search mechanism that produces difference of randomly selected pairs of solutions in the population and provides diversity, while a crossover operator maintains the uniform distribution of population. The selection process ensures that the fitness value of the population always improves or remains unchanged but does not get deteriorate. The key steps of DE [24] algorithm is:

- Initialize
- Evaluate
- Repeat
    - Mutate
    - Crossover
    - Evaluate
    - Select
- Until requirements are met

Mutation: A mutant solution vector $\hat{x}_i$ is produced by equation

$$\hat{x}_i = x_{r1} + F\,(x_{r2} - x_{r3}) \quad \ldots\ldots (14).$$

Where F is the scaling factor in the range of [0, 1], and solution vectors x_{r1}, x_{r2}, x_{r3} are randomly chosen and must satisfy the condition x_{r1}, x_{r2}, x_{r3} | $r_1 \neq r_2 \neq r_3 \neq I$, where I is the index of the current solution.

Crossover: The parent vector is mixed with mutant vector to produce a trail vector using

$$y^j_i = \hat{x}^j_i \text{ if } R_j \leq CR, \text{ otherwise } x^j_i \qquad \ldots\ (15).$$

CR is a crossover constant which controls the recombination and varies between [0, 1]. And R_j is a random number between [0, 1].

Selection: All solutions of the population have equal chance of being selected as parents irrespective of their fitness value. The child produced after the mutation and crossover operations is evaluated. And then, the performance of the child vector and its parent vector is compared and the better one will be stored. If the parent is still better, it is retained in the population [24].

3.20 Artificial Bee Colony – Differential Evolution (ABC-DE) algorithm

Striking perfect balance between exploration and exploitation of any random population based search algorithm is a challenging task, because such balance determines the success of any evolutionary algorithm. Although ABC algorithm exhibits reasonably good balance between exploration and exploitation, however it suffers from certain inherent limitations, as is the case of many evolutionary algorithms. It is observed that, since ABC algorithm does not use crossover operator, that is employed in GA or DE, the distribution of good information between solutions may be lost or it is not at the required level. This causes the convergence performance of ABC for local minimum to be slow [19]. Further it is also observed, as DE algorithm employs a crossover operator which makes it converge faster in initial generations but which may lead to premature convergence in case of multi-modal problems since it does not have an operator maintaining sufficient diversity [25]. A comprehensive survey of state of the art of DE [26] indicates that for large problem landscapes DE get stuck in local optima and has limited ability to move its population to larger distances of solution spaces and therefore DE also requires improvement. It is an observed fact that no one algorithm can find best solution to all optimization problems, the advantageous features of two different algorithms may be used to form a hybrid or augmented technique so as to improve the quality of the solutions [27]. Though, ABC and DE algorithms appear to be using similar kind of operators that weigh the difference of solutions, but the DE algorithm uses a constant scaling factor which is greater than zero, while the ABC algorithm uses a random number in the range [−1, 1], therefore they are intrinsically different. Therefore, in order to improve the quality of the solution and the convergence rate of ABC algorithm it is proposed hybridize ABC algorithm by embedding a mutation and crossover operators of DE algorithm in onlooker bee phase. And thus is this paper we present ABC algorithm's integration with DE's mutation and crossover operators in its onlooker phase, since onlooker phase represent global search. The hybrid ABC algorithm called, ABC-DE algorithm features the following modifications to the original ABC algorithm, keeping all its other steps unaffected.

1. In step 9, for each onlooker bee produce new solutions v_i using differential mutation operator and crossover operators, i.e., using equations (4) and (5), in the neighborhood of the solutions selected depending on p_i and evaluate it.
2. The scaling factor F, employed in mutation operator of DE which is a constant and greater than 0 (F>0) has been changed to vary in the range of [0, 1] and thus making the algorithm self adaptive.

3.21 Handling of constraints

Constraint handling is the huge concern in solving the constrained manufacturing optimization problems, as the choice of the constraint handling technique has a definite impact on efficiency of any stochastic search technique. Among many constraint handling mechanisms [28], because of its simplicity and effectiveness Deb's feasibility rules [29] are adopted for handling constraints in the current optimization problem. In this method tournament selection is used, that is two solutions are compared against each other, according to the following order of preference of feasible solution.

- If both solutions are feasible, the one with better objective function value wins.
- If one solution is feasible and the other unfeasible, the feasible one wins.
- If both solutions are infeasible, the one with lower constraint violation wins.

