An Investigation of Grinding Process Optimization via Evolutionary Algorithms

T.S. Lee[1], T.O. Ting[2] and Y.J. Lin[3]

[1]Faculty of Engineering and Technology, Multimedia University, Jalan Ayer Keroh Lama, 75450 Malacca, Malaysia
[2]Department of Electrical Engineering, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong
[3]Department of Mechanical Engineering, University of Akron, Akron, Ohio 44325, USA

Abstract – In this paper, the performance of some evolutionary algorithms on grinding process optimization of silicon carbide (SiC) is investigated. The grinding of SiC is not an easy task due to its low fracture toughness, therefore making the material sensitive to cracking. The efficient grinding involves the optimal selection of operating parameters to maximize the Material Removal Rate (*MRR*) while maintaining the required surface finish and limiting surface damage. In this work, optimization based on the available model has been carried out to obtain optimum parameters for silicon carbide grinding via three prominent evolutionary algorithms. They are Particle Swarm Optimization (PSO), Differential Evolution (DE) and Genetic Algorithm (GA). The objective of this optimization process is to maximize the *MRR*, subject to surface finish and damage constraints of the grinding process. Numerical results show that PSO is comparatively superior in comparison with DE and GA algorithms for grinding process optimization in terms of its accuracy and convergent capability.

I. INTRODUCTION

Currently, the usage of advanced structural ceramics such as silicon carbide (SiC), silicon nitride, alumina or partially stabilized zirconiums are demanding for engineering applications. These materials feature a high strength at elevated temperatures, resistance to chemical degradation, wear resistance and low density. The effective use of these structural ceramics in such functional applications demands the grinding of ceramic components with good surface finish and low surface damage. The main concern about the use of ceramics in industry is the complexity involved in machining due to the high hardness and low fracture toughness which involves a high cost of machining. Therefore, the process optimization in ceramic processing and manufacturing technology are necessary for the commercialization of ceramic use.

SiC is a non-oxide ceramic which consists of 9:1 ratio of covalent to ionic bonding. The special features of ceramics with covalent bonding are that they have a low thermal coefficient of expansion and a relatively high thermal conductivity. In regards to these properties, SiC is expected to be used increasingly for heat resistant parts in the work done by Inasaki [1]. The damage introduced during grinding has strong influence on the performance and reliability of ceramic components. Two possible cracks, namely median and radial cracks produced during grinding are responsible for chip formation and eventually material removal. Malkin and Malkin [2] explained that these cracks are due to the energy introduced in the layers close to the surface.

Models are pre-requisite for producing an efficient and high quality grinding tool. Tonshoff et al. [3] described the state-of-the-art modeling and a simulation of the grinding processes, making comparison of the different approaches to modeling. Suresh et al. have developed a surface roughness prediction model and optimize this model using Genetic Algorithms (GAs) in [4]. Meanwhile, Konig and Wemhoner [5] have conducted experiments to determine the optimal conditions for the surface grinding of sintered SiC that result in high *MRR* while improving the strength of the ground components. In their study, no mathematical model was developed and hence it would be difficult to optimize the process. The optimum parameters suggested in their work is based on experimental results, which might not be very accurate. Mayer and Fang [6] have carried out empirical studies on the grinding of hot pressed silicon nitride to obtain the relationships of grit depth of cut and grind direction with respect to the strength and surface characteristics of the ground sample. It was reported that no loss of strength for grinding in longitudinal direction as the grit depth of cut increases. However, for transverse grinding as the grit depth of cut goes beyond a critical grit size value, there was a decrease in strength. This work could have been further strengthened by modeling and optimization for the optimal parameters of the process.

