Proceedings of COBEM 2003 17th International Congress of Mechanical Engineering
COBEM2003 - 0513 Copyright © 2003 by ABCM November 10-14, 2003, São Paulo, SP

DETERMINATION AND OPTIMIZATION OF PARAMETERS IN MILLING BY GA APPROACH

Franc Cus
University of Maribor, Faculty of Mechanical Engineering, Smetanova 17, 2000 Maribor, Slovenia
franc.cus@uni-mb.si

Joze Balic
University of Maribor, Faculty of Mechanical Engineering, Smetanova 17, 2000 Maribor, Slovenia
joze.balic@uni-mb.si

Janez Kopac
University of Ljubljana, Faculty of Mechanical Engineering, Aškerceva 6, 1000 Ljubljana, Slovenia
janez.kopac@fs.uni-lj.si

Abstract. *The paper proposes a new optimization technique based on genetic algorithms for the determination of the cutting parameters in machining operations. In metal cutting processes, cutting conditions have an influence on reducing the production cost and time and deciding the quality of a final product. This paper presents a new methodology for continual improvement of cutting conditions with GA (Genetic Algorithms). It performs the following: the modification of recommended cutting conditions obtained from a machining data, learning of obtained cutting conditions using neural networks and the substitution of better cutting conditions for those learned previously by a proposed GA. Experimental results show that the proposed genetic algorithm-based procedure for solving the optimization problem is both effective and efficient, and can be integrated into an intelligent manufacturing system for solving complex machining optimization problems.*

Keywords *Genetic Algorithm, Cutting Parameters, Manufacturing, Simulation*

1. Introduction

In today's manufacturing environment, many large industries have attempted to introduce flexible manufacturing systems (FMS) as their strategy to adapt to the ever-changing competitive market requirements. To ensure the quality of machining products, and to reduce the machining costs and increase the machining effectiveness, it is very important to select the machining parameters when the machine tools etc. are selected in CNC machining.

The traditional methods for solving this kind of optimization problem include calculus-based searches, dynamic programming, random searches, and gradient methods whereas modern heuristic methods include, artificial neural networks (Pandey at al. 1995), Lagrangian relaxation approaches (Hsu at al. 1995), and simulated annealing (Pandey at al. 1995). Some of these methods are successful in locating the optimal solution, but they are usually slow in convergence and require much computing time. Other methods may risk being trapped at a local optimum which fails to give the best solution. In this paper, a novel approach, genetic algorithms (GA), based on the principles of natural biological evolution, have received considerable and increasing interest over the past decade, will be used to tackle this kind of problem. Compared to traditional optimization methods, a GA is robust, global and may be applied generally without recourse to domain-specific heuristics. It can be used not only for general optimization problems, but also in indifferent optimization problems and unconventional optimization problems, etc. So GA's are widely used for machine learning, function optimising and system modelling etc. (Goldberg 1989, Hui at al. 1996). Although GA is an effective optimization algorithm, it usually takes a long time to optimise machining parameters because of its slow convergence speed. In this paper genetic algorithm for optimization of cutting parameters GA is proposed based on traditional genetic algorithms. The operating domain is defined and changed to be around the optimal point in its evolutionary processes so that the convergence speed and accuracy are improved. The genetic algorithm is used for the optimization of cutting parameters and simulation and experimental results show improved performance.

2. Genetic algorithms

Genetic Algorithms are a family of computational models inspired by evolution. These algorithms encode a potential solution to a specific problem on a simple chromosome-like data structure and apply recombination operators to these structures so as to preserve critical information. Genetic algorithms are often viewed as function optimizers, although the range of problems to which genetic algorithms have been applied is quite broad.

An implementation of a genetic algorithm begins with a population of (typically random) chromosomes. One then evaluates these structures and allocates reproductive opportunities in such a way that those chromosomes which represent a better solution to the target problem are given more chances to "reproduce" than those chromosomes which are poorer solutions. The "goodness" of a solution is typically defined with respect to the current population. This particular description of a genetic algorithm is intentionally abstract because in some sense, the term genetic algorithm has two meanings. In a strict interpretation, the genetic algorithm refers to a model introduced and investigated by John Holland (Holland 1975) and by students of Holland (e.g., DeJong, (DeJong 1975)). It is still the case that most of the existing theory for genetic algorithms applies either solely or primarily to the model introduced by Holland, as well as variations of genetic algorithms.