3.30 Working of ABC-DE algorithm

Step 0: Set the control parameters such as colony size (CS), number of food sources (SN), number of dimension of the optimization problem (D), maximum limit of scout production period (Limit), modification rate (MR), maximum cycle number (MCN) for which the algorithm would repeat its process, and crossover rate (CR).

Step 1: Initialize required number random solutions as equal to the size of CS for all the number of variables that are equal to the size of D.

Step 2: Now generate random original solutions by evaluating the fitness function for all the randomly initialized solution variables in the previous step and store it.

Step 3: Now initialize MCN counter to 1.

Step 4: Now generate a random number raging [0, 1] and compare it with a control parameter MR, if the random number is less than the MR then, by employing the neighborhood operator of ABC algorithm generate neighborhood sources equal to the size of SN and store the neighborhood solutions, which are employee bees.

Step 5: Now check for boundary violations of solution variables and if there are any repair them accordingly using a repair algorithm

Step 6: Now evaluate the fitness of the neighborhood solution variables

Step 7: Now check for any constraint violations and if there are any such violations apply Deb's constraint handling rules, and store the best feasible solution.

Step 8: Now apply a greedy selection process between the randomly generated original solutions and their neighborhood solutions generated in the previous step using neighborhood operator of ABC algorithm, and best value is stored.

Step 9: Now compute the probability of a particular solution that may selected for the assignment of onlooker bee phase, by dividing the every fitness value stored in the previous step with the maximum fitness value of the stored fitness values. And, now compare the computed probability of an individual solution with a random number ranging from [0, 1] and if the probability of an individual solution is less than the random number then assign or consider such solution as an onlooker bee.

Step 10:	Now at this stage of onlooker bee phase process, generate new solution for each onlooker bee by employing the mutation and crossover operators of DE.
Step 11:	Now check for boundary violations of solution variables and if there are any repair them accordingly using a repair algorithm
Step 12:	Now evaluate the fitness of the neighborhood solution variables
Step 13:	Now check for any constraint violations and if there are any such violations apply Deb's constraint handling rules, and store the best feasible solution.
Step 14:	And now activate greedy selection process between the randomly generated original solutions and the solutions that are produced by employing the DE operators in the previous step.
Step 15:	Now at this stage check whether there are any redundant solutions that have not improved after certain number of cycles, i.e., the scout production period Limit is exhausted. Now produce new random solutions using the neighborhood operator of ABC algorithm's scout bee phase.
Step 16:	Now memorize the best solutions achieved so far and continue the process by incrementing the cycle number until the requirements are met.

3.31 Parameters of algorithm

ABC and ABC-DE algorithms are implemented in MATLAB 7.0, and tested on a Laptop machine equipped with Intel Centrino Duo processor, 512 MB RAM, and 150 GB HDD. The following control parameters are adopted for this study.

Parameters of algorithms

Parameter	ABC	ABC-DE
Colony size (CS)	40	40
Number of food sources (SN)	20	20
Modification rate (MR)	0.9	0.9
Limit	MCN/4	MCN/4

Crossover rate (CR)	-	0.5
Scaling Factor (F)	-	[0-1]
Maximum cycle number (MCN)	500	500
Number of independent runs	100	100

4.0 RESULTS

Optimal cutting parameters such as feed rate, cutting speed for different depth of cut values and the corresponding total production time, obtained using ABC and ABC-DE are compared with RCGA-LXPM, and DE, in Table 1. The results indicate that ABC and ABC-DE algorithms outperformed RCGA-LXPM and DE. It is further observed that, ABC-DE has performed highly competitive to ABC, and in fact at fifth digit level ABC-DE has reduced the total production time which indicates its increased potential in finding quality solution in precision manufacturing environment.

Table no 1. Comparison of optimal parameters predicted by RCGA-LXPM, DE, ABC and ABC-DE for minimum production time

d	**RCGA- LXPM [18]**			**DE [18]**			**ABC**			**ABC-DE**		
	v	***f***	***T_U***	***v***	***f***	***T_U***	***v***	***f***	***T_U***	***v***	***f***	***T_U***
2	139.26	0.762	2.78	139.26	0.761	2.77	150.7987	0.7620	2.54344	150.8053	0.7620	**2.543400604242**
2.5	129.07	0.762	2.87	129.07	0.761	2.87	129.0802	0.7620	2.68014	129.0799	0.7620	**2.680134103801**
3	122.72	0.686	3.06	121.56	0.685	3.06	121.5745	0.6858	2.86215	121.5658	0.6858	**2.862125025945**
3.5	122.43	0.585	3.30	122.61	0.585	3.31	122.6193	0.5854	3.06109	122.6166	0.5854	**3.061077613434**
4	134.54	0.517	3.55	123.53	0.510	3.57	123.5359	0.5104	3.25924	123.5343	0.5104	**3.259206144301**
4.5	127.92	0.454	3.82	124.34	0.452	3.83	124.3534	0.4523	3.45667	124.3494	0.4523	**3.456617014652**
5	132.15	0.410	4.08	125.08	0.405	4.09	125.0974	0.4059	3.6535	125.0831	0.4059	**3.653392503927**

