ANNALS OF FACULTY ENGINEERING HUNEDOARA
– International Journal of Engineering
Tome XI (Year 2013) – FASCICULE 3 (ISSN 1584 – 2673)

1. Nikolaos A. FOUNTAS, 2. Nikolaos M. VAXEVANIDIS

A MODIFIED VIRUS EVOLUTIONARY GENETIC ALGORITHM FOR ROUGH MACHINING OPTIMIZATION OF SCULPTURED SURFACES

1-2. SCHOOL OF PEDAGOGICAL & TECHNOLOGICAL EDUCATION (ASPETE) – DEPARTMENT OF MECHANICAL ENGINEERING EDUCATORS – LAB. OF MANUFACTURING PROCESSES & MACHINE TOOLS (LMProMaT), N. Heraklion Athens, GREECE

ABSTRACT: Genetic and Evolutionary Algorithms have been implemented so far to manufacturing tasks as optimization modules. Genetic and Evolutionary Algorithms are based on the concept of natural selection of species to later produce new individuals. However, different selection mechanisms between species may occur when other populations are incorporated to already existed ones. This paper presents a Genetic Algorithm based on the Virus theory of Evolution. The main work-flow of the algorithm is described and implemented to a benchmark mathematical function. Moreover, the module is incorporated to a CAM system's platform in order to address several machining optimization scenarios for sculptured surfaces. The integration is achieved through the development of automation routines which handle appropriate sets of CAM objects exposed to the Application Program Interface (API). The quality characteristic in this study is the volume remaining after roughing phase. The results verified that the incorporation of virus operators to a genetic algorithm can considerably increase its optimization abilities by producing more effective schemata, hence; minimizing computational time whilst converging to global optimum.
KEYWORDS: Virus Evolutionary Genetic Algorithm (VEGA), Sculptured Surface Machining, Optimization, CAM systems

INTRODUCTION

A vast number of intelligent algorithms and meta heuristics have been associated to manufacturing optimization problems, usually combinatorial. Such algorithms are Genetic and Evolutionary Algorithms (GAs-EAs) [Goldberg, 1989], Simulated Annealing (SA) [Kirkpatrick, Gelatt and Vecchi, 1983], Tabu search (TS) [Glover, 1989-90], Ant Colony Optimization (ACO) [Dorigo and Blum, 2005] and Particle Swarm Optimization (PSO) [Kennedy and Eberhart, 1995]. Each of these approaches have been implemented by researchers to several fields of manufacturing optimization. [Davis, 1985] moved towards the optimum job shop scheduling formulation through a genetic algorithm. [Pare, Agnihorti and Krishna, 2011] controlled the surface finish by optimizing cutting parameter selections in end milling through a (PSO) algorithm. [Rao, Pawar, and Davim 2010] employed SA techniques to optimize process parameters of mechanical type advanced machining. ACO was implemented by [Cus, Balic and Zuperl, 2009] for the optimization of process parameters when turning.

All the approaches mentioned above simulate natural phenomena while others mimic mechanics of processes such as the annealing of metals (Simulated Annealing). GAs-EAs represent the most popular artificial intelligence and evolutionary computation systems, yet; based on Darwin's evolution theory. With the progress of molecular biology, various theories of evolution such as Neo-Darwinism, neutral theory of molecular evolution, Imanishi's evolution theory, serial symbiosis and virus theory of evolution [Anderson, 1970]; [Nakahara and Sagawa 1989]; [Ridley, 1993] research conducted aiming at contributing to optimization through the development of artificial intelligent algorithms based on these theories.

This paper proposed a modified virus evolutionary genetic algorithm (mVEGA) for the rough machining optimization of sculptured surfaces with the use of CAM software. Typical process parameters are examined (cutting speed, feed rate, axial and radial cut depths, cutting tool, etc) whilst the quality objective is the volume remained on the ideal model yet; to be removed by finishing operations.