In a broader usage of the term, a genetic algorithm is any population-based model that uses selection and recombination operators to generate new sample points in a search space. Many genetic algorithm models have been introduced by researchers largely working from an experimental perspective. Many of these researchers are application oriented and are typically interested in genetic algorithms as optimization tools. Modelling, machining, selection of cutting parameters and monitoring often have to deal with the problem of optimization.

2.1. Evolution and genetic algorithms

Holland had a double aim: to improve the understanding of natural adaptation process, and to design artificial systems having properties similar to natural systems (Holland 1975).

The basic idea is as follow: the genetic pool of a given population potentially contains the solution, or a better solution, to a given adaptive problem. This solution is not "active" because the genetic combination on which it relies is split between several subjects. Only the association of different genomes can lead to the solution. No subject has such a genome, but during reproduction and crossover, new genetic combination occur and, finally, a subject can inherit a "good gene" from both parents.

Holland method is especially effective because he not only considered the role of mutation (mutations improve very seldom the algorithms), but he also utilized genetic recombination, (crossover): these recombination, the crossover of partial solutions greatly improve the capability of the algorithm to approach, and eventually find, the optimum.

2.2. Defining genetic algorithms

What exactly do we mean by the term Genetic Algorithms? Goldberg (Goldberg 1989) defines it as: Genetic algorithms are search algorithms based on the mechanics of natural selection and natural genetics.

GA's exploit the idea of the survival of the fittest and an interbreeding population to create a novel and innovative search strategy. A population of strings, representing solutions to a specified problem, is maintained by the GA. The GA then iteratively creates new populations from the old by ranking the strings and interbreeding the fittest to create new strings, which are (hopefully) closer to the optimum solution to the problem at hand. So in each generation, the GA creates a set of strings from the bits and pieces of the previous strings, occasionally adding random new data to keep the population from stagnating. The end result is a search strategy that is tailored for vast, complex, multimodal search spaces.

GA's are a form of randomized search, in that the way in which strings are chosen and combined is a stochastic process. This is a radically different approach to the problem solving methods used by more traditional algorithms, which tend to be more deterministic in nature, such as the gradient methods used to find minima in graph theory.

The idea of survival of the fittest is of great importance to genetic algorithms. GA's use what is termed as a fitness function in order to select the fittest string that will be used to create new, and conceivably better, populations of strings. The fitness function takes a string and assigns a relative fitness value to the string. The method by which it does this and the nature of the fitness value does not matter. The only thing that the fitness function must do is to rank the strings in some way by producing the fitness value. These values are then used to select the fittest strings. The concept of a fitness function is, in fact, a particular instance of a more general AI concept, the objective function.

2.3. Genetic algorithms: A natural perspective

The population can be simply viewed as a collection of interacting creatures. As each generation of creatures comes and goes, the weaker ones tend to die away without producing children, while the stronger mate, combining attributes of both parents, to produce new, and perhaps unique children to continue the cycle. Occasionally, a mutation creeps into one of the creatures, diversifying the population even more. Remember that in nature, a diverse population within a species tends to allow the species to adapt to it's environment with more ease. The same holds true for genetic algorithms.

2.4. The iteration loop of a basic genetic algorithm

The following flowchart shows the interative cycle of a basic genetic algorithm. Firstly, an initial population of strings is created. The process then iteratively selects individuals from the population that undergo some form of transformation (via the recombination step) to create new a population. The new population is then tested to see if it fulfills some stopping criteria. If it does, then the process halts, otherwise another iteration is performed. (Diagram taken from Blickle, (Blickle 1995)).

2.5. Basic genetic algorithm operations

With GA's having such a solid basis in genetics and evolutionary biological systems, one might think that the inner workings of a GA would be very complex. In fact, the opposite is true. Simple GA's are based on simple string copying and substring concatenation, nothing more, nothing less. Even more complex versions of GA's still use these two ideas as the core of their search engine. All this will become clear when we walk through a simple GA optimization problem.

There are three basic operators found in every genetic algorithm: reproduction, crossover and mutation.

