Optimization of operation sequencing based on GA-Jaya algorithm

Kai Zheng*, Rui Zhang, Zhen-wei Zhu and Hua-dong Zhao

College of Mechanical and Power Engineering, Zhengzhou University, Zhengzhou 450001, China

Corresponding author’s e-mail: zk1996@gs.zzu.edu.cn

Abstract: To solve the operation sequencing problem in CAPP that is a difficult problem, combining the idea of genetic algorithm, an GA-Jaya algorithm is proposed to minimize the total cost. In the GA-Jaya, the population is initialized according to the procedure priority adjacency matrix which makes the population all meet the process priority relationship. Mutation iteration operator and two kinds of crossover iteration operator are designed for process sequence and processing resource evolution. The GA-Jaya algorithm is applied to a typical case, and compared with the existing genetic algorithm, particle swarm optimization algorithm and ant colony optimization algorithm. The results show that the average quality of the solution obtained by the GA-Jaya algorithm is better than the existing genetic algorithm, particle swarm optimization algorithm and ant colony optimization algorithm.

1. Introduction

Computer Aided Process Design (CAPP) is an important part of computer integrated manufacturing system (CIMS) and an important link between COMPUTER aided design (CAD) and computer aided manufacturing (CAM). Its essence is to match the manufacturing capacity of an enterprise with the design information of parts, automatically determine and optimize the process route of designing parts [1], and convert the part design information obtained by CAD system into a series of processing operations that can be used by CAM system. CAPP is a multi-level, multi-task complex system that integrates selection, calculation, planning and drawing. It generally includes the following continuous processes: firstly, manufacturing features are identified from CAD system; Secondly, according to the manufacturing resources of the workshop, the manufacturing feature processing scheme is generated, and the processing steps and applicable manufacturing resources of each manufacturing feature are determined. Finally, priority constraints are considered and appropriate manufacturing resources are selected to prioritize processing processes to achieve an optimal plan, such as the lowest processing cost or the shortest processing time, so as to improve production efficiency and reduce manufacturing costs [2].

One of the two key technologies and difficulties in CAPP system is the process sequencing optimization of the whole parts. The essence of the process sequencing optimization of parts is the process of matching the design information of parts with the manufacturing resources of enterprises. In the past decade, many optimization algorithms have been used in process sequencing optimization, such as genetic algorithm (GA) [1-3], simulated annealing algorithm (SA) [4], ant colony algorithm (ACO) [5,6], particle swarm optimization (PSO) [7,8] and various hybrid algorithms [9,10]. However, process sequencing optimization has the potential to be further improved to improve efficiency. Jaya algorithm
is a method for solving continuous optimization problems proposed by Rao in 2016 [11]. Like genetic algorithm and particle swarm optimization algorithm, Jaya algorithm belongs to swarm intelligence algorithm. Its advantages are that it does not need to adjust algorithm parameters, and it has good robustness and high efficiency [12]. Recently, there are some researches based on Jaya algorithm, such as Bayesian network structure learning [13], flexible Job Shop Scheduling [14].

In view of the characteristics of process sequencing optimization problem, this paper studies the use of this new optimization algorithm to solve the process sequencing problem, on the Jaya algorithm idea fusion of genetic algorithm variation and crossover, developed three new iteration operators to realize the iterative process of GA-Jaya algorithm, formed a GA-Jaya algorithm to solve the process sequencing optimization problem. The proposed algorithm is used to solve typical examples to verify the feasibility and superiority of the proposed algorithm.

2. Process sequencing optimization in CAPP

2.1 Problem description

The process sequencing of parts is to sort and optimize the processing schemes of each feature of parts under certain technological constraints and the condition of meeting optimization objectives, so as to obtain the optimal or approximate optimal processing sequence.

