PRODUCTIVITY IMPROVEMENT BY SA AND GA BASED MULTI-OBJECTIVE OPTIMIZATION IN CNC MACHINING

Prof. (Dr). Rachayya.R.Arakerimath
Department of Mechanical Engineering, G H Raisoni College of Engg and Mgmt
Dighi Hills, Pune - 411015, India. Email : Rachayya_aict@rediffmail.com

Prof (Dr).V.A.Raikar
Principal Govt College of Engineering, Karawar, Karnataka, India

ABSTRACT

The manufacturing automation aims for high quality of product with minimum production cost and machining time. The production cost can be minimized by reducing the machining time using optimized machining conditions and proper setting of various parameters during machining. Therefore there is a vital need to optimise and correlate machining parameters for economic machining at higher quality of the product. This paper presents a multi-objective optimization technique based on Simulated annealing (SA) and Genetic Algorithm (GA) using MATLAB 7.0. It optimizes the cutting parameters in CNC turning process using three conflicting objectives, like surface roughness, Cutting force and Cost of production simultaneously. The proposed model uses SA and GA as optimization method for C45 and SS410 work pieces with carbide tool inserts on CNC machine. An application sample is developed and its results are analyzed for several different production conditions. It has been found that the results of multi-objective optimization will reduce the machining cycle times by 27 to 38% and machining cost by 25-38%. The optimized results also increase the tool utilization by reducing tool breakage. The optimized results can be used as data base in machining expert system.

Keywords
CNC Turning, Multi objective optimization (MOO), multiple regressions Analysis simulated annealing (SA), Genetic algorithm (GA). Production cost.

1. Introduction
The optimization of metal cutting in general has a long history starting from first decade to the present century. A few of the prominent researchers are F.W.Taylor, Brewer, Armerego, and Russell. Basu S.K, Battacharya, Ramaswamy and lambarty. Cook and Peters. Consideration of machining parameter optimization started by Taylor (1907) and has increased since 1950. Ermer and kromodihardjo (1981)
considered turning operations in their study and concluded that double and triple pass operations are more economical than single pass operations. Prasad and Rao (1997) used geometric programming and linear programming model to determine machining parameters in turning process for CAPP. Shunmugam, Reddy et al (2000) used genetic algorithm to optimize milling process parameters in multi pass milling operation. Ahmari (2001) presented a non linear programming model to optimize machining parameters in multi pass turning process for minimum production cost with constraints. Based on the above points and development in computer based optimization, SA and GA based multi-objective optimization (MOO) is explained in this paper.

In traditional SA/GA, the fitness function deals only with one optimization objective. Many practical problems however are concerned with several equally important objectives. These types of problems are called multi-objective or multi-criteria optimization problems (MOO). The MOO can be performed by using SA or Genetic algorithm, Because of the drawbacks involved in the traditional methods. The objectives may be minimum production time per piece, Minimum surface roughness, minimum power consumption etc. The machining parameters affecting the process objectives are cutting speed, feed, and depth of cut and time of machining and nose radius. The optimum set of these input variables and objectives are determined for Tungsten carbide insert tool with C45, and SS410 work pieces during single pass CNC Turning process. The optimized results of GA can be compared with the any other optimization techniques for accuracy.

CNC machining can be fully optimized through the implementation of add-on adaptive control systems with MOO, which continuously calculates cutting conditions in real time. Such optimization and machine automation technology systems are indispensable if expensive CNC machines are ever to run at their full capacity and if cutting tools are to be utilized up to their maximum life rather than incurring in-process catastrophic breakage and production disruption. Similarly, machine operators will not be required to intervene in the machining process to watch and manually fine-tune the process. In this way, true automation is made a reality and programmers may be more aggressive, knowing the optimized values to adjust the feed rate based on the load. The optimized data will replace the age old practice of manufacturing industry in selecting cutting tool solely on experience of engineers and machine operators.

2. Simulated annealing

The concept of Simulated Annealing (SA) was first published in 1953 by Metropolis et al [5]. Annealing is a material cooling process that first heats the material to melting point and then decreases the temperature. Material can be formed into the preferable structure by controlling the temperature change during the cooling process. SA is a variant of a local search that solves optimization problems using the same concept. SA gives flexible search directions by first assigning higher temperature and then reduces the flexibility to identify preferable solutions by decreasing the temperature. Empirical experiments show that near optimal results for both linear and non-linear problems can be found by SA. The methodology approximates the objective function of a simulation model and solves the objective function by the SA algorithm.