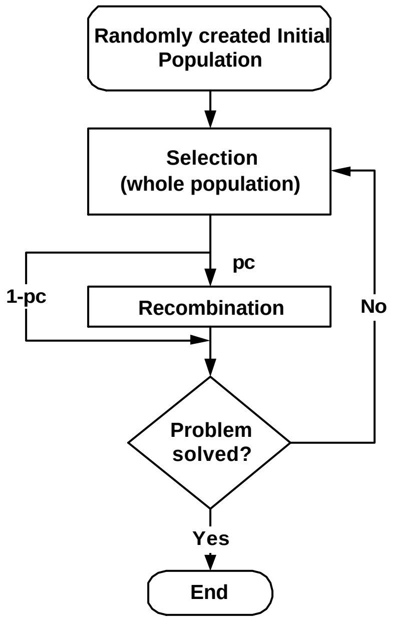
Randomly created Initial Population
Selection (whole population)
pc
1-pc
Recombination
No
Problem solved?
Yes
End

Figure 1. Flowchart of the basic genetic algorithm.

Reproduction

The reproduction operator allows individual strings to be copied for possible inclusion in the next generation. The chance that a string will be copied is based on the string's fitness value, calculated from a fitness function. For each generation, the reproduction operator chooses strings that are placed into a mating pool, which is used as the basis for creating the next generation.

There are many different types of reproduction operators

- Proportional Selection (This method will only work with fitness values above zero (non-negative) and scaling may sometimes be necessary. It has been shown that proportional selection performs poorly compared with other selection schemes in many GA problems.),
- Tournament Selection (Choose t individuals at random from the population and copy the best individual from this group into the new population. Repeat N times.),
- Truncation Selection (With truncation selection that has a threshold of T between 0 and 1, only the fraction T best individuals can be selected. They all have the same selection probability.),
- Linear Ranking Selection (The individuals are sorted according to their fitness values and the rank N is assigned to the best individual, the rank 1 assigned to the worst. The selection probability is linearly assigned to the individuals according to their rank and a selection equation.),
- Exponential Ranking Selection (This follows the same methodology of linear ranking selection, the only difference being that the probabilities of the ranked individuals are exponentially weighted.).

One always selects the fittest and discards the worst, statistically selecting the rest of the mating pool from the remainder of the population. There are hundreds of variants of this scheme. None are right or wrong. In fact, some will perform better than others depending on the problem domain being explored. For a detailed, mathematical comparison of reproduction/selection strategies for genetic algorithms.

Crossover

Once the mating pool is created, the next operator in the GA's arsenal comes into play. Remember that crossover in biological terms refers to the blending of chromosomes from the parents to produce new chromosomes for the offspring. The analogy carries over to crossover in GA's.

The GA selects two strings at random from the mating pool. The strings selected may be different or identical, it does not matter. The GA then calculates whether crossover should take place using a parameter called the crossover probability.

If the GA decides not to perform crossover, the two selected strings are simply copied to the new population. If crossover does take place, then a random splicing point is chosen in a string, the two strings are spliced and the spliced regions are mixed to create two (potentially) new strings. These child strings are then placed in the new population.

As an example, say that the strings 10000 and 01110 are selected for crossover and the GA decides to mate them. The GA selects a splicing point of 3.

The following then occurs :

$$\begin{array}{c|c} 100 & 00 \\ 011 & 10 \end{array} \longrightarrow \begin{array}{cc} 100 & 10 \\ 011 & 00 \end{array}$$

Figure 2. Example of the crossover operation.

The newly created strings are 10010 and 01100.

Crossover is performed until the new population is created. Then the cycle starts again with selection. This iterative process continues until any user specified criteria are met.

Mutation

Selection and crossover alone can obviously generate a staggering amount of differing strings. However, depending on the initial population chosen, there may not be enough variety of strings to ensure the GA sees the entire problem space. Or the GA may find itself converging on strings that are not quite close to the optimum it seeks due to a bad initial population.

Some of these problems are overcome by introducing a mutation operator into the GA. The GA has a mutation probability, m, which dictates the frequency at which mutation occurs. Mutation can be performed either during selection or crossover (though crossover is more usual). For each string element in each string in the mating pool, the GA checks to see if it should perform a mutation. If it should, it randomly changes the element value to a new one. In our binary strings, 1s are changed to 0s and 0s to 1s. For example, the GA decides to mutate bit position 4 in the string 10000.

$$10000 \longrightarrow 10010$$

Figure 3. Example of the mutation operation.

The resulting string is 10010 as the fourth bit in the string is flipped. The mutation probability should be kept very low (usually about 0.001%) as a high mutation rate will destroy fit strings and degenerate the GA algorithm into a random walk, with all the associated problems.

But mutation will help prevent the population from stagnating, adding "fresh blood", as it were, to a population. Remember that much of the power of a GA comes from the fact that it contains a rich set of strings of great diversity. Mutation helps to maintain that diversity throughout the GA's iterations.