Other optimization methods that have published in literatures and applied to the related machining processes are deterministic optimization approach [8] applied to peripheral milling processes; fuzzy optimization [9] for turning; evolutionary computation [10] for end milling and abductive networks [11] for drilling. The machining of ceramics with a low surface damage and good surface finish is utmost important for its applications in industries. A model available in literature [7] is adopted as a test problem to be solved by PSO [12, 13] and comparison of results are made with the solutions obtained using Differential Evolution (DE) and Genetic Algorithm (GA).

This paper is organized as follows: in Section II the relevant evolutionary algorithms are briefly described. The objective function formulation and constraints handling are detailed in Section III. Experimental settings for all the algorithms and parameter sensitivity analysis are available in Section IV. This is followed by Section V which presents the numerical experimental results and discussions concerning the convergent capability and efficiency of the algorithms. In Section VI, statistical evaluation for each of the algorithm namely GA, DE and PSO are carried out and analyzed and finally the conclusions are derived in Section VII.

1-4244-0708-7/07/$20.00 ©2007 IEEE

II. Overview of Evolutionary Algorithms

In this work we employ some prominent evolutionary algorithms to investigate their performance in the grinding process optimization. Three well-know evolutionary algorithms namely Particle Swarm Optimization (PSO), Differential Evolution (DE) and Genetic Algorithm (GA) are chosen for performance analysis in grinding process optimization. As the information of GA is available from [14] and in many resources, the explanation is omitted here, thereby only PSO and DE are elaborated in the sub-section below.

A. Particle Swarm Optimization (PSO)

Kennedy and Eberhart [12] first introduced the particle swarm optimization (PSO) method. Similar to evolutionary computation, a population of candidate solutions is used. The method has been found to be robust in solving real-world problems featuring non-differentiability, high dimension, multiple optima and non-linearity. PSO algorithm is a model that mimics the movement of individuals (fishes, birds, or insects) within a group (school, flock, and swarm). Similar to GA, a PSO consists of a population refining its knowledge of the given search space. PSO is inspired by models of flocking behaviour.

Instead of using evolutionary operators such as selection, mutation and crossover, each particle in the population moves in the search space with velocity which is dynamically adjusted and all particles are assumed to be of no volume. In short, the whole concept of PSO can be concluded in a sentence that is "A population consisting N particles, each particles has d variables (dimensions) which have its own ranges for each value, velocities and positions are updated every iteration until maximum iteration is reached". Each particle keeps track of its coordinates in the search space, which are associated with the best solution it has achieved so far. This value is known as *pbest*. Another best value that is tracked by the global version of the particle swarm optimizer is the overall best value or the best solution in the population is called *gbest*.

The PSO concept consists of, at each time step, changing the velocity of each particle toward its *pbest* and *gbest* solutions. The movement is weighted by a random term, with separate random numbers being generated toward *pbest* and *gbest* values. For example the i^{th} particle consisting d dimensions is represented as $X_i = (X_{i,1}, X_{i,2}, X_{i,3}, \ldots, X_{i,d})$. The same notation applied to the velocity, $V_i = (V_{i,1}, V_{i,2}, V_{i,3}, \ldots, V_{i,d})$. The best previous position of the i^{th} particle is recorded and represented as $pbest_i = (pbest_{i,1}, pbest_{i,2}, pbest_{i,3}, \ldots pbest_{i,d})$. In the case of minimization that we consider in this paper, the value of $pbest_i$ with lowest fitness is known as *gbest*. The modification of velocity and position can be calculated using the current velocity and the distance from $pbest_i$ to *gbest* as shown in the following formulas:

$$V_{i,j}^{t} = wV_i^{t-1} + \rho_1 r_1 (gbest_{i,j} - X_{i,j}^{t-1}) + \rho_2 r_2 (pbest_{i,j} - X_{i,j}^{t-1}) \quad (1)$$

$$X_{i,j}^{t} = X_{i,j}^{t-1} + V_{i,j}^{t} \quad (2)$$

where $i \in 1...N$, $j \in 1...d$, $t \in 1...T$ with N is the number of population size, d is the number of dimension and T is the number of maximum generation.

