ISSN 1330-3651 (Print), ISSN 1848-6339 (Online)

https://doi.org/10.17559/TV-20211012084632
Original scientific paper

An Integrated Intelligent CAD/CAPP Platform: Part II - Operation Sequencing Based on Genetic Algorithm

Stevo BOROJEVIĆ*, Dragan MATIĆ, Miroslav DRAGIĆ

Abstract: We present a platform for integrated CAD/CAPP part design based on Elementary Machining Features (EMF) and intelligent approach for setup planning and operation sequencing based on a genetic algorithm through two papers. In this paper, as Part II of this platform, CAD/CAPP integration was realized via information from the enriched EMF, as well as production rules and a genetic algorithm. This is done for the purpose of the automated machining operation sequencing. Operation sequencing was conducted by using the improved genetic algorithm (GA).The improved GA uses integer representation for operations and implements modified genetic operators, enabling the achievement of high results in a reasonable computational time. In the paper we present a comprehensive case study applied to some existing and one new industrial example, confirming a high level of usability of the proposed GA and overall platform. Experimental results show that the improved GA algorithm gives slightly better results than similar algorithms in literature. For industrial example, we use body of the hydraulics cylinder which consists of 52 EMF. After implementation of the proposed methodology, the optimal machining operation sequence was identified, as well as the total machining cost of 142.49 BAM.

Keywords: elementary machining features; genetic algorithm; operation sequencing

1 INTRODUCTION

Today, artificial intelligence is applied in almost all fields of human activities. The ranges of artificial intelligence in the fields of robotics, machine vision, medicine, software engineering, automotive and transport industry and many other fields are well known [1-6].

Integration of CAD/CAPP (CAD - Computer Aided Design, CAPP - Computer Aided Process Planning) activities today cannot be imagined without the application of various forms of artificial intelligence (expert systems, artificial neural networks, fuzzy logic, algorithms based on heuristics, agent-based technologies, etc.). The reason for the presence of artificial intelligence in this area is the domain of knowledge used in the field of CAPP. Knowledge in the field of CAPP is expert-type knowledge (i.e., experience-based knowledge), which is difficult to describe algorithmically and execute its precisely mathematical modelling. Precisely, such knowledge is suitable for presentation using some of the methods of artificial intelligence.

Part II of this paper presents the integrated intelligent CAD/CAPP platform for defining operation sequencing based on GA. The concept of this approach can be resumed as generation of all necessary information (geometrical and technological) from enriched EMF (EMF - Elementary Machining Features), developed mechanism for generation of EMO (EMO - Elementary Machining Operations) for each EMF, and transferring this information to GA mechanism for determination of the operation sequencing. Previously mentioned generation of all information, ready for implementation in GA, is described in detail in Part I of this paper.

A review of the literature is given in Chapter 2 of this paper. A detailed description of the improved GA is provided in Chapter 3. The improved GA, together with its operators, was developed for the purpose of determining operation sequencing based on the minimum function of machining cost. Thereafter, the improved GA was tested on existing examples from the literature. The test results of the improved GA show encouraging results in relation to the literature, and qualify the improved GA as a reliable mechanism for determining operation sequencing. Within Chapter 4, a determination of the operation sequencing was performed on an industrial example of a body of the hydraulic cylinder for press brake. Chapter 5 of this paper draws conclusions regarding the overall development of an integrated intelligent CAD/CAPP platform.

2 LITERATURE REVIEW

CAD/CAPP integration in this paper was implemented by applying the hybrid technology based on the use of elementary machining forms and genetic algorithms as a heuristic method in the context of artificial intelligence. In this literature review we present achievements in the field of the implementation of the heuristic algorithm mainly in solving tasks regarding process planning design and operation sequencing.

The problem of identification and optimization of machining operation sequencing within the CAD/CAPP integration belongs to the class of NP hard problems, so it cannot be optimally solved by exact algorithms in a reasonable time in a general case [7]. Specially, exact algorithms fail to optimally solve large-scale problem instances which most often occur in practice. The reason for the hardness of the problem lies in the large amount of empirical data that are difficult to formalize, as it was not easily possible to cover all the factors which affect the implementation of the activities of machining operation sequencing. For these reasons, the implementation of heuristic algorithms has found its full application in the CAD/CAPP integration. Heuristic algorithms that were commonly used in the field of CAD/CAPP integration include: Genetic Algorithms (GA), algorithms based on Simulated Annealing (SA), TABU search algorithms, Ant Colony Optimization (ACO) algorithms and other hybrid algorithms. Some achievements in this area are presented below. Research results by Reddy et al. [8] can be classified among the first of this kind in the field of determining the optimal or nearly optimal operation sequencing as part of CAPP activities. The authors have applied genetic algorithms in order to quickly identify the optimal or nearly optimal operation sequence. The genetic algorithm was implemented as a general technique for searching of operation sequence in a dynamic production environment. Qiao et al. [9] used a GA-based approach to determine the sequence of machining operation for prismatic parts. For the purpose of generating alternative operation sequence, the calculation of objective function was based on four types of process planning rules, namely: preceding rules, grouping rules, rules of neighbouring operation sequencing and optimization rules. The context of research by Lee et al. [10] contains six algorithms for the local-heuristic search methodology in the operation sequencing space. The algorithms were based on techniques of simulation annealing (SA) and TABU search. The results of randomly generated problems show, at the total average, that algorithms based on TABU search generate better results than the algorithm based on SA. Li et al. [11] modelled computer-aided process planning as a constrained optimization problem. In order to solve this problem, they proposed a hybrid approach based on GA and SA. Criteria evaluation or objective function was the total production cost of a part, which consists of the machine cost, tool cost, machine change costs, tool change cost and setup change cost. The GA was implemented in the first phase in order to generate the initial and satisfactory operation sequence. In the second phase, the SA algorithm was applied for the purpose of identifying the optimal or near-optimal operation sequence. The hybrid approach for optimization of process planning activities, based on the integration of GA and SA, was also described by Ma et al. [12] and Falih et al. [13]. One of the most outstanding papers, in the field of process planning design, which strongly focuses on the integration of setup planning and operation sequencing, was presented in [14]. The authors in this paper suggest that the key processes in CAPP systems, such as the selection of resources, setup planning and operation sequence, must be observed at the same time in order to generate a globally optimal solution. Framework of this paper includes modelling and integration of these processes in the form of optimization problems based on constraints. In order to solve this issue effectively, the authors proposed an approach based on the TABU search. Li et al. [15] describe a flexible optimization of process planning activities by using genetic algorithms. The authors describe three levels of flexibility, namely the level of operations, operation sequencing and setup changes. For the purpose of the implementation of GA, the flexibility of the process planning activities was described in a form of the network flexibility which is transformed into a tree form. Salehi et al. [16] present an approach in which process planning activities were divided into preliminary and detailed design phases. The analysis of constraints and generating of feasible process plans were carried out in the preliminary design phase, while the optimization of the process plan was carried out in the context of the detailed design phase. The techniques based on genetic algorithms were applied during the implementation of a preliminary as well as a detailed design phase of process planning activities. Kafashi [17] proposes a methodology for simultaneous optimization of setup planning and operation sequencing. The proposed methodology for optimization was based on constrains analysis, such as the tool approach directions, feature precedence relationship and tolerance relations between the machining features. For the purpose of generating all possible setup plans and operation sequences, the analysis of constraints was conducted through the implementation of resources database. In [18, 19] Ant Colony Optimization was used for finding the process plans. The process planning problem was mapped to a weighted graph consisting of vertices and edges. The optimization strategy includes directing of the ant colony through necessary nodes on the graph to reach solutions.

