Journal of Intelligent Manufacturing (2020) 31:1079–1099
https://doi.org/10.1007/s10845-019-01496-7

Hybrid constrained permutation algorithm and genetic algorithm for process planning problem

Abdullah Falih[1] · Ahmed Z. M. Shammari[2]

Received: 17 March 2019 / Accepted: 21 September 2019 / Published online: 14 October 2019

Abstract

In this research, a hybrid constrained permutation algorithm and genetic algorithm approach is proposed to solve the process planning problem and to facilitate the optimisation process. In this approach, the process planning problem is represented as a graph in which operations are clustered corresponding to their machine, tool, and tool access direction similarities. A constrained permutation algorithm (CPA) developed to generate a set of optimised feasible operations sequences based on the principles of minimising the number of setup changes and the number of tool changes. Due to its strong capability in global search through multiple optima, genetic algorithm (GA) is used to search for an optimal or near optimal process plan, in which the population is initialised according to the operations sequences generated by CPA. Furthermore, to prevent premature convergence to local optima, a mixed crossover operator is designed and equipped into GA. Four comparative case studies are carried out to evidence the feasibility and robustness of the proposed CPAGA approach against GA, simulated annealing, tabu search, ant colony optimisation, and particle swarm optimisation based approaches reported in the literature, and the results are promising.

Keywords Process planning · Genetic algorithm · Constrained permutation algorithm · Operation sequencing

Introduction

Computer-aided process planning (CAPP) is substantial linkage between computer-aided design (CAD) and computer-aided manufacturing (CAM). It aims to transform acquired part design specification from CAD into a sequence of machining operations that are used by CAM to manufacture a part economically and competitively. Although much effort has been exerted to evolve CAPP systems, they are still lagging behind in comparison with CAD and CAM systems. One of the reasons CAPP is falling behind is due to the tremendously complex nature of its tasks (Al-wswasi et al. 2018). Generally, CAPP is concerned with maintaining three main tasks, namely, feature recognition, operation selection, and operation sequencing. Feature recognition is the act of recognising the machining features from a CAD file that a part is comprised of. Operation selection is the act of selecting the necessary operations needed to produce a part. Operation sequencing is the act of determining the sequence of operations by which a part should be produced while satisfying the precedence constraints among operations.

It is well-known that operation sequencing is regarded as an NP-hard problem, which is very difficult to solve using conventional techniques. Even though many metaheuristic-based approaches have been proposed to solve the operation sequencing optimisation problem, it is difficult to handle the precedence constraints among operations efficiently. In general, precedence constraints handling methods and strategies can be categorized into additional adjustment strategies and repairing strategies. The additional adjustment strategies, such as edge selection strategy (Su et al. 2018) and intelligent search strategy (Salehi and Bahreininejad 2011), evades infeasible solutions in initialisation. Premature convergence in some complicated precedence constraints cases is considered its main drawback (Su et al. 2018; Li et al. 2004). On the other hand, the repairing strategies, such as a penalty matrix strategy (Liu et al. 2013; Nallakumarasamy

✉ Abdullah Falih
abdullah.ali.falih@hotmail.com

Ahmed Z. M. Shammari
drahmed@kecbu.uobaghdad.edu.iq

1 Mechanical Engineering Department, College of Engineering, University of Baghdad, Baghdad, Iraq

2 AL-Khwarizmi College of Engineering, University of Baghdad, Baghdad, Iraq

et al. 2011) and a topological storing and encoding strategy (Huang et al. 2012; Yun and Moon 2011), tries to rectify the infeasible solutions that conflict with the precedence constraints. Unreliability and low efficiency are its main drawbacks (Yun and Moon 2011; Moon et al. 2002). In this research, a novel constrained permutation algorithm has been established and combined with Genetic algorithm (GA) to solve the operations sequencing and operation selections problems simultaneously. GA effectiveness has been practically demonstrated in solving sophisticated combinatorial optimization problems. As opposite to other meta-heuristics, GA has the ability to direct the search toward relatively promising regions in the problem's search space (Zacharia et al. 2015). The contribution of this research is as follows:

1. A new graph-based representation of the process planning problem has been developed. In which, operations are clustered corresponding to their machine, tool, and tool access direction (TAD) similarities. The graph consists of a set of operations that can be machined in one setup (OSS) nodes, a set of operations that can be machined by one tool in one setup (OTS) nodes, operation (O) nodes, directed arcs, and undirected arcs. Undirected arcs represent the inclusion relationship between OSS, OTS, and O nodes, on the other hand, directed arcs represent precedence constraints between OSS nodes, OTS nodes, and O nodes.
2. A novel constrained permutation algorithm has been established to generate an initial set of operations sequences. First, the algorithm is used to initiate the OSS, OTS, and O nodes. Second, several graphs depending on the part complexity are generated by applying the part precedence constraints among OSS nodes, OTS nodes, and O nodes. Finally, for each graph, a constrained permutation is performed among OSS nodes and OTS nodes, and the operations sequences are acquired by replacing OTS nodes by O nodes.
3. A mixed crossover operator that comprises an order crossover operator and a new designed three strings crossover operator is developed to avoid premature convergence to local optima and to enhance the performance of GA in searching for an optimal or near optimal process plan.