The position, X of each particle is updated for every dimension for all particles in each iteration. This is done by adding the velocity vector to the position vector, as described in equation (2) above. In equation (1), w is known as the inertia weight. This parameter was introduced by Shi and Eberhart [13] to accelerate the convergence of PSO. Suitable selection of w provides a balance between global and local explorations, thus requiring less iteration on average to find sufficiently optimal solution. Low values of w limits the contribution of the previous velocity to the new velocity, limiting step sizes and therefore, limiting exploration. On the other hand, high values result in abrupt movement toward target regions.

The parameters ρ_1 and ρ_2 are set to constant values, which are normally given as 2.0 whereas r_1 and r_2 are two random values, uniformly distributed in [0, 1]. The constants, ρ_1 and ρ_2 represent the weighting of the stochastic acceleration terms that pull each particle toward *pbest* and *gbest* positions. The general flow of PSO is illustrated in Fig. 1 below.

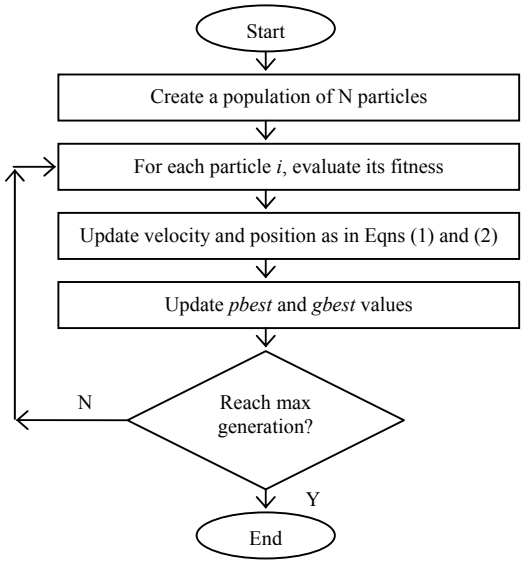

Fig. 1. A general flow chart of PSO

B. Differential Evolution (DE)

DE is a new heuristic approach for optimizing non-linear and non-differentiable continuous space function. This evolutionary algorithm is a powerful tool for optimizing real-valued and multi-modal functions. In previous work, the algorithm has been shown to be superior to other methods presented [15]. The simplicity and robustness of this algorithm is further testified and applied to a wide variety of test functions [16]. DE has a simple adaptive mutation scheme whereby random vector differentials are perturbed around the best solution. DE always accepts improved

solution and neither tournament selection nor annealing criteria are needed. Beneath the superiority of DE algorithm is the mutation strategy of manipulating the vector with the knowledge of the best solution so far. Each dimension of a vector is manipulated in the following steps:

1. For a vector, choose four vectors randomly from the population. Let those vectors be $\boldsymbol{X}^{a}$, $\boldsymbol{X}^{b}$, $\boldsymbol{X}^{c}$ and $\boldsymbol{X}^{d}$. These four vectors are then combined to form $\boldsymbol{X}^{abcd}$:

$$\boldsymbol{X}^{abcd} = (\boldsymbol{X}^{a} - \boldsymbol{X}^{b}) + (\boldsymbol{X}^{c} - \boldsymbol{X}^{d}) \quad (3)$$

2. For j[th] dimension of the first vector, X_j, generate a random number r. If r < cr, apply Eqn (4.1), else apply Eqn (4.2).

$$Trial_j = Best_j + F \times X_j^{abcd} \quad (4.1)$$

$$Trial_j = X_j \quad (4.2)$$

where F is the scaling factor($0 \le F \le 1.2$).

3. Go to Step 2 for the next dimension, $(j+1)^{th}$ until all dimensions are evaluated.

4. Evaluate the trial vector, ***Trial***. If the fitness is better than the fitness of current vector, $f(\boldsymbol{X})$, assign current vector, **X** to vector **Trial**. (**X**=**Trial**). Also compare $f(\boldsymbol{Trial})$ with $f(\boldsymbol{Best})$. If better, **Best** = **Trial**.

