Pergamon

PII: S0305-0548(96)00077-9

Computers Ops Res. Vol. 24, No. 7, pp. 647–657, 1997

0305-0548/97 $17.00+0.00

MACHINING CONDITION OPTIMIZATION BY GENETIC ALGORITHMS AND SIMULATED ANNEALING

Z. Khan†‡, B. Prasad§ and T. Singh¶

Department of Mechanical and Aerospace Engineering 1009 Furnas Hall, Suny at Buffalo, Buffalo, NY 14260, U.S.A.

(Received July 1995;in revised form September 1996)

Scope and Purpose—Knowledge of optimal cutting parameters for machining operations is required for process planning of metal cutting operations. Numerous nonlinear and non-convex machining models have been developed with the objective of determining optimal cutting conditions. Traditionally these problems have been solved using gradient based algorithms. The purpose of this article includes studying three non gradient based stochastic optimization algorithms to test their efficiency in solving several benchmark machining models.

Abstract—Optimal machining conditions are the key to economical machining operations. In this work, some benchmark machining models are evaluated for optimal machining conditions. These machining models are complex because of non-linearities and non-convexity. In this research, we have used Genetic Algorithms and Simulated Annealing as optimization methods for solving the benchmark models. An extension of the Simulated Annealing algorithm, Continuous Simulated Annealing is also used. The results are evaluated and compared with each other as well as with previously published results which used gradient based methods, such as, SUMT (Sequential Unconstrained Minimization Technique), Box's Complex Search, Hill Algorithm (Sequential search technique), GRG (Generalized Reduced Gradient), etc. We conclude that Genetic Algorithms, Simulated Annealing and the Continuous Simulated Annealing which are non-gradient based optimization techniques are reliable and accurate for solving machining optimization problems and offer certain advantages over gradient based methods. © 1997 Elsevier Science Ltd

1. INTRODUCTION

The most important interface between product design and manufacture is Process Planning. Process plans typically contain the specified sequence of operations to be performed, specifying various parameters that aid in producing the part like machining dimensions, tolerances etc. and machine and tool selection with machining conditions to be used. In traditional process planning, the process planner who is usually knowledgeable and experienced, generates the process plan on an ad-hoc basis. This may lead to inconsistencies in manufacturing production. It is also impossible to achieve any kind of optimization which results in the increase of planning and manufacturing costs. With today's advanced computers, this task can be routinely performed. Computer Aided Process Planning (CAPP) is one of the most important advances in the area of manufacturing engineering.

A part of the CAPP process is the selection of an economically optimal combination of machining conditions. This is achieved by using optimization models for machining conditions. These models are usually a combination of variables such as production time, production cost, metal removal rate, profit etc. which are either maximized or minimized as appropriate. Like any other optimization model, these are in the format of an objective function and a set of constraints. These models are usually non-linear and the optimization method that are suitable to solve them are problem dependent [1].

The primary objectives in the solution process for an optimization problem are reliability, accuracy of result, insensitiveness to the initial conditions and small computational effort. A number of researchers have used a host of algorithms to solve the machining models which are discussed further in Section 2. A fact to be noted is one method is always better than the other for a certain problem. So it is hard to determine which method is right for a given model.

† To whom all correspondence should be addressed.

‡ Z. Khan is a graduate student in the Dept. of Mechanical & Aerospace Engineering, SUNY at Buffalo. He is currently working on a Ph.D degree in the area of System Identification and Control. He obtained his MS degree in the area of Production Systems in the Dept. of Industrial Engineering, SUNY at Buffalo

§ B. Prasad is a graduate student in the Dept. of Mechanical & Aerospace Engineering, SUNY at Buffalo. He obtained his MS degree in the area of Fuzzy Logic Control of ABS

¶ T. Singh is an Assistant Professor in the Dept. of Mechanical & Aerospace Engineering, SUNY at Buffalo. He obtained his M.E. degree from the Indian Institute of Science and his Ph.D from the University of Waterloo. His areas of interest are Dynamics and Control of Structures.

Thus, there exists a need to determine a standard method in the CAPP process which can generate the optimal machining parameters given any machining model. In Section 2, we present Genetic Algorithms (GA), Simulated Annealing (SA) and a modification of the SA which will be referred to as Continuous Simulated Annealing (CSA), as solution methods for these complex problems. SA's and GA's are non-traditional methods which offer the maximum advantages when the problem is highly non-linear and non-convex. The performance of the algorithms for a number of machining models are presented and compared with the results available in literature, which include optimization methods, such as SUMT, Box, Hill, GRG, etc. [1].