The rest of this article is organised as follows. Section in "Previous related work" gives an overview of previous related work about metaheuristic-based approaches applied to the process planning problem. Section in "Problem modelling" elaborates on the process planning problem representation. Details of the proposed CPAGA approach is presented in "Hybrid CPA and GA Approach" section. Section in "Comparative case studies" illustrates the effectiveness of the proposed CPAGA approach through four comparative case studies. Section in "Conclusions" concludes the article.

Previous related work

In the past two decades, many metaheuristic-based approaches proposed to solve the process planning problem, which can be categorised as genetic algorithm (GA) based approach (Salehi and Bahreininejad 2011; Huang et al. 2012; Zhang et al. 1997; Reddy et al. 1999; Dou et al. 2018b; Salehi and Tavakkoli-Moghaddam 2009; Musharavati and Hamouda 2011; Su et al. 2015), simulated annealing (SA) based approach (Ma et al. 2000; Nallakumarasamy et al. 2011), GA-SA based approach (Li et al. 2002; Huang et al. 2017; Xu et al. 2014) , ant colony optimisation (ACO) based approach (Liu et al. 2013; Krishna and Mallikarjuna Rao 2006; Hu et al. 2017; Wang et al. 2015), particle swarm optimisation (PSO) based approach (Dou et al. 2018a; Petrović et al. 2016; Wang et al. 2012; Guo et al. 2006; Li et al. 2013), and tabu search (TS) based approach (Li et al. 2004).

Zhang et al. (1997) presented a novel GA based approach to handle the operations sequencing and the operation resources selection of the process planning problem simultaneously. Optimal or near optimal process plans were achieved through specially designed crossover and mutation operators. Reddy et al. (1999) used GA as a global search by developing a novel chromosome representation schema for the process plan. In their approach, operation sequences were obtained quickly, and a special crossover operator to maintain them in the chromosomes was well-intended. Salehi and Tavakkoli-Moghaddam (2009) proposed an approach in which the process planning problem was handled in two stages: preliminary and secondary. In the preliminary stage, a set of feasible operation sequences was generated, while in the secondary stage, the manufacturing resources selected for each specific operation were handled. GA was used as an optimisation technique in the two stages. In the same manner, but instead, using GA in the preliminary stage, Salehi and Bahreininejad (2011) developed an intelligent search algorithm to generate a set of feasible operation sequences. The results show better computation time in comparison with the two stages of GA. Su et al. (2015) presented a hybrid approach that incorporated GA with a local search to minimise the machining cost and maximise the utilisation of the machine tools, based on a 0–1 mixed integer model. First, the feasible initial process plans were generated by using an operation precedence graph (OPG), then the hybrid GA and the local search were used to find the optimal process plan.

Moreover, Huang et al. (2012) embedded the graph theory escort with the matrix theory into GA to deal with precedence constraints and generate initial feasible operation sequences. A special crossover operator and two types of mutation operators were developed and a heuristic algorithm to modify the infeasible solutions generated by the mutation was also introduced. Furthermore, a modified GA based on a cyclic crossover operator and a neighbourhood search mutation operator were developed by Musharavati and Hamouda (2011) to obtain a near optimal process plan of multiple parts manufacturing lines. Su et al. (2018) proposed an edge selection strategy to produce feasible operation sequences in the initialisation of GA for the purpose of improving GA convergence. The operation sequences kept up through a special crossover, furthermore, a mutation operator was designed to optimise the resources selection. Dou et al. (2018b) presented an improved genetic algorithm (IGA) to minimise the cost of process plans. The feasible operation sequences were encoded by permutation. Furthermore, fragment crossover and fragment mutations were designed to maintain the operation sequences in the chromosomes. A comparison of IGA against GA, ACO, and PSO was carried out through two case studies, and a better solution was reported.

In regard to SA, Ma et al. (2000) developed SA based approach to find the optimal process plan by exploring the entire solution space. Tool cost, machine cost, tool change cost, machine change cost, and setup change cost were regarded as objective functions. Nallakumarasamy et al. (2011) used the SA algorithm to minimise machine, tool, and setup costs as an objective. A reward penalty matrix and a cost matrix were used to generate the feasible operation sequences. Their experiments showed lesser computational time and better generated operation sequences. In addition to the use of SA, Li et al. (2002) proposed a hybrid GA-SA approach to produce a set of optimal or near optimal process plans, with consideration of a dynamic job shop environment such as the unavailability of machines and tools. The GA was used as a global search, initially to explore better process plans, while SA was used as a local search to find optimal or near optimal process plans. Furthermore, Xu et al. (2014) developed genetic simulated annealing (GSA) algorithm to find a near optimal process plan with a high complexity precedence constraint cylinder block. Huang et al. (2017) proposed a hybrid GA-SA approach, in which a graph search algorithm was embedded and the precedence constraints formulated as an OPG directed graph. GA with a stochastic topologic sort algorithm was used as a global search to generate the initial feasible operation sequences, and then SA was used as a local search to find the optimal process plan.