GENETIC AND EVOLUTIONARY ALGORITHMS

Genetic Algorithms (GAs) are optimization modules based on the concept of the natural selection through evolution, as introduced by Darwin. Genetic Algorithms operate on a population which reflects a set of candidate solutions by applying the principle of "survival of the fittest" [Holland, 1992] to produce better approximations to a solution of a given optimization problem. GAs *have stochastic characteristics that offer the ability of tracing optimized solutions within a given solution space. This ability is known as "exploration". Populations evolved in iterations, called "generations". A new population (the off-spring), is occurred in each generation by applying mathematical operators on the old population. Such operators are crossover, mutation, inversion, etc. Evolutionary Algorithms (EAs) are in fact generalized GAs and include more sophisticated operators than the ones operated in GAs and may possibly employ other deterministic or heuristic modules to reduce evaluation cost or increase the overall system's performance in terms of solution quality and of high solution exploitation. Exploitation is another ability of Genetic or Evolutionary Algorithms that characterizes the rapid arrival at global optima after tracing local optima within the solution space. Exploration and exploitation characteristics of GAs-EAs are depicted in Figure 1.*

The criterion by which candidate solutions in EA populations are evaluated is the objective function. When it comes to a scalar type of an objective function the optimization approach is a single-objective one; whilst when it comes to a vector type of an objective function the optimization approach is a multi-objective one. The solutions ranking in the latter approach is often based on Pareto optimality. As optimization modules, GAs and EAs perform specific operations like working on a population of strings. These strings contain in an encoded type, the candidate solutions. GAs and EAs do not need derivative information; they need only fitness value. Common genetic operators existed in a simple GA structure are one of the selection schemes, a simple crossover type (i.e. one-point crossover) and a simple mutation. It has to be mentioned that for each operator several types are existed and described in a later section of this study.

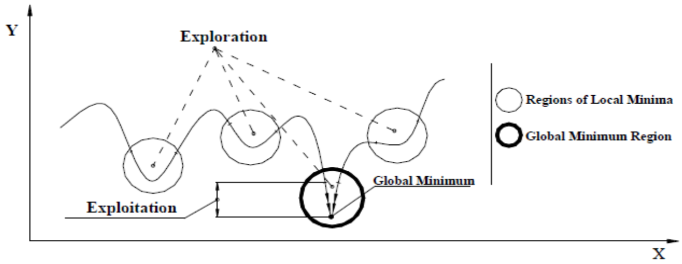

Figure 1. Representation of exploitation and exploration characteristics of GAs-EAs

The overall performance of Genetic and Evolutionary Algorithms is strongly influenced by artificial functions like the encoding type of chromosomes, the genetic operators (mainly crossover and mutation) and ranking of fitness values. In machining optimization where problems become more complicated multi-tasked and conflicted, an implementation of a simple GA-EA would not be robust enough, hence; a means of reinforcing a GA's – EA's structure is sought so as to meet the demands and arrive at global optimum. Several mechanisms have been proposed so far in order to enhance Genetic and Evolutionary Algorithms [Syswerda, 1991], [Manderick and Spoessens 1991].

Prior to the implementation of a GA or an EA to an optimization problem, potential solutions should be properly encoded to facilitate their computational processing. Three basic types for encoding are found in the literature, Binary; Real valued and Gray-binary encoding [Wright, 1991]; [Fogel, 1995]. In the case of binary encoding type the chromosomes are represented by stings consisted of "0" and "1" digit sequences. Each digit in the sequence reflects a value of a probable solution. Binary encoding type is the most common to represent information contained. Encoding using real values for a candidate solution represents chromosomes as a sequence of real numbers [Wright, 1991]; [Michalewicz, 1992]. This type of chromosome representation may be applied when the optimization problem is such that there is no need to convert chromosomes to phenotypes (their real numbers). Gray-binary encoding is quite similar to Binary one with the only difference that two successive digits differ in only one bit. Gray-binary encoding type can be beneficial in an optimization task when mutations permit incremental modifications, thus a single digit-change may cause strong changes which may lead to different solutions.

GENETIC OPERATORS

Genetic diversity (position of candidate solutions within the search space) can be maintained through the respective genetic operators. The genetic operators a Genetic or an Evolutionary Algorithm may utilize are Selection, Crossover and Mutation [Goldberg, 1989]; [Holland, 1992]. Genetic Operators are applied on a population (number of chromosomes in one generation) or a group of populations. If the size of a population is small enough, then the possibility of each genetic operator to be performed (especially the crossover) becomes very low. As a matter of fact, only a small segment of the search space may be explored and the global optimum may never be reached.