2. REVIEW OF MATHEMATICAL PROGRAMMING-BASED METHODS

The main goal of optimization is to find the best possible combination of factors which can be termed as design variables to extremize a given cost function. Calculus based solution methods are the most commonly used techniques. These methods require that the optimization be started from a number of initial points to avoid convergence to a local minima. Thus, calculus based methods are best when the solution space is convex. For non-convex solution spaces, the results might not be globally optimal. Review of the optimization methods used by other researchers in the area of machining optimization is given as follows. Duffuaa, Shuaib and Alam [1] have compared the results of a number of gradient based optimization algorithms with different machining models. Their approach is limited because of the use of gradient based methods which are not ideal for non-convex problems. They have concluded that the Generalized Reduced Gradient method is the most suitable for solving machining optimization models. Petropoulos [2] has used geometric programming for optimization of machining parameters. Multipass turning optimization has been addressed by Ermer and Kromodihardjo [3]. They use a combination of linear and geometric programming. Iwata *et al.* [4] have used a stochastic approach to solve for optimal machining parameters. Eskicioglu and Eskicioglu [5] have demonstrated the use of Non-Linear Programming for machining parameter optimization. Hati and Rao [6] have used SUMT to solve a multi-pass turning operation.

The above review clearly shows that most of the researchers have used one or two methods to compare their model performance. The performance of optimization algorithms is problem dependent for non-linear problems, implying that there are advantages and disadvantages of using a particular optimization method depending on the area of application. The potential of applying Simulated Annealing and Genetic Algorithms to machining optimization problems is explored in this work. Techniques that emulate nature have led to the development of methods like Genetic Algorithms, Simulated Annealing and Neural Networks. These methods are ideal for global optimization for non-linear systems with non-convex solution spaces.

2.1. Genetic Algorithms

Genetic Algorithms are used extensively for the solution of optimization problems and were first developed by Holland [7] in the 1970s. These algorithms are based on the biological evolution process. A similar analogy is used to evolve solutions to complex optimization problems. The notable feature of Genetic Algorithms is that it emulates the biological system's characteristics like self repair and reproduction.

It is generally well known that the human being is a very good example of a decision maker. So researchers began experimenting with the natural systems and have developed methods like Genetic Algorithms and Neural Networks. The actual differences between the GA's and other methods of optimization are briefly summarized below. GA move through the solution space starting from a population of points and not from a single point. This is similar to the calculus based methods where we have to restart the solution from a number of points to ensure global convergence. GA's work with the objective function information directly and not with any other auxiliary information like derivatives. Constraints are included in the objective function using some *Penalty* function. GA's use probabilistic rules and not deterministic rules. The GA's also differ from Simulated Annealing in the fact that the optimal solution is selected from a population of solutions and not from one solution which is computed based on a probability.

The theory behind the Genetic Algorithms is explained in this paragraph. A potential solution to a problem may be represented by a set of parameters known as *genes*. These genes are combined together to form a string which is referred to as a *chromosome*. It is widely believed that ideally a binary string should be used for the chromosome. The set of parameters represented by a particular chromosome is

called as *genotype*. This genotype contains the information required to construct an organism called the *phenotype*. A fitness function is analogous to the objective function in an optimization problem. The fitness function returns a single numerical 'fitness' which is proportional to the 'utility' or 'ability' of the individual which that chromosome represents. During reproduction in the GA, individuals are selected from the population and recombined, producing off-springs which will comprise the next generation. Two parents are selected and their chromosomes are recombined, typically using the mechanisms of crossover and mutation. Crossover is the operation when two individuals are taken and their chromosomes are cut at some randomly chosen position, to produce two head and tail segments. These segments are swapped to produce two new full length chromosomes. The offsprings inherit some genes from each parent. This is known as a *single point crossover. Mutation* is the technique used to randomly alter the genes with a small probability and is typically applied after crossover. Crossover is more important for rapidly exploring a search space. Mutation provides only a small amount of random search. If the GA has been implemented correctly, the population will evolve over successive generations so that the fitness of the best and the average individual in each generation increases towards the global optimum. A gene is said to have converged when 70% of the population share the same value of the fitness function.

