Optimization, genetic algorithm,
cutting parameters, manufacturing,
simulation

Franci CUS
Joze BALIC
Uros ZUPERL

GENETIC ALGORITHM BASED OPTIMISATION OF END MILLING PARAMETERS

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. Operators usually select the machining parameters according to handbooks or their experience, and the selected machining parameters are usually conservative to avoid machining failure. Compared to traditional optimisation methods, a GA is robust, global and may be applied generally without recourse to domain-specific heuristics. 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.

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 [1], Lagrangian relaxation approaches [5], and simulated annealing [3]. 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, that received considerable and

* University of Maribor, Faculty of Mechanical Engineering, Slovenia, E-mail: franc.cus@uni-mb.si

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. [3, 5].

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 and simulation of cutting parameters. 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.the limitation equations and balances the conflicting objectives.

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 [4] and by students of Holland (e.g., DeJong, [4]). 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.

3. THE ITERATION LOOP OF A BASIC GENETIC ALGORITHM

The following flowchart (Fig. 1) 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, [1], pc - number of combinations).

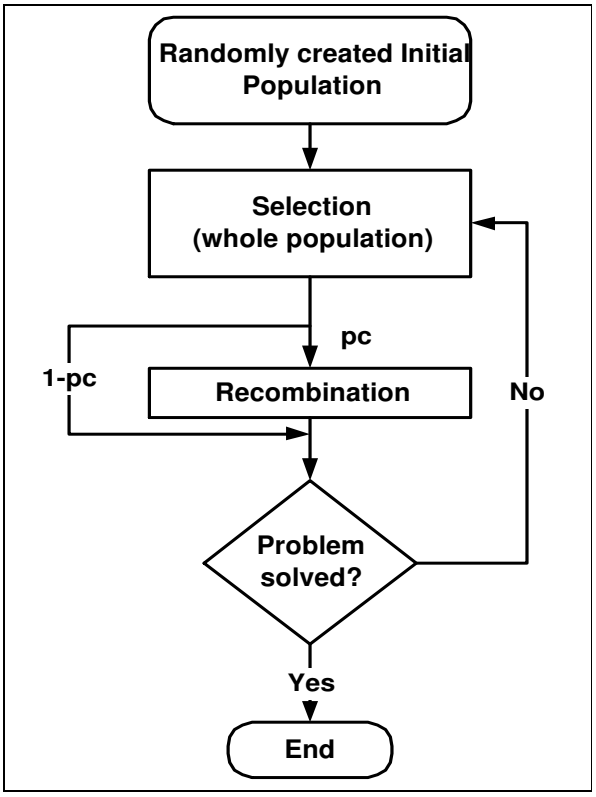

Fig. 1. Flowchart of the basic genetic algorithm

4. 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.

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

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.

Once the mating pool is created, the **crossover** operator in the GA's arsenal comes into play. Remember that 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 (Fig. 2), 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 :

Fig. 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.

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 (Fig. 3). For example, the GA decides to mutate bit position 4 in the string 10000

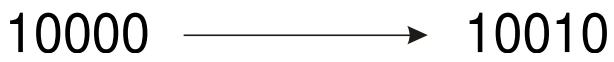

Fig. 3. Example of the mutation operation

The resulting string is 10010 as the fourth bit in the string is flipped. The mutation probability should be kept very low (usually about 0.01%) 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. Mutation helps to maintain that diversity throughout the GA's iterations.

5. 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.

6. 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, maximum 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 relations. 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 Cus [2] 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. In this experiment the following manufacturer's implicit value function [6] is selected:

$$\begin{aligned} y\left(T_p, C_p, R_a\right) = {} & 2{,}3 \cdot e^{(-1.62 Tp)} + 0.016 \cdot e^{(0.62 Cp)} + 0.17 \cdot e^{(-0.8 Ra)} + \\ & + 0.72 / \left(1 + 2.12 \cdot T_p \cdot C_p \cdot R_a\right) \end{aligned} \tag{1}$$

Usually, the **production rate** is measured as the entire time necessary for the manufacture of a product T_p.

$$T_p = T_s + V \times (1 + T_c / T) / MRR \quad + T_i \tag{2}$$

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.

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

$$MRR = 1000 \times v \times f \times a \tag{3}$$

The tool life T measured as the average time between the tool changes or tool sharpenings.

$$T = k_T \times v^{\alpha_1} \times f^{\alpha_2} \times a^{\alpha_3} \tag{4}$$

where the constant parameters, are determined statistically [6].

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 [6] are distinguished:

$$C_p = T_p \times \left(C_t / T + C_1 + C_0\right) \tag{5}$$

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.