With respect to ACO, Krishna and Mallikarjuna Rao (2006) modelled the process planning problem as a constrained travelling salesman problem (TSP) and adopted ACO for the first time to solve it. An ACO algorithm was used as a global search for finding the optimal operations sequence by considering various feasibility constraints. In a similar manner, Liu et al. (2013) used an ACO algorithm to search for the lowest cost sequence of operations. They embedded a constraint matrix and a state matrix into the ACO algorithm to show the operations state. Two study cases were investigated to show the advantage of this approach over GA, SA, and TS, and the results were promising. Hu et al. (2017) developed a novel modified ACO to solve the combinatorial optimisation problem of operation sequencing. An adaptive updating method and a local search mechanism are embedded into ACO to enhance the global search capability and to avoid the local optima. A comparison with GA, SA, TS and PSO was carried out to ensure feasibility and robustness, as well as to show the advantages of their proposed approach. In addition, Wang et al. (2015) proposed two-stage ACO algorithms (TSACO) to find the minimum total production cost. The process planning problem was modelled as a directed AND/OR graph, and one of the ACOs was used to optimise the nodes in the first stage, while the other was used to optimise the weighted arcs in the second stage.

In regard to PSO, Guo et al. (2006) developed a PSO algorithm to solve the combinatorial optimisation problem of operation sequencing. New designed crossover, mutation and shift operators were employed to avoid the local optima, and the result was satisfactory in comparison with GA and SA. Wang et al. (2012) developed a PSO with a local search strategy to avoid any unnecessary convergence that may occur in early generations. A solution scheme was offered to decode the discrete nature of the process planning problem in order to remedy the obstacle that PSO is designed for the continuous optimisation problem. Petrović et al. (2016) introduced chaos theory into PSO to prevent premature convergence and to enlarge the solution space of the AND/OR process plans network. Their approach was extensively verified through a comparison with GA, SA, GA-SA, and PSO in four experimental studies. Dou et al. (2018a) presented a novel feasible sequence discrete PSO (FSDPSO) to search for feasible operation sequences. Updating mechanism crossover, fragment mutation, and uniform greedy mutations were developed to enhance FSDPSO performance. Finally, the results showed an advantage of their approach against GA and two-stage ACO. Furthermore, efficient encoding, updating, and random search methods were developed and incorporated into PSO by Li et al. (2013). A comparison of their approach with GA and SA in seven cases was reported, and the results were satisfactory.

In addition to GA, SA, ACO, and PSO based approaches, Wen et al. (2014) proposed honey bees mating optimisation (HBMO) for the first time to solve the process planning problem. Three experiments were reported and HBMO showed better performance in comparison with well-known metaheuristic algorithms. Li et al. (2004) modelled the process planning problem as a constraint-based optimisation problem, in which the resources selection and operation sequences were treated simultaneously. A hybrid constraint-handling method is developed and embedded in a TS algorithm to handle the operation sequencing problem and conduct the search efficiently in a large-sized constraint-based space. Lian et al. (2012) utilised a novel metaheuristic-inspired imperialist competitive algorithm (ICA) to obtain a near optimal process plan. The process planning problem was modelled considering various flexibilities, i.e., machine, tool, TAD, process, and sequence flexibilities. The proposed algorithm emphasises its efficiency in comparison with other metaheuristic algorithms. Ding et al. (2005) developed a hybrid approach that incorporates an artificial neural network (ANN), an analytical hierarchical process (AHP), and GA for optimising operation sequences. A multi-objective function was established that incorporates minimising manufacturing time and cost to evaluate candidates' process plans.

Through the above analysis, it could be deduced that each approach has its own advantages and disadvantages. Nevertheless, two issues are still ongoing and necessitate more attention. The first issue is that the flexibility of machines, tools, and TADs produces a large solution space for the process planning problem. The second issue is the precedence constraints handling strategies. Repairing strategies are unreliable, inefficient, and time consuming since they try to modify infeasible solutions after the process of generating them. In comparison to repairing strategies, additional adjustment strategies try to initiate a feasible solution before the process of searching for an optimal or near optimal one. In addition to suffering from the premature convergence problem, the initiated solutions are feasible, but they are not the best in the solution space, which could result in much time spent searching for the optimal or near optimal process plan. Much effort is required to design a model for the process planning problem that minifies the initial solution space to be restricted to good solution candidates only. Furthermore, there is a critical need for a new method to handle precedence constraints more effectively and efficiently.

Problem modelling

In CAPP, a process plan is described as a set of machining features that can be machined in a specific features sequence using specific manufacturing resources. First, machining features are recognised by the topological and geometrical information gained from a CAD file, such as dimension, surface finish, and position. Then an operation (O) or several operations are assigned to each machining feature, depending on its information. An operation is comprised of a set of candidate operations (CO), which is the use of a specific tool (T) on a specific machine (M) with a specific tool access direction (TAD) to produce a feature or sub-feature. Therefore, a process plan can be presented as follows:

$$PP = \{O_1, O_2, \ldots, O_i\} \tag{1}$$

where PP is the process plan, O_i is the ith operation of the process plan, which is defined as:

$$O_i = \{CO_{i,M,T,TAD}\} \tag{2}$$

Where $CO_{i,M,T,TAD}$ is the Mth, Tth, and TADth alternative operation of the ith operation of the process plan. M, T, and TAD are the indices of the machine, tool, and TAD, respectively, by which CO_i is formed by its machine flexibility list (M []), tool flexibility list (T []), and TAD flexibility list (TAD []). on other hand, a setup defined as a set of operation that can be machined continuously on same machine with the same TAD (Li et al. 2004). That's it:

$$PP = \{OSS_1, OSS_2, \ldots, OSS_s\} \tag{3}$$

Where OSS_s is the sth set of operations of the process plan that can be machined in one setup, s is the index of setup that indicate M and TAD similarities of operations. OSS can be defined as:

$$OSS = \{CO_{1,s}, CO_{2,s}, \ldots, CO_{i,s}\} \tag{4}$$

Furthermore, within OSS there are operations that may be machined using same tool, therefore:

$$OSS = \{OTS_1, OTS_2, \ldots, OTS_T\} \tag{5}$$

Where OTS is the Tth set of operations of OSS that can be machined using one tool which is defined as:

$$OTS = \{O_{1,s,T}, O_{2,s,T}, \ldots, O_{i,s,T}\} \tag{6}$$

Finally, according Eq. (3), Eq. (5), and Eq. (6) a process plan can be represented as follows:

$$PP = \{OSS_s(OTS_T(O_i))\} \tag{7}$$

In order to implement the proposed approach, the process planning problem has to be represented as a graph, as shown in Fig. 1. The graph is denoted as $G = (N, R)$, where N is a set of OSS, OTS, and O nodes and R is a set of directed and undirected arcs.Undirected arcs represent the inclusion relationships between OSS, OTS, and O nodes. For example, OSS_1 is a parent of OTS_1 and OTS_2, OTS_1 is a parent

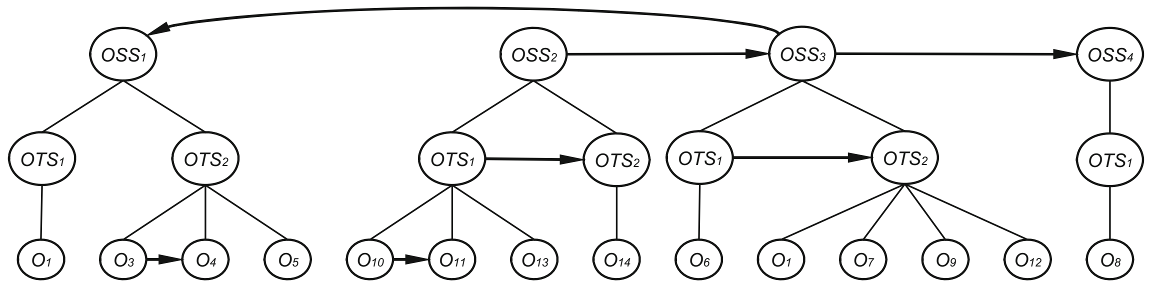


Fig. 1 Graphical representation of the process planning problem

Table 1 Production rules for OSS and OTS construction and precedence constraints analysis

Rule	Description	
Rule 1	IF	There is an operation with more one TAD in its TAD [ ]
	AND	One of the TADs doesn't match any one TAD operations
	THEN	Remove that TAD from the operation TAD [ ]
Rule 2	IF	There are operations with one TAD in its TAD [ ]
	AND	These TADs similar
	AND	There is a similar M in its M [ ]
	THEN	Create an OSS for these operations
Rule 3	IF	There is an operation with one TAD in its TAD [ ]
	THEN	Create an OSS for this operation
Rule 4	IF	There is an operation with more than one TAD in its TAD flexibility list
	AND	These TADs match the TAD of the OSSs constructed in Rule 2 and Rule 3
	THEN	Permute the operation into OSSs, and create permuted OSS (POSS) sets
Rule 5	IF	There are operations within OSS with one similar tool in its T [ ]
	AND	There is a similar machine in their M [ ]
	THEN	Create an OTS for these operations
Rule 6	IF	There is a bidirectional precedence constraint between any OSS nodes
	AND	One the operation that cause the bidirectional precedence constraint between the OSS permuted by Rule 4
	THEN	Remove that POSS set
	ELSE	Decompose OSS with minimum number of conflicting operations, and create a new OSS of these conflicting operations
Rule 7	IF	There is a bidirectional precedence constraint between two OTS nodes
	THEN	Decompose OTS with minimum number of conflicting operations, and create a new OTS of these conflicting operations
Rule 8	IF	A node is a predecessor of two other nodes and there is a precedence constraint between these two nodes
	THEN	Remove the precedence constraint between the node that are successor of the two nodes and the node that are predecessor of the two nodes

of O_1, and OTS_2 is a parent of O_3, O_4, and O_5, and so on. On the other hand, directed arcs represent the precedence constraints between OSS nodes, OTS nodes, and O nodes. For example, the directed arc between O_3 and O_4 represents a part precedence constraint between these two operations, and the directed arc between OTS_1 and OTS_2 of OSS_2 represents a part precedence constraint between one (or more) of the OTS_1 child nodes and O_{14}, a child of OTS_2.

