Available online at www.sciencedirect.com

ScienceDirect

Procedia CIRP 93 (2020) 1508–1513

www.elsevier.com/locate/procedia

53rd CIRP Conference on Manufacturing Systems

Optimizing index positions on CNC tool magazines considering cutting tool life and duplicates

Kaveh Amouzgar[a,*], Amos H.C. Ng[a,c], Goran Ljustina[b]

[a]*School of Engineering Science, University of Skövde, 541 28, Skövde, Sweden*
[b]*Volvo Car Corporation, ME PS Research & Technology, 541 34, Skövde, Sweden*
[c]*Division of Industrial Engineering and Management, Uppsala University, PO Box 534, Uppsala 75121, Sweden*

Abstract

Minimizing the non-machining time of CNC machines requires optimal positioning of cutting tools on indexes (stations) of CNC machine turret magazine. This work presents a genetic algorithm with a novel solution representation and genetic operators to find the best possible index positions while tool duplicates and tools life are taken in to account during the process. The tool allocation in a machining process of a crankshaft with 10 cutting operations, on a 45-index magazine, is optimized for the entire life of the tools on the magazine. The tool-indexing time is considerably reduced compared to the current index positions being used in an automotive factory.

Keywords: tool indexing; cutting tools; genetic algorithm; non-machining time, tool life

1. Introduction

Production times of machining processes can be shortened by combining multiple operations onto one machine, where each operation can require different sets or types of tools. This is possible by using tool turret magazines or (automatic tool changers, ATC) which are indexing tool holders for CNC (computerized numerical control) lathe machines. The tool turrets allow the machine to carry multiple cutting tools at the same time and automatically deliver a sequence of tools to the cutting position based on the process plan. This is done by turrets, rotating along a vertical axis switching from a tool placed on one of the turret's index (stations) to successive tools mounted on another index (station). An illustration of a turret magazine with 8 indexing stations is shown in figure 1 adopted from [7].

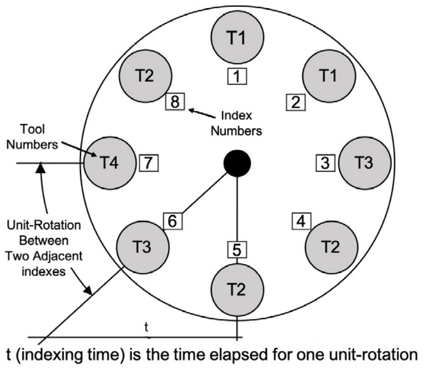
T1
1
T2
8
T1
2
Index Numbers
Tool Numbers
T4
7
3
T3
6
4
Unit-Rotation Between Two Adjacent indexes
T3
5
T2
T2
t
t (indexing time) is the time elapsed for one unit-rotation

Fig. 1. Tool indexing on a 8-index turret magazine [7]

The time it takes for a turret magazine to rotate from one tool slot to another one is known by *turret-indexing time* or *tool-indexing time*. It is obvious that cutting operations can not be carried out during the time when the turret is rotating. The turret-indexing time is hence called non-machining time. The efficiency of machining processes can be further improved by minimizing the non-machining time and increasing the time spent in cut, which results in shorter cycle time. This is possible by optimally allocating the cutting tools on indexes of turret magazine to reduce the number of unit-rotations (i.e., rotation of turret magazine between two adjacent indexes) during a process.

The rotation of the turret magazine can be uni-directional or bi-directional, wherein the later is mostly preferred because of the possibility to find the shortest path from the current index to the target index.The relation between the number of required tools (tool types) for a sequence of operations of a part or sev-

* Corresponding author. Tel.: +46-500-44-8595 ; fax: +46-500-41-6325.