5. Continue with step 1 for the next vector until all vectors are evaluated.

Repeat Steps 1-5 for the next generation until maximum number of generation is reached. In the step 3 above, a checking is done so as to ensure that the trial value in equation (4) does not exit the specified boundaries. Whenever the boundary is violated, the related value will be reinitialized within the specified range. This will introduce a more diversified population, and therefore increases the performance of the DE algorithm.

III. Methodology and Implementation

In this section, we formulate the objective function, which is the fitness function, used to evaluate every particle in the population. The technique of constraint handling and encoding scheme of the variables are discussed here.

A. Formulation of the Objective Function

The objective of the optimization problem of the ceramic grinding process can be described as maximizing the *MRR* subject to a set of constraints on surface roughness, number of flaws and input variables. The constrained optimization problem is formulated as the maximization of the following objective function:

$$F(\bar{X}) = MRR = f \times d_c \quad (5)$$

where f = table feed rate [m/min] and d_c = depth of cut [μm]

The maximization of (5) is subject to some inequality constraints:

$$R_a \le R_{a(\max)}, \quad R_a = 0.145 d_c^{\,0.1939} f^{0.7071} M^{-0.2343} \quad (6)$$

$$N_c \le N_{c(\max)}, \quad N_c = 29.67 d_c^{\,0.4167} f^{-0.8333} \quad (7)$$

$$5\mu m \le d_c \le 30\mu m \quad (8)$$

$$8.6\,\text{m/min} \le f \le 13.4\,\text{m/min} \quad (9)$$

$$120 \le M \le 500 \quad (10)$$

where $R_{a(max)}$ and $N_{c(max)}$ are the maximum allowable values of surface roughness and number of flaws respectively while M is the grit size. As for the objective function above, three variables are present which are depth of cut (d_c), table feed rate (f) and grit size (M). Therefore, this is a 3-dimensional problem whereby the values of d_c, f and M are to be determined through optimization process.

B. Constraints Handling

To apply the evolutionary algorithms for searching the optimal parameters, we replace the three variables, d_c, f and M with X_1, X_2 and X_3, equivalent to the X values in equations (1) to (4). Hence, the dimension of this problem is only three. We use real values representation for this X vector. As the number of dimension is 3, there are three values in a candidate solution that changes continuously until it converges. In a population consisting of a pool of N candidate solutions, only the best solution will be the choice after a number of N iterations.

In order to ensure the feasibility of each particle, checking of constraint violation is employed. Constraints (8), (9) and (10) are handled by the algorithm itself through re-initialization whenever any of these values exceed the specified range. The advantage of this technique is that it creates more diversity in the population which will help the algorithm to find the optimal values. However, the drawback of using this technique is that the algorithm will not be able to find optimal solution if it is located exactly at the boundaries. To tackle this problem, we generate a random number and if this random number is less than 0.5, re-initialization is done, else assign the current X to boundary value. The pseudocode below illustrates this scenario:

```
If X violates upper or lower range
    If r > 0.5 then
       Re-initialize X value within the
       boundary
    Else
       Set the X value to its boundary
       value
    End if
End if
```

The method above has been found to be effective in handling boundary constraints in the aim of maximizing the objective function described in Eqn. (5) as some of the variables are found to be exactly at the boundary when optimal solutions are obtained. Results using this methodology are better compared to the previous work reported in [17]. Further details of the solution are available from the results in Section V.