Most of the research work in GA has been concentrated in the area of Numerical optimization. GA's have been shown to outperform conventional optimization methods on difficult, discontinuous and noisy functions. GA's are also used in the field of image processing, combinatorial optimization and machine learning. With medical X-rays or satellites images, there is often a need to align two images of the same area, taken at different times. By comparing a random sample of points on the two images, a GA can be used to find a set of equations which transform one image to another. Combinatorial optimization is a class of problems where the solution space is discrete. So traditional methods of optimization are not well suited to such problems. Traveling salesman problem, job shop scheduling etc., are examples of such problems.

In this work the Genetic Algorithm has been implemented in C++. Each variable was represented by 32 bits. Reproduction is done by *1-point crossover* [10] with a likelihood of 0.65 (normally between 0.6 and 1) and *mutation* [9] with a likelihood of 0.001. The population is assumed to have converged when 70% of the population have the same fitness. Constraints are augmented to the objective function using different *Penalty* functions, such as, exterior, interior, extended interior, quadratic extended etc. in this work, the type of penalty function used did not seem to make any difference in the final results. The results presented in this article are based on the exterior penalty approach.

2.2. *Simulated Annealing*

Simulated Annealing is also an emerging technique that is being extensively used in optimization of complex systems. The backbone of the theory lies in the fact that there exists a very strong analogy between statistical mechanics (the behavior of systems with many degrees of freedom in thermal equilibrium at a finite temperature) and multivariable optimization. In addition, the annealing process in solids provides a framework for optimization of the properties of very large and complex systems. This was introduced in the early 1980s by Kirkpatrick, Gellat and Vecchi [8]. The salient features of this method are its general applicability and its ability to obtain solutions arbitrarily close to an optimum. A major drawback of this method is that finding high quality solutions may require large computational effort.

The SA can be applied to generate a sequence of solutions of a combinatorial optimization problem [12]. The analogy between a physical multiple particle system and a combinatorial problem is based on the following equivalences.

- Solutions in a combinatorial optimization problem are equivalent to the states of a physical system.
- The cost of a solution is equivalent to the energy of the state.
- The temperature is analogous to the control parameter which is explained below.

The Simulated Annealing algorithm can be viewed as an iteration of the Metropolis algorithm, which is evaluated at decreasing values of the control parameter. A feature of the Simulated Annealing algorithm is that, besides accepting improvements in the cost, it also, to a limited extent, accepts deterioration in cost. Initially, when the control parameter is large, a large percent of deterioration will be accepted and as the value of the control parameter decreases, the rate of accepting deterioration will be smaller and finally as the control parameter becomes zero, the probability of accepting deteriorations is almost nonexistent. This shows that the Simulated Annealing algorithm can escape from the local minima which

a gradient based approach cannot emulate. Simulated Annealing can be viewed as a generalization of the local search. It may become identical to some forms of the local search or greedy methods when the value of the control parameter is taken to be zero. Deterministic minimization algorithms could get trapped in local minima. However, pseudo random methods, such as Simulated Annealing can overcome this problem, that is, in theory, a global minima can be found if certain sufficient conditions are satisfied [12].

The Simulated Annealing algorithm is very easy to program. Typically it takes only a few hundred lines of computer code. Implementation of a new problem often only takes very little modifications of the existing code. The amount of computational effort required by the Simulated Annealing algorithm is very large for convergence to a near-optimum solution. This may vary depending on the nature and size of the optimization problem.

Simulated Annealing has been applied to a variety of problems like Traveling Salesman Problems, Graph Partitioning Problems, Matching Problems, Scheduling Problems, VLSI Design, Facilities Layout and Image Processing.

2.3. Continuous Simulated Annealing

Corana *et al.* [13] proposed a modification to the SA approach for combinatorial optimization. The proposed algorithm includes adaptive modification of the search space in each coordinate direction based on the ratio of accepted to rejected configurations. A ratio of 1:1 is selected since a lower rate implies larger number of rejections, thus wasting computational time and a higher rate implies that the test points lie close to the starting point, thus limiting the search space to a small domain of the feasible region. The search space is a n dimensional cuboid whose center is the starting point. Random numbers are generated consistent with the bounds of the search space in each coordinate direction sequentially, to generate successive search coordinates. The sides of the cuboid are modified based on the number of accepted and rejected configurations along each coordinate direction in an attempt to maintain an accepted to rejected ratio of 1:1. The adaptation criterion is