3 AN INTEGRATED APPROACH FOR OPTIMIZING THE OPERATION SEQUENCING BASED ON GA

The structure of the developed platform for integrated CAD/CAPP design based on EMF is shown in Fig. 1. Detailed description of this platform was made in Part I of this paper. Here we will pay attention at the part of the platform which was connected with operation sequencing based on improved GA.

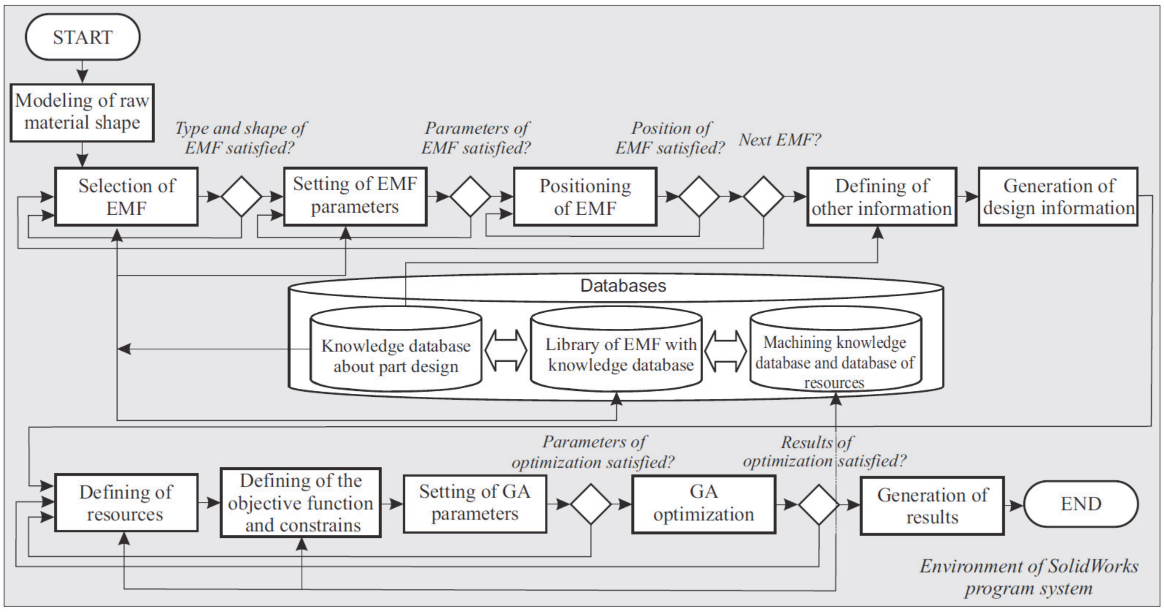
START
Modeling of raw material shape
Type and shape of EMF satisfied?
Parameters of EMF satisfied?
Position of EMF satisfied?
Next EMF?
Selection of EMF
Setting of EMF parameters
Positioning of EMF
Defining of other information
Generation of design information
Databases
Knowledge database about part design
Library of EMF with knowledge database
Machining knowledge database and database of resources
Parameters of optimization satisfied?
Results of optimization satisfied?
Defining of resources
Defining of the objective function and constrains
Setting of GA parameters
GA optimization
Generation of results
END
Environment of SolidWorks program system

Figure 1 Structure of the integrated intelligent CAD/CAPP platform based on EMF and GE

After identification of operations via elementary machining forms, the next stage is to organize the setup process and operations sequencing. Process planning is usually defined as a combinatorial optimization problem with constraints and includes two main tasks: determining the setup process planning and operation sequencing. The setup process planning includes the identification of the machine, tool and the tool approach direction (TAD) for each operation. The task of operation sequencing includes the determination of the sequence of operations required to produce a part, considering the dependence between operations.

Since the whole problem is NP hard, there is no exact algorithm which can solve it in a reasonable time. Therefore, to identify the sequence of operations and setup process plans in the best possible way, non-deterministic algorithms were used, especially in cases of large data inputs. As it was already mentioned in the introductory sections, there are many different approaches for solving the problem, mostly based on the heuristics.

In this work, we developed an improved genetic algorithm which optimizes the process plan for a design part, in the sense of practical and economic aspects. Two types of constraints were usually involved: precedent constraints and resource constraints.

In almost all situations, there exists a dependence between operations: some operations cannot be performed before some others were finished. For example, drilling the hole must be performed before the reaming. Also, process planners sometimes decide to perform some type of operations before others, e.g., they perform all roughing operations before finishing or localize operations, considering accessibility aspects and so on. Another aspect of restricting the operation schedule can be based on clustering of the operations: two operations should be performed together, i.e., one after another, satisfying the constraints related to perpendicularity, angularity, concentricity, circular run-out, total run-out, symmetry and similar. The precedent constraints were usually represented by a 0-1 matrix: the element (*i*; *j*) is equal to 1 if the operation *i* must be performed before the operation *j*, otherwise is equal to 0. Another approach is to represent the constraints as a directed graph: from the node *i* there is an arc to the node *j* if the operation *i* must be performed before the operation *j*.

The second type of constraints was related to determining the lists of available machines, tools and TADs for each operation. In practice, a typical production environment comprises the usage of many different machines and tools. Also, each operation can be performed from different directions. In this paper, we consider the 3-axis machine centre and therefore there are six different TADs (+*x*; −*x*; +*y*; −*y*; +*z*; −*z*).