Where, d is Depth of cut in mm; v is Cutting speed in m/min; f is Feed rate in mm/rev; T_U is Total production time in min

Standard performance metrics of algorithm such as mean, standard deviation, number of function evaluations of RCGA-LXPM, DE, ABC and ABC-DE are presented in tables numbered 2, 3 and 4 respectively. Mean represents the average fitness value achieved by the algorithm for number of independent runs. Standard deviation indicates the robustness and stability as it is the deviation from the best fitness value achieved by the algorithm for number of independent runs. Average numbers of fitness evaluations reflect the rate of convergence of the algorithm.

Table no 2. Comparison of standard performance metrics of RCGA-LXPM and DE on CNC turning process optimization

Depth of cut (*d*) in mm	**RCGA-LXPM [18]**				**DE [18]**			
	Mean fitness value (T_U)	Standard deviation	Average function evaluations	Average computational time in seconds	Mean fitness value (T_U)	Standard deviation	Average function evaluations	Average computational time in seconds
2	2.780401	4.135e-05	308	0.0453	2.78	0.00147145	801.8	0.0489
2.5	2.8733757	1.40e-04	338	0.0468	2.87276	2.16583e-005	826.2	0.0542
3	3.0660640	0.0024726	405	0.0460	3.06573	0.0022352	993.2	0.0558
3.5	3.336070	0.0294902	465	0.0474	3.31947	0.00303094	1000	0.0681
4	3.5686617	0.0108513	426	0.0462	3.57587	0.000988971	997.8	0.0586
4.5	3.8364324	0.017543	474	0.1045	3.83784	0.0214352	999.2	0.0599
5	4.0990033	0.012372	414	0.0752	4.09841	0.00449684	996.6	0.0574

As we may observe from the table no. 2 that RCGA-LXPM's performance is better than DE [18]. Table no.1 and 2 suggest that RCGA-LXPM has not only achieved competitive results in terms of its fitness value with respect to DE but also the convergence rate is also improved. In order to ascertain the performance of ABC algorithm on CNC turning process optimization problem and improve the quality of the solution we have first applied ABC algorithm and the results are presented in table given below.

From following table no. 3, it may be observed that ABC can produce not only more quality solution but also it is more stable and faster than RCGA-LXPM. However, it required slightly more number of fitness evaluation to converge to the best solution in the only first case where depth of cut is at its minimum. Average computational time in seconds is also tabulated for every case and it may be observed that the performance of ABC is significantly better than RCGA-LXPM. In all other cases of different depth of cuts ABC's performance is better than the RCGA-LXPM.

Table no 3. Comparison of standard performance metrics of RCGA-LXPM and ABC on CNC turning process optimization

Depth of cut (*d*) in mm	**RCGA-LXPM [18]**				**ABC**			
	Mean fitness value (T_U)	Standard deviation	Average function evaluations	Average computational time in seconds	Mean fitness value (T_U)	Standard deviation	Average function evaluations	Average computational time in seconds
2	2.780401	4.135e-05	308	0.0453	2.54342	3.50497e-005	332.44	0.001087
2.5	2.8733757	1.40e-04	338	0.0468	2.6802	5.53916e-005	276.0600	0.001169
3	3.0660640	0.0024726	405	0.0460	2.86246	0.000212219	289.1100	0.001075
3.5	3.336070	0.0294902	465	0.0474	3.06144	0.000231517	291.5200	0.001027
4	3.5686617	0.0108513	426	0.0462	3.25965	0.000308054	261	0.001302
4.5	3.8364324	0.017543	474	0.1045	3.45725	0.000408364	293.6400	0.001197
5	4.0990033	0.012372	414	0.0752	3.65414	0.000382429	270.3300	0.001131

The proposed new hybrid algorithm named ABC-DE's performance is ascertained by evaluating it on standard performance metrics and compared with ABC in table no. 4. It is observed that ABC-DE has outperformed ABC algorithm in terms of standard metrics such as standard deviation, number of fitness evaluations besides maintaining and producing the very competitive and quality of the solution in every case. These findings suggest that ABC-DE is more stable, faster, and reliable and can produce highly comparable quality solutions with respect to RCGA-LXPM, DE and ABC.