Simulated annealing is a stochastic algorithm used for optimization problems where the objective function corresponds to the energy of the states of a solid [6]. The SA algorithm requires the definition of the neighborhood structure as well as the parameters for the cooling schedule. The temperature parameter distinguishes
between large and small changes in the objective function. Large changes occur at high temperatures and small changes at low temperatures. It is an evolutionary process moving in small steps from one stage to another, avoiding the problem of getting stuck in a local minimum by allowing uphill and downhill moves for the temperature.

SA is a good tool for solving optimization problems. Some examples on the capabilities of the simulated annealing algorithm to solve optimization problems can be found in [13, 14]. According to [14] the four basic requirements for using the SA algorithm in combinatorial optimization problems are:
(a) concise description of the problem. (b) Random generation of the changes from one configuration to another. (c) an objective function containing the utility function of the trade-offs (d) the initial state, the number of iterations to be performed at each temperature and its annealing scheme (for a detailed discussion of the algorithm see [6, 14]).

2.1. SA Implementation

For the implementation of the SA algorithm we need to choose the following parameters: how to generate a state y a neighbor of x; the aggregation function - Miu; the number of neighbor generated - N (t); the decreasing temperature function - T (t); and finally the stopping criterion.

The choice of how to generate a state y as a neighbor of x, is done by defining a new state which is a random point y where the distance to the point x is random and less than t, defined by
\[ F(t) = \begin{cases} 
1 & t < 100 \\
2 & 100 \leq t < 150 \\
3 & 150 \leq t < 250 \\
5 & 250 \leq t < 350 \\
15 & t \geq 350 
\end{cases} \]

The aggregation function selected is the intersection, and the operator used in this implementation is the t-norm min. The intersection represents the logical ‘and’ to signify that all the constraints must be satisfied. As defined by Bellman and Zadeh [12] a decision is represented by the confluence of goals and constraints (intersection) and the best decision is the one with the maximum value.

The choice for number of neighbors to be generated, N (t), follows the heuristic of generating 200 neighbors, if t < 400, and generating 250, if t > 400. This heuristic takes in account that more sons should be generated when the temperature, T, decreases to have more options to test. The temperature's function T (t) is a function which uses a decreasing factor of 0.99. The stopping criterion used for the implementation is reached when the temperature is less than 0.0001.

2.2. Steps used in SA

In 1953, Metropolis and coworkers [5] first incorporated these kinds of principles into numerical calculations. Offered a succession of options, a simulated thermodynamic system was assumed to change its configuration from energy \( E_1 \) to energy \( E_2 \) with probability \( p = \exp \left[ - \frac{(E_2 - E_1)}{kT} \right] \). Notice that if \( E_2 < E_1 \), this probability is greater than unity; in such cases the change is arbitrarily assigned a probability \( p = 1 \), i.e., the system always took such an option. This general scheme, of always taking a downhill step while sometimes taking an uphill step, has come to be known as the Metropolis algorithm. To make use of the Metropolis algorithm for other than thermodynamic systems, one must provide the following elements:
1. Description of possible system configurations.
2. A generator of random changes in the configuration; these changes are the “options” presented to the system.
3. An objective function $E$ (analog of energy) whose minimization is the goal of the procedure.
4. A control parameter $T$ (analog of temperature) and an annealing schedule which tells how it is lowered from high to low values, e.g., after how many random changes in configuration is each downward step in $T$ taken, and how large is that step. The meaning of “high” and “low” in this context, and the assignment of a schedule, may require physical insight and/or trial-and-error experiments.

3. Genetic Algorithm

A genetic algorithm (GA) is a search technique used in engineering to find approximate solutions to the optimization and search problems. Genetic algorithms are a particular class of evolutionary algorithms [7]. They use techniques inspired by evolutionary biology such as inheritance, mutation, selection, and crossover (also called recombination). Genetic algorithms are typically implemented as a computer simulation, in which a population of string representations (called chromosome) the candidate solution (called individuals) to an optimization problem that evolves towards better solutions. Traditionally, solutions are represented in binary as strings of 0s and 1s, but different encodings are also possible. The evolution starts from a population of completely random individuals and happens in generations. In each generation, the fitness of the whole population is evaluated, the multiple individuals are stochastically selected from the current population (based on their fitness), and modified (mutated or recombined) to form a new population. The new population is then used in the next iteration of the algorithm.