10.1016/j.procir.2020.03.044

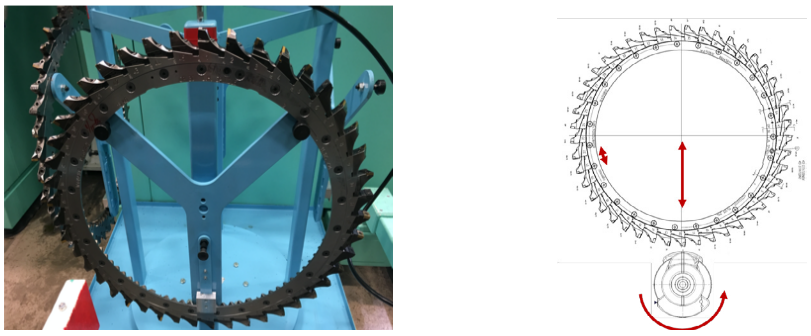
(a) (b)

Fig. 2. (a) The tool turret magazine with 45 index positions used in OP30; (b) the drawing of the OP30 turret

eral parts and the number of available positions (indexes) on a turret can be divided into three scenarios:

1. number of index positions = number of required tools
2. number of index positions > number of required tools
    (a) without tool duplicates
    (b) with tool duplicates
3. number of index positions < number of required tools

The first two scenarios fall into the tool indexing problem, where the main aim is to place the tools on the indexes in a way that the total number of unit-rotations of the turret is minimized. In scenario 2(b), it is possible to place duplicates of some or all tools based on the number of vacant positions to reduce the turret-indexing time. The last scenario needs an extra optimization problem, called tool switching problem (ToSP), to be solved in addition to the tool indexing problem. It is possible to sub-divide scenario 2(b) and 3 by considering filling all unloaded indexing positions with duplicates or leaving some indexes unloaded.

One of the first attempts in the literatures on optimally allocating the cutting tools on index positions of the turret of the CNC machines and the importance of this issue in reducing non-machining time in manufacturing was initiated by applying a genetic algorithm (GA) to minimize the total indexing time [7]. In another study, a meta-heuristic optimization system is developed to determine the best index position of cutting tools on the tool turret [2]. The difference between these two earlier works is the consideration of tool duplicates (allocating more than one instance for some of tool types on the turret) in the later study, which can reduce the total indexing time and improve the productivity of the process. The literature includes a similar study presenting an ant colony (AC) algorithm to minimize the total indexing time [9]. A previous work [2], is improved by integrating a shortest path algorithm in evaluating the objective function, which is a sub-optimization problem of the ATC indexing problem [3]. The same authors present their work on combining the tool indexing with ToSP in a later study [1]. They attempted to solve both problems simultaneously by using a SA algorithm. The dynamic operating condition is also taken into account in the most recent study of the same authors [4]. They propose a solution strategy for solving the indexing and ToSP problems simultaneously in dynamic environments.

In all previous studies of tool indexing problem, it is assumed that the tool-life of the cutting tools is long enough to carry out the operation or a sequence of operations of a *single part*. This assumption simplifies the optimization problem into a single step problem, while in real-world industrial cases the optimal allocation of cutting tools considering the entire life-span (for several parts) of all tools is essential, rather than just a single part.

The present paper contributes to the existing literature by presenting a optimization procedure using GA for a complex real-world tool indexing problem in automotive industry taking into account the entire life-span of the tools on a turret magazine, which is the machining of *1304 crankshafts*, instead of just a single crankshaft. With consideration of the life-span of all tools into the proposed solution, a more realistic condition of the problem consequently a well-defined tool-indexing time is calculated which can hopefully be implemented in the machining center of the factory. Moreover, a customized solution representation (encoding-decoding) and novel genetic operators constitute the proposed GA, as to be described with details in the forthcoming sections.

The industrial case used in the present paper, is the machining operations on crankshafts for 4-cylinder engines, called OP30 (operation 30) carried out at an automotive engine plant. The CNC machining center is a two axes lathe, performing in total of 19 turning operation using 15 different cutting tools on two circular bi-directional turret magazines each with a holding capacity of 45 tools, with the indexing time of approximately 0.2s. The turrets run in parallel opposite each other, the left turret is responsible for 9 cutting operations using 8 different tool types and the right magazine performs 10 operations using 7 cutting tool types. Here we only focus on the right-hand side turret magazine, the methodology and the algorithm can be readily applied to the other turret. One of the tool turret magazines and the related drawing is shown in figure 2. The sequence of operations (denoted by O1 to O10) along with the cutting tool (shown by T1 to T7) required for each operation and the fixed tool-life (the number of crankshafts a tool can machine before it is worn out) is given in table 1. Since removing the