2.6. Encodings and optimization problems

Usually there are only two main components of most genetic algorithms that are problem dependent: the problem encoding and the evaluation function. Consider a parameter optimization problem where we must optimize a set of variables either to maximize some target such as profit, or to minimize cost or some measure of error. We might view such a problem as a black box with a series of control dials representing different parameters; the only output of the black box is a value returned by an evaluation function indicating how well a particular combination of parameter settings solves the optimization problem.

3. Optimization of cutting parameters with GA

Intelligent manufacturing achieves substantial savings in terms of money and time if it integrates an efficient automated process-planning module with other auto-mated systems such as production, transportation , assembly, etc. Process planning involves determination of appropriate machines, tools for machining parts, cutting fluid to reduce the average temperature within the cutting zone and machining parameters under certain cutting conditions for each operation of a given machined part. The machining economics problem consists in determining the process parameter, usually cutting speed, feed rate and depth of cut, in order to optimize an objective function. A number of objective functions by which to measure the optimality of machining conditions include: minimum unit production cost, maxi-mum production rate, maximum profit rate and weighted combination of several objective functions. Several cutting constraints that should be considered in machining economics include: tool-life constraint, cutting force constraint, power, stable cutting region constraint, chip-tool interface temperature constraint, surface finish constraint, and roughing and finishing parameter re lations.

The main objective of the present paper is to determine the optimal machining parameters that minimize the unit production cost without violating any imposed cutting constraints. Consequently, the mathematical formulation of the machining optimization problem is similar to that of Chen and Tsai (Chen at al. 1996) having 20 cutting constraints. A new local search optimization based on genetic algorithm approach is developed to solve the machining optimization model.

The entire development of planning of the machine processes is based on the optimization of the economic criteria by taking the technical and organizational limitations into account. In the cutting operations the economic criteria are the costs and the manufacturing time. The objectives of the described process are: maximization of the production rate, reduction of the costs and improvement of the surface quality.

GA computes score (objective) function for each string of the solution space so that the string that has the maximum score function value is determined. The goal of optimization problems is to minimize some cost function. In GA approach, the cost function being optimized is usually mapped to a score function.

3.1. Production rate

Usually, the production rate is measured as the entire time necessary for the manufacture of a product T_p. It is the function of the metal removal rate *MRR* and of the tool life *T* (Malakooti at al. 1990):

$$T_p = T_s + V \times (1 + T_c / T) / MRR \quad + T_i \tag{1}$$

where T_s, T_c, T_i and V are the tool set-up time, the tool change time, the time during which the tool does not cut and the volume of the removed metal. In some operations the T_s, T_c, T_i and V are constants so that T_p is the function of *MRR* and *T*.

The metal removal rate *MRR*.

MRR be expressed by analytical derivation as the product of the cutting speed, feeding and cutting depth:

$$MRR = 1000 \times v \times f \times a \tag{2}$$

Tool life *T*.

The tool life *T* measured as the average time between the tool changes or tool sharpenings. The relation between the tool life and the parameters is expressed with the well known Taylor's formula:

$$T = k_T / v^{a_1} \times f^{a_2} \times a^{a_3} \tag{3}$$

where the k_T, a_1, a_2 and a_3, which are always positive constant parameters, are determined statistically (Hayes at al. 1979).

3.2. Operation cost

The operation cost can be expressed as the cost per product C_p. In the cost of the operation two values connected with the cutting parameters T, T_p (Malakooti at al. 1990) are distinguished:

$$C_p = T_p \times \left(C_t / T + C_l + C_0 \right) \tag{4}$$

where C_t, C_l and C_0 are the tool cost, the labour cost and the overhead cost respectively. In some operations the C_t, C_l and C_0 are independent of the cutting parameters.

3.3. Cutting quality

The most important criterion for the assessment of the surface quality is roughness calculated according to:

$$R_a = k \times v^{x_1} \times f^{x_2} \times a^{x_3} \tag{5}$$

where x_1, x_2, x_3 and k are the constants relevant to a specific tool-workpiece combination.

3.4. Limitations

There are several factors limiting the cutting parameters. Those factors originate usually from technical specifications and organizational considerations. The following limitations are taken into account.

Due to the limitations on the machine and cutting tool and due to the safety of machining the cutting parameters are limited with the bottom and top permissible limit.