The main drawbacks of traditional methods lies in the fact that there is a chance of solutions getting trapped into the local minima, where as GA is population based search and optimization technique the chances of its solutions getting trapped into local minima is very less and also the various advantages of GA Techniques are:

1) GA’s do not require extensive problem formulation.
2) The objective function need not be continuous and differentiable for GA applications.
3) GA is able to find the global optimum solution easily.
4) GA is population based search and optimization techniques.
5) Multiple optimal solutions can be obtained.
6) Multi-modal functions can be solved easily by GA.

3.1. Steps in GA

1. Start: Generate random population of ‘n’ chromosomes (suitable solutions for the problem)
2. Fitness: Evaluate the fitness $f(x)$ of each chromosomes in the population
3. New population: Create a new population by repeating following steps until the new population is complete: a) Selection: Select two parent chromosomes from a population according to their fitness (the better fitness, the bigger chance to be selected). b) Crossover: With a crossover probability cross over the parents to form new offspring (children). If no crossover was performed, offspring is the exact copy of parents. c) Mutation: A mutation probability, mutate new offspring at each locus (position in chromosome). d) Accepting: Place new offspring in the new population
4. Replace: Use new generated population for a further run of the algorithm
5. Test: If the end condition is satisfied, stops and returns the best solution in current population

The final problem is to decide when to stop execution of algorithm. There are two possible [7] solutions to this problem: First approach is stop after production of definite number of generations. Second approach is stop when the improvement in average fitness over two generations is below a threshold value.

4. Multi objective optimization

In a traditional method of SA/GA the fitness function deals only with one optimization objective. Many practical problems, however, are concerned with several equally important objectives. These types of problems are called multi-objective or multi-criteria [8] optimization problems (MOO). The simplest and most common way to tackle MOO is to combine its several objectives into one scalar function as the fitness function. Different objectives in the problem are given different weights based on some prior knowledge. In dealing with multi objective optimization problems, classical optimization methods (weighted sum methods, goal programming, min–max methods etc) are not efficient, because they cannot find multiple solutions in a single run, thereby requiring them to be applied as many times as the number of desired optimal solutions. On the contrary, studies on evolutionary algorithms have shown that these methods can be efficiently used to eliminate most of the above-mentioned difficulties of classical methods [9]. In this paper a multi-objective optimization method based on SA and genetic algorithm is proposed to obtain the optimal parameters in turning processes.

4.1. Mathematical modeling

The multi-objective optimization (MOO) approach obtains a set of solutions by generating the optimal solution for the objective functions while holding all others as constant. The constant value for the objective functions are moved to cover the full rage of their possible values and the optimal solutions obtained for these conditions generate a trade–off values (satisfaction curves are also used). Different satisfaction criterion would lead to different acceptable solutions. A general optimization problem for multi-objective function can be stated as:

Find F(X) which minimizes, \[ f1(x) + f2(x) + f3(x) \]. Subjected to the constraints:
\[ G_j(x) = 0 \quad j = 1, 2 \ldots \ldots \ldots n \]
\[ L_k(x) = 0 \quad k = 1, 2 \ldots \ldots \ldots m \]

In the above context the optimization problem for the turning operation can be presented as follows:
The design variable vector may be taken as,
\[ X = [V \ f \ d \ r] = [x1 \ x2 \ x3 \ x4] \]

An experiment was conducted on CNC machining centre (lathe machine) using various work pieces of Size 300mm x Dia 40mm. (C45 and SS410) and Carbide tipped tools. (Insert was-CNMG120404/08-HM). The process variables used during experimental work are shown in table-1. Using design of experiments the observations are made and then by using multiple regression analysis the mathematical models are developed for various machining objectives are noted.

The mathematical models are formulated for Surface roughness, cutting force and Cost of production for C45 and SS410 work pieces with Tungsten carbide tool insert are listed.

4.1 a) Mathematical models for MOO (C45 with carbide insert tool):

\[ Ra = 168.781(V)^{1.7632}(f)^{1.3595}(d)^{0.766}(r)^{0.03338} \]
\[ Fc = 39.0288(V)^{-1.0192}(f)^{-0.0567}(d)^{0.0635}(r)^{-0.0812} \]
Cp = cost of production per piece = Co.n.3.142 DL/1000Vf + [n.3.142 DL/1000α] (Tc. Co + Ct) (V)^1.25 (f)^0.75 (d)^0.25 (r)^0.25 + Co.Th.