Selection simulates the process of natural selection and GAs need a similar mechanism to make a population evolve towards a better direction of optimal solutions. The main procedure of a selection module is to reproduce a population by pre-selecting an individual with a selection probability proportional to its fitness value. In this kind of selection, an individual with a higher fitness can reproduce more offspring. Some of the most commonly used selection schemes are: Roulette wheel selection, Elitist selection, Tournament selection, Ranking selection, and Expected value selection.

Crossover generates new individuals as solution candidates in Gas. GAs can search the solution space mainly by using one of the crossover operators. With the absence of crossover operators, GAs *would be random search algorithms. The crossover operator exchanges each substring between two individuals and replaces old individuals with others of a new genotype. The recombination between two strings is performed according the type of crossover operator. Depending on the number of break-points among individuals, the crossover mechanism recombines the strings of the genotype. Some of the crossover operators are, one-point crossover, multi-point crossover, uniform crossover, cycle crossover, and partially matched crossover.*

Mutation occurs as the replicating error in nature. In GAs, the mutation operator replaces a randomly selected character on the string with the other one. Mutation is performed regardless of individual fitness values. A classic mutation operation is the one-point changing per individual. Several mutation types are occurred in nature such as inversion, translocation and duplication. These are also the mutating mechanisms applied to GAs to simulate these phenomena. Mutation operators have great influence on a GA's-EA's performance because they seriously affect populations. GAs-EAs are able to search a global solution space, since the mutation mechanism randomly changes the strings of individuals. All types of genetic operators (selection, crossover and mutation) are extensively discussed in [Chipperfield, Fonseca and Fleming, 1992].

A MODIFIED VIRUS EVOLUTIONARY GENETIC ALGORITHM (mVEGA)

Gas-EAs have a major drawback; that is the premature convergence when finding local optima within the search space and presenting it as the global one. The most important reason why this premature local convergence occurs in a population is that proportional selection operators applied to the mechanisms of GAs-EAs increase all schemata (effective and ineffective) thus; making them less efficient and reliable. In general, the Virus theory of evolution is relied on "Transduction" operation. Transduction is the process of transporting DNA segments across species. Hence, genetic changes occurred to a bacterium's DNA chain when a becteriophage carries DNA segments from another bacterium and locates them to its DNA chain. Viruses found in nature can be such bacteriophages. Viruses have the ability to penetrate to species' genetic material (DNA chain) and being transmitted directly from individuals of a phylum to another. This special ability of viruses to be transmitted directly from one kind to another is known as horizontal propagation. The incorporation of a host's DNA segments into effective viruses and subsequent transfer to other cells is widely known. Besides, entire virus genomes can be incorporated into germ cells and transmitted from generation to generation as "vertical Inheritance" [Anderson, 1970]; [Nakahara and Sagawa, 1989]. The natural mechanism of "vertical inheritance" among genomes was successfully simulated by Kubota et al., [Kubota, Fukuda and Shimojima, 1996] through "reverse transcription" operator. The proposed algorithm's special features and operators involve functions presenting host and virus populations, reverse transcription and transduction operators. The rest of the functions (selection, crossover and mutation) operate as in conventional GAs - EAs.

As a standard procedure of GAs-EAs, initialization process involves the random generation of host population. As a parameter representation scheme, binary encoding was selected. Virus individuals are formulated as substrings of host individuals through transduction operation. In the original VEGA transduction operator randomly selects the hosts from which substrings to formulate viruses will be cut; whilst in the proposed algorithm the initial virus population is created partially from some of the best host individuals and partially randomly. To increase effective schemata through the virus evolution process, reverse transcription operator is applied to produce new substrings for viruses. Reverse transcription overwrites a virus' string on a host's string. As a result a new host, (infected host), is generated (see Figure 2a). Transduction operator generates a new virus from a host string by taking out (cutting) a sub-string from the string of a host individual in order to generate new viruses (virus individuals) as Figure 2b illustrates.