Knowledge-based production rules established to construct OSS and OTS nodes and to analyse the precedence constraints are placed in Table 1. Rules 1 to 4 are used to maintain different cases that may occur in OSS nodes construction, while Rule 5 is used to construct the OTS nodes. Regarding the precedence constraints analysis, Rules 6 and 7 are used to handle the problem of bidirectional precedence constraints that may occur when generalising the part precedence constraints from operation level into OTS or OSS

level. Finally, Rule 8 is used to handle complex precedence constraints between the OSS nodes or OTS nodes.

Hybrid CPA and GA approach

GA starts with an initial set of random solutions called population (chromosomes). Each chromosome in the population representing a solution to the problem that satisfies the problem constraints. During each iterative procedure, the chromosomes are evaluated according to the problem objective or objectives. The best chromosomes are selected from the current population to form the population of the next generation by the crossover process. A mutation process is also performed on the chromosomes to prevent premature convergence through slight deformation that is to be conducted of the chromosome structure (Zacharia et al. 2018). Generally, the GA after number of iterations converges to the best chromosome, which represents the best solution to the problem. The proposed approach is comprised of CPA and GA As shown in Fig. 2. First, CPA is used to reduce the solution space through generating an optimised feasible operation sequence based on the part database and the principles of minimising the number of machine changes, the number of setup changes, and the number of tool changes. Then GA is used to optimise the manufacturing resources for these generated sequences, with the objective of finding the optimal process plan.

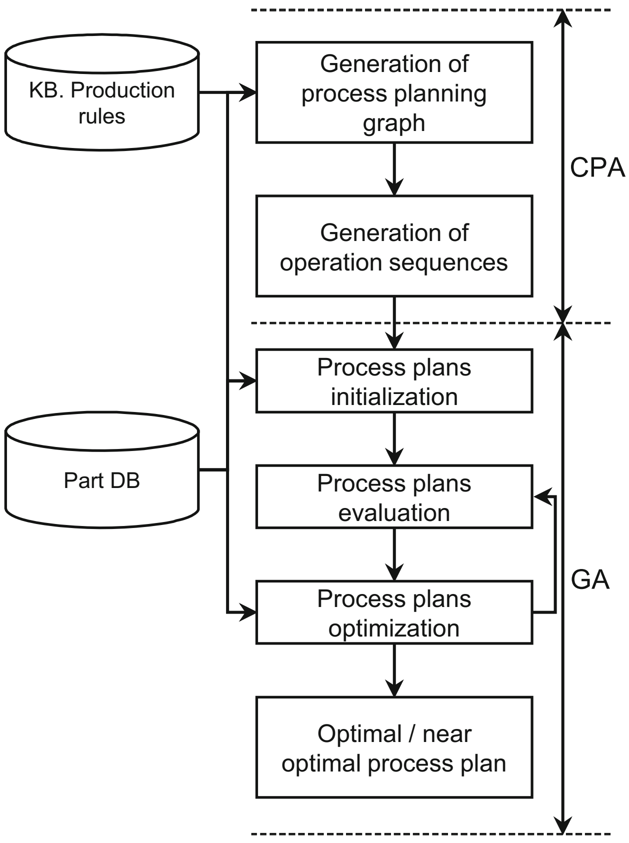

KB. Production rules
Part DB
Generation of process planning graph
Generation of operation sequences
Process plans initialization
Process plans evaluation
Process plans optimization
Optimal / near optimal process plan
CPA
GA

Fig. 2 Block diagram of the proposed CPAGA

Constrained permutation algorithm (CPA)

In CPA, the production rules, established "Problem modelling" in section were used to construct OSS and OTS nodes and to analyse the precedence constraints between them. Furthermore, constrained permutation was performed for both OSS and OTS nodes to explore all the operation sequences alternatives. The idea is to generate an OSS sequence ($OSSS$), then each OSS node in $OSSS$ is replaced by OTS sequences ($OTSS$). Finally, by replacing the OTS nodes by CO, the operation sequences are obtained. The CPA algorithm steps are as follows:

Step 1: Initialize OSS nodes using Rule 1, Rule 2, Rule 3, and Rule 4 respectively, and generate a $POSS$.
Step 2: For $POSS_i$, $(i = 1, \ldots, n)$, if $i > n$ stop, otherwise go to Step 3.
Step 3: Apply the part precedence constraints between OSS nodes.
Step 4: Apply Rule 6, if Rule 6 two conditions achieved go to Step 2 otherwise go to Step 3.
Step 5: Apply Rule 5 to create OTS nodes, and create OTS node for each operation that doesn't follow Rule 5.
Step 6: Apply the part design precedence constraints between OTS nodes.
Step 7: Apply Rule 7, and go to Step 6.
Step 8: Apply Rule 8.
Step 9: Initialize

Step 9.1: $G(OSS, R)$ graph.
Step 9.2: Computation matrix $A_{ij} = [OSS_s, s]$, (OSS_s is the vector of OSS nodes in $G(OSS, R)$, s is the number of OSS nodes).
Step 9.3: Permitted nodes vector OSS_p (OSS_p is the set of OSS predecessors' nodes in $G(OSS, R)$).