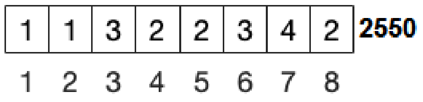

Fig. 3. Solution representation of the 8-index turret magazine shown in figure 1

magazine from the CNC machine and replacing the worn out tools with new ones is a complex and time consuming task, it is beneficial to maximize the time each magazine is used for cutting operation before the replacement of tool should be performed. Therefore, the number of duplicates for each of the 7 tool-types is optimized based on the tool-lives with the aim of maximizing the number of crankshafts that can be machined without removing the turret magazine, which is 1304 parts. The tool types, number of duplicates and the station (index number) each tool is placed on the magazine, according to the current setting in the factory, are given in table 2. The existing allocation of cutting tools on the indexes in the factory is based on experience of the production and tooling engineers, and no previous study on evaluating the current tool-indexing time nor optimization of the allocation has been done before. Figure 2(b) shows a schematic diagram of the turning process in OP30.

This problem falls into scenario 2(b), where no index position is left empty and unloaded. Solving this problem using Brute-force Search algorithms is very difficult due to $7.5e^{35}$ possible alternative tooling set-ups that may need to be evaluated. The tool indexing and ToSP claimed to be NP-hard problems in all related articles mentioned above, which render the finding an optimal solution within a reasonable computational time, especially in our case with a large decision space, using deterministic methods not possible. Therefore, meta-heuristics methods are very well suited for this type of problems.

In the following Section, the customized solution representation and calculation of the objective values is explained. The GA and the novel genetic operators developed for this problem are presented in Section 3 . Finally, the results of this study along with concluding remarks are given in Sections 4 and 5.

2. Solution Representation

A representation of solutions needs to be carefully structured when designing a meta-heuristic algorithm. The solution representation must be designed in such a way that new feasible solution could be easily generated, and all possible feasible solutions could be represented by the technique. We present a unique encoding method specifically designed for this type of problems, but can be readily generalized to solve other similar problems.

Table 1. Operations with the required tools and the tool life (the number of crankshaft each tool can machine before it wears out)

Oper. No.	O1	O2	O3	O4	O5	O6	O7	O8	O9	O10
Tool No.	T1	T2	T6	T5	T7	T5	T6	T7	T4	T3
Tool life	154	131	326	326	326	326	326	326	326	326

Table 2. Number of duplicates for each tool and the list of index position on the turret magazine that each tool is allocated to, based on the current setting in the factory

Tool No.	No. of Duplicates	Index positions
T1	9	1,3,5,8,12,16,20,24,28
T2	10	2,4,6,10,14,18,22,26,30,32
T3	4	7,19,21,33
T4	10	9,11,13,15,17,23,25,27,29,31
T5	4	34,36,40,42
T6	4	35,37,41,43
T7	4	38,39,44,45

A solution is represented by a row vector. The columns represent the index positions on the turret magazine, starting from the top station as index No.1 continued by moving clockwise to next indexes subsequently to the last index (No.45).

Figure 3 shows the solution representation corresponding to the 8-index turret magazine illustrated in figure 1. It can be seen that a T1 is placed in index No.1, another T1 on index No.2, until a T2 on index No.8. The allocation of duplicates is possible by inserting the corresponding tool number in several columns. For example, there are 3 instances of T2, which are positioned on indexes 4,5, and 8.

2.1. Objective function

In this tool indexing problem the objective is to minimize the tool turret-indexing time. The value of the objective function can be calculated by multiplying the indexing time of the CNC machine (0.2s) by the total number of unit-rotations, based on the given operation sequence for the entire life-span of all tools placed on the turret. Because the indexing time is constant, we can assume that minimizing the total number of unit-rotations the turret magazine has to go through to the operations for all the 1304 crankshafts is our objective function.