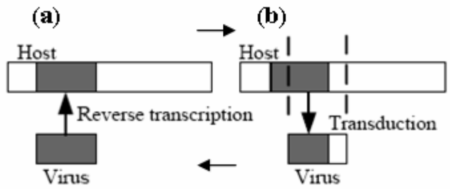

Figure 2. Virus operators: (a) Reverse transcription; (b) Transduction

Unlike the conventional Virus Evolutionary Genetic Algorithm, the virus population size for the proposed algorithm is 1/10 of the host population. This is done so as to simultaneously maintain low computational time whilst achieving high exploitation-exploration rates. Even though the size of a host chromosome is constant, the size of the virus individual grows larger as evolution occurs during the iterations. That is; each of the virus individuals has a variable string growing every few generations.

During virus infection, an elitist scheme which acts as a replacement mechanism; is maintained. This particular scheme removes low fitted individuals from the population by replacing them with those having better fitness values. Fitness is specified according to the problem's nature. Should the optimization problem is about maximization; the replacement mechanism removes the individual with the least fitness value; otherwise the individual with the highest fitness value is removed. If none of the infected hosts has a positive fitness then the original host is preserved. Figure 3 illustrates the work-flow of virus infection with elitism.

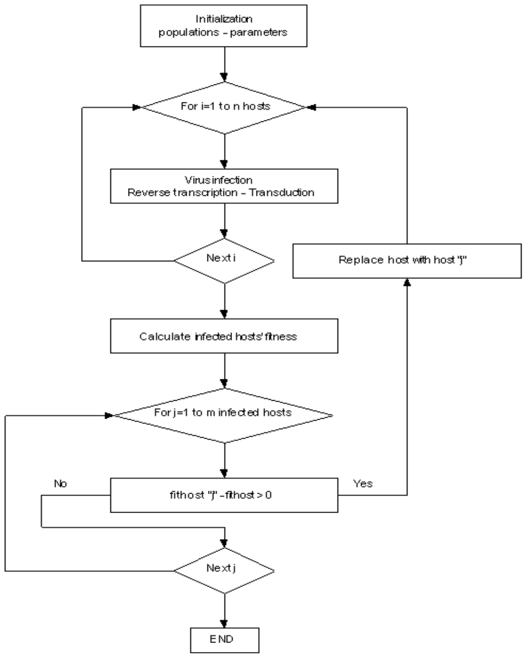

Figure 3. Work-flow of virus infection with elitism

A virus individual (virus "i") has a fitness value (fitvirus (i)) which is calculated for its effectiveness. The fitness of each virus is determined as the sum of the fitness of each infection caused by the current virus to the host population.

$$fitvirus(i) = \sum_{j \in S} fitvirus(i, j) \qquad (1)$$

The fitness of each infection is the difference between the fitness value of the original host (before its infection) and the infected one. This is mathematically described as follows:

$$fitvirus(i, j) = fithost(j') - fithost(j) \qquad (2)$$

According to equations 1 and 2, each virus has a measuring parameter for its infection strength; that is fitvirus (i). It is also assumed that fithost (j) and fithost (j') are fitness values of a host "j" before and after its infection, respectively. The indicator fitvirus (i,j) denotes the difference between the fitness values fithost (j) and fithost (j') which is equal to the improvement value obtained through the infection of a host individual. To the equations presented above, "i" denotes the virus number and "S" symbolizes the set of the host individuals infected by the virus "i".

Infection rate (infrate (i)) controls the number of infections caused by each virus. If a virus has a positive fitness, the infection rate is increased according to a constant "a". In contrast, if the virus has a negative value the infection rate is decreased. [Kubota, Fukuda and Shimojima, 1996] suggested that maximum infection rate to be 0.1 and initial infection rate to be 0.05. The infection rate of each virus should satisfy 0≤ infrate (i)≤1.0 in order to perform a reverse transcription operation to a given host population. Infection rate operates differently on a given search space, regarding the searching ratio. That is, if inf rate is high then local search through virus operators is performed, whereas global search through traditional genetic operators is achieved should virus infection rate is low. Note that virus infection plays the simultaneous role of crossover and mutation. Equation 3 represents the formulation of these settings for the determination of infection rate by Kubota et al., and is depicted below:

$$\inf rate_{i,t+1} = \left\{(1+a) \times \inf rate_{i,t}\right\},\ fitvirus(i) \geq 0 \qquad (3)$$
$$\inf rate_{i,t+1} = \left\{(1-a) \times \inf rate_{i,t}\right\},\ fitvirus(i) \leq 0$$