For (6) and (7) inequality constraints, the penalty method is employed here. Individuals that violate any of these constraints will be penalized heavily with a penalty factor, *pf* =10000 as shown in the pseudocode shown below. The following test is employed in the fitness evaluation of a particle, assuming $R_{a(max)} = 0.3$ and $N_{c(max)} = 7$ and X_1, X_2 and X_3 represent d_c, f and M respectively.

```
Ra = 0.1453 × X1^0.1939 × X2^0.7071 × X3^-0.2343
   If Ra > 0.3 Then
   Error1 = abs(Ra-0.3)
   End If

Nc = 29.67 × X1^0.4167 × X2^-0.8333
   If Nc > 7 Then
   Error2 = abs(Nc-7)
   End If

SumError = Error1 + Error2
Fitness = Fitness – SumError × pf
```

IV. Experimental Settings

In this experiment, the population size is fixed at 20 particles for all the algorithms in order to keep the computational requirements low. The maximum number of iterations or the maximum number of generations is set to 500 and a total of 50 runs/trials are performed. Table I summarizes the parameter settings of the relevant evolutionary algorithms. These are optimal values found by parameter sensitivity analysis to ensure optimum performance of each algorithm.

TABLE I
Parameter Settings for Evolutionary Algorithms

Evolutionary Algorithm	Parameter Settings
PSO	Inertia weight, w=0.5 ρ_1 and ρ_2 = 2.0
DE	Crossover rate, cr = 0.5 Scaling, F=0.5
GA	Dynamic Mutation [14] with probability, mp = 0.3 Arithmetic crossover with probability, cr = 0.85 Tournament based selection with 5 candidates in each cycle

V. Numerical Simulation Results and Discussion

First, the PSO algorithm is applied on a total of 9 problems with different combinations of constraints and results are recorded in Table II. From this table, all the constraints from (6) to (10) are not violated. It can be observed from the optimization results that the *MRR* is remaining constant (*MRR* = 75.1342) at a lower range of number of flaws (7 in this case) even with different values of roughness. Thus, it will be more advantageous to grind SiC at a depth of cut, d_c=5.6070 μm and feed, f =13.40 m/min with a grit size, M=500 mesh. As for the higher range of number of flaws ($N_{c(max)}$), an increase of 30% to 60% in *MRR* can be observed by increasing roughness value from 0.3 μm to 0.4 μm. This can be a good approach to achieve the specified *MRR* with a constraint on surface roughness and surface damage.

The same problem is run and solved using DE and GA algorithms. The results are shown in Tables III and IV. Results of DE algorithm as shown in Table III is very similar to the one obtained by PSO except for the case of $R_{a(max)} = 0.4$ and N_c=11 whereby results obtained by PSO is slightly better with *MRR* = 222.2834. The *MRR* recorded by DE method is 222.2833. Further analysis in the next section shows that PSO has faster convergence capability and is very consistent (very low standard deviation values). Both algorithms are able to find the optimal values of the variables without any violation to the specified constraints. However, the results of GA seems discouraging as all the *MRR* values recorded are lower compared to either PSO or DE. This is due to the fact that for the same N_c for these three methods, the PSO and DE are able to suggest either to use finer grain size to grind SiC or use the higher R_a values but still without violating $R_{a(max)}$ constraint. Therefore, we can cut with deeper depth or faster speed. As a result of this, the *MRR* values obtained using PSO and DE are better as compared to GA.

TABLE II
Optimization Results of PSO (50 Trials)

No	Constraints		Value of Variables			Values of Constraints		*MRR*
	$R_{a(max)}$ [μm]	$N_{c(max)}$	d_c [μm]	f [m/min]	M	R_a [μm]	N_c	
1	0.30	7	5.6070	13.4000	500.0000	0.2965	7	75.1342
2	0.30	9	8.4557	12.1715	500.0000	0.3000	9	102.9204
3	0.30	11	11.5405	11.1765	500.0000	0.3000	11	128.9838
4	0.35	7	5.6070	13.4000	258.2402	0.3461	7	75.1342
5	0.35	9	10.2484	13.4000	430.7846	0.3451	9	137.3293
6	0.35	11	15.2934	12.8663	500.0000	0.3500	11	196.7708
7	0.40	7	5.6070	13.4000	238.4232	0.3527	7	75.1342
8	0.40	9	10.2484	13.4000	295.0714	0.3771	9	137.3293
9	0.40	11	16.5883	13.4000	452.2157	0.3746	11	222.2834