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

$$R_a = k \times v^{x_1} \times f^{x_2} \times a^{x_3} \tag{6}$$

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

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.

$$\begin{aligned} v_{min} \le v \le v_{max} \\ f_{min} \le f \le f_{max} \\ a_{min} \le a \le a_{max} \end{aligned} \tag{7}$$

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

$$P = F \times v / (6122.45 \times \eta), \; F = k_F \times f^{\beta_2} \times a^{\beta_3} \tag{8}$$

where K_f is obtained experimentally and η is the mechanical efficiency of the machine.

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

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

7. RESULTS

For the experiment the genetic algorithms were used. The genetic algorithm give more accurate results, but they require more time for calculating the objective function than neural network. The programme containing this genetic algorithm is slow. Precision of results is very reliable. The Table 1 shows the selected optimum cutting conditions and the corresponding values of variables based on maximization of the implicit function obtained by genetic algorithm. The first line shows the optimal cutting conditions determined by mathematical tool, whereas the second line shows the cutting conditions determined by genetic algorithm approach.

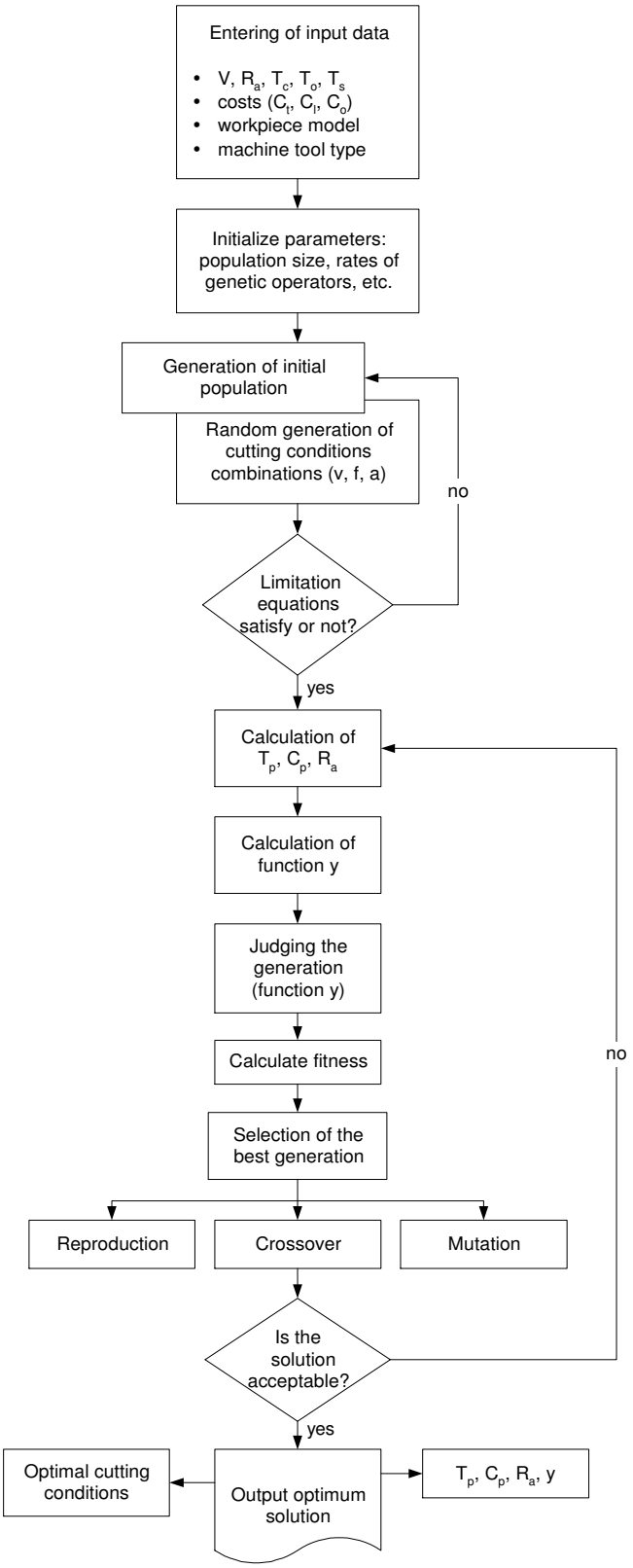

Fig. 4. Flowchart of the GA solution

Clearly, the genetic algorithm-based optimization approach (Fig. 4) provides a sufficiently approximation to the true optimal solution. Fig. 5 shows the extreme of the optimization function with relevant optimum cutting conditions.