Step 10: Generate OSS sequences ($OSSS_k$) in co-factor expression, $OSSS_k = OSS_p + A_{ij}$, for each matched node between OSS_p and the first row of A_{ij}.
Step 11: For $OSSS_k$, update $G(OSS, R)$ by removing the matched nodes between OSS_p and A_{ij}, and update OSS_p and A_{ij}.
Step 12: Set $OSSS_k = OSSS_k + OSS_p$.
Step 13: If A_{ij} is empty store $OSSS_k$ and go to Step 14, otherwise go to Step 11.
Step 14: For OSS_i, $(i = 1, \ldots, s)$, if $i > s$ go to Step 22.
Step 15: Initialize

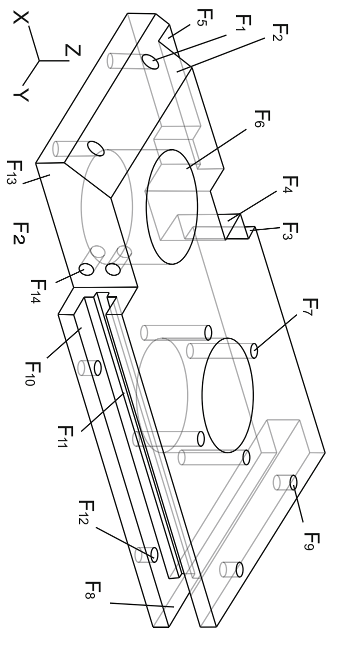


Fig. 3 Part 1 with 14 manufacturing feature (Li et al. 2004)

Step 15.1: $G(OTS, R)$ graph.
Step 15.2: Computation matrix $B_{ij} = [OTS_T, T]$, (OTS_T is the vector of OTS nodes in $G(OTS, R)$), T is the number of OTS nodes).
Step 15.3: Permitted nodes vector OTS_p (OTS_p is the set of OTS predecessors' nodes in $G(OTS, R)$)

Step 16: Generate OTS sequences ($OTSS_h$) in co-factor expression, $OTSS_h = OTS_p + B_{ij}$, for each matched node between OTS_p and the first row of B_{ij}.
Step 17: For $OTSS_h$, update$G(OTS, R)$ by removing the matched nodes between OTS_p and B_{ij}, and update OTS_p.
Step 18: Set $OTSS_h = OTSS_h + OTS_p$.
Step 19: If B_{ij} is empty go to Step 20, otherwise go to Step 17.
Step 20: Update $OSSS_k$ via permutation of $OTSS_h$ into OSS_i of $OSSS_k$.
Step 21: Clear $OTSS_h$, and go to Step 14.
Step 22: Replace OTS nodes in $OSSS_k$ by its comprising operations randomly with consideration of precedence constraints.
Step 23: Generate operation sequences (OS_i), set $OS_i = OS_i + OSSS_k$, clear $OSSS_k$, and go to 2.

The prismatic Part 1 in Fig. 3 reported by Li et al. (2004) is used as an example to clarify some CPA steps. The part manufacturing information and the precedence constraints are listed in Table 2. The part was reported with two types of precedence constraints: hard and soft. Hard constraints influence the manufacturing feasibility, and a process plan should be consistent with these constraints. On the other hand, soft constraints influence only the cost, quality, or efficiency of a process plan. To attain the lowest production cost, soft constraints can be violated in case they conflict with hard constraints (Li et al. 2004). In Part 1, the soft constraints 8 $\rightarrow$ 9 and 10 $\rightarrow$ 12 conflict with the hard constraints 9 $\rightarrow$ 8 and 12 $\rightarrow$ 10 respectively, and, since soft constraints can be violated, four possibilities occur for applying soft con-

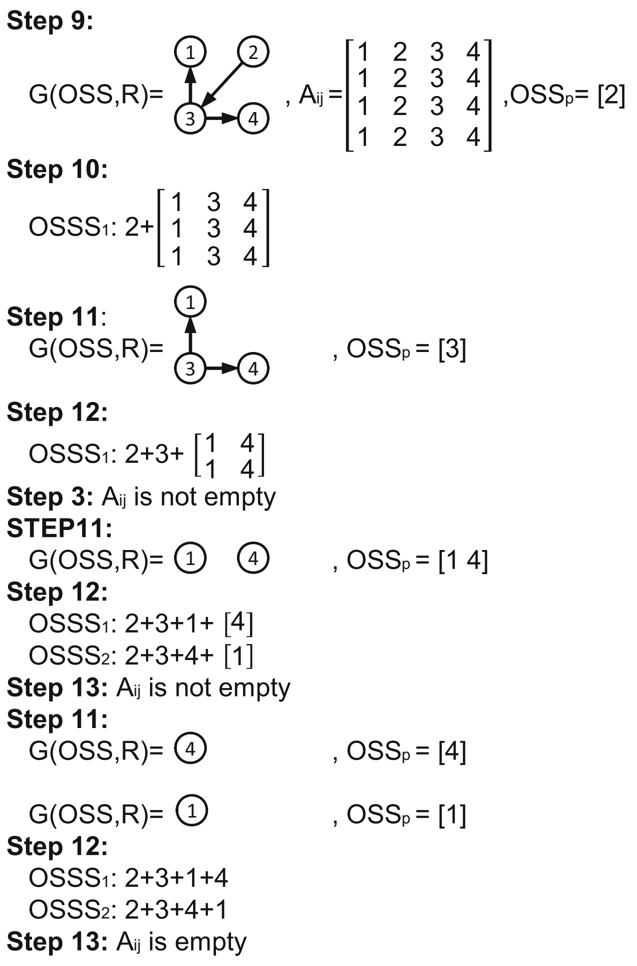
Table 2 Features, operations, manufacturing resources flexibilities, and precedence constraints of Part 1 (Li et al. 2004)