Permissible range of cutting conditions.

$$\begin{aligned} v_{\min} &\le v \le v_{max} \\ f_{\min} &\le f \le f_{max} \\ a_{\min} &\le a \le a_{max} \end{aligned} \tag{6}$$

Implied limitations issuing from the tool characteristics and the machine capacity.

For the selected tool the tool maker specifies the limitations of the cutting conditions. The limitation on the machine is the cutting power and the cutting force. Similarly, the machining characteristics of the workpiece material are determined by physical properties.

3.4. Cutting power and force

The consumption of the power can be expressed as the function of the cutting force and cutting speed:

$$P = \frac{F \times v}{6122.45 \times h} \tag{7}$$

where h is the mechanical efficiency of the machine and F is given by the following formula:

$$F = k_F \times f^{b_2} \times a^{b_3} \tag{8}$$

When the equation 8 is introduced into the equation 7 the following is obtained:

$$P = \frac{k_n \times v}{6122.35 \times h} \times f^{b_2} \times a^{b_3}\text{ ; where: } k_n = \frac{k_F}{(6122.45 \times h\,)} \tag{9}$$

The limitations of the power and cutting force are equal to:

$$\begin{gathered} P(v, f, a) \le P_{max} \\ F(v, f, a) \le F_{max} \end{gathered} \tag{10]}$$

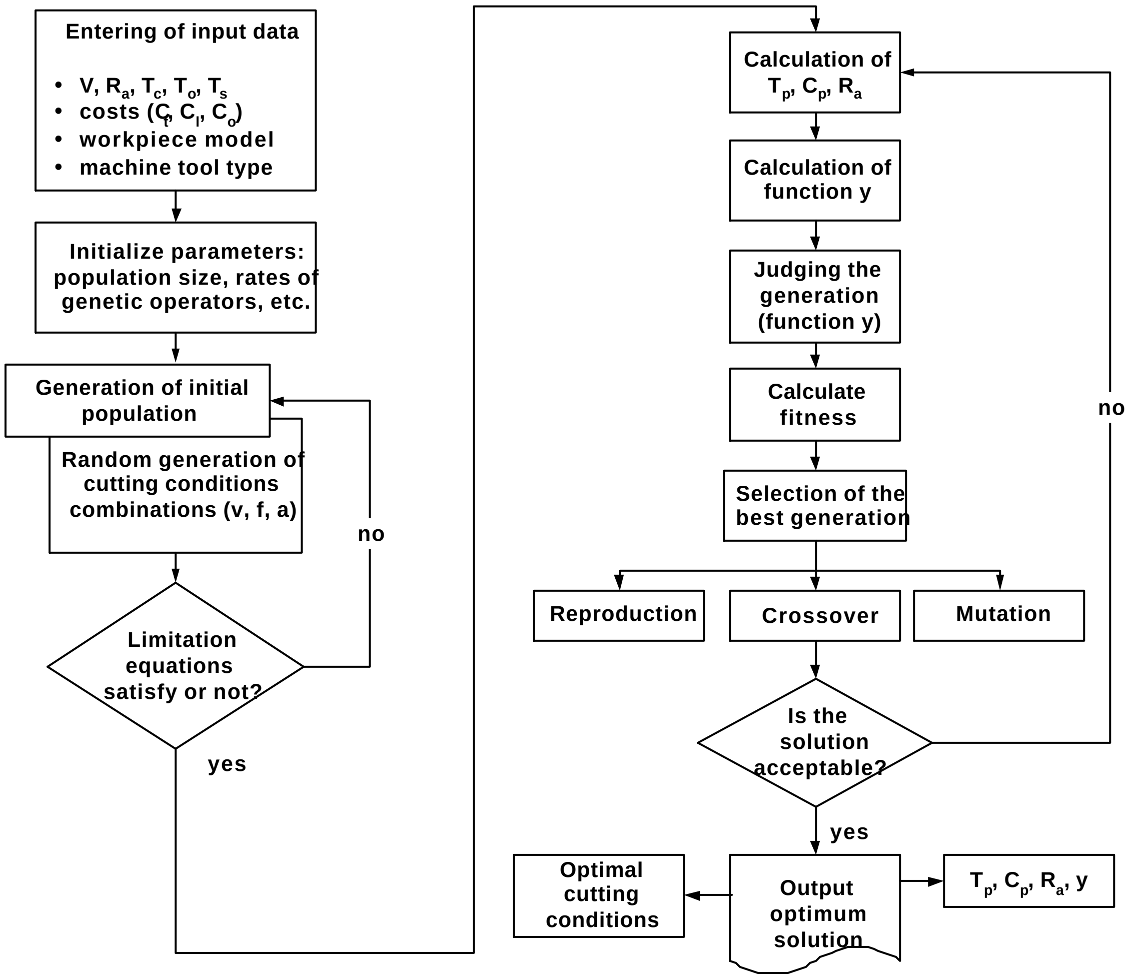
Entering of input data
• V, R_a, T_c, T_o, T_s
• costs (C_t, C_l, C_o)
• workpiece model
• machine tool type
Initialize parameters: population size, rates of genetic operators, etc.
Generation of initial population
Random generation of cutting conditions combinations (v, f, a)
Limitation equations satisfy or not?
no
yes
Calculation of T_p, C_p, R_a
Calculation of function y
Judging the generation (function y)
Calculate fitness
Selection of the best generation
no
Reproduction
Crossover
Mutation
Is the solution acceptable?
yes
Optimal cutting conditions
Output optimum solution
T_p, C_p, R_a, y