Table 3: Results obtained by genetic algorithm

Basis	v (m/min)	f (mm/rev)	a (mm)	T_p (min)	C_p ($)	R_a (μm)
Ideal solut.	86.837	1.8601	0.50	0.459051	0.8	0.7201
GA	86.8549	1.8622	0.5068	0.4938	0.826	0.7203

Basis	*MRR* (mm^3/min)	T (min)	F (N)	P (kW)
Ideal solut.	777820.74231	32.00	177.507	0.70
GA	777820.74236	32.81	177.512	0.71

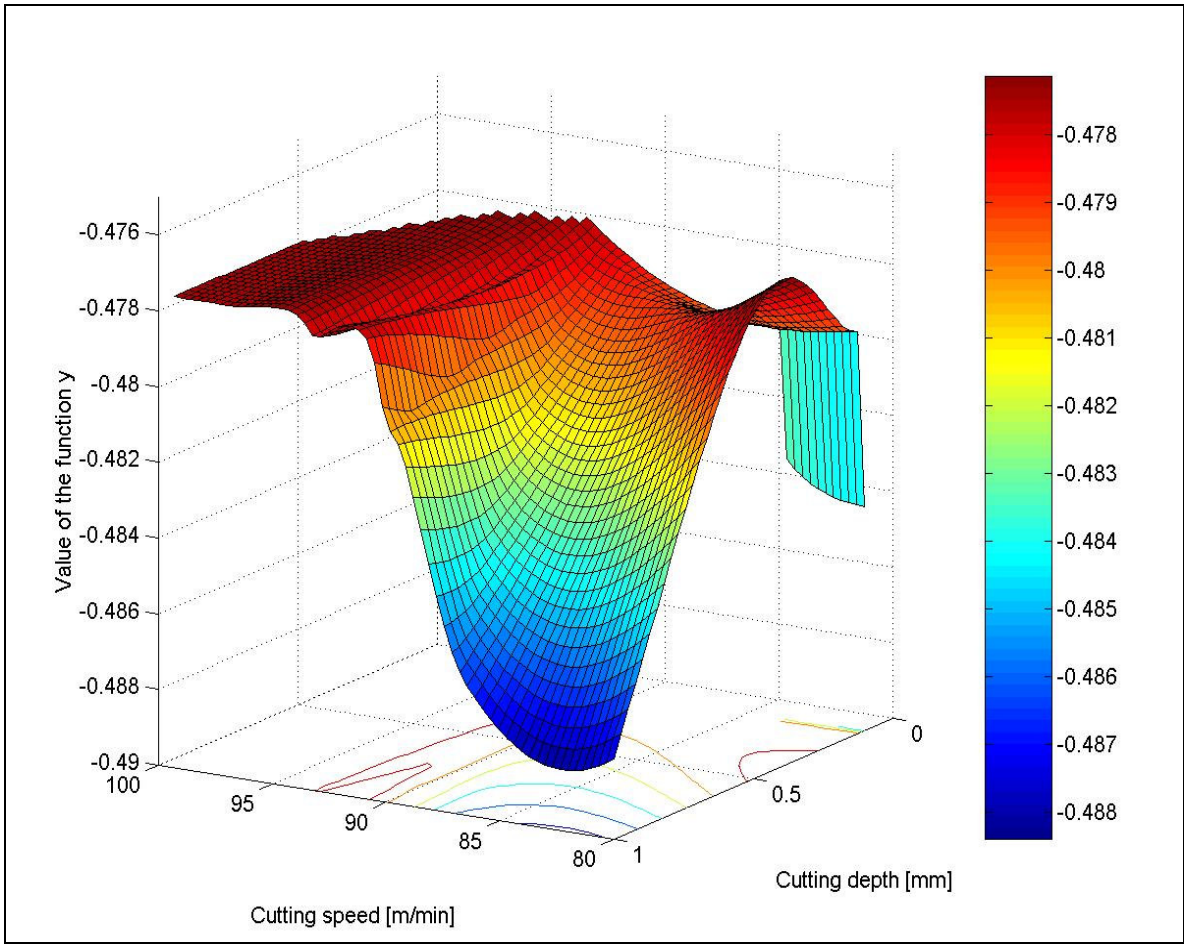

Fig. 5. Optimization function y with the optimal cutting conditions

Advantages of developed GA 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.

8. CONCLUSION

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.

REFERENCES

[1] BLICKLE T., *A Comparison of Selection Schemes used in Genetic algorithms*, TIK_Report, Vol. 11, 1995.

[2] CUS F., *Selection of cutting conditions and tool flow in flexible manufacturing system*, The international journal for manufacturing science & technology, 2000, Vol. 2, pp. 101-106.

[3] GOLDBERG E. E., *Genetic Algorithm in Searching, Optimization, and Machine Learning*, Addison-Wesley, 1989.

[4] HOLLAND J., *Adaptation In Natural and Artificial Systems*, University of Michigan Press, 1975.

[5] HUI W. J., Xi Y. G. *Operation mechanism analysis of genetic algorithm*, Control Theory and Application, Vol. 13(3), pp. 297–303, 1996.

[6] KARRI V., *Performance in Oblique Cutting using Conventional Methods and Neural Networks*, Neural Computing & Applications, 1999; Vol. 8, pp. 196-205.

[7] LEE B.Y., TARNG Y.S., *Cutting-parameter selection for maximizing production rate or minimizing production cost in multistage milling operations*, Journal of Materials Processing Technology, 2000; Vol. 7, No. 105, pp. 61-66.