Therefore, the input data for the optimization algorithm were identified by combining the data obtained in the design phase: the list of operations and TADs for each EMF and data manually added by the designer, i.e., the list of machines and tools for each operation and the precedent constraints.

Machine and tool costs for each machine and tool, as well as machine, tool and setup change costs, were also identified before the algorithm was executed. The calculation of the total costs was described in the next subsection. After the GA was executed, the operation sequence, exact machine, tool and TAD for each operation and the total production costs were identified.

3.1 Calculating the Objective Function

The objective function of the minimization problem was defined as minimizing the total production costs. A standard approach mainly used in literature takes into account time required to complete the production process, combined with the machines and tools utilization costs.

In order to determine the final formula for calculating the total costs, the following indices are introduced: machine cost change index (MCCI), tool cost change index (TCCI), set-up change cost index (SCI). Calculation of the machining costs was based on the utilization costs for the machines and tools for each operation. For each machine *i*, the machine cost index MCI_i and for each tool *j*, the tool cost index TCI_j were introduced.

Let the list of operations be a feasible sequence of the operations and let *nOps* be to total number of operations. Let the *op:machine*, *op:tool* and *op:tad* for each operation be the assigned machine, tool and tool approach direction. In the following formulas, two auxiliary functions will be used:

$$\Omega_1(X,Y)=\begin{cases}1, X\neq Y\\0, X=Y\end{cases} \tag{1}$$

and

$$\Omega_2(X,Y)=\begin{cases}0, X=Y=0\\1, \text{otherwise}\end{cases} \tag{2}$$

Calculation of the final formula is based on the following five factors: total machine cost (*MC*), total tool cost (*TC*), total machine change costs (*MCC*), total tool change costs (*TCC*) and total setup change costs (*SCC*). A similar way is also presented in literature, for example, in [20]. The calculation is as follows:

1. Total machine cost (*MC*)

$$MC=\sum_{i=1}^{nOps} operations[i].machine\cdot MCI_i \tag{3}$$

2. Total tool costs (*TC*)

$$TC=\sum_{i=1}^{nOps} operations[i].tool\cdot TCI_i \tag{4}$$

3. Total machine change costs. We first calculate the total number of changes of the machines.

$$NMC=\sum_{i=1}^{nOps-1}\Omega_1\begin{pmatrix}operations[i].machine,\\ operations[i+1].machine\end{pmatrix} \tag{5}$$

Total machine change costs are now calculated as:

$$MCC=NMC\cdot MCCI \tag{6}$$

4. Total tool change costs. Tool change cost for two consecutive operations was taken into account if:
(a) different tools were used on the same machine;
(b) different tools were used on different machines;
(c) same tools were used on different machines.

By using the auxiliary function Ω_1 , introduced by the Eq. (1), we firstly define the total number of tool changes:

$$NTC = \sum_{i=1}^{nOps-1} \Omega_2 \left(\begin{array}{l} \Omega_1 \left(\begin{array}{l} operations[i].machine, \\ operations[i+1].machine \end{array} \right), \\ \Omega_1 \left(\begin{array}{l} operations[i].tool, \\ operations[i+1].tool \end{array} \right) \end{array} \right) \tag{7}$$

Then, total tool change costs were calculated as:

$$TCC = NTC \cdot TCCI \tag{8}$$

5. Total setup change costs. Similarly, like counting the number of tool changes, we identify the cases of two consecutive operations, when setup change was counted:
a. different TADs were used on the same machine;
b. different TADs were used on different machines;
c. same TADs were used on different machines.

We firstly define the total number of TAD changes:

$$NSC = \sum_{i=1}^{nOps-1} \Omega_2 \left(\begin{array}{l} \Omega_1 \left(\begin{array}{l} operations[i].machine, \\ operations[i+1].machine \end{array} \right), \\ \Omega_1 \left(\begin{array}{l} operations[i].tad, \\ operations[i+1].tad \end{array} \right) \end{array} \right) \tag{9}$$

and the total TAD change cost were then calculated as:

$$SCC = NSC \cdot SCCI \tag{10}$$

Finally, the total costs were calculated as the sum of these five types of cost:

$$TC = MC + TC + MCC + TCC + SCC \tag{11}$$

3.2 Description of the Genetic Algorithm

The genetic algorithm was a metaheuristic method based on the population which simulates some aspects of natural evolution. Each element of the population is an individua and represents a potential solution to the problem. The concept of the GA has been introduced by Holland [21]. In the last three decades a large number of problems have been successfully solved by various GA variants and implementations. The GA was a population-based optimization technique. The main principles of this method include the continuous improvement of the population quality, with the hope that the best individual will achieve the optimal value at the end of the optimization process.

More precisely, the GA is an iterative process which applies genetic operators: crossover, mutation and selection on the individuals in each iteration (also called generation). The quality of each individual was determined by a numerical value called fitness, which indicates the ability of the individual to survive to the next generation. Generally, better fitness indicates greater probability that the individual will survive. The function which maps the searching space to the fitness values is called a fitness function. The fitness function is in a strong relation with the objective function of the optimization problem, i.e., the individual with better fitness represents a solution with better objective function.

The basic objects in the GA are population and single individuals. The population consists of individuals and the number of them is usually several tens, up to several hundreds. Each individual is relating to one solution of the problem. Individuals are represented by a genetic code which is usually written by binary, integral or some other finite alphabet. Genes are usually represented in a way which allows easier implementation of genetic operators. In order to ensure the diversity of the genetic material, the starting population is usually chosen randomly. Also, there are other approaches, like usage of some other heuristics for generating the starting solutions, usually with the intention to determine some better starting solutions. General pseudo-code for the proposed GA algorithm for the purpose of optimization of operation sequence is presented in the following Tab. 1.

Table 1 General pseudo-code for proposed GA algorithm

```
1: Input_Data_Generating();
2: Generating_of_Initial_Population();
3: while not Stopping_Criteria_GA()do
4:          for i=1 to Npop do
5:          obj[i] = Goal_Function(i);
6:          end for
7: Fitness Function();
8:        Selection();
9:        Crossover();
10:       Mutation();
11:        end while
12: Print_Output_Data();
```

In the optimization process, the application of genetic operators was iteratively applied to the individuals, until the stopping criteria were satisfied. It was usually based on the limitation of the overall execution time, number of generations, or their combination. The following subsections contain more detailed description of the proposed GA.