Figure 4. Flowchart of the GA solution.

The problem of the optimization of cutting parameters can be formulated as the following multi-objective optimization problem.

$$\begin{gathered} \min Tp\,(v, f, a) \\ \min Cp\,(v, f, a) \\ \min Ra\,(v, f, a) \end{gathered} \tag{11}$$

By the described algorithm it is possible to solve the optimization problems which can be transformed into the expression 12 by which the maximum of the optimization function is determined.

max_x ($f(x)$) = max_x (algorithm); so that the following expressions are satisfied:

$$\begin{gathered} c(x) \le 0, \\ c_{equation}(x) = 0, \\ A \cdot x \le b, \quad A_{equation} \cdot x = b_{equation}, \qquad \text{where: } u_{min}, u_{max}, b \in R \\ u_{min} \le x \le u_{max}; \end{gathered} \tag{12}$$

In the algorithm the genetic algorithm is the optimization function.

4. An illustrative example

On the NC lathe we want to machine a cast steel blank by means of the tool made from HSS. The task is to find optimum cutting conditions for the process of turning. The values of coefficients are statistically determined on the basis of the data measured experimentally (tool life, roughness, time of manufacture, cutting force).

Values of coefficients:

T_s = 0.12 min,	T_c = 0.26 min,	T_i = 0.04 min,	V = 231376 mm^3,
C_t = 13.55 \$,	C_1 = 0.31 \$/min,	C_0 = 0.08 \$/min,	h = 36 %,
k = 1.001,	x_1 = 0.0088,	x_2 = 0.3232,	x_3 =0.3144,
k_T = 1575134.21,	a_1 = 1.70,	a_2 = 1.55,	a_3 = 1.22,
k_F = 1.38,	b_1 = 0,	b_2 = 1.18,	b_3 = 1.26,

The objective functions are as follows:

$$\min \mathrm{Tp} = 0.12 + 231276\ (1 + 0.26/\mathrm{T})/\mathrm{MRR}$$

$$\min \mathrm{Cp} = (13.55/\mathrm{T} + 0.39) \times \mathrm{Tp}$$

$$\min \mathrm{R_a} = 0.0088 \cdot \mathrm{v} + 0.3232 \cdot f + 0.3144 \cdot a$$

$$\mathrm{T} = 1575134.21 \times \mathrm{v}^{-1.70} \times \mathrm{f}^{-1.55} \times \mathrm{a}^{-1.22}$$

$$\mathrm{MRR} = 1000 \times 9.81 \times \mathrm{v} \times \mathrm{f} \times \mathrm{a}$$

The limitation functions are as follows:

$$\mathrm{v_{min}} \le v \le v_{max} \qquad \mathrm{P(v, f, a)} \le P_{max}$$

$$\mathrm{f_{min}} \le f \le f_{max} \qquad \mathrm{F(v, f, a)} \le F_{max}$$

$$\mathrm{a_{min}} \le a \le a_{max}$$

$$\mathrm{F} = 1.38 \times f^{1.18} \times a^{1.26}$$

$$\mathrm{P} = 0.000626 \times v \times f^{0.24} \times a^{0.11}$$

v_{min} = 70 m/min,	v_{max} = 100 m/min
f_{min} = 0.1 mm/rev,	f_{max} =1 mm/rev
a_{min} = 0.1 mm,	a_{max} = 5.0 mm
F_{max} = 230 N,	P_{max} = 5 kW