4.1 b) Mathematical models for MOO (SS410 with Carbide insert tool):
Ra = 10.446 (V)^0.8880 (f)^0.313 (d)^0.446 (r)^-0.285.
Fc = 38.2675(V)^-0.9435 (f)^0.04035 (d)^0.04041 (r)^-0.0096.
Cp = Co.n.3.142 DL/1000Vf + [n.3.142 DL/1000α] (Tc. Co + Ct) (V)^1.415 (f)^0.65 (d)^-0.215 (r)^0.24 + Co.Th.

Where, D=39mm.L=210mm.Co=10Paise/min. Ct=50paise/edge.Tc=0min.Th=1min.α =60x10^{-10}.

5. Steps in MOO
1. Determine the design objectives such as Cutting force, Surface roughness, Power consumption, and Production cost/time etc.
2. Identify the system parameters affecting the objective function.
3. Build statical/experimental model of the process and express objective function as a function of system parameters using ANOVA.
4. Formulate Multi-objective function f(x):
Minimize……f1(x) =Surface roughness (Ra).
Minimize……f2(x) =Cutting force (fc).
Minimize……fn (x) =Cost of production (Cp).
Hence for MOO, f(x) =Ra+ fc+ Cp.
5. Formulate the constraints of objective f(x) as:
a) 62<=v<=113, b) 0.1<=f<=0.2,
c) 1<=d<=2 and d) 0.4<=r<=0.8.
e) T>1400>Tmax. F) Pc>Pmax.
6. Develop the multi-objective optimization algorithm using Turbo C/C++ or use MATLAB SA tool and GA tool.
7. Use the MATLAB SA tool and GA tool for MO Optimization
8. Naturally GA is a maximization problem. This can be converted into minimization problem.

6. Results and discussions
The outputs of the SA are obtained by running the SA tool from MATLAB7.0 programme using inputs shown in table-1. The optimized results by SA are shown in table-6 and GA results are in table-7. From the optimized SA outputs the variation of number of iteration with Best and current fitness are drawn as shown in figure-1 and 2. The variation of number of generation with Best and current fitness obtained by GA are drawn in figure-3 and 4.

We found that initially the fitness was varying in nature. But as number of generation increases the relation remains constant. This means that the obtained results are optimum [10]. The optimized values obtained by SA are compared by GA or any of the optimization method for correctness. The theoretical machining time(Tm) in CNC turning [11] is given by 3.142.D.L/1000.Vo.fo. Then by using the theoretical values of V and f the range of machining time is calculated as 1.139-4.15 minutes. But by the optimized values of SA/GA, the optimized machining times are: 2.075 and 1.1386 minutes for C45 and SS410 respectively as in table-5.

From obtained results we conclude that multi-objective optimization helps in saving the machining time by 27-48% with lower production cost (table-5). This means that optimized machining parameters helps to improve productivity. It is also
found that MATLAB based optimization is faster and accurate method to calculate optimum values of complex machining objectives at different operating conditions.

7. Conclusion

The optimum results obtained by multi-objective optimization (MOO) are shown in table-4. And found that the optimized values of SA or GA are within the selected theoretical range. We note that, the fitness value for C45 Carbide insert tool = 96.809, whereas for SS410-Carbide insert =69.8820 (table-4). Multi objective evolutionary algorithms can be used to deal very complex problems involving many machining objectives. This may not be solved or rather very difficult to solve mathematically by other methods. The combined optimization of Surface roughness, cutting force and Production cost will help as a data base in expert system of adaptive control (optimization) machining.

The GA based multi objective optimization helps to reduce the machining cycle times by 27 to 38% depending upon the application and work-tool combination. The MOO results also help in the following:

a) Will increase the tool utilization and prevent the tool breakage
b) Helps to reduce the machining cost and increases the machining capacity.
c) The optimized results help to overcome the use of costlier probes in adaptive control optimisation.
d) GA results are optimum in terms of speed, accuracy and calculations
e) Further Study can also be extended for optimization of various machining operations and different objectives.

8. References

11) Juneja and Sekhan. Fundamentals of metal cutting and machine tools, New Age
9. List of tables and figures

Table-1: Cutting parameters used

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Cutting Speed (V) in m/min</th>
<th>Feed (f) in mm/rev</th>
<th>Depth of cut (d) in mm</th>
<th>Nose radius (r) in mm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max</td>
<td>113</td>
<td>0.1</td>
<td>2</td>
<td>0.4</td>
</tr>
<tr>
<td>Mid</td>
<td>88</td>
<td>0.15</td>
<td>1.5</td>
<td>0.8</td>
</tr>
<tr>
<td>Min</td>
<td>62</td>
<td>0.2</td>
<td>1</td>
<td>0.8</td>
</tr>
</tbody>
</table>