3.3 Data Representation and Population Initialization

Each machining operation was represented by the following elements: *operation ID,* lists of the allowed machines, tools and TADs, and also with three parameters used for storing information to which machine, tool and TAD were assigned later to this operation. All operations were collected in the list, called *operations.*

In the entire population, all individuals were collected in a list called *individuals*. Total number of individuals represent the size of population.

Each individual represents a potential solution of the GA. It was represented by the list of operations and by an integer array called *code* of the length *nOps*, where *nOps* is the total number of operations. In fact, the array *code* contains a permutation of numbers 1, 2, …, *nOps*, indicating the ordering of performing the operations: each element of the array represents one operation and contains the *operation ID*. For each $i, j = 1, 2, \ldots, nOps$, if $i < j$ then the operation with the ID *code*[*i*] is going to be performed before the operations with the ID *code*[*j*]. Data representation of this process was graphically illustrated in Fig. 2.

The initial population was created as follows. For each individual, two steps were performed:

1. For each operation, the machine, tool and TAD were randomly assigned from the lists of allowed values. After the assignment of the machine, tool and TAD, the operation was added to the list operations.

2. The order of operations was determined in a way which ensures the correctness of each individual. The correctness of the individuals was provided by the following procedure:

Based on the problem inputs, for each operation *op*, the list of preceding operations which were to be performed before *op* were executed. At the beginning of the procedure, two empty auxiliary lists were also formed: The list *candidates*, used for storing the operations which can be coded without the violation of the precedent constraint and the list *coded*, needed for storing already coded operations.

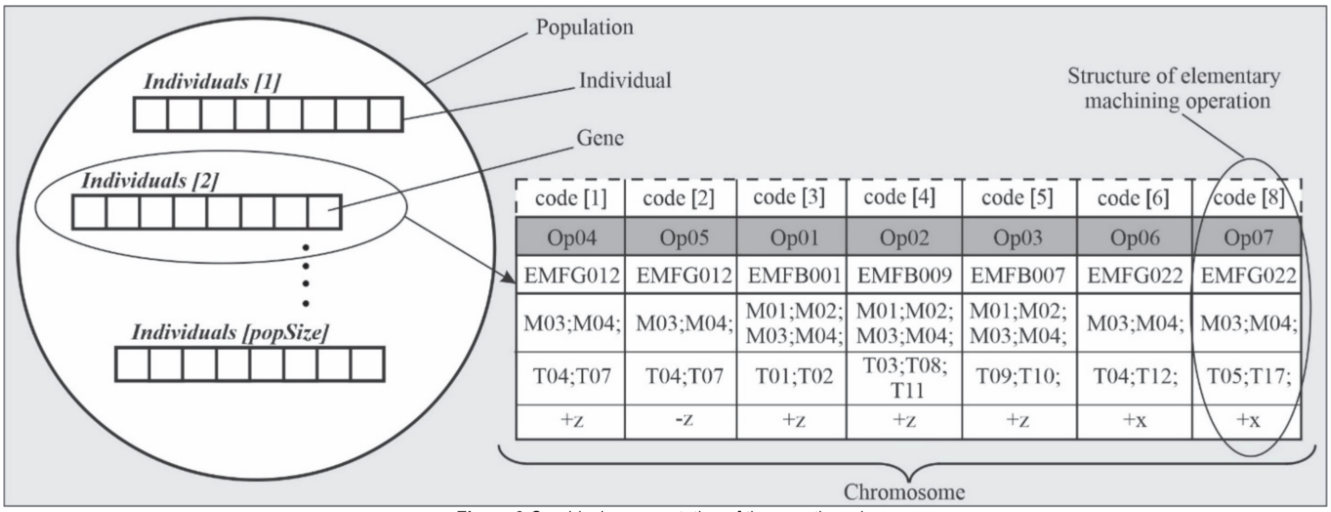

Figure 2 Graphical representation of the genetic code

In the *k*-th, $k = 1, 2, \ldots, nOps$ iteration, for each non-coded operation the procedure looks up the list of its precedent operations. If that list was empty, the operation was added to the list of candidates, becoming a candidate for coding.

After the list *candidates* was updated, an operation from that list is randomly chosen and coded in the individual's array *code*, at the *k*-th place. That operation was moved from the list *candidates* to the list *coded*, in order to exclude it in further steps. For the rest of non-coded operations, the lists of precedent operations were updated and the iteration process continues with next *k*. The procedure finishes when all operations were coded. Once an individual was created, the fitness function, described in subsection 5.4, was called.

When an individual was created, the initialization process continues with the next one, until the entire initial population was created.

3.4 Calculating the Fitness Function

The fitness function was used to determine the criterion for comparison of the quality of individuals. In the minimizing optimization problems, the fitness function was sometimes set to be inversely proportional to the objective function, in order to achieve the effect of increasing the fitness of the individuals during the optimization process. The other option was to leave the direct proportion between the objective function and fitness. In this case, which was also the case in our approach, the objective function (11) was directly employed to calculate the fitness function, so we can get the effect that the individual was considered "better" if its fitness gets lower value. Therefore, for each individual the fitness was defined by the Eq. (12):

$$individual.fitness = TC\left[individual\right] \tag{12}$$

where *TC*[*individual*] was the value of the objective function applied to individual.

3.5 Genetic Operators

3.5.1 Crossover

The crossover operator was applied to two individuals called parents. After selecting a pair of parents, the crossover operator was applied, producing two offsprings. In the proposed GA, one-point crossover operator is used: a random point, i.e., number from the interval [0, *nOps*] was randomly chosen. The chosen point splits the genetic code of each parent into two parts: left and right. According to the existence of preceding constraints, it was obvious that simple swapping of left and right parts of the two parent individuals can produce infeasible offspring. To avoid this, an order crossover operator (OX) [22] was applied, which preserves the local precedence on both offsprings. The left part of the first new offspring was formed by the left part of the genetic code of the first parent. The ordering of the remaining operations was based on the ordering of these operations in the genetic code of the second parent. The second offspring was formed similarly, where the roles of the first and the second parent were exchanged. This operator was also used in many other GA variants in the literature, like in [20, 23, 24].

3.5.2 Mutation

A typical role of a mutation operator was to restore the lost or unexplored genetic material in the population, which can avoid premature convergence into local suboptimal solutions. Mutation includes small changes of the genetic code: usually modification of a single gene, with some predefined small probability, called mutation rate. In most cases, mutation rate was not increased 10%, but there were some GA implementations which allow different mutation rates along the genetic code. The proposed i.e., improved GA uses two types of mutations.