$$v_i' = \begin{cases} v_i\left(1+\dfrac{c}{0.4}\left(\dfrac{n_i}{N}-0.6\right)\right) & \text{if} \quad n_i > 0.6N \\ v_i\left(1+\dfrac{c}{0.4}\left(0.4-\dfrac{n_i}{N}\right)\right)^{-1} & \text{if} \quad n_i < 0.4N \\ v_i & \text{otherwise} \end{cases} \tag{1}$$

where v_i is the current bound on the search space in the i^{th} direction, n_i is the number of accepted configurations, N is the total number of searches in the i^{th} direction, and v_i' is the adapted bound. c is a number greater than one and is generally selected to be 2. Corana *et al.* [13] have tested their algorithm on some benchmark problems successfully.

3. MACHINING MODELS USED IN THE PERFORMANCE STUDY

In this section, we review the different machining models we have used in this study.

3.1. Hati and Rao model

Hati and Rao [6] have used this model in multi-pass turning optimization of mild steel work-piece using a carbide tool. The objective function used is minimum production cost in dollars/piece.

$$Min.\ Cost = n(3141.59V^{-1}f^{-1}d^{-1} + 2.879\times 10^{-8}V^4 f^{0.75}d^{-0.025} + 10) \tag{2}$$

Subject to the following constraints:

(i) Maximum and Minimum cutting speeds

$$50 \le V \le 400 \text{ m/min} \tag{3}$$

(ii) Maximum and minimum feed rates

$$0.30 \le f \le 0.75 \text{ mm/rev} \tag{4}$$

(iii) Range of allowable depths of cut

$$1.20 \leq d \leq 2.75 \text{ mm} \tag{5}$$

(iv) Cutting force

$$F_c \leq 85 \text{ kg} \tag{6}$$

where

$$F_c = (28.10V^{0.07} - 0.525V^{0.5})d \times f\left(1.59 + 0.946 \frac{(1+x)}{\sqrt{(1-x)^2+x}}\right) \tag{7}$$

and

$$x = \left(\frac{V}{142} \exp(2.21f)\right)^2 \tag{8}$$

(v) Cutting power

$$P_c \leq 2.25 \text{ kW} \tag{9}$$

where

$$P_c = \frac{0.746F_cV}{4500} \tag{10}$$

(vi) Tool life

$$25 \leq TL \leq 45 \text{ min} \tag{11}$$

where

$$TL = 60\left(\frac{10^{10}}{V^5 f^{1.75} d^{0.75}}\right) \tag{12}$$

(vii) Temperature

$$T \leq 1000°\text{C} \tag{13}$$

where

$$T = 132V^{0.4} f^{0.2} d^{0.105} \tag{14}$$

(viii) Limitations on the value of the depth of cut in removing 'A' in 'n' passes:

$$\frac{A}{d} = n \tag{15}$$

3.2. Ermer's model

Ermer's model [11] minimizes production cost in dollars/piece for single pass turning

$$\text{Min. Cost} = 1.25V^{-1}f^{-1} + 1.8 \times 10^{-8}V^3 f^{0.16} + 0.2 \tag{16}$$

Subject to the following constraints:

(i) Surface finish

$$SF \leq 100 \ \mu \text{ in} \tag{17}$$

where

$$SF = 1.36 \times 10^{8} V^{-1.52} f^{1.004} \tag{18}$$

(ii) Feed rate

$$f \leq 0.01 \text{ in/rev} \tag{19}$$

(iii)

$$HP \leq 2.0 \text{ hp} \tag{20}$$

where

$$HP = 3.58 V^{0.91} f^{0.78} \tag{21}$$

3.3. Petropoulos model

In this model [2], the objective function minimizes the production cost in pence/piece for single pass turning of medium carbon steel workpiece using a carbide tool.

$$\text{Min. Cost} = 452 V^{-1} f^{-1} \times 10^{-5} V^{2.33} f^{0.4} \tag{22}$$

Subject to the following constraints:

(i) Cutting Power

$$P_c \leq 5.5 \tag{23}$$

where

$$P_c = 10.6 \times 10^{-2} V f^{0.83} \text{kW} \tag{24}$$

(ii) Surface finish

$$R_a \leq 2 \ \mu\text{m} \tag{25}$$

where

$$R_a = 2.2 \times 10^{4} V^{-1.52} f \ \mu\text{m} \tag{26}$$

3.4. Ermer and Kromodihardjo model

This model [3] minimizes the production cost in dollars/piece for single pass turning.