Features	Feature describtions	O_i	TAD []	M []	T []	Prior O_i
F1	Two holes arranged as a replicated feature	Drilling (O_1)	+ z, − z	M01, M02, M03	T01	
F2	A chamfer	Milling (O_2)	x, + y, − y, − z	M02, M03	T08	O_1(H)
F3	A slot	Milling (O_3)	+ y	M02, M03	T05, T06	
F4	A slot	Milling (O_4)	+ y	M02	T05, T06	O_3(S)
F5	A step	Milling (O_5)	+ y, − z	M02, M03	T05, T06	
F6	Two holes arranged as a replicated feature	Drilling (O_6)	+ z, − z	M01, M02, M03	T02	
F7	Four holes arranged as a replicated feature	Reaming (O_7)	+ z, − z	M01, M02, M03	T01	O_6(H)
F8	A slot	Boring (O_8)	+ x	M02, M03	T05, T06	O_9(S)
F9	Two holes arranged as a replicated feature	Drilling (O_9)	− z	M01, M02, M03	T01	O_8(S)
F10	A slot	Drilling (O_{10})	− y	M02, M03	T05, T06	O_{12}(S)
F11	A slot	Reaming (O_{11})	− y	M02, M03	T05, T07	O_{10}(H)
F12	Two holes arranged as a replicate feature	Boring (O_{12})	+ z, − z	M01, M02, M03	T01	O_{10}(S)
F13	A step	Milling (O_{13})	− y, − x	M02, M03	T05, T06	
F14	Two holes arranged as a replicate feature	Drilling (O_{14})	− y	M01, M02, M03	T01	O_{13}(H)/ (S)

H hard constraints, *S* soft constraints

Table 3 POSS of Part 1 generated by Step 1 of CPA

POSS	OSS	TAD	O
$POSS_1$	OSS_1	+ y	O_3, O_4, O_5, O_2
	OSS_2	− y	$O_{10}, O_{11}, O_{13}, O_{14}$
	OSS_3	− z	$O_1, O_6, O_7, O_9, O_{12}$
	OSS_4	+ x	O_8
$POSS_2$	OSS_1	+ y	O_3, O_4, O_5
	OSS_2	− y	$O_{10}, O_{11}, O_{13}, O_{14}, O_2$
	OSS_3	− z	$O_1, O_6, O_7, O_9, O_{12}$
	OSS_4	+ x	O_8
$POSS_3$	OSS_1	+ y	O_3, O_4, O_5
	OSS_2	− y	$O_{10}, O_{11}, O_{13}, O_{14}$
	OSS_3	− z	$O_1, O_6, O_7, O_9, O_{12}, O_2$
	OSS_4	+ x	O_8
$POSS_4$	OSS_1	+ y	O_3, O_4, O_2
	OSS_2	− y	$O_{10}, O_{11}, O_{13}, O_{14}$
	OSS_3	− z	$O_1, O_6, O_7, O_9, O_{12}, O_5$
	OSS_4	+ x	O_8
$POSS_5$	OSS_1	+ y	O_3, O_4
	OSS_2	− y	$O_{10}, O_{11}, O_{13}, O_{14}, O_2$
	OSS_3	− z	$O_1, O_6, O_7, O_9, O_{12}, O_5$
	OSS_4	+ x	O_8
$POSS_6$	OSS_1	+ y	O_3, O_4
	OSS_2	− y	$O_{10}, O_{11}, O_{13}, O_{14}$
	OSS_3	− z	$O_1, O_6, O_7, O_9, O_{12}, O_5, O_2$
	OSS_4	+ x	O_8



Step 9:

G(OSS,R)= $$\begin{tikzcd} 1 & 2 \arrow[dl] \\ 3 \arrow[u] \arrow[r] & 4 \end{tikzcd}$$, $A_{ij}=\begin{bmatrix}1&2&3&4\\1&2&3&4\\1&2&3&4\\1&2&3&4\end{bmatrix}$,OSS_p= [2]

Step 10:

$OSSS_1$: 2+$\begin{bmatrix}1&3&4\\1&3&4\\1&3&4\end{bmatrix}$

Step 11:

G(OSS,R)= $$\begin{tikzcd} 1 & \\ 3 \arrow[u] \arrow[r] & 4 \end{tikzcd}$$, OSS_p = [3]

Step 12:

$OSSS_1$: 2+3+ $\begin{bmatrix}1&4\\1&4\end{bmatrix}$

Step 3: A_{ij} is not empty

STEP11:

G(OSS,R)= ① ④ , OSS_p = [1 4]

Step 12:

$OSSS_1$: 2+3+1+ [4]

$OSSS_2$: 2+3+4+ [1]

Step 13: A_{ij} is not empty

Step 11:

G(OSS,R)= ④ , OSS_p = [4]

G(OSS,R)= ① , OSS_p = [1]

Step 12:

$OSSS_1$: 2+3+1+4

$OSSS_2$: 2+3+4+1

Step 13: A_{ij} is empty

Fig. 4 The constrained permutation illustration of $POSS_1$

straints to each other. The possibilities for applying the soft constraints are 9 → 8 and 10 → 12, 8 → 9 and 10 → 12, 9 → 8 and 12 → 10, or 8 → 9 and 12 → 10. The clarification of CPA confines the first possibility that implies applying constraints 9 → 8 and 10 → 12 and neglecting constraints 8 → 9 and 12 → 10.