(a) Mutation 1. Swapping two operations. This operator randomly chooses two genes (operations) and exchanges their positions in the genetic code. It is obvious that this approach can produce infeasible solutions. In general, there are four options for handling the infeasible solutions: preventing their appearance, discarding infeasible solutions, the usage of the penalty function and repairing the feasible solutions [25-27]. While in [20] the latter approach, i.e. repairing the infeasible solutions was applied, in the GA proposed in this paper we decided to use another approach, i.e. to discard the infeasible solution. More precisely, after the operator Mutation 1 was applied, a special procedure called *checkFeasibility* was used. If the procedure finds out that the mutated individual was infeasible, it was completely discarded and replaced by a newly created individual. This approach has its advantages and disadvantages.

The main advantage was speeding up the overall mutation operator, since the procedure of correcting the individuals was a time-consuming operation. A disadvantage was reflected in the fact that good genetic material, held by an (even infeasible) individual, may be lost. This situation can especially happen in late phases of the optimization process. On the other hand, correcting the individual based on, for example, the principle presented in [20] can significantly change the overall genetic code, so it was disputable if a relatively good, but infeasible individual will still remain good after the repairing procedure. Another approach of discarding the individual is also implemented in [23], i.e., the fitness of the infeasible individual was set to a very large value, avoiding the individual to survive to the next generation.

(b) Mutation 2, which consists of three segments: changing of the machine, tool and TAD for some operations. In practical considerations, these changes can have significant influence for decreasing the total production costs. Changing the machine was based on the following: for the given operation, let the previously assigned machine be m_i. With some mutation rate, another machine m_j from the list of available machines was assigned. Further, with some probability p_{mut2}, the same change was performed for all operations which satisfy the following conditions:

1. currently set machine is m_i
2. machine m_j was available in the list.

Comparing to the similar mutation operator presented in [20], which is ultimately performed if the conditions (1) and (2) were satisfied, our implementation considers additional aspect, i.e. changing the machine for other operations is not performed ultimately, but with some probability factor p_{mut2}. This enables more probabilistic nature of the operator, in contrast to fully deterministic approach used in [20]. Although this approach was more customizable, experimental results still indicate that better solutions were achieved when the probability factor p_{mut2} was closer to 1. The same principles were applied to the other two segments of the Mutation 2, for changing the tool and TAD.

3.5.3 Selection

Selection operator was based on a modified tournament selection, where the size of tournament was two, i.e., two individuals were compared in each tournament. The best individual can be selected into the next generation in two ways: automatically, if it was considered as an elite one, or by "winning" a tournament, since its fitness was better than that of the others. Elite individuals were those which automatically survive to the next generation. In our approach, the best individual was not considered as elite one and therefore it was not automatically selected to the next generation. This strategy helps to preserve the diversity of genetic material, which keeps the algorithm away from the local optimum. Although it gives a (small) possibility that the best individual was not selected to the next generation, experimental results indicate that the optimization process was not disrupted, and the algorithm easily reconstructs that individual in a few following generations.

3.6 Experimental Results

This section contains the experimental results performed to test the efficiency of the proposed GA. The computational experiments were carried out on a PC based on Intel i7 processor @3.4 GHz, and with 8 MB memory on Windows 7 operating system. The algorithm was implemented in C++ programming language. For each execution, only one thread/processor was used.

The population size was $N_{\text{pop}} = 90$. In each generation 30 crossovers were performed. The experiments were conducted with various values for the mutation rates, chosen from the interval [0.3, 0.7]. Although it seems that mutation rates were rather high and can cause the divergence of the optimization process, in the experiments we show that the values of the mutation rates equal or close to 0.5 give good quality solutions.

For the experiments performed on smaller problem instances, the number of generations was set to 300. For each problem instance the algorithm was run 10 times, with different random seeds.

In the experiments two parts commonly used in the literature were tested. The first prismatic part (Part 1) used in an early work [25] consists of 14 features and machining operations ($nOps$ = 14). Detailed information about the resources and hard and soft precedence constraints can be found in the literature, for example in [14]. The second prismatic part (Part 2), used in [11] contains more complex features and constraints. It consists of 14 features and 20 machining operations ($nOps$ = 20). The relevant information about this part can also be found for example in [11, 14]. In the literature, three different conditions were tested under the Parts 1 and 2:

(a) All machines and tools were available. The costs were calculated according to Eq. (11).

(b) All machines and tools were available. Tool costs and tool changing costs were set to 0, i.e., total costs were calculated only on machines costs, machine changes costs and TAD changes costs.

(c) Applied only to the Part 2. Machine m_2 and the tool *T*7 were down and the total costs were calculated the same as in the Condition (b).

In order to make the comparison of the proposed GA to other methods as fair as possible, we adopted our improved GA and tested it on all three conditions. In Tab. 2 we show the results obtained for the two parts. In the first three columns the label of the part and the condition, the number of features and the number of operations were given. The next two columns contain the best GA value of the objective function among all executions (column GA_{best}) and the average value (column GA_{avg}). The last two columns contain the information about the number of generations (column gen) and the average execution time in seconds (column time / s). For the Part 1, we report the solutions obtained by the improved GA satisfying only hard precedent constraints described in [14].

Table 2 Results obtained on two sample test parts under various conditions

1	2	3	4	5	6	7	8
Part	Condition	#Fetaures	#Operations	GA_{best}	GA_{avg}	gen	time / s
Part 1	con (a)	14	14	1128	1137	300	7.8801
	con (b)	14	14	970	982	300	8.1909
Part 2	con (a)	14	20	2404	2406.5	300	10.9378
	con (b)	14	20	1960	1960	300	12.4048
	con (c)	14	20	2500	2500	300	12.4481

From Tab. 2, one can conclude that the proposed GA successfully finds quality solutions for two sample parts in a relatively short time. The difference between the best and the average values is rather small which indicates high reliability of the proposed method. Even more, for the part 2, conditions (b) and (c), the improved GA achieved the best value (1960 and 2500) in all of 10 runs.

In order to further demonstrate the performances of our algorithm, we compare the results against other heuristic methods from the literature, applied to the Part 1 and the Part 2. In Tabs. 2 and 3 we show the results of these comparisons. The first column contains information on which condition was considered, also indicating which of three values was considered in the row: Mean, Maximum or Minimum. In Tab. 3, the results obtained by using the following algorithms were presented: column IACO: improved ant colony optimization [19], column ACO: ant colony optimization [18], column TS: tabu search [14], SA: simulated annealing [11] and column GA: genetic algorithm [11]. The last column GA [this] contains results obtained by the improved GA proposed in this paper.