Table-2: Inputs to Multi Objective Optimization (SA)

```matlab
>> satool
>> x0=[0 0 0 0];     >> lb=[62 .1 1 .4];   >> ub=[113 .2 2 .8];
>> options=saoptimset('plotfcns',{@saplotbestf,@saplotbestx,@saplotf,@saplotx,
@saplotstopping,@saplottemperature}, 'display','iter', 'hybridfcn', @fmincon, 'maxiter',
1500);>> [x, fval,exitflag,output]=simulannealbnd(@cmmo, x0, lb, ub, options)
```

Table-3: Inputs to Multi Objective Optimization (GA)

Population size : 10, Cross over probability : 0.8000, Mutation probability (binary) : 0.02000, Mutation probability (real) : 0.02000, Total String length : 32, Number of binary-coded variables : 4, Number of real-coded variables : 4, Exponent (n for SBX) : 1.00, Exponent (n for Mutation) : 1.00, Termination criteria : Number of generations.

| Lower and Upper bounds | x_bin[1] | <= 113.0000, string length = 8, 0.1000 | <= x_bin[2] | <= 0.2000, string length = 8, 1.0000 | <= x_bin[3] | <= 2.0000, string length = 8, 0.4000 | <= x_bin[4] | <= 0.8000, string length = 8, 62.0000 | <= x_real[1] | <= 113.0000, 0.1000 | <= x_real[2] | <= 0.2000, 1.0000 | <= x_real [3] | <= 2.0000, 0.4000 | <= x_real [4] | <= 0.80. |
Table 4: Optimum values (MOO) by SA and GA

<table>
<thead>
<tr>
<th>Optimization</th>
<th>Work pieces</th>
<th>V</th>
<th>f</th>
<th>d</th>
<th>R</th>
<th>Optimum Outputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>SA-CMMO1</td>
<td>C45</td>
<td>62</td>
<td>0.2</td>
<td>1</td>
<td>0.8</td>
<td>96.8093</td>
</tr>
<tr>
<td>GA-CMMO1</td>
<td>C45</td>
<td>62</td>
<td>0.2</td>
<td>1</td>
<td>0.4</td>
<td>97.7</td>
</tr>
<tr>
<td>SA-CMMO2</td>
<td>SS410</td>
<td>113</td>
<td>0.2</td>
<td>1</td>
<td>0.4</td>
<td>69.82</td>
</tr>
<tr>
<td>GA-CMMO2</td>
<td>SS410</td>
<td>113</td>
<td>0.2</td>
<td>1</td>
<td>0.8</td>
<td>69.558</td>
</tr>
<tr>
<td>Theoretical</td>
<td>C45/SS410</td>
<td>62-113</td>
<td>0.1-0.2</td>
<td>1-2</td>
<td>0.4-0.8</td>
<td>---------------</td>
</tr>
</tbody>
</table>

Table 5: Machining time (Tm) comparison

<table>
<thead>
<tr>
<th>Work-tool insert type</th>
<th>Theoretical ‘Tm’ in minute by low and high values of V and f.</th>
<th>Optimum ‘Tm’ by SA from the optimized V and f.</th>
<th>Optimum ‘Tm’ by GA from the optimized V and f.</th>
<th>Percentage saving of ‘Tm’</th>
</tr>
</thead>
<tbody>
<tr>
<td>C45-carbide</td>
<td>1.139-4.15</td>
<td>2.0275</td>
<td>2.0275</td>
<td>48%</td>
</tr>
<tr>
<td>SS410-carbide</td>
<td>1.139-4.15</td>
<td>1.1386</td>
<td>1.1386</td>
<td>27.436%</td>
</tr>
</tbody>
</table>

Figure-1, Iteration v/s Best function value in C45 (SA)  
Figure-2, Iteration v/s Best function value in SS410 (SA)
Symbols used

SA = Simulated Annealing, GA= Genetic algorithm,  V= Cutting speed in m/min. n = No of finish passes. f= Feed in mm/rev. d= Depth of cut in mm. r= Nose radius in mm, I= Current consumed in Ampere. Fc= Cutting force in Newton. Ra= Surface roughness in mm or micron. Pc= Power consumed in cutting in watts. Cp = Production cost in Paisa. Co= Operating cost in paisa/edge Ct= Tooling cost in paisa/edge. Th= Handing cost in paisa. Tc= Tool change cost in paisa. Tm= Machining time in min. MOO= Multi-objective optimization. L= length of job in mm. CNC= Computer Numerical Controlled.