It is obvious that, the higher the infection strength of a virus "i" (fitvirus (i)) is, the higher its infection capability (thus the infection rate) will be. Apart from the aforementioned parameters and indicators, a virus has also an indicator that indicates its performance duration, or its life. Actually, the virus life indicator shows the positive contribution of a virus to the host population. Equation 4 shows this indicator:

$$life_{i,t+1} = r \times life_{i,t} + fitvirus_i \qquad (4)$$

where, "r" is the life reduction rate, "t" is the generation. If life is negative then the virus individual transduces a new substring from a random host. If life is positive the virus individual transduces a partially new substring from one of the infected host individuals for evolving for itself. Hence, both populations (hosts and viruses) co-evolve through the implementation of genetic operators and virus infection operators. This particular indicator of a virus life may be initialized as: infrate $_{i,0}$ = init infrate while, life$_{i,0}$=0.

Traditional operators found in GAs-EAs (selection, crossover and mutation) are still applied; should infection operators fail to produce effective chromosome schemata. For this GA, stochastic sampling with partial replacement (roulette wheel selection), one-point crossover and classic mutation (one-point changing per individual) with 0.05 probability were programmed. Due to the stochastic nature of the mVEGA its termination is achieved after a predetermined number of generations, or when no further improvement is reached.

ROUGH MACHINING OPTIMIZATION WITH mVEGA AND CAM SOFTWARE

Rough-machining operations are applied in order to remove most of the original block material yielding a much more convenient shape for the finish machining operations that follow. In rough machining, the difference between stock and final part volume has to be divided into Z-heights to be sequentially machined. Remained volume is critical objective for optimization owing to its outcome on *later machining stages. The lower the difference between rough machined and ideal part is, the finer surface finish is achieved.*

A rough machining modelling was prepared for a sculptured part representing a hip joint used in orthopaedics as artificial implants. The advanced machining workbench of Dassault Systemes CATIA® V5 R18 was the machining modelling environment. A sweeping machining strategy for roughing was specified in order to take advantage of producing more uniform Z heights as intermediate tool trajectories are applied to models, thus; getting closer to final geometry.

Optimization parameters were cutting speed – V_c (m/min) through its relation to spindle – n (rpm), feed rate – f(mm/min), stepover (radial tool engagement) – a_e (%Ø, mm), stepdown (axial cutting depth) – a_p (mm), and tool path style (choosing among three different tool paths). Cooperation among CAM software and mVEGA optimization algorithm was achieved by programming in Visual Basic for Applications (VBA) though an "in-house" software application program interface (API) development. CAM software played the role of "evaluator" (extraction of remaining volume) whilst mVEGA produced optimal values for machining parameters (phenotypic values). Thereby, loops of operations among these two systems were performed until reaching global minimum for the quality objective of remaining volume. To test performance and virus infection contribution to machining optimization, virus operators were isolated hence; resulting to a conventional GA. The results obtained by the two algorithms (GA and mVEGA) for their loops of evaluations indicated that mVEGA achieved faster convergence to global optimum than the traditional GA; thus revealing the important contribution of virus infection mechanism to optimization.

RESULTS AND DISCUSSION

A number of CAM software evaluations was executed to conduct the machining simulations and obtain data for remaining volume. Remaining volume was calculated by subtracting the ideal model's volume from the machined model's volume. As 50 generations were programmed for mVEGA, 50 evaluations were to be conducted. Prior to the final evaluations a number of machining experiments were carried out to several sculptured models so as to check optimization efficiency, premature convergence avoidance and computational speed.