Performance comparison of nature inspired algorithms and traditional optimization methods on CNC turning process optimization is presented in table no. 5, where NMS, BSP, GA, SA, PSO, DE, RCGA-LXPM, ABC, and ABC-DE are compared in terms of their best fitness value achieved. The above results clearly point out that ABC-DE algorithm outperformed all the techniques such as NMS, BSP, GA, SA, PSO, DE, RCGA-LXPM, and ABC in terms of its fitness value.

Table no 4. Comparison ABC and ABC-DE on CNC turning process optimization

Depth of cut (d) in mm	ABC				ABC-DE			
	Mean fitness value (T_U)	Standard deviation	Average function evaluations	Average computational time in seconds	Mean fitness value (T_U)	Standard deviation	Average function evaluations	Average computational time in seconds
2	2.54342	3.50497e-005	332.44	0.001087	2.5434	6.24857e-015	114.8500	0.001095
2.5	2.6802	5.53916e-005	276.0600	0.001169	2.68013	5.35592e-015	156.5500	0.0011
3	2.86246	0.000212219	289.1100	0.001075	2.86213	0	193.0900	0.001491
3.5	3.06144	0.000231517	291.5200	0.001027	3.06108	4.02165e-015	201.3100	0.001376
4	3.25965	0.000308054	261	0.001302	3.25921	4.41841e-016	203	0.001212
4.5	3.45725	0.000408364	293.6400	0.001197	3.45662	5.35592e-015	216.7100	0.001137
5	3.65414	0.000382429	270.3300	0.001131	3.65339	8.92653e-016	197.3000	0.001177

Table no. 5 Comparison of ABC-DE algorithm with different traditional and nontraditional optimization methods for CNC turning process optimization [23]

Depth of cut (d) in mm	ABC-DE fitness value (T_U)	ABC fitness value (T_U)	RCGA-LXPM fitness value (T_U)	DE fitness value (T_U)	PSO fitness value (T_U)	SA fitness value (T_U)	GA fitness value (T_U)	NMS fitness value (T_U)	BSP fitness value (T_U)
2.0	2.5434006	2.54344	2.78	2.77	2.78	2.85	2.85	2.87	2.84
2.5	2.6801341	2.68014	2.87	2.87	2.87	2.93	3.12	2.97	2.93
3.0	2.86212503	2.86215	3.06	3.06	3.04	3.15	3.13	3.15	3.11
3.5	3.06107761	3.06109	3.3	3.31	3.29	3.34	3.46	3.44	3.34
4.0	3.25920614	3.25924	3.55	3.57	3.55	3.59	3.51	3.69	3.59
4.5	3.45661701	3.45667	3.82	3.83	3.82	3.85	3.96	3.88	3.84
5.0	3.6533925	3.6535	4.08	4.09	4.08	4.12	4.14	4.23	4.1

Further, an attempt has been made to pictorially present the effectiveness of the ABC-DE algorithm over its counterparts such as ABC, RCGA-LXPM and DE, by plotting graphs that represent key standard performance metrics, such as number of fitness evaluations required to converge to optimal value, the repeatability or stability of the algorithm for 100 different runs by plotting the standard deviation form the mean fitness, and the convergence behavior of the proposed algorithm during process of finding the best value for different depth cuts. Figure no. 1, depicts the how fast ABC-DE converges against other nature inspired algorithms such as ABC, RCGA-LXPM and DE. Figure no. 2 displays the robustness and stability of the ABC-DE algorithm when compared with ABC, RCGA-LXPM and DE. In figure no.3 convergence behaviour of ABC-DE algorithm for different depth of cut values is presented.