$$\text{Min. Cost} = 1.2566 V^{-1} f^{-1} + 1.77 \times 10^{-8} V^{3} f^{0.16} + 0.2 \tag{27}$$

Subject to the constraints:

(i) Feed rate

$$f \leq 0.1 \text{ in/rev} \tag{28}$$

(ii) Horsepower

$$HP \leq 4 \text{ hp} \tag{29}$$

where

$$HP = 2.39 V^{0.91} f^{0.78} d^{0.75} \tag{30}$$

(iii) Surface finish

$$SF \leq 50 \ \mu \text{ in} \tag{31}$$

where

$$HP = 204.62 \times 10^{6} V^{-1.52} f^{1.004} D^{0.25} \tag{32}$$

3.5. Iwata, Oba and Murotsu model

Iwata, Oba and Murotsu [4] have proposed this model for multi-pass turning operation of medium carbon steel using carbide tool where the objective is production cost/piece in yens/piece.

$$\text{Min. Cost} = \sum_{i=1}^{n} 3927V_i^{-1}f_i^{-1} + 1.95\times 10^{-8}V_i^{2.88}f_i^{-1}\exp(5.884f_i)d_i^{-1.117} + 60 \tag{33}$$

Subject to the following constraints:

(i) Maximum and minimum feed rates.

$$0.001 \le f \le 5.6 \text{ mm/rev} \tag{34}$$

(ii) Maximum and minimum cutting speeds

$$14.13 \le V \le 1005.3 \text{ m/min} \tag{35}$$

(iii) Maximum and minimum depth of cut

$$0 \le d \le A \text{ mm} \tag{36}$$

where 'A' is the depth of material to be cut.

(iv) Maximum cutting force

$$F_c \le 170 \text{ kg} \tag{37}$$

where

$$F_c = 290.73V^{-0.1013}f^{0.725}d \tag{38}$$

(v) Stable cutting region related to the cutting surface

$$fV^2 \ge 2230.5 \tag{39}$$

(vi) Maximum allowed surface roughness

$$0.356f^2 \le H_{\max} \tag{40}$$

where

$$H_{\max} \text{ranges from 0.01 to 0.06 mm} \tag{41}$$

(vii) The maximum power consumption

$$P_c = 7.5 \text{ kW} \tag{42}$$

where

$$P_c = \frac{F_c V}{4896} \tag{43}$$

(viii) The sum of depths of cut of the 'n' passes used to remove the total depth 'A' of the material.

$$\sum_{i=1}^{n} d_i = A \tag{44}$$

4. PERFORMANCE STUDY OF GENETIC ALGORITHMS AND SIMULATED ANNEALING METHODS ON THE MACHINING MODELS

In this section, we present the performance study and evaluation of GA, SA and the CSA on the machining models discussed in Section 3. All the models were tested using the three algorithms with three different initial conditions to check for the consistency of the results. The results are presented in the form of tables with optimal cutting conditions and cost ($(.)_0$ and $(.)^*$ represent initial and optimal conditions respectively). All the results are discussed in the next section. The GA and SA programs were developed on an IBM PC using C++. For the GA, the population size was 240 per generation and the number of generations allowed were 250. The maximum number of function and constraints evaluations were 240×250. After considerable amount of parametric study on the solutions, we concluded that for the five models under consideration, we can reduce the population size to 100 per generation and increase the maximum number of generations allowed to 300 (considerable reduction in number of function and constraints evaluations) and still get similar results. But for other problems, results might be different.

Table 1. Results for Hati and Rao model

Method	Run #	V_0 (m/min)	f_0 (mm/rev)	V^* (m/min)	f^* (mm/rev)	Cost (yens/piece)	Function Evaluations
Simulated	1	255.0	0.525	148.215	0.3617	79.544	49500
Annealing	2	112.5	0.262	148.192	0.3617	79.545	70500
	3	375.0	0.875	148.176	0.3616	79.548	36000
Continuous	1	255.0	0.525	148.219	0.3617	79.542	36000
Simulated	2	112.5	0.262	148.219	0.3617	79.542	37000
Annealing	3	375.0	0.875	148.219	0.3617	79.542	41000
Genetic	1			145.703	0.3598	79.722	25000
Algorithm	2			147.710	0.3614	79.569	50000
	3			146.472	0.3589	79.821	40500