When a manufacturing feature of a part is machined, a path allowing the Tool to enter the feature is required, which is called TAD (Tool Approach Direction). Geometric reasoning is used to identify effective TADs in feature recognition[15,16]. Process sequencing involves machining sequence, machine tool selection, cutter selection and TAD selection. A good part process plan is based on two elements[7]: (1) optimal selection of machine tools, cutters and TAD in each process; (2) The optimal processing sequence of parts. Therefore, the algorithm developed needs to address these two aspects.

Among all the process constraints, there are mainly priority relationship constraints and process clustering constraints [17]. Priority relationship constraints are mandatory constraints, and priority relationship constraints must be first met in process planning. Clustering constraints generally do not need to be fully met, but the cost of violating such constraints is high. The priority relationship mainly includes datum first, coarse first, fine first, principal first, second order first, surface first and hole first, etc. The priority constraint relationship can be described by Operation Precedence Graph (OPG), which is a directed acyclic Graph, as shown in figure 1. It consists of two sets: vertex (procedure) set and directed edge (priority constraint) set. In the figure: the node represents a procedure, and the direction of the directed edge represents the sequential relationship between the two procedures.

Because computers are better at processing numbers rather than recognizing images, OPG[17] is represented by adjacency matrix $P_{n \times n}$ to realize the transfer of graphics to computers. The sum of elements in each row of the matrix is the number of constraints imposed by the procedure, and the sum of elements in each column is the number of constraints imposed by the procedure. The adjacency matrix $P_{8 \times 8}$ transformed from Figure 1 is shown in figure 2.

![Figure 1. Process priority diagram](image)
After the parts are clamped once, a series of processing of the same TAD on the same machine tool is regarded as a process, and each process corresponds to a number of machine tools, tools and TAD combinations [3]. Therefore, the process sequencing optimization problem referred to in this paper is a single objective multi-constraint combinatorial optimization problem. The objective of optimization is process sequence, machine tool, cutter and TAD corresponding to each process. Constraints include priority constraint and machine tool function constraint, and the objective function is to minimize the total processing cost.

![Figure 2. Adjacency matrix diagram](image)

2.2 Objective function and calculation
The objective function of process sequencing optimization in this paper is to minimize the total machining cost (TC), including the following five types: total machine tool cost (TMC), total tool cost (TTC), total machine tool replacement cost (TMCC), total tooling replacement cost (TSCC) and total tool replacement cost (TTCC). Refer to reference [15] for specific calculation formula.

3. Ga-Jaya algorithm is used to solve the process sequencing optimization problem

3.1. Basic theory of Jaya
Jaya algorithm is Rao, a meta heuristic algorithm is put forward in 2016 came [11], the algorithm also considers the optimal individual and the worst, the effect of the individual to search the optimal solution is a combination of evolutionary algorithm and the characteristics of swarm intelligence, and do not like other evolutionary algorithms need to adjust the parameters of the structure is simple, adaptive, flexibility, improve and completeness, It is a typical swarm intelligence optimization algorithm. The iterative formula of the algorithm is as follows:

$$X'_{j,k,i} = X_{j,k,i} + r_{1,j,i} \left( X_{j,best,i} - X_{j,k,i} \right) - r_{2,j,i} \left( X_{j,worst,i} - X_{j,k,i} \right)$$  \hspace{1cm} (1)

In the formula, $X'_{j,k,i}$ is the $j^{th}$ variable of the $k^{th}$ individual in the $i^{th}$ iteration, $X_{j,best,i}$ is the variable corresponding to the fittest individual in iteration $i^{th}$, $X_{j,worst,i}$ is the variable corresponding to the individual with the worst fitness in iteration $i^{th}$, $r_{1,j,i}$ and $r_{2,j,i}$ are two random numbers in the range [0,1], $X'_{j,k,i}$ is the updated value of $X_{j,k,i}$, If the fitness of $X'_{j,k,i}$ is superior to that of $X_{j,k,i}$, $X'_{j,k,i}$ is substituted for $X_{j,k,i}$.