Table 3 Comparative results obtained on the Part 1

1	2	3	4	5	6	7	8
Condition		IACO [19]	ACO [18]	TS [14]	SA [11]	GA [11]	GA [this]
(a)							
	Mean	1129*	1329.5	1342	1373.5	1611	1337
	Maximum	1137*	1343	1378	1518	1778	1343
	Minimum	1128*	1328	1328	1328	1478	1328
(b)							
	Mean	-	1170	1194	1217	1482	1182
	Maximum	-	1170	1290	1345	1650	1190
	Minimum	-	1170	1170	1170	1410	1170

In order to make comparison as fair as possible for the part 1, we considered both hard and soft precedent constraints. In [14], due to the thin-wall and material removal interactions, two pairs of soft constraints appear to be opposite one to another. Therefore, each feasible solution was additionally penalized by the value 200. We calculated these penalty values for our solutions in Tab. 3.

In [19], the authors presented a wider set of results obtained for the Part 1 by the improved ant colony algorithm, depending on various values of the algorithm parameters. In Tab. 3, we report their best result found in Tab. 7 from [19], for the weighting parameter $\alpha = 1$ and the heuristic information $\beta = 1$. However, it was not reported if two pairs of opposite soft constraints were considered, so we cannot be sure if the result from [19] was obtained under the same conditions as in other papers. Therefore, we mark their result by the asterisk mark (*). The results of the IACO algorithm under the condition (b) were not reported in [19] at all.

In Tab. 4, an additional column HGA was added, containing information about the result obtained by the hybrid GA algorithm from [11]. In [11], only the best result for the condition (a) was reported, while conditions (b) and (c) were not analysed.

From Tab. 4 one can see that the proposed GA achieves the best found result for the Part 1, condition (a), also reported in most papers. Under the condition (b), we report the result 1170, which is the result 970 (presented in Tab. 3), based on considering hard constraints and increased by the penalty value 200. It should be noted that some other algorithms (for example ACO [18] also found a solution containing the same production costs (970), but with two violated soft precedent constraints. That solution is therefore penalized with 200, so we could ascertain that our solution and the solutions achieved by other heuristic algorithms from literature [18] were of a similar quality.

From Tab. 4, one can conclude that the proposed improved GA slightly outperforms other methods under each condition. It is important to emphasize that the average value obtained by the proposed improved GA was even smaller than the best values obtained by other methods, which indicates that each improved GA run can obtain solutions of a good quality on this instance.

Table 4 Comparative results obtained on the Part 2

1	2	3	4	5	6	7	8	9
Condition		IACO [19]	ACO [18]	TS [14]	SA [11]	GA [11]	HGA [11]	GA [this]
(a)								
	Mean	2456.1	2490	2609.6	2668.5	2796	-	2406.5
	Maximum	2527	2500	2690	2829	2885	-	2409
	Minimum	2435	2450	2527	2535	2667	2527	2404
(b)								
	Mean	2115.4	2117	2208	2287	2370	-	1960
	Maximum	2380	2120	2390	2380	2580	-	1960
	Minimum	1970	2090	2120	2120	2220	-	1960
(c)								
	Mean	2600	2600	2630	2630	2705	-	2500
	Maximum	2740	2600	2740	2740	2840	-	2500
	Minimum	2580	2600	2580	2590	2600	-	2500

4 VERIFICATION OF THE DEVELOPED CAD/CAPP PLATFORM BY OPERATION SEQUENCING VIA GENETIC ALGORITHM

Verification of the intelligent CAD/CAPP platform was conducted on the industrial example of the body of hydraulic cylinder for a hydraulic press brake. Design of the body of the hydraulic cylinder was made in compliance with the developed structure of the system, shown in Figure 1 and presented in detail in the Part I of this paper.

The final solid model of the body of the hydraulic cylinder, as a result of the verification process described in Part I of this paper, was shown in Fig. 3. The solid model of the body of the hydraulic cylinder consists of 52 child EMFs.

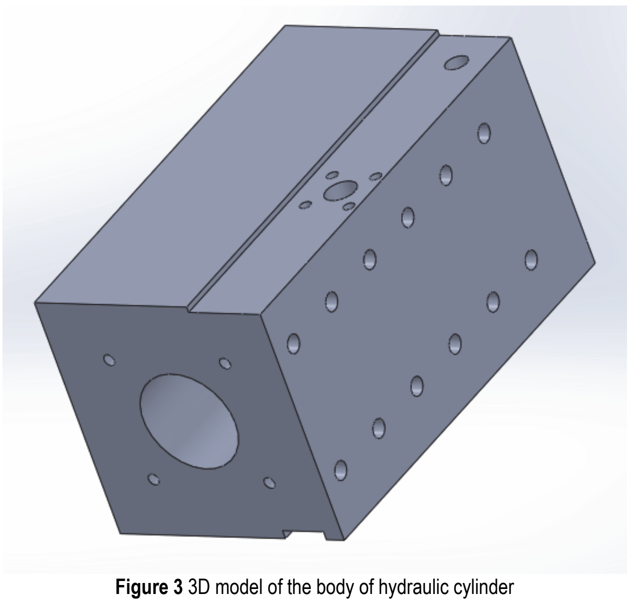

Figure 3 3D model of the body of hydraulic cylinder

For machining of the body of the hydraulic cylinder, the total of 6 machines and 51 tools were identified. The methodology of calculating the costs and more detailed data were out of the scope of this paper and they can be also found on the first authors website accessible by the link https://mf.unibl.org/Stevo/ (*Cost calculation*). In short, the utilization costs for each machine and tool were calculated upon the estimation of real costs, based on the analysis of real production costs [28].

An overview of the costs was shown in Tab. 5. In the *Costs* column, the utilization costs for each machine were shown. Since the total number of available tools is 51, we report the range of the tool utilization costs. In the last three rows the machine, tool and set-up cost change indices were shown. The last column (Unit) contains information about unit costs, given in the local currency BAM/eo (1 EUR = 1.95583 BAM, co-elementary operation).