The cutting conditions are generated at random inside the specified limits. The other values are calculated according to equations 1-5 with selected cutting conditions. The following manufacturer's implicit value function (Malakooti at al. 1990) is selected:

$$z(T_p, C_p, R_a) = 0.42 \cdot e^{(-0.22Tp)} + 0.36 \cdot e^{(-0.32Cp)} + 0.17 \cdot e^{(-0.26Ra)} + \\ + 0.05 / (1 + 1.22 \cdot T_p \cdot C_p \cdot R_a) \tag{13}$$

The columns 2 and 3 of the table 1show the values of the function z, calculated according to equation (13), and the values predicted by the algorithm y. The first half of the data is used for training and the second half for testing of ANN. With any cutting conditions the trained algorithm approximates the function z with satisfactory accuracy, therefore it is used for continuation of the optimization process. Searching for the extreme of the function y, simulated by genetic algorithm, follows. All steps of the process are executed automatically within the time of 1 second.

Table 1 Comparison of results.

TRAINING				TESTING			
No.	$z\ [T_p, C_p, R_a]$	$y\ (T_p, C_p, R_a)$	$z - y$	No.	$z\ [T_p, C_p, R_a]$	$y\ (T_p, C_p, R_a)$	$z - y$
1.	0.7985	0.8008	-0.0023	11.	0.7911	0.7932	-0.0021
2.	0.7872	0.7855	0.0017	12.	0.8817	0.8835	-0.0018
3.	0.7999	0.7975	0.0024	13.	0.8076	0.8060	0.0016
4.	0.8103	0.8103	0.0000	14.	0.7865	0.7867	-0.0002
5.	0.7880	0.7871	0.0009	15.	0.8146	0.8136	0.0010
6.	0.8173	0.8184	-0.0011	16.	0.7849	0.7860	-0.0011
7.	0.8104	0.8099	0.0005	17.	0.8132	0.8174	-0.0042
8.	0.8119	0.8123	-0.0004	18.	0.8192	0.8170	0.0022
9.	0.8069	0.8070	-0.0001	19.	0.7985	0.7966	0.0019
10.	0.8148	0.8146	0.0002	20.	0.8176	0.8187	-0.0011

5. Discussion of results

For the experiment the genetic algorithms were used. The genetic algorithm give more accurate results, but they require more time for training and testing. The programme containing this genetic algorithm is slow. Precision of results is very reliable. The table 2 shows the selected optimum cutting conditions and the corresponding values of variables based on maximization of the implicit function *y* predicted by genetic algorithm. The first line shows the optimal cutting conditions calculated by mathematical tool, whereas the second line shows the cutting conditions determined by genetic algorithm approach. Clearly, the genetic algorithm-based optimization approach provides a sufficiently approximation to the true optimal solution. Figure 3 shows the extreme of the optimization function with relevant optimum cutting conditions.