Table 1. Optimal settings for machining parameters and intelligent operators for mVEGA and GA

Optimum machining program 1								
Phase	Optimal settings for machining parameters						Quality objective	Machining time
Roughing	Machining strategy	Tool Ø (mm)	a_p (%Ø)	a_e (mm-%Ø)	f (mm/min)	n (rpm)	RV (mm^3)	t_{rm} (min)
	Z-Offset	10	29.85%	37%	600	4516	64483.14	76.35
Optimization Algorithm	Settings for genetic and virus infection operators						Convergence time for mVEGA	
mVEGA	encoding type	selection scheme	crossover probability (p_c)	mutation probability (p_m)	infection rate	generations		$t_{convergence}$ (hours)
	binary	SSPR	1	0.001	0.1	50		5.14
Optimum machining program 2								
Phase	Optimal settings for machining parameters						Quality objective	Machining time
Roughing	Machining strategy	Tool Ø (mm)	a_p (%Ø)	a_e (mm-%Ø)	f (mm/min)	n (rpm)	RV (mm^3)	t_{rm} (min)
	Z-Offset	10	31.12%	32.45%	628.13	4483	64501.28	77.12
Optimization Algorithm	Settings for genetic operators						Convergence time for GA	
GA	encoding type	selection scheme	crossover probability (p_c)	mutation probability (p_m)		generations		$t_{convergence}$ (hours)
	binary	SSPR	1	0.001		50		6.27

In all experiments mVEGA shown great operability without trapping to local minima. mVEGA algorithm tended to maintain vertical convergences during the evolution process; indicating its high exploration rate. Figure 4 illustrates the evolution diagram among the results obtained for mVEGA and GA. The former algorithm found optimal solution to the 35th iteration whilst the latter found the same result for the optimum at 47th iteration.

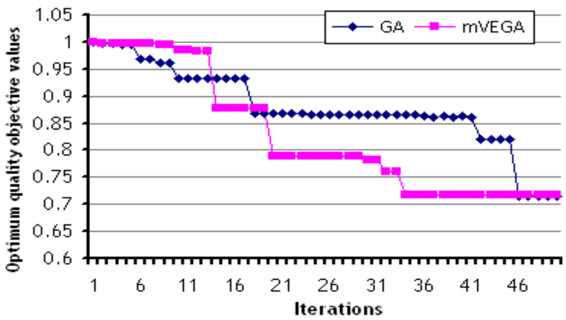

Figure 4. Comparative evolution diagram for resulting outputs among GA and mVEGA

Based on the phenotypes the two optimization algorithms generated for each of the optimal values for parameters, two machining programs for roughing were formulated using CAM software and suitable post-processor. Table 1 presents the parameter settings for both optimization algorithms and machining programs and Figure 5 depicts the rough machining simulation using the optimal settings for the parameters involved to optimization (with mVEGA).

CONCLUSIONS

A modified Virus Evolutionary Genetic Algorithm for sculptured surface machining optimization using CAM software was presented in this paper. Optimization effort was referred to rough machining where the least amount of remaining volume need to be existed on an ideal model's surface to ease further machining processes. Typical process parameters available in a typical commercial CAM system were handled by the optimization algorithm, whilst measurements to extract remaining volume for simulations performed through API automation.

The proposed algorithm was developed as a host application existed in CAM software and applied on a sculptured part for rough machining optimization. Results obtained indicated that intelligent systems based on other evolution theories than Darwin's not only may efficiently be applied to manufacturing optimization but also may perform better when compared to other systems. In particular, the proposed algorithm proved capable of rapidly converging to global optimum in terms of the quality objective studied. It was observed that virus operators promote only effective schemata with the ability to perform both local and global search without trapping to local optima.

As a future perspective finish machining operations will be investigated and optimized through proper problem definition in terms of process parameters and quality objectives. In addition the proposed algorithm will be subjected to multi-objective optimization using Pareto optimality.

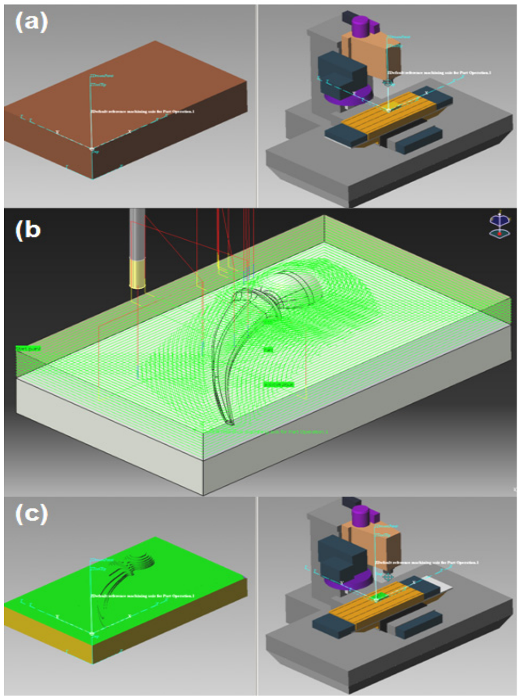

Figure 5. Machining simulation performed in CAM software using optimum parameter settings obtained by mVEGA: (a) Machining set-up, (b) tool path simulation and (c) roughed model