In the formula, the second term "$r_{1,j,i} \left( X_{j,best,i} - X_{j,k,i} \right)"$ represents the tendency of the solution to converge towards the optimal solution, And the third term "$-r_{2,j,i} \left( X_{j,worst,i} - X_{j,k,i} \right)$" indicates the tendency of the solution to avoid the worst solution. Therefore, Jaya algorithm always moves towards the best individual while keeping away from the worst individual in the process of individual iteration update.
3.2. Solution of the coding

The solution of process sequencing is expressed as chromosomes composed of ordered genes. All candidate solutions form a solution space. In the chromosome, a gene needs to contain the process ID, machine tool ID, tool ID, and TAD ID, as well as the candidate machine tool sequence, tool sequence, and TAD sequence. Figure 3 shows the encoding of a solution for the case part.

![Figure 3. Chromosome coding schematic of a solution of a case part](image)

3.3. Population initialization

Initialization of the population to the quality and speed of the algorithm has an important influence, in order to solve the speed, avoid the process sequence of random against preference constraint and not feasible, therefore, when making the population initialization, will preference constraint into consideration, through an initialization method, to ensure that the initial population contains process sequences are feasible.

In this paper, the OPG graph information of parts is transformed into adjacency matrix \( P_{nnn} \), and the population initialization is realized by using topological sorting method.

3.4. GA-Jaya iteration

The traditional Jaya algorithm is suitable for solving continuous optimization problems, while process sequencing is a typical discrete optimization problem. This paper makes full use of the influence of \( X_{best} \) and \( X_{worst} \) on the individual iteration. Combining the variation and crossover ideas of genetic algorithm, a discrete GA-Jaya iterative method applied to process sequencing is proposed. The iterative method contains three iterators:

3.4.1 Mutation iteration operator

The mutation iteration operator changes the partial order of the process and reduces the number of types of processing resources (machine tools, cutters and TAD) by means of topological ordering. It includes two processes, process variation and processing resource variation, and the specific steps are as follows:

**Process variation:** In the process of process variation, the processing resources under the process change with the process.

- **Step 1:** Select a random position according to the number of procedures and divide the procedure sequence into A and B parts;
- **Step 2:** generate a random number \( r \) in the interval of \((0,1)\). If \( r \leq 0.5 \), copy part A of the procedure sequence to an initially empty array \( C \). If \( r > 0.5 \), copy part of the procedure sequence to an initially empty array, as shown in Figure 4(a);
- **Step 3:** Generate a zero matrix \( R_{jxj} \) (or \( R_{(n-j)(n-j)} \)) according to the number of elements \( j \) (or \( n - j \)) in array \( C \);
- **Step 4:** Access to an array of two elements of \( C, C \{d\} \) and \( C \{e\} \), and obtain its process ID (\( oper_u \) and \( oper_r \)), the two process ID combination (\( oper_u, oper_r \)), as access to bottom corner of the element in the adjacency matrix \( P_{nnn} \), obtain the values in the matrix, and copies the value to the local adjacency matrix.
matrix $R_{j\times j}$ (or $R_{(n-j)\times(n-j)}$), until the combination of any two elements in the array are complete, as shown in figure 4 (b);

Step 5: Compute the sum of the elements of each column of the local adjacency matrix $R_{j\times j}$ (or $R_{(n-j)\times(n-j)}$) to generate an array $D = [\text{NOR}(\text{oper}_1), \ldots]$;

Step 6: Select a random element which value is 0 from array $D$, assuming $\text{NOR}(\text{oper}_g)$, take $\text{oper}_g$ into the initially empty array $E$, $E = [\text{oper}_g]$, and remove $\text{oper}_g$ from array $C$;

Step 7: Delete row $\text{oper}_g$ and column $\text{oper}_g$ from local adjacency matrix $R_{j\times j}$ (or $R_{(n-j)\times(n-j)}$) to generate a new matrix $R_{(j-1)\times(j-1)}$ (or $R_{(n-j-1)\times(n-j-1)}$), as shown in figure 4(c);

Step 8: Calculate the sum of each column of the new matrix $R_{(j-1)\times(j-1)}$ (or $R_{(n-j-1)\times(n-j-1)}$) and put it back into array $D$;

Step 9: Select a random element which value is 0 from array $D$, assuming $\text{NOR}(\text{oper}_h)$, put $\text{oper}_h$ into array $E$, that is, $E = [\text{oper}_g, \text{oper}_h]$, and delete it from the array $C$.