Let us discuss the objective function evaluation process of the simple example illustrated in figure 1 in which 5 cutting operations using 4 different tool types with a sequence of T4-T2-T3-T4-T1, placed on a turret magazine with 8 index positions are carried out to machine a workpiece. The number of duplicates and tool life for T1 to T4 are 2,3,2,1 and 150,100,150,300 parts, correspondingly.

First, we focus on evaluating the total number of rotations required to machine the first workpiece, based on the given operation sequence. We need to find the shortest route, the turret has to rotate to position the tools $T4, T2, T3, T4, T1$ in sequence, to the cutting position. Such an objective function evaluation of each solution by it self is a separate optimization sub-problem of the master problem (i.e., optimal allocation of tools on the turret), which can be converted to a shortest path problem. For this purpose, we adopt the methodology proposed in [3], which applies the Dijkstra shortest path algorithm [8] with additional *dummy-in* (D_{in}) and *dummy-out* (D_{out}) nodes. Readers can refer to the original paper for the full details of the method.

The objective value (the shortest path which can be translated into the number of unit-rotations) for machining the first

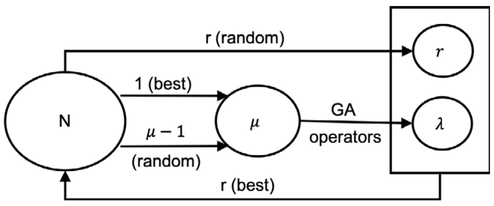

Fig. 4. The flowchart for G3 model

workpiece with the tool allocation shown in figure 1 is 6 unit-rotations, and the shortest turret index rotation is D_{in}-7-8-6-7-1-D_{out}.

All the methods and algorithms presented in existing studies, will stop here and report 6 as the objective function value for this specific solution. Their optimization algorithms search through different alternative allocations with the aim of finding the minimum of this objective value. This procedure is not realistic since the tools on the turret will not be replaced with a new one after each part (for our case, crankshaft) is machined. Hence the life of the tools included in the shortest path used for machining one workpiece (part or crankshaft) will be reduced by one. In other words, we have to reduce one from the tool life of the cutting tools placed on indexes 7-8-6-7-1. For instance, after machining 100 parts with the same shortest route mentioned above, T2 placed on index No.8 cannot be used for machining any more parts, because according to the given tool lives, T2 has the life-span of 100 parts. In this situation the Dijkstra algorithm must find another shortest path without considering the tool on index No.8. This process will continue until all the tools on the turret magazine are worn-out, consequently 300 parts are machined. The objective function value is thus calculated by the sum of multiplication of shortest distance by the number of parts for each solution. The objective function value for this simple 8-index turret is calculated by:$(6*100)+(6*50)+(11*50)+(11*100) =$ **2550** unit-rotations, where the first number in each parenthesis is the shortest distance (number of rotations) of the turret magazine, and the second number is the number of parts that can be machined with the number of rotations given in the same parenthesis.

3. Genetic Algorithm

For this study the Generalization Generation Gap (G3) model [6], introduced for real parameter optimization, is modified with the new solution representation and a novel and customized genetic operators. The flowchart for G3 algorithm is illustrated in figure 4, and the steps in the algorithm are as follows:

Step 1: From a population N, The best parent and $\mu - 1$ other parents are selected.
Step 2: By applying the genetic operators on the chosen μ parents, λ offspring are generated.
Step 3: r random parents are chosen from the population N
Step 4: The r solutions in step 3 are replaced with r best solutions chosen from the combined r random parents and λ offspring.

3.1. Crossover

In a GA crossover operator is required for the creation of new solutions from parents. A novel crossover operator has been designed for this work. In addition, a repair process is designed with the aim of repairing offspring, generated by crossover, which does not satisfy the fixed number of tool duplicates which is required in this problem.

The concept of the crossover operator is to exchange some portion of strings (solutions represented as strings) between the two parents to create new strings (offspring). In general if the nature of the problem is known to the algorithm designer, and the solutions are represented in a clever way that the stronger portions of the strings (the good bits of the strings) are exchanged in crossover operation, it is more likely that the created offspring are good strings with the a better fitness value.