REFERENCES

[1.] Anderson, N.: Evolutionary Significant of Virus Infection, Nature, volume (227), p.1346-1347, 1970.
[2.] Chipperfield, AJ; Fonseca, CM and Fleming PJ: Development of genetic optimization tools for multi-objective optimization problems in CACSD, In IEE Colloq. on Genetic Algorithms for Control Systems Engineering, p.3/1-3/6. The Institution of Electrical Engineers Digest, volume (106), 1992.
[3.] Cus, F; Balic, J and Zuperl, U: Hybrid ANFIS-ants system based optimization of turning parameters, Journal of Achievements in Materials and Manufacturing Engineering, volume 36(1), p.79-86, 2009.
[4.] Davis, L: Job shop scheduling with genetic algorithms. in Proc. 1st Int. Conf. Genetic Algorithms Appl., J. J. Grefenstette, Ed. Hillsdale, NJ: Lawrence Erlbaum, p.136-140, 1985.
[5.] Dorigo, M and Blum, C: Ant Colony Optimization Theory: A survey, Journal of Theoretical Computer, Science, volume (344), p.243-278, 2005.
[6.] Fogel, DB: Phenotype, Genotype and Operators in Evolutionary Computation, The 1995 IEEE International Conference on Evolutionary Computation, p.193-198, 1995.
[7.] Glover, F: Tabu Search, Part I, ORSA, Journal on Computing, volume 1(3), p.190-206, 1989.
[8.] Glover, F: Tabu Search, Part II, ORSA, Journal on Computing, volume 2(1), p.4-32, 1990.
[9.] Goldberg, DE: Genetic Algorithms in Search, Optimization, and Machine Learning, Reading, Massachusetts, Addison-Wesley, 1989.
[10.] Holland, J: Adaptation in Natural and Artificial Systems, 2nd ed., The MIT Press, Massachusetts, 1992.
[11.] Kennedy, J and Eberhart, R: Particle swarm optimization, Proceedings of the IEEE International Conference on Neural Networks, IV, p.1942-1948, 1995.
[12.] Kirkpatrick, S; Gelatt, Jr-C and Vecchi, M: Optimization by Simulated Annealing, Science, volume (2200, p. 671-680, 1983.
[13.] Kubota, N; Fukuda, T and Shimojima, K: Virus-evolutionary algorithm for a self-organising manufacturing system, Computers and Industrial Engineering, volume 30(4), p.1015-1026, 1996.
[14.] Manderick, B and Spoessens, P: Fine-grained parallel Genetic Algorithms, The 4th International Conference on Genetic Algorithms, Virginia, p.428-433, 1991.
[15.] Nakahara, H and Sagawa, T: Virus Theory of Evolution, Tokyo, Tairyusha, 1989.
[16.] Pare, V; Agnihotri, G and Krishna, CM: Optimization of Cutting Conditions in End Milling Process with the Approach of Particle Swarm Optimization, International Journal of Mechanical and Industrial Engineering, volume 1(2), 2011.
[17.] Rao, RV; Pawar, PJ and Davim JP: Optimisation of process parameters of mechanical type advanced machining processes using a simulated annealing algorithm, International Journal of Materials and Product Technology, volume 37 (1-2), p.83-101, 2010.
[18.] Ridley, M: Evolution 1^{rst} ed., Massachusetts, Blackwell Scientific Publications Inc.,1993.
[19.] Syswerda, G: A Study Reproduction in Generational and Steady-State Genetic Algorithms, Foundations of Genetic Algorithms, (ed. Rawlins GJE), San Mateo, CA, Morgan Kaufmann Publishers, p.94-101, 1991.