Step 10: Repeat steps 6, 7, 8 and 9 until there is only one element in array $C$, as shown in figure 4(d). Add the last element in array $C$ to the last bit of array $E$, and the sort in array $E$ is the mutated sort.

Step 11: Combine the sorting in array $E$ with A (or B) to form A complete operation sequence, as shown in figure 4(e);

Variation of processing resources: The variation process of machine tool, tool and TAD is similar, the variation of machine tool is first, the variation of tool is second, and the variation of TAD is last. Take machine tool variation as an example.

Step 1: Find the largest number of machine tools $M_{\text{max}}$ and the smallest number of machine tools $M_{\text{min}}$ in the sequence;

Step 2: find out the process set using the machine tool $M_{\text{min}}$;

Step 3: Judge successively whether $M_{\text{max}}$ is included in the candidate machine tool sequence of process $\text{opers}$ in process set $\text{oper}_x$. If so, set the machine tool in process $\text{opers}$ to $\text{oper}_x$; if not, select a machine tool randomly from the candidate machine tool sequence of process $\text{opers}$.

3.4.2. Best cross iteration operator

The sequence of the current solution is compared with that of the best solution, and the same genes in the sequence are preserved. Part of the sequence is in the order of the best solution, and part of the sequence is in the order of the current solution. The processing resources change with the process, as shown in figure 5. The specific steps are as follows:

Step 1: Select a position $j$ randomly according to the length of the procedure sequence and divide the procedure sequence into left and right parts;

Step 2: Copy the left part of $X_{\text{best}}$ to become the left part of $X_{k+1}$;

Step 3: find the genes with the same sequence in the right part of $X_k$ and $X_{\text{best}}$, and store them in the corresponding position of $X_{k+1}$;

Step 4: Find out the sequence of other genes in $X_{\text{best}}$ in $X_k$, and generate the right part of $X_{k+1}$ according to the sequence of $X_k$;
Step 5: Output $X_{k+1}$:

\[
\begin{array}{c|cccc}
\hline
j & A & B & j & \left\lceil r \leq 0.5 \right. \\
\hline
1 & 8 & 4 & 3 & \left\lceil r \leq 0.5 \right. \\
2 & 7 & 6 & 5 & \left\lceil r \leq 0.5 \right. \\
\hline
\end{array}
\]

(a)

Figure 4. Schematic diagram of process variation

3.4.3. Worst cross iteration operator

The current process and worst process sequences comparison, marking the position of the sequence of the same process, choose the best process sequence in the same location of the gene as a new individual, the remaining part of the order of the sequence with the current solution, the other part of the order of the sequence with the worst, as shown in figure 6, specific steps are as follows:

Step 1: Identify genes with the same sequence in $X_k$ and $X_{\text{worst}}$ and mark their locations.

Step 2: Copy the gene located at the step 1 marker in $X_k$ to the corresponding location in $X_{k+1}$.  

Step 3: Select a position $j$ randomly according to the length of the procedure sequence and divide the procedure sequence into left and right parts.
Step 4: Save the remaining genes of the left part of $X_k$ into $X_{k+1}$ according to the sequence in $X_k$.

Step 5: Find out the sequence of genes in the right part of $X_k$ in $X_{\text{worst}}$, and form the right part of $X_{k+1}$ according to the sequence.

Step 6: Output $X_{k+1}$.