Table 5 The overall utilization costs for the hydraulic cylinder

No.	Cost type	Costs	Unit
1	Machine utilization costs		
	1.1 Vertical milling machine	0.98	BAM/eo
	1.2 Horizontal milling machine	0.91	BAM/eo
	1.3 Horizontal machine centre	2.12	BAM/eo
	1.4 Column drilling machine	0.71	BAM/eo
	1.5 Coordinate drilling machine	0.77	BAM/eo
	1.6 Universal grinding machine	2.44	BAM/eo
2	Tool utilization costs	0.007-0.99	BAM/eo
3	Machine cost change index	1.1	BAM/setup
4	Tool cost change index	0.06	BAM/setup
5	Set-up change cost index	0.58	BAM/setup

The next step is defining the constraints. *Operation ID,* operation name and TADs were automatically assigned. The lists of available machines, tools and precedent constraints were assigned by the executive designer through the software interface.

For the Part, total of 147 operations were identified. Detailed information about the Part design results was also available on the first author website.

In order to achieve the solution as best as possible, we decided to prolong the execution time of the algorithm and set the termination criterium to be 10000 generations achieved.

Experimental verification of the proposed methodology was performed by executing the improved GA in 12 consecutive runs, with different random seeds. The results of the verification were shown in Tab. 6. The columns were organized as follows: ordering number of executions, population size, number of generations, number of crossovers in a generation, mutation rates for the first and the second mutation, random seed, execution time and total costs (the value of the objective function).

From Table 6 it can be concluded that the proposed GA succeeds to find solutions in a reasonable time. The best solution is found in the 6-th execution, with the smallest value of the objective function 142.49. The average value of the objective function is 144.8. The highest relative error is 1.6%, which indicates a very high level of reliability. The detailed information about the best-found solution can also be found on the first author website (Results for the Hydraulics Cylinder).

Table 6 The results of the experimental verification

1	2	3	4	5	6	7	8	9
No.	PSize	#gen	#cross	prob$_{m1}$	prob$_{m2}$	Seed	Time / s	Costs BAM
1.	90	10000	30	0.4	0.4	13579	4188.64	143.71
2.	90	10000	30	0.4	0.4	15713	4260.90	145.39
3.	90	10000	30	0.4	0.4	73981	4271.28	146.55
4.	90	10000	30	0.4	0.4	85793	4191.78	146.83
5.	90	10000	30	0.4	0.4	68745	4181.1	143.47
6.	90	10000	30	0.4	0.4	12345	4184.05	142.49
7.	90	10000	30	0.4	0.4	52461	4218.35	146.95
8.	90	10000	30	0.4	0.4	49127	4146.69	145.81
9.	90	10000	30	0.4	0.4	31213	4200.2	142.95
10.	90	10000	30	0.4	0.4	91217	4247.96	143.77
11.	90	10000	30	0.4	0.4	24713	4210.22	145.15
12.	90	10000	30	0.4	0.4	17953	4200.02	144.53

To confirm the usability of the results obtained for the Part in the real environment, the sequence of the operations was manually checked in the Laboratory for CAPP systems at the Faculty of Mechanical Engineering University of Banja Luka. By following the results, the production of the part was proved to be possible. Specially, the set of the output data obtained for the Part was useful for the laboratory or enterprise staff because of the following information:

- Identification of the first operation. This information was of great usefulness for the technical preparation of the production, as well as for the start of setup, which was identified as a challenging task.

- Identification of the operation sequencing. As the main result of the algorithm, this information was useful to determine the scheduling of the machines and the organization of other resources, like tools, additional equipment, staff and so on.

- Optimizing the technological resources. It was based on identifying minimal preliminary costs which were obtained as the result of the GA. Identifying machines and tools which generate minimal costs was also a base for realization of the technical preparation, machine schedule in the production environment, maintenance and procurement of other technological resources.

- Total preliminary costs. Based on the objective function, it was possible to generate the overall preliminary costs. This information was crucial in all phases of manufacturing which were performed before the production. Being aware of preliminary costs in earlier phases of the production enables easier calculation of the final production costs during and after the production.

5 CONCLUSIONS

In the comprehensive study we introduced an integrated CAD/CAPP platform based on the elementary machining features and the genetic algorithm. In this platform we established a link between part design phase and process planning phase. This link was based on using the information obtained by enriched Elementary Machining Features in the part design phase and production rules, and also genetic algorithm in the phase of process setup planning and operation sequencing.

Improved GA was used for determining the sequence of operations according to objective function of the minimal production costs. The improved GA uses integer representation for the sequence of operations and implements modified crossover, mutation and selection operators. The performances of the algorithm were tested on the instances used in literature. Experimental results indicate that the proposed GA can find quality solutions in a reasonable time, so it can be used in practical environments.

In a comprehensive case study, the proposed approach was verified on an industrial example of the body of hydraulic cylinder for hydraulic press brake. In the first phase of the process, described in part I of the paper, generic elementary machining forms from the EMF library were applied on a blank part thus creating the CAD model of the industrial example. This CAD model now contains a great number of the pieces of information used in CAPP phase of the operation sequencing. After the part was designed in the first phase, the proposed GA was applied to identify the sequence of operations and the setup process plan. Obtained results indicate a high level of usability of the proposed integrated platform.

6 REFERENCES

[1] Istokovic, D., Perinic M., Vlatkovic M., & Brezocnik M. (2020). Minimizing Total Production Cost in a Hybrid Flow Shop: a Simulation-Optimization Approach. *International Journal of Simulation Modelling, 19*(4), 559-570. https://doi.org/10.2507/IJSIMM19-4-525

[2] Pajak, M. (2020). Fuzzy Model of The Operational Potential Consumption Process of a Complex Technical System. *Universitatis-Series Mechanical Engineering, 18*(3), 453-472. https://doi.org/10.22190/FUME200306032P

[3] Savkovic, B., Kovac, P., Rodic, D., Strbac, B., & Klancnik, S. (2020). Comparison of artificial neural network, fuzzy logic and genetic algorithm for cutting temperature and surface roughness prediction during the face milling process. *Advances in Production Engineering & Management, 15*(2), 137-150. https://doi.org/10.14743/apem2020.2.354

[4] Aboulissane, B., El Bakkali, L., & El Bahaoui, J. (2020). Workspace Analysis and Optimization of The Parallel Robots Based on Computer-Aided Design Approach. *Facta Universitatis-Series Mechanical Engineering, 18*(1), 79-89. https://doi.org/10.22190/FUME190428006A

[5] Chen, D. & Zhao, X. R. (2021). Production Management of Hybrid Flow Shop Based on Genetic Algorithm.