TABLE III
Optimization Results of DE (50 Trials)

No	Constraints		Value of Variables			Values of Constraints		*MRR*
	$R_{a(max)}$ [μm]	$N_{c(max)}$	d_c [μm]	f [m/min]	M	R_a [μm]	N_c	
1	0.30	7	5.6070	13.4000	476.6414	0.2998	7	75.1342
2	0.30	9	8.4558	12.1716	500.0000	0.3000	9	102.9024
3	0.30	11	11.5405	11.1766	500.0000	0.3000	11	128.9838
4	0.35	7	5.6070	13.4000	255.7044	0.3470	7	75.1342
5	0.35	9	10.2485	13.4000	483.4113	0.3359	9	137.3293
6	0.35	11	15.2935	12.8663	500.0000	0.3500	11	196.7708
7	0.40	7	5.6070	13.4000	256.5515	0.3467	7	75.1342
8	0.40	9	10.2485	13.4000	404.9179	0.3502	9	137.3293
9	0.40	11	16.5883	13.4000	381.3980	0.3899	11	222.2833

TABLE IV
OPTIMIZATION RESULTS OF GA (50 TRIALS)

No	Constraints		Value of Variables			Values of Constraints		*MRR*
	$R_{a(max)}$ [μm]	$N_{c(max)}$	d_c [μm]	f [m/min]	M	R_a [μm]	N_c	
1	0.30	7	5.6067	13.3996	492.4720	0.2975	7	75.1285
2	0.30	9	8.4520	12.1688	499.4811	0.3000	9	102.8518
3	0.30	11	11.5332	11.1730	499.2599	0.3000	11	128.8612
4	0.35	7	5.6067	13.3996	467.0716	0.3012	7	75.1278
5	0.35	9	10.2477	13.3995	451.8736	0.3412	9	137.3147
6	0.35	11	15.2933	12.8662	499.9865	0.3500	11	196.7674
7	0.40	7	5.6069	13.3999	368.3301	0.3185	7	75.1335
8	0.40	9	10.2480	13.3997	335.3639	0.3659	9	137.3201
9	0.40	11	16.5879	13.3998	373.7518	0.3917	11	222.2769

VI. STATISTICAL EVALUATION

Further, we record the average values of *MRR* for GA, DE and PSO out of 100 trials as depicted in Table V. Careful analysis of the results in the table proves that PSO is very stable and consistent as it has very low standard deviation values for all 9 cases. The standard deviation here can be regarded as zero as we examine the similar results printed to the file from the program which is written in Visual Basic 6.0 language. It means that all the solutions of 100 trials are exactly the same. The performance of GA is inconsistent and unstable as it recorded high standard deviation for all the cases. The highest standard deviation is 1.0011 for case number 3 which can be regarded as discouraging value. DE is very competent as it recorded the same average values except for the last row. However the DE algorithm lacks slightly in terms of its consistency. This is shown by higher standard deviation which means that the final solution by some trials differs slightly. Further, the convergence graphs for GA, DE and PSO are plotted in Figs. 2, 3 and 4 respectively for all the three cases with $R_{a(max)} = 0.4$μm. From these graphs, the PSO depicts fastest convergence in all three cases. This is followed by DE and GA respectively.