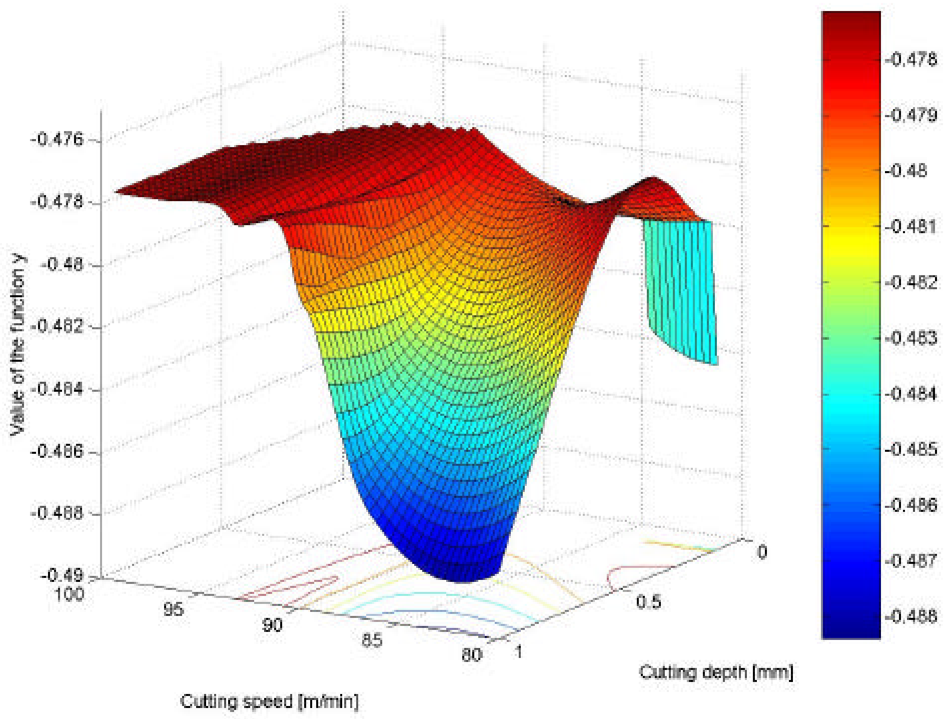

Figure 5. Optimization function *y* with the optimal cutting conditions.

Table 2 Results obtained by genetic algorithm.

Basis	v (m/min)	f (mm/rev)	a (mm)	T_p (min)	C_p ($)	R_a (μm)
z [T_p, C_p, R_a]	86.837	1.8601	4.30	0.459051	0.3114	0.7201
y [T_p, C_p, R_a]	86.8549	1.8622	4.3068	0.4938	0.3233	0.7203

Basis	MRR (mm^3/min)	T (min)	F (N)	P (kW)
z [T_p, C_p, R_a]	777820.74231	42.00	177.507	0.0070
y [T_p, C_p, R_a]	777820.74236	42.81	177.512	0.0071

Advantages of developed algorithm:

- simple complementing of the model by new input parameters without modifying the existing model structure,
- automatic searching for the non-linear connection between the inputs and outputs,
- fast and simple optimizing.

Disadvantages:

- time-consuming determination of training parameters,
- experience is necessary for conceiving of the algorithm,
- repeatability of training is not assured.

6. Conlusion

This paper presents a genetic algorithm optimization approach for solving the machining operations problem with milling. The results obtained from comparing the proposed genetic algorithm optimization approach with those taken from recent literature prove its effectiveness. The results of the proposed approach are compared with results of simulated annealing, fuzzy possibilistic-genetic algorithm, linear-programming approaches. The implication of the encouraging results obtained from the present approach is that such approach can be integrated on-line, with an intelligent manufacturing system for automated process planning. Since the genetic algorithm-based approach can obtain near-optimal solution, it can be used for machining parameter selection of complex machined parts that require many machining constraints. Integration of the proposed approach with an intelligent manufacturing system will lead to reduction in production cost, reduction in production time, flexibility in machining parameter selection, and improvement of product quality. This research definitely indicates some directions for future work. First, is the application of the genetic algorithm-based approach in complex machining systems and automated process planning-system. Second, is comparing the genetic algorithm-based approach with a number of other emerging optimization-techniques.

7. References

Blickle, T., 1995, "comparison of selection schemes used in genetic algorithms", TIK_Report, 11.

Chen, M. C., and Tsai, D. M., 1996, A simulated annealing approach for optimization of multi-pass turning operations", Int. Journal of Production Research, 34, 2803-2825.

DeJong, K., 1975, "An analysis of the behavior of a class of genetic adaptive systems", PhD Dissertation, Dept. of Computer and Communication Sciences, Univ. of Michigan, Ann Arbor. 35.

Goldberg, E. E., 1989, "Genetic Algorithms in Search, Optimization and Machine Learning Reading", (MA: Addison-Wesley).

Hayes, G. M., and Davis, R. P., 1979, "A discrete variable approach to machine parameter optimization", IIE Transactions, 11, 155-159.

Holland, J., 1975, "Adaptation In Natural and Artificial Systems", (University of Michigan Press).

Hsu, V. N., and Daskin, M., and Jones, P. C., and Lowe, T. J., 1995, "Tool selection for optimal part production: a Lagrangian relaxation approach", IIE Transactions, 27, 417–426.

Hui, W. J., and Xi, Y. G., 1996, "Operation mechanism analysis of genetic algorithm", Control Theory and Application, 13(3), 297–303.

Malakooti, B., and Wang, J., and Tandler, E. C., 1990, "A sensor-based accelerated approach for multi-attribute machinability and tool life evaluation", Int. Journal of Production Research, 28, 2373-2392.

Pandey, P. P. C., and Pal, S., 1995, In Proceedings of the 3rd International Conference in Computer Integrated Machining, Singapore, vol. 1, pp. 812–819.