For SA, the temperature was reduced from 0.5 to 0.001 in 123 steps. For each temperature, there were 1000 iterations. This results in total number of function and constrains evaluations to be 123×1000, which is much higher than GA. But the SA results are better than the GA results for the given five models. The number of iterations at any given temperature can be reduced if the saturation phenomenon occurs at that temperature, i.e., if the cost function is not changing at that temperature, then proceed to next temperature. The number of variables to be perturbed for each iteration is selected randomly (each variable has a 60% chance of being selected). The percentage of perturbation to be applied to each variable, at each iteration, is also selected randomly (between -25% and $+25\%$). All random numbers are generated from a *uniform distribution random number generator.*

The CSA algorithm was designed to perturb the current solution 20 times in each coordinate direction before the bound along that direction is adapted based on the ratio of accepted and rejected solutions. The temperature is reduced by a factor of 0.85 for every 2000 adaptations of the search bounds. The CSA consistently generated better results than the GA and SA in addition to using smaller number of function evaluations, for all the benchmark problems.

The first model used in the study was the Hati and Rao model. Three runs were made with different initial vectors and the results are presented in Table 1. A depth of cut of 5 mm was used i.e., 2.5 mm for each pass.

The second model used in the study was Ermer's model. A depth of 0.2 inch in one pass was removed starting from three different points and the results are given in Table 2.

The third model used in the study was the Petropoulos model. Here a depth of cut of 3 mm in one pass was used with three starting points and the results are given in Table 3.

The fourth model used was Ermer and Kromodihardjo model. The results are shown in Table 4 where

Table 2. Results for Ermer's model

Method	Run #	V_0 (in/min)	f_0 (in/rev)	V^* (in/min)	f^* (in/rev)	Cost (dollars/pc)	Function Evaluations
Simulated	1	725.5	0.0050	143.917	0.001439	6.2553	65900
Annealing	2	376.2	0.0025	143.908	0.001439	6.2550	120000
	3	1368.0	0.0091	143.908	0.001439	6.2550	68500
Continuous	1	725.5	0.0050	143.9140	0.001439	6.2551	37000
Simulated	2	376.2	0.0025	143.9407	0.001439	6.2528	43000
Annealing	3	1368.0	0.0091	146.5218	0.001411	6.2531	31000
Genetic	1			145.068	0.001423	6.2758	65500
Algorithm	2			145.156	0.001416	6.2992	41000
	3			150.098	0.001423	6.3142	40250

Table 3. Results for Petropoulos model

Method	Run #	V_0 (m/min)	f_0 (mm/rev)	V^* (m/min)	f^* (mm/rev)	Cost pence/piece	Function Evaluations
Simulated	1	625.0	0.705	174.394	0.2321	12.097	89000
Annealing	2	250.0	0.282	174.402	0.2321	12.098	62900
	3	1041.0	1.175	174.402	0.2321	12.098	51000
Continuous	1	625.0	0.705	174.2229	0.2321	12.096	55000
Simulated	2	250.0	0.282	174.4328	0.2322	12.100	12000
Annealing	3	1041.0	1.175	174.2229	0.2320	12.097	38000
Genetic	1			174.622	0.2316	12.111	51500
Algorithm	2			174.446	0.2320	12.102	25250
	3			174.399	0.2321	12.099	49500

Table 4. Results of Ermer and Kromodihardjo model

Method	Run #	V_0 (in/min)	f_0 (in/rev)	V^* (in/min)	f^* (in/rev)	Cost (dollars/pc)	Function Evaluations
Simulated	1	755.5	0.05	433.809	0.003811	1.5528	42000
Annealing	2	591.25	0.01	433.980	0.003814	1.5526	88400
	3	1258.75	0.083	433.663	0.003810	1.5527	68900
Continuous	1	755.5	0.05	440.8529	0.003907	1.5526	80000
Simulated	2	591.25	0.01	441.2849	0.003908	1.5526	72000
Annealing	3	1258.75	0.083	433.1774	0.003803	1.5526	46000
Genetic	1			434.375	0.003814	1.5536	22000
Algorithm	2			434.398	0.003816	1.5533	41400
	3			435.371	0.003827	1.5537	36500