Here, a customized and novel crossover that exchanges the strongest portion of the parents' string is designed. The crossover operator for the simple 8-index turret mentioned earlier can be described by using the illustration in figure 5.

Two parents are shown in the left-hand side of figure 5(a), the objective values of the parents are 2550 and 2800 unit-rotations. The crossover-repair procedure is as follows:

Step1: Find the best index rotation path (shortest route) from the two parents. The index positions used in this path are highlighted in the figure for both parents. P1={1,6,7,8}, P2={2,6,7,8}.
Step2: Allocate the tools on the best indexes of P2 (highlighted tools on P2), to the same index numbers of P1, and keep the other tool allocations of P1 unchanged. This means T1,T3,T2 and T4 should be placed on index numbers 2,6,7 and 8 of P1, to create an offspring shown in the right-hand side of the figure. The same procedure will create the second offspring.
Step3: Check the constraints on the number of instances of all tools in the offspring, to determine if repair is needed. Here we can observe that the second offspring has three duplicates of T1, while based on problem input only 2 instances of T1 is required. Hence, offspring2 need repair to be a feasible solution.
Step4: (*Repair*) Figure 5(b) illustrates the repair process. We understand that one of the 3 instances of T1s (positioned on indexes 1,2,4) should be replaced by a T2. Since T1 in index 1 was active in the crossover process (was exchanged from P1), assuming it is a strong gene of the string, the next instance of T1 which did not participate in crossover, i.e., the T1 placed on index 2, will be replaced by a T2.

To summarize, the integrated corssover-repair operator recombines good substrings from two strings (parents) to create better substrings (offspring). It is interesting to observe that our operator did, in fact, create two stronger offspring, the fitness

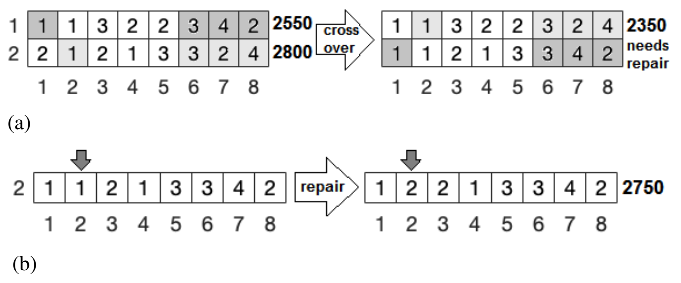

Fig. 5. An illustration of the customized crossover-repair operator by using the simple 8-index turret magazine in figure 1 (a) the best indexing rout of the two parents are swapped, keeping the other indexes unchanged; (b) the repair process on the second offspring

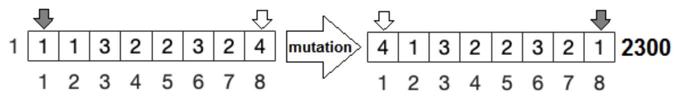

Fig. 6. In the mutation operator, the index position of the first instance of two random tools, that did not participate in the crossover process, are swapped creating a fitter offspring

of both offspring (2350 and 2750) is better than their related parents. The repair process is applied to offspring which violate the constraints of the problem.

3.2. Mutation

The customized mutation operator designed for this problem alters two string locally from a parent, while keeps the feasibility of the mutated solution. Basically, two random tool numbers are selected from the available tools. Then, the index positions of the first instances of the two randomly chosen tools are swapped. For example, considering the first offspring created from the crossover operation, figure 6 illustrates the mutation operator with the process described as follows:

Step1: Two random number from 1 to 4 are chosen $\{1, 4\}$. the first instances of T1 and T4 are placed on index numbers 1 and 8, correspondingly.

Step2: The position of the two tools are swapped, i.e. T1 will replace T4 on index 8 and T4 will replace T1 on index 1, creating the new mutated offspring with a better objective value of 2300 unit-rotations .

It should be noted that because this operator does not change the number of duplicates, the mutated offspring will be a feasible solution and a repair process is not needed. A mutation probability (ρ_μ) is defined, which will limit the number of crossover offspring that undergoes the mutation process.