Figure 5. Schematic diagram of Best cross iteration

Figure 6. Schematic diagram of Worst cross iteration
Begin
Initialize population $N$ and set the number of iterations $NOI$
Calculate the fitness value of each solution to find the best and worst solution
$q=1$, $noi=1$
Pick the $q$th solution
randi(3) is equal to 1?
randi(3) is equal to 2?
Perform mutation iteration
Perform Best cross iteration
Perform the Worst cross iteration
fitness $(q) >$ fitness $(q'$)\
N
Y
The iterative solution replaces the original solution
Do not replace

$q = q + 1$
$q < N$?

$q = 1$
$noi = noi + 1$

$noi < NOI$?
The maximal fitness value is the optimal solution
End

Figure 7. Algorithm flow chart

4. The case study
In order to verify the effectiveness of the algorithm, a case jointly adopted with literature [3,7,17] is adopted to verify the effectiveness and advantages of the algorithm through calculation comparison. The case part shown in figure 8 is composed of 11 manufacturing features that can be divided into 14 machining processes.

Figure 8. 11 manufacturing features of the case part

The algorithm in this paper is written by Matlab. The number of population in the algorithm is $N = 150$, and the maximum number of iterations is $NOI = 200$. Table 1 shows the comparison of
results of GA-Jaya algorithm, improved genetic algorithm [3], genetic algorithm [7], particle swarm optimization algorithm [7] and ant colony algorithm [17]. The improved Jaya algorithm has been run for 10 times. Figure 9 and 10 are convergence curves for obtaining the best solution.

| Algorithm    | Optimal solution cost | Solving cost mean | Worst solution cost | Earliest algebra of convergence |
|--------------|-----------------------|-------------------|---------------------|-------------------------------|
| GA-Jaya      | 1364                  | 1390.8            | 1437                | 22                            |
| FOSOGAF      | 1357                  | 1424.7            | 1459                | 73                            |
| GA           | 1381                  | 1459.4            | ——                  | ——                            |
| PSO          | 1361                  | 1430.0            | ——                  | ——                            |
| ACO          | 1357                  | 1419.0            | ——                  | ——                            |

Figure 9. Convergence curves of the best solution obtained by GA-Jaya algorithm for the first five times

Figure 10. Convergence curves of the optimal solution obtained by ga-Jaya algorithm for the last five times

The optimal process planning scheme obtained by the algorithm in this paper is shown in Table 2, and its total cost is $TC = 1364$. The cost of the optimal solution obtained by GA in literature [7] is
1381 (see Table 16 in literature [7]). The optimal solution obtained by GA-Jaya algorithm is better than that obtained by GA. Although the best solutions obtained by SOFOGAF, PSO and ACO are slightly superior to GA-Jaya algorithm, ga-Jaya algorithm is far superior to SOFOGAF, GA, PSO and ACO in terms of average quality of solutions. And the earliest convergence algebra of GA-Jaya algorithm is better than the improved GA algorithm. By comparing the results of the five algorithms, the ga-Jaya algorithm in this paper is superior to the existing GA and PSO algorithms in terms of solution quality.

| Table 2. a process planning scheme for case parts |
|-----------------------------------------------|
| Process ID | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 |
| Machine ID | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 |
| Tool ID | 6 | 6 | 6 | 2 | 9 | 10 | 7 | 7 | 7 | 2 | 9 | 1 | 5 |
| TAD ID | +z | -z | -z | -z | -z | -z | -z | -z | -z | +z | +z | +y | +y |

5. Conclusion
In this paper, GA-Jaya algorithm is proposed to optimize the process sequencing problem which is difficult to solve in CAPP by combining the crossover and mutation ideas of genetic algorithm. The algorithm generates feasible solutions according to the procedure priority adjacency matrix, and the solution is always in the feasible domain regardless of mutation iteration or crossover iteration. Thus, the search space of the algorithm is reduced and the efficiency of the optimization algorithm is improved. This algorithm is applied to the diamond roller CAPP system developed in practice. The case comparison results show that the average quality of solutions obtained by the improved Jaya algorithm proposed in this paper is better than SOFOGAF, GA, PSO and ACO.