Table 5. Results for Iwata, Oba and Murotsu model

Method	Run #	V_0 (m/min)	f_0 (mm/rev)	V^* (m/min)	f^* (mm/rev)	Cost (yens/piece)	Function Evaluations
Simulated	1	275.25	0.865	216.019	0.3885	108.0334	39000
Annealing	2	110.10	0.322	216.013	0.3886	108.0327	70500
	3	458.75	1.340	216.013	0.3886	108.0332	51000
Continuous	1	275.25	0.865	216.0618	0.3886	108.0177	40000
Simulated	2	110.10	0.322	216.0601	0.3886	108.0181	36000
Annealing	3	458.75	1.340	216.0006	0.3886	108.0319	42000
Genetic	1			216.108	0.3879	108.093	59250
Algorithm	2			215.953	0.3878	108.137	27500
	3			215.948	0.3883	108.086	47000

0.2 inches of material was removed in one pass.

The fifth model in the study was the Iwata, Oba and Murotsu model. SA, CSA, and GA were used to solve this model for removing a depth of 2 mm and a surface roughness of H_{max}=0.006 mm. The results are presented in Table 5.

5. COMPARISON WITH CALCULUS BASED SEARCH METHODS RESULTS

In this section we compare the results of SA, CSA, and GA with calculus based search methods published by Duffuaa *et al.* [1] The next five tables show the comparison. For the GA, the initial values of the variables shown are not used and the population is initialized by randomly generating values in the given range of each variable. The cells showing **** indicate that the input vector led to the failure of the corresponding algorithm.

It is evident from Tables 6–10 that the stochastic optimization algorithms have consistently performed better than the gradient based approaches. It should be noted that very often the gradient based method did not converge to the optimal solution.

Table 6. Comparison table for Ermer and Kromodihardjo model

Run #	V_0 (m/min)	f_0 (mm/rev)	CSA (dollars/pc)	SA (dollars/pc)	GA (dollars/pc)	SUMT (dollars/pc)	Box (dollars/pc)	Hill dollars/pc	GRG dollars/pc
1	320	0.0018	1.5526	1.5528	1.5532	1.553	1.629	1.555	1.553
2	320	0.0039	1.5526	1.5526	1.5533	****	1.195	****	1.553
3	440	0.0018	1.5526	1.5527	1.5536	1.553	2.057	1.554	1.553
4	440	0.0039	1.5527	1.5527	1.5527	****	1.500	****	1.553

Table 7. Comparison table for Ermer and Kromodihardjo model

Run #	V_0 (m/mi)n	f_0 (mm/rev)	CSA (dollars/pc)	SA (dollars/pc)	GA (dollars/pc)	SUMT (dollars/pc)	Box (dollars/pc)	Hill (dollars/pc)	GRG (dollars/pc)
1	135	0.0011	6.251	6.2551	6.2758	6.26	6.29	6.31	6.26
2	135	0.0035	6.251	6.2551	6.2985	****	****	****	6.26
3	170	0.0011	6.250	6.2563	6.6259	6.26	6.30	6.27	6.26
4	170	0.0035	6.251	6.2557	6.3042	****	****	****	6.26

Table 8. Comparison table for Petropoulos model

Run #	V_0 (m/min)	f_0 (mm/rev)	CSA (pence/pc)	SA (pence/pc)	GA (pence/pc)	SUMT (pence/pc)	Box (pence/pc)	Hill (pence/pc)	GRG (pence/pc)
1	185	0.15	12.096	12.0979	12.099	12.10	13.08	11.74	12.10
2	185	0.20	12.096	12.0981	12.103	12.10	12.51	11.45	12.10
3	215	0.15	12.096	12.0979	12.112	12.10	13.41	11.79	12.10
4	215	0.20	12.092	12.0977	12.100	****	****	****	12.10

Table 9. Comparison table for Hati and Rao model

Run #	V_0 (m/min)	f_0 (mm/rev)	CSA (yens/pc)	SA (yens/pc)	GA (yens/pc)	SUMT (yens/pc)	Box (yens/pc)	Hill (yens/pc)	GRG (yens/pc)
1	146	0.375	79.545	79.5442	79.677	****	****	****	151.55
2	196	0.575	79.543	79.5443	79.722	****	****	****	151.55
3	146	0.375	79.542	79.5457	79.719	****	****	****	151.55
4	196	0.575	79.544	79.5465	79.581	****	****	****	151.55