A systematic study on the parameters of the G3 algorithm [5] recommended $N = 100$, a parent size of $\mu = 3$, number of offspring of $\lambda = 2$, and $r = 2$. However, the crossover-repair and the mutation operators developed for this work will create two offspring from two parents. Therefore, the parameters are slightly tuned to fit the our circumstances: $\mu = 2$ and $\lambda = 2$.

4. Results

The proposed algorithm for optimal tool allocation of OP30 is programmed in MATLAB_ R2019a, and run on an Intel Core i5-4310U CPU@ 2.00GHz with 16.0GB of RAM laptop.

The modified G3 model is started with $N = 100$ initial random population, by shuffling the existing allocation of tools on the turret magazine, i.e, how it is now allocated in the automotive engine factory under study. The mutation probability (ρ_μ) introduced in the previous section need to be determined, but lack of benchmarks for this problem, hinders a proper parameter tuning. Therefore, we ran the G3 algorithm for 37 times, with the stopping criteria $STD(obj) < 1$, while keeping the initial population unchanged, each time using a mutation probability within the range of $\rho_\mu = [0.05, 0.95]$ with 0.025 intervals. The total computation time recorded for all 37 runs is $2643s$, hence, each run will take around $71s$. The best objective function values were achieved for $\rho_\mu = 0.55$ with the total unit-rotations equal to 19560, dividing this value to a total of 1304 crankshafts, gives an average of 15 unit-rotations for each crankshaft, i.e., a significantly improved tool turret-indexing time of only $3s$.

Table 3 clarifies the superior results from this study when compared to the numbers obtained from the current tool allocation in the factory. The reduction of tool-indexing time from 9 to 3 seconds and an average unit-rotations per part from 45 to 15 seconds, are the highlights of the results. This saving in the total turret-indexing time can have a significant effect on high volume production costs. Moreover, production and tooling engineers can use this saving in improving the quality of machined parts, e.g., surface roughness or dimensional accuracy. The tool allocation from this study and the current allocation of tools in the factory are shown in figure 7.

Table 3. The results obtained from the best tool allocation recommended by this study compared to current tool allocation in the automotive engine factory

	total unit-rotations	unit-rotations per part	non-maching time (s)
This study	19560	15	3
Factory	58398	45	9

Figure 8, illustrates the number of unit-rotations per part throughout the life-span of the turret magazine. The tool allocation found in this study, will be able to machine 262 crankshaft with the minimum possible number of rotations, 10 unit-rotations. This is the lower bound of the problem, because there are 10 operations in the sequence and one unit-rotation per operation will lead to a total of 10 unit-rotations. Comparing, even if OP30 is now using the shortest path, it can machine the first 131 crankshafts with 23 unit-rotations, which is more than twice the number of rotation we found in this study. Looking at the far end of the plot in figure 8, in which most of the tools are worn out, the allocation from this study will machine the last crankshaft with 41 unit-rotations while the current allocation needs 95 unit-rotations.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
This study	4	7	5	6	2	1	2	4	1	2	1	2	6	5	7	4	4	3	1	2	1	2	1
Factory allocation	1	2	1	2	1	2	3	1	4	2	4	1	4	2	4	1	4	2	3	1	3	2	4

	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	
This study	2	6	5	7	4	3	4	4	3	4	4	7	5	6	2	1	2	1	2	1	4	3	**19560**
Factory allocation	1	4	2	4	1	4	2	4	2	3	5	6	5	6	7	7	5	6	5	6	7	7	**58938**

Fig. 7. Solution representation of the best tool allocation found in this study and the current allocation in the automotive engine factory under study