Although ga-Jaya algorithm proposed in this paper obtains high-quality solutions for a given case, with more complex geometric shapes, the number and types of features to be produced will increase exponentially, and the constraint relationship between each feature will become more complex, which will be a challenge to the search ability of the algorithm. Therefore, in the future, new sequence adjustment methods are needed to reduce computing time and improve the performance of the algorithm in more complex process sequencing problems.

References
[1] Gao Gang. Research on decision Technology in Intelligent CAPP [D]. Yanshan University, 2018.
[2] Hua Guangru, Wang Huifeng, Zhang Zhenhuan, et al. Comprehensive Optimization Method of Machining Scheme Selection and Operation Sequencing Based on Genetic Algorithm %J journal of Shanghai Jiaotong University, 2006(02): 195-200.
[3] Dou Jianping, Li Jun, Su Chun. Process Sequencing Optimization based on Feasible Process Sequence Genetic Algorithm for %J Computer Integrated Manufacturing System [J], 2019, 25(08): 1981-1990.
[4] Zhai Dehui. Research on Key Technology of 3d Process Design of Box Parts Based on Feature [D]. Beijing Institute of Technology, 2015.
[5] Gao Tianwei. Research on Key Technology of Intelligent Process of Large Marine Engine Cylinder Based on MBD Feature [D]. Chongqing University of Technology, 2020.
[6] Pan Qinggu. Research and Application of Process Planning Method for Typical Automobile Parts Based on PLM Platform [D]. Wuhan University of Technology, 2019.
[7] Guo Y W, Mileham A R, Owen G W, et al. Operation sequencing optimization using a particle swarm optimization approach %J Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture[J], 2006, 220(12).
[8] Particle Swarm Optimization; Reports from North China Electric Power University Provide New Insights into Particle Swarm Optimization (A simulation approach to the process planning problem using a modified particle swarm optimization) %J Journal of Technology &
Science[J], 2016.

[9] Huang Weijun. Research on Key Technology of Process Plan Optimization of Complex Parts [D]. Huazhong University of Science and Technology, 2012.

[10] Li W D, Ong S K, Nee A Y C. Hybrid genetic algorithm and simulated annealing approach for the optimization of process plans for prismatic parts %J International Journal of Production Research[J], 2002, 40(8).

[11] Venkata Rao R. Jaya: A simple and new optimization algorithm for solving constrained and unconstrained optimization problems[J]. International Journal of Industrial Engineering Computations, 2016: 19-34.

[12] Abu Z R, Azmi A M, A A M, et al. An Intensive and Comprehensive Overview of JAYA Algorithm, its Versions and Applications. %J Archives of computational methods in engineering: state of the art reviews[J], 2021.

[13] Yan Zhi, Zhang Peng, Xie Chuan. Research on Bayesian Network Structure Learning Algorithm based on Jaya %J Computer Engineering and Applications, 2019, 55(19): 173-177+184.

[14] Lian Yuxiang, Zhang Chaoyong, Meng Leilei, et al. Flexible Job-shop Scheduling Problem %J Computer Integrated Manufacturing System Based on Improved Jaya Algorithm [J]: 1-19.

[15] Huang W, Hu Y, Cai L. An effective hybrid graph and genetic algorithm approach to process planning optimization for prismatic parts %J The International Journal of Advanced Manufacturing Technology[J], 2012, 62(9-12).

[16] Su Y, Chu X, Chen D, et al. A genetic algorithm for operation sequencing in CAPP using edge selection based encoding strategy %J Journal of Intelligent Manufacturing[J], 2018, 29(2).

[17] Hu Q, Qiao L, Peng G. An ant colony approach to operation sequencing optimization in process planning %J Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture[J], 2017, 231(3).