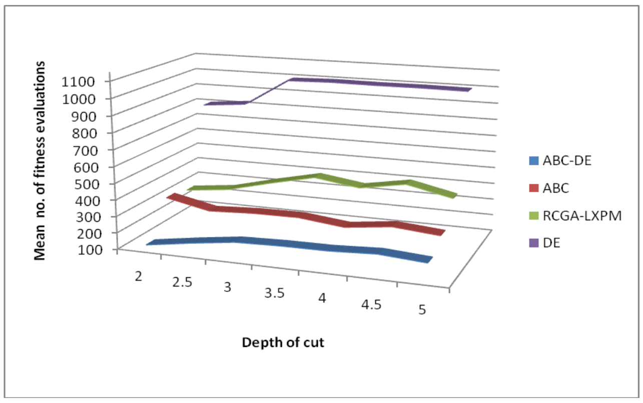

Figure no. 1 Comparision of ABC-DE algorithm interms of its convergence aginast ABC. RCGA-LXPM and DE algorithms

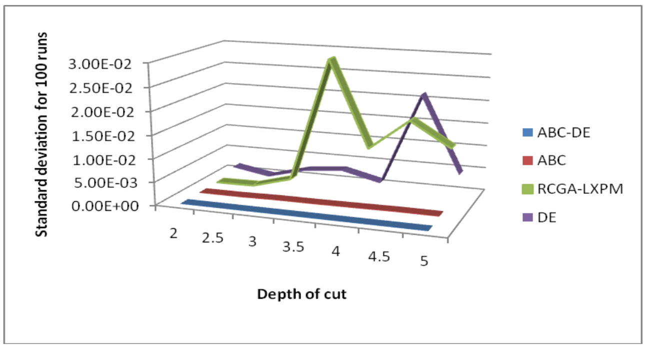

Figure no. 2 Comparision of ABC-DEalgorithm interms of its satbility aginast ABC. RCGA-LXPM and DE algorithms

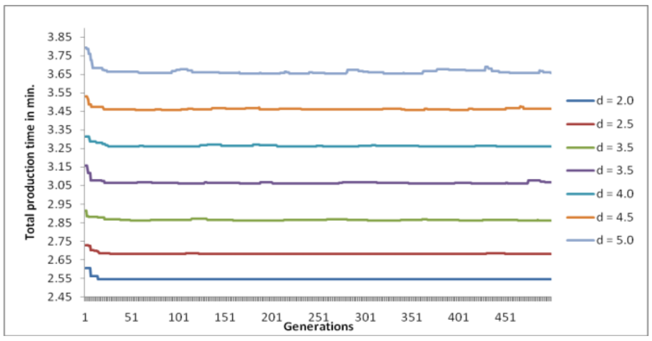

Figure no. 3 Convergence behaviour of ABC-DEalgorithm for different depth of cut values

5.0 CONCLUSIONS

Artificial bee colony - differential evolution (ABC-DE) algorithm is implemented in MATLAB 7.0., and applied on CNC turning process optimization problem to predict optimal cutting parameters such as cutting speed, feed rate for different depth of cut values and estimate minimum total production time. The performance of ABC-DE algorithm on the aforesaid optimization problem is evaluated on standard performance metrics such as mean fitness value, standard deviation and number of fitness evaluation to ascertain its ability to find quality solution, stability and rate of convergence. The results suggest that ABC-DE algorithm, not only achieved improved results in terms of quality solutions but also in terms of its speed, convergence behavior, and stability, when compared to other nature inspired algorithms such as ABC, RCGA-LXPM and DE algorithms. The results indicate the potential and suitability of the ABC-DE algorithm for precession oriented intelligent manufacturing systems.

6.0 ACKNOWLEDGEMENTS

Authors gratefully acknowledge the inspiration and guidance of Most Revered Prof. P. S. Satsangi, the Chairman, Advisory Committee on Education, Dayalbagh, Agra, India.

REFERENCES

1. R. Saravanan, P. Asokan, M. Sachithanandam, Comparative Analysis of Conventional and Non-Conventional Optimization Techniques for CNC Turning Process, Int J Adv Manuf Technol. *17*, 2001, 471-476.