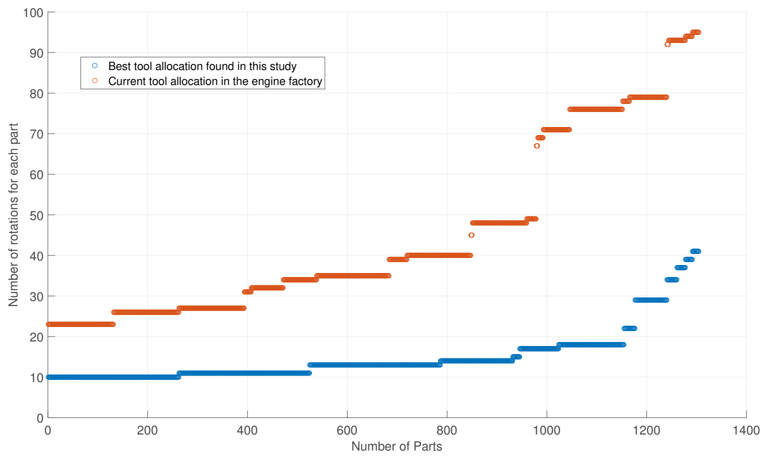

Fig. 8. The number of unit-rotations for OP30, throughout the life-span of the 45-index turret magazine

Squeezing findings out of the same figure, the deviation of the number of rotations along x-axis (i.e., the life-span of the tools on the turret) is lower in the results obtained in this study compared to the existing allocation. The lower the deviation, the lower the deviation in tool-indexing time in machining 1304 crankshafts, ultimately results in a better production planning.

5. Conclusion

The main purpose of this study was to find the optimal allocation of cutting tools on a 45-index turret magazine for machining crankshafts of automotive engines, with the objective of minimizing the tool-indexing time considering the entire life-span of all tools on the magazine. For this purpose, a customized encoding-decoding scheme is designed and a GA with novel crossover-repair and mutation operator is developed. The optimization algorithm was able to find a considerably better tool allocation compared to the current allocation of cutting tools on the turret magazine running the engine factory. The tool-indexing time was reduced by three time, from 9 to 3 seconds.

Analysing the collision detection of the obtained index positions in this study is being planned for future work. Furthermore, this work can be extended by including more objectives such as the number of tool duplicates and the number of changes of index position from the current allocation in the factory.

Acknowledgements

This work was partially financed by the Knowledge Foundation (KKS), Sweden, through the Synergy project, Knowledge-Driven Decision Support via Optimization (KDDS).

References

[1] Adil, B., Ozsoydan, F.B., 2017. Minimizing tool switching and indexing times with tool duplications in automatic machines. The International Journal of Advanced Manufacturing Technology 89, 1775–1789. doi:10.1007/s00170-016-9194-z.

[2] Baykasoglu, A., Dereli, T., 2004. Heuristic optimization system for the determination of index positions on CNC magazines with the consideration of cutting tool duplications. International Journal of Production Research 42, 1281–1303. doi:10.1080/00207540310001622557.

[3] Baykasolu, A., Ozsoydan, F.B., 2016. An improved approach for determination of index positions on CNC magazines with cutting tool duplications by integrating shortest path algorithm. International Journal of Production Research 54, 742–760. doi:10.1080/00207543.2015.1055351.

[4] Baykasolu, A., Ozsoydan, F.B., 2018. Minimisation of non-machining times in operating automatic tool changers of machine tools under dynamic operating conditions. International Journal of Production Research 56. doi:10.1080/00207543.2017.1357861.

[5] Deb, K., 2005. A population-based algorithm-generator for real-parameter optimization. Soft Computing 9, 236–253.

[6] Deb, K., Anand, A., Joshi, D., 2002. A computationally efficient evolutionary algorithm for real-parameter optimization. Evolutionary computation 10, 371–395.

[7] Dereli, T., Filiz, I.H., 2000. Allocating optimal index positions on tool magazines using genetic algorithms. Robotics and Autonomous Systems 33, 155–167. doi:10.1016/S0921-8890(00)00086-5.

[8] Dijkstra, E.W., et al., 1959. A note on two problems in connexion with graphs. Numerische mathematik 1, 269–271.

[9] Gopala, A., Rao, K.K.M., 2006. Optimal allocation of index positions on tool magazines using an ant colony algorithm. Int J Adv Manuf Technol 30, 717–721. doi:10.1007/s00170-005-0099-5.