TABLE V
STATISTICAL COMPARISON BETWEEN GA, DE AND PSO (100 TRIALS)

No	Constraints		Material Removal Rate, *MRR*					
			GA		DE		PSO	
	$R_{a(max)}$ [μm]	$N_{c(max)}$	Ave	Std Dev	Ave	Std Dev	Ave	Std Dev
1	0.30	7	75.02	0.0995	75.1342	0	75.1342	7.18×10^{-14}
2	0.30	9	101.67	0.8839	102.9204	7.19×10^{-6}	102.9204	1.44×10^{-14}
3	0.30	11	127.31	1.0011	128.9838	1.08×10^{-5}	128.9838	0
4	0.35	7	75.05	0.0557	75.1342	0	75.1342	7.18×10^{-14}
5	0.35	9	136.99	0.2958	137.3293	4.60×10^{-6}	137.3293	8.61×10^{-14}
6	0.35	11	195.42	0.9215	196.7708	6.85×10^{-6}	196.7710	1.44×10^{-13}
7	0.40	7	75.06	0.0505	75.1342	0	75.1342	7.18×10^{-14}
8	0.40	9	137.10	0.2007	137.3293	4.60×10^{-6}	137.3293	8.61×10^{-14}
9	0.40	11	221.90	0.3140	222.2833	1.01×10^{-3}	222.2834	1.15×10^{-13}