2. R V Rao, P J Pawar, Grinding process parameter optimization using non-traditional optimization algorithms, Proc Inst Mech Eng Part B-J Eng Manuf, *224(B6)*, 2010, 887–898.
3. S Deb, US Dixit, *Intelligent machining: computational methods and optimization. In: Davim JP (ed) Machining: fundamentals and recent advances*. (Springer, London, 2008)
4. J Kennedy, R Eberhart, Particle swarm optimization. In: Proceedings of the IEEE International Conference on Neural Networks (ICNN'95), Perth, Australia, 1995.
5. E Bonabeau, M Dorigo, G Théraulaz , *Swarm intelligence: from natural to artificial systems*. (Oxford University Press; 1999)
6. D Karaboga, *An idea based on honey bee swarm for numerical optimization. Technical report TR06*, (Erciyes University, Engineering Faculty, Computer Engineering Department, 2005) 21
7. D Karaboga, B Basturk, A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (abc) algorithm, Journal of Global Optimization, *39(3)*, 2007, 459–471.
8. Karaboga, D., & Basturk, B, On the performance of artificial bee colony (abc) algorithm, Applied Soft Computing, *8(1)*, 2008, 687– 697.
9. T Srikanth, V Kamala, Experimental determination of optimal speeds for alloy steels in plane turning, Proceedings of the 9th Biennial ASME, 2008.
10. T Srikanth, V Kamala, A Real Coded Genetic Algorithm for Optimization of Cutting Parameters in Turning, International Journal of Computer Science and Network Security (IJCSNS), *8(6)*, 2008, 189-193.
11. R S S Prasanth, K Hans Raj, Application of Differential evolution algorithm for optimizing orthogonal cutting, Proceedings of International Conference on Systemics, Cybernetics and Informatics, ICSCI-2011, Hyderabad, 2011, 122-126.
12. R S S Prasanth, K Hans Raj, Process optimization of plane turning using artificial bee colony algorithm, Proceedings of National Systems conference (NSC- 2011), IIT Bhubaneswar, 2011, 112 – 118.

13. R S S Prasanth, K Hans Raj, Estimation of optimal cutting parameters of plane turning using quantum inspired evolutionary algorithm, International Journal of Modern Engineering Research (IJMER), *2(1)*, 2012, 304 –312 .
14. Ali Rıza Yıldız, A novel particle swarm optimization approach for product design and manufacturing, Int J Adv Manuf Technol, *40*, 2009, 617–628.
15. V JanakiramanV. R Saravanan, Concurrent optimization of machining process parameters and tolerance allocation, Int J Adv Manuf Technol, *51*, 2010, 357–369.
16. Antonio Costa, Giovanni Celano, Sergio Fichera, Optimization of multi-pass turning economies through a hybrid particle swarm optimization technique, Int J Adv Manuf Technol, *53*, 2011, 421–433.
17. Shutong XIE, Yinbiao GUO, Intelligent Selection of Machining Parameters in Multi-pass Turnings Using a GA-based Approach, Journal of Computational Information Systems, *7(5)*, 2011, 1714-1721.
18. Chauhan Pinkey, Kusum Deep, Millie Pant, Optimization of CNC turning process using real coded genetic algorithm and differential evolution, Transaction on Evolutionary algorithm and Continuous Optimization, 2011. ISSN: 2229-8711.
19. D Karaboga, B Aka., A comparative study of artificial bee colony algorithm, Applied Mathematics and Computation, 214, 108–132, (2009).
20. Ajith Abraham, Ravi Kumar Jatoth, A. Rajasekhar, Hybrid Differential Artificial Bee Colony Algorithm, Journal of Computational and Theoretical Nanoscience, *(9)*, 2012, 1–9.
21. Bin Wu, Cun hua Qian, Differential Artificial Bee Colony Algorithm for Global Numerical Optimization, Journal of Computers, 6*(5)*, 2011.
22. Deep K., Thakur M., A new mutation operator for real coded genetic algorithms, Appl. Math. Comput., *193*, 2007, 21-230.
23. Deep K., Bansal J.C., Performance Analysis of Turning Process via Particle Swarm Optimization, Studies in Computational Intelligence, *129*, 2008, 453-460 (2008)
24. R. Storn, K. Price, Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces, Journal of Global Optimization, *11*, 1997, 341–359.

25. Bahriye Akay, Dervis Karaboga, Artificial bee colony algorithm for large-scale problems, and engineering design optimization, J Intell Manuf, 2010. DOI 10.1007/s10845-010-0393-4
26. Swagatam Das, Ponnuthurai Nagaratnam Suganthan, Differential Evolution: A survey of the state of the art, IEEE Transactions on Evolutionary Computation, *15 (1)*, No. 1, 2011, 4-31.
27. C Grosan, A. Abraham, Hybrid Evolutionary Algorithms: Methodologies, Architectures, and Reviews", Studies in Computational Intelligence (SCI), *75*, 2007, 1-17.
28. Efrén Mezura-Montesa, Carlos A. Coello Coello: Constraint - handling in nature inspired numerical optimization: past present and future, Swarm and Evolutionary Computation, *1*, 2011, 173-194.
29. K. Deb, An efficient constraint Handling Method for Genetic Algorithms, Comput. Methods Appl. Mech. Engrg, *186*, 2000, 311-338.