International Journal of Simulation Modelling, *20*(3), 571-582. https://doi.org/10.2507/IJSIMM20-3-CO12

[6] Liu, Y. F. & Zhang, Q. S. (2018). Multi-objective production planning model for equipment manufacturing enterprises with multiple uncertainties in demand. *Advances in Production Engineering & Management*, *13*(4), 429-441. https://doi.org/10.14743/apem2018.4.301

[7] Liu, Q., Li, X., & Gao, L. (2020). Mathematical modeling and a hybrid evolutionary algorithm for process planning. *Journal of Intelligent Manufacturing*, *32*, 781-797. https://doi.org/10.1007/s10845-020-01703-w

[8] Reddy, S. V. B. (1999). Operation sequencing in CAPP using genetic algorithms. *International Journal of Production Research*, *37*(5), 1063-1074. https://doi.org/10.1080/002075499191409

[9] Qiao, L., Wang, X. Y., & Wang, S. C. (2000). A GA-based approach to machining operation sequencing for prismatic parts. *International Journal of Production Research*, *38*(14), 3283-3303. https://doi.org/10.1080/002075400418261

[10] Lee, D. H., Kiritsis, D., & Xirouchakis, P. (2001). Branch and fathoming algorithms for operation sequencing in process planning. *International Journal of Production Research*, *39*(8), 1649-1669. https://doi.org/10.1080/00207540010028100

[11] Li, W. D., Ong, S. K., & Nee, A. Y. C. (2002). Hybrid genetic algorithm and simulated annealing approach for the optimization of process plans for prismatic parts. *International Journal of Production Research*, *40*(8), 1899-1922. https://doi.org/10.1080/00207540110119991

[12] Ma, G. H., Zhang, Y. F., & Nee, A. Y. C. (2002). An automated process planning system based on genetic algorithm and simulated annealing. *Proceedings of the ASME design engineering technical conference*, 2671-2687. https://doi.org/10.1080/002075400411420

[13] Falih, A. & Shammari, A. Z. M. (2020). Hybrid constrained permutation algorithm and genetic algorithm for process planning problem. *Journal of Intelligent Manufacturing*, *31*, 1079-1099. https://doi.org/10.1007/s10845-019-01496-7

[14] Li, W. D., Ong, S. K., & Nee, A. Y. C. (2004). Optimization of process plans using a constraint based TaBu search approach. *International Journal of Production Research*, *42*(10), 1955-1985. https://doi.org/10.1080/00207540310001652897

[15] Li, X. Y., Shao, X. Y., & Gao, L. (2008). Optimization of flexible process planning by genetic programming. *The International Journal of Advanced Manufacturing Technology*, *38*(1), 143-153. https://doi.org/10.1007/s00170-007-1069-x

[16] Salehi, M. & Tavakkoli-Moghaddam, R. (2009). Application of genetic algorithm to computer aided process planning in preliminary and detailed planning. *Engineering Applications of Artificial Intelligence*, *22*(8), 1179-1187. https://doi.org/10.1016/j.engappai.2009.04.005

[17] Kafashi, S. (2011). Integrated setup planning and operation sequencing (ISOS) using genetic algorithm. *The International Journal of Advanced Manufacturing Technology*, *56*(5), 589-600. https://doi.org/10.1007/s00170-011-3202-0

[18] Liu, X. J., Yi, H., & Ni, Z. H. (2013). Application of ant colony optimization algorithm in process planning optimization. *Journal of Intelligent Manufacturing*, *24*(1), 1-13. https://doi.org/10.1007/s10845-010-0407-2

[19] Wang, J., Fan, X., & Ding, H. (2014). An improved ant colony optimization approach for optimization of process planning. *The Scientific World Journal*, *2014*, 15. https://doi.org/10.1155/2014/294513

[20] Huang, W., Hu, Y., & Cai, L. (2012). An effective hybrid graph and genetic algorithm approach to process planning optimization for prismatic parts. *The International Journal of Advanced Manufacturing Technology*, *62*, 9-12. https://doi.org/10.1007/s00170-011-3870-9

[21] Holland, J. (1992). *Adaption in Natural and Artificial Systems.* University of Michigan. A Bradford Book.

[22] Su, Y., Chu, X., Chen, D., & Sun, X. (2018). A genetic algorithm for operation sequencing in CAPP using edge selection based encoding strategy. *Journal of Intelligent Manufacturing*, *29*, 313-332. https://doi.org/10.1007/s10845-015-1109-6

[23] Salehi, M. & Bahreininejad, A. (2011). Optimization process planning using hybrid genetic algorithm and intelligent search for job shop machining. *Journal of Intelligent Manufacturing*, *22*(4), 643-652. https://doi.org/10.1007/s10845-010-0382-7

[24] Ma, G. & Zhang, F. (2012). *Genetic Algorithms for Manufacturing Process Planning*. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-23424-8_7

[25] Younes, A., Elkamel, A., & Areibi, S. (2010). Genetic algorithm in process engineering: Development and implementation issues. *Advances in Process Systems EngineeringStochastic Global Optimization*, 111-145. https://doi.org/10.1142/9789814299213_0004

[26] Zapfel, G., Braune, R., & Bogl, M. (2010). *Metaheuristic Search Concepts: A Tutorial with Applications to Production and Logistics, chap. Metaheuristics Based on Solution Recombination*. Springer-Verlag Berlin Heidelberg.

[27] Milošević M., Lukić D., Antić A., Lalić B., Ficko M., & Šimunović G. (2017). e-CAPP: A distributed collaborative system for internet-based process planning. *Journal of Manufacturing Systems*, *42*, 210-223. https://doi.org/10.1016/j.jmsy.2016.12.010

[28] Lukić D., Milošević M., Antić A., Borojević S., & Ficko M. (2017). Multi-criteria selection of manufacturing processes in the conceptual process planning, *Advances in Production Engineering and Management*, *12*(2), 151-162. https://doi.org/10.14743/apem2017.2.247

Contact information:

Stevo BOROJEVIĆ, PhD, Associate Professor
(Corresponding author)
University of Banja Luka, Faculty of Mechanical Engineering,
Stepe Stepanovića 71, 78 000 Banja Luka, Bosna i Hercegovina
E-mail: stevo.borojevic@mf.unibl.org

Dragan MATIĆ, PhD, Associate Professor
University of Banja Luka, Faculty of Natural Sciences and Mathematics,
Mladena Stojanovića 2, 78 000 Banja Luka, Bosna i Hercegovina
E-mail: dragan.matic@pmf.unibl.org

Miroslav DRAGIĆ, PhD, Assistant Professor
University of Banja Luka, Faculty of Technology,
Stepe Stepanovića 73, 78 000 Banja Luka, Bosna i Hercegovina
E-mail: miroslav.dragic@tf.unibl.org