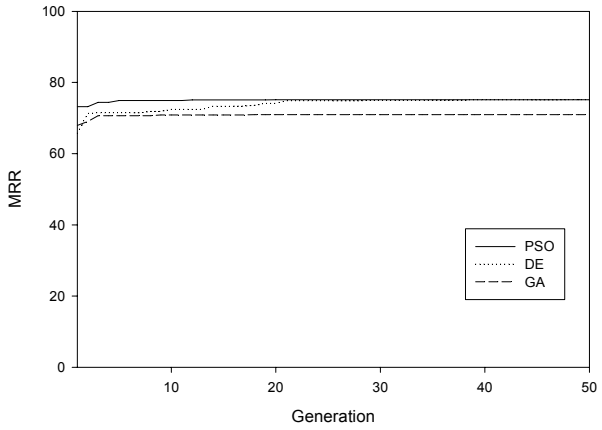

Fig.2. Convergence graph for $R_{a(max)} = 0.4$ and N_c=7

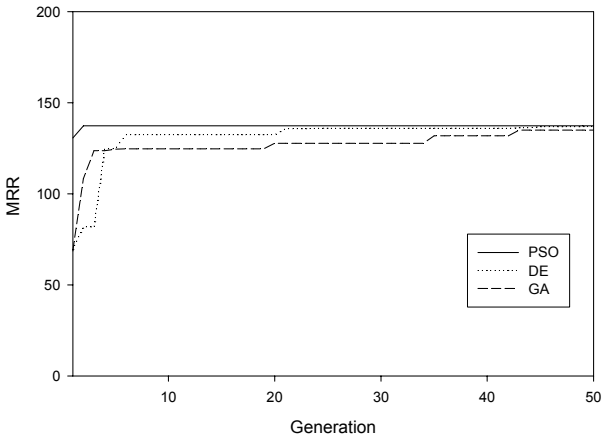

Fig. 3. Convergence graph for $R_{a(max)} = 0.4$ and N_c=9

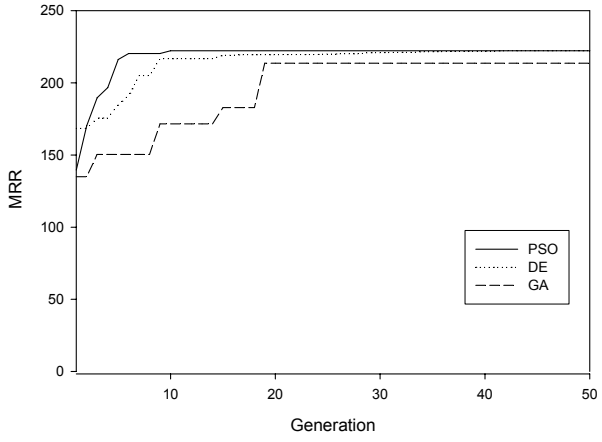

Fig. 4. Convergence graph for $R_{a(max)} = 0.4$ and N_c=11

VII. CONCLUSION

The effect of parameters such as depth of cut (d_c), feed rate (f) and grit size (M) has been studied in silicon carbide grinding. In this work, three prominent evolution algorithms namely GA, DE and PSO are investigated for the grinding process optimization with the objective of maximizing the Maximum Removal Rate, *MRR* subject to some operating constraints. The similar constraints handling are applied to all the algorithms. From the numerical results, PSO methodology is superior in comparison with other optimization algorithms such as DE or GA. Thus, we recommend the usage of PSO for better quality ceramics production in industrial applications.

REFERENCES

[1] I. Inasaki, "Grinding of hard and brittle materials", *Annals CIRP*. 36 (2) (1987) 463-471

[2] S. Malkin, T.W. Hwang, "Grinding mechanisms for ceramics", *Annals CIRP*. 45(2) (1996) 569-580

[3] H.K. Tonshoff, J. Peters, I. Inasaki, T. Paul, "Modeling and simulation of grinding processes", Annals *CIRP*. 41(2) (1992) 677-688

[4] P.V.S. Suresh, P.V. Rao, S.G. Deshmukh, "A genetic algorithm approach for optimization of surface roughness prediction model", *Int J Mach Tools and Manufact*. 42 (2002) 675-680

[5] W. Konig, J. Wemhoner, "Optimizing grinding of SiC", *Ceram Bullet* 68(3) (1989) 545-548

[6] J.E. Mayer, G.P. Fang, "Effect of grinding parameters on surface finish of ground ceramics", *Annals CIRP*. 44(1) (1995) 279-282

[7] A.V. Gopal, P.V. Rao, "The optimization of the grinding of silicon carbide with diamond wheels using genetic algorithms", *Int J Adv Manuf Technol* 22 (2003) 475-480

[8] E.J.A. Armarego, A.J.R. Smith and J. Wang, "Constrained optimization strategies and CAM software for single-pass peripheral milling", *Int J Prod Res* 31 (1993) 2139-2160

[9] X.D. Fang and I.S. Jawahir, "Predicting total machining performance in finish turning using integrated fuzzy-set models of the machinability parameters", *Int J Prod Res* 32 (1994) 833-849

[10] V. Tandon, H. El-Mounary and H. Kishawy, "NC end milling optimization using evolutionary computation", *Int J Machine Tools & Manuf.* 42 (2002) 595-605

[11] B.Y. Lee, H.S. Liu and Y.S. Tarng, "Modeling and optimization of drilling process", *J of Materials Processing Technology*. 74 (1998) 149-157

[12] J. Kennedy and R. C. Eberhart, "Particle swarm optimization," in Proc.1995 *IEEE Int. Conf. Neural Networks IV*, 1995, pp. 1942–1948.

[13] Y. Shi and R.C. Eberhart, "A modified particle swarm optimizer", *Proceedings of the IEEE International Conference on Evolutionary Computation*, Anchorage, Alaska, May 4-9, 1998.

[14] Z. Michaelwicz, Genetic algorithms + Data structures = Evolutionary programs, Springer, New York, 1996

[15] R. Storn, K. Price, "Differential Evolution – A simple and efficient adaptive scheme for global optimization over continuous spaces", Technical Report, *TR-95-012, ICSI*

[16] K. Price, "Differential Evolution – A Fast and Simple Numerical Optimizer", *Biennial Conference of the North American Fuzzy Information Processing Society*. (1996)

[17] T.O. Ting, T.S. Lee and H Than, "Performance Analysis of Grinding Process via Particle Swarm Optimization", *Proc. of the IEEE Int. Conf. Computational Intelligence & Multimedia Applications (ICCIMA'05)*, Las Vegas, Nevada, 16-18 Aug 2005, pp. 92-97.