Table 10. Comparison table for Iwata, Oba, and Murotsu model

Run #	V_0 (m/min)	f_0 (mm/rev)	CSA (yens/pc)	SA (yens/pc)	GA (yens/pc)	SUMT (yens/pc)	Box (yens/pc)	Hill (yens/pc)	GRG (yens/pc)
1	190	0.23	108.0180	108.0328	108.056	108.03	108.03	108.12	108.03
2	190	0.32	108.0180	108.0203	108.091	108.03	108.03	108.91	108.03
3	250	0.23	108.0176	108.0324	108.086	108.03	108.03	108.38	108.03
4	250	0.32	108.0135	108.0323	108.086	108.03	108.03	108.75	108.03

6. DISCUSSION OF RESULTS AND CONCLUSIONS

From the results obtained, we conclude that all three methods, i.e., SA, CSA, and GA are highly reliable and converge consistently to the optimum solutions for the five benchmark machining optimization problems. Significantly, all three methods were unaffected by the choice of the input vector, although, sometimes this led to an increase in the convergence time. The GA, SA and CSA only required changes which include programming the objective function and constraints for the different benchmark problems. SA gives high precision and the code can be run longer to get higher precision. For GA, the precision is limited by the number of bits used to represent each variable. We used 32 bits to represent each variable where the maximum precision depends on the parameter range which is resolved into 2^{32} parts. If the range is large, e.g., velocity has range of about 1450 in the given five models, then the precision goes down. For GA, to improve the precision we can increase the number of bits used to represent each variable. The CSA achieved high precision because of its adaptive capability.

All three methods converge to global minima and do not require any gradient information. This property makes these methods suitable for discontinuous functions. The disadvantage of these methods is the number of function evaluations required per run (i.e., the time required to converge) may be long. Total number of function and constraints evaluations for GA, CSA, and SA are much higher than any gradient based method. This results in longer convergence times and make these methods not very attractive for real-time parameter optimization.

REFERENCES

1. Duffuaa, S. O., Shuaib, A. N. and Alam, A., Evaluation of Optimization methods for machining economics models. *Comp. Oper. Res.* , 1992, **20**, 227–237.
2. Petropoulos, P., Optimal selection of machining variables using geometric programming. *Int. J. Prod. Res.* , 1973, **11**, 305–314.
3. Ermer, D. S. and Kromodihardjo, S., Optimization of multipass turning with constraints. *J. Engng Industry, Trans. ASME*, 1981, **103**, 462–468.
4. Iwata, K., Murotsu, Y. and Oba, F., Optimization of cutting conditions for multipass operations considering probablistic nature in machining processes, *J. Engng Industry, Trans. ASME Ser. B*, 1977, pp. 210–217.
5. Eskicioglu, A. M. and Eskicioglu, H., Optimization of machining conditions in metal cutting with non-linear programming, *Engng Systems Design and Analysis, ASME 1992*, **PD-47-1**, 1992, pp. 95–100.
6. Hati, S. K. and Rao, S. S., Determination of machining conditions using probablistic and deterministic approaches, *J. Engng Industry, Trans. ASME* Paper No. 75-Prod.-K.
7. Holland, J. H., *Adaptation in Natural and Artificial Systems*. MIT press, 1975, pp. 9–16.
8. Kirkpatrick, S., Gellat, C. D. and Vecchi, M. P., Optimization by Simulated Annealing. *Science*, 1983, **220**, 671–680.
9. Beasly, D., Bull, D. R. and Martin, R. R., An Overview of Genetic Algorithms: Part 1 Fundamentals. *Univ. Comp.* , 1993, **15**, 2 58–69.
10. Beasly, D., Bull, D. R. and Martin, R. R., An Overview of Genetic Algorithms Part 2 Research Topics. *Univ. Comp.* , 1993, **15** , 4 170–181.

11. Ermer, D. S., Optimization of the constrained machining economics problem by geometric programming. *Trans. ASME*, 1971, **93**, 1067–1072.
12. Metropolis, N., Rosenbluth, A., Rosenbluth, M., Teller, A. and Teller, E., Equations of state calculations by fast computing machines, *J. Chem. Phys.*, 1953, pp. 1087–1092.
13. Corana, A., Marchesi, M., Martini, C. and Ridella, S., Minimizing Multimodal Function of Continuous Variables with the 'Simulated Annealing' Algorithm. *ACM Transaction on Mathematical Software*, 1987, **13**, 262–280.