In Step 1, Rule 1 removes TADs + z and − x from the TAD [] of operations O_1, O_6, O_7, and O_{12} and the operations O_2, O_{13} respectively. Rule 2 constructs the operation setup sets OSS_1, OSS_2, and OSS_3 while OSS_4 is constructed by Rule 3. Two operations did not follow Rules 2 and 3 (O_2 and O_3) because the TADs in their TAD [] already exist in the constructed OSS nodes. Therefore, Rule 4 generates a permuted OSS set ($POSS$) by the permutation of O_2 into OSS_1, OSS_2, and OSS_3, and the permutation of O_5 into OSS_1 and OSS_3. The result is six POSS, as shown in Table 3.

In regard to $POSS_1$, Fig. 1 represents the alternative process plans graph by applying Step 3 to 8. As shown in Fig. 1, some of the part precedence constraints between operations are generalised to be between OTS nodes or OSS nodes, for example, the precedence constraints between OSS_3 and OSS_1 caused by the part precedence constraints between O_1 and O_2. Furthermore, a constrained permutation of $POSS_1$ to generate $OSSS$ is performed through Step 9 to 13, as shown in Fig. 4. The same procedure is repeated on OTS nodes through Step 15 to 19 to generate $OTSS$. Finally, four operation sequences, as shown in Table 4, are obtained by the permutation of $OTSS$ into the OSS of $OSSS$ and replacing the OTS nodes by their comprising O operations. As shown in Table 4, CPA generates 36 operation sequences. $POSS_2$ and $POSS_5$ are neglected because the two conditions of Rule 6 are achieved. The reason is that O_2, which caused the bidirectional constraints between OSS_2 and OSS_3, has already permuted into other OSS nodes, so generating additional OSS is not considered worthwhile, since it will increase the number of setups.

Process plans initialization

The chromosome presented as four strings of integers in which each gen indicates a $O_{i,M,T,TAD}$. Usually, the population initialised randomly in GA, in CPAGA, the population size and the i string are obtained by CPA, and the M string,

Table 4 The generated operations sequences of the first possibility of Part 1

POSS	Operations sequences
$POSS_1$	$O_{10} \rightarrow O_{11} \rightarrow O_{13} \rightarrow O_{14} \rightarrow O_6 \rightarrow O_1 \rightarrow O_7 \rightarrow O_9 \rightarrow O_{12} \rightarrow O_3 \rightarrow O_4 \rightarrow O_5 \rightarrow O_2 \rightarrow O_8$
	$O_{10} \rightarrow O_{11} \rightarrow O_{13} \rightarrow O_{14} \rightarrow O_6 \rightarrow O_1 \rightarrow O_7 \rightarrow O_9 \rightarrow O_{12} \rightarrow O_8 \rightarrow O_3 \rightarrow O_4 \rightarrow O_5 \rightarrow O_2$
	$O_{10} \rightarrow O_{11} \rightarrow O_{13} \rightarrow O_{14} \rightarrow O_6 \rightarrow O_1 \rightarrow O_7 \rightarrow O_9 \rightarrow O_{12} \rightarrow O_2 \rightarrow O_3 \rightarrow O_4 \rightarrow O_5 \rightarrow O_8$
	$O_{10} \rightarrow O_{11} \rightarrow O_{13} \rightarrow O_{14} \rightarrow O_6 \rightarrow O_1 \rightarrow O_7 \rightarrow O_9 \rightarrow O_{12} \rightarrow O_8 \rightarrow O_2 \rightarrow O_3 \rightarrow O_4 \rightarrow O_5$
$POSS_3$	$O_3 \rightarrow O_4 \rightarrow O_5 \rightarrow O_{10} \rightarrow O_{11} \rightarrow O_{13} \rightarrow O_{14} \rightarrow O_6 \rightarrow O_1 \rightarrow O_7 \rightarrow O_9 \rightarrow O_{12} \rightarrow O_2 \rightarrow O_8$
	$O_{10} \rightarrow O_{11} \rightarrow O_{13} \rightarrow O_{14} \rightarrow O_3 \rightarrow O_4 \rightarrow O_5 \rightarrow O_6 \rightarrow O_1 \rightarrow O_7 \rightarrow O_9 \rightarrow O_{12} \rightarrow O_2 \rightarrow O_8$
	$O_{10} \rightarrow O_{11} \rightarrow O_{13} \rightarrow O_{14} \rightarrow O_6 \rightarrow O_1 \rightarrow O_7 \rightarrow O_9 \rightarrow O_{12} \rightarrow O_2 \rightarrow O_3 \rightarrow O_4 \rightarrow O_5 \rightarrow O_8$
	$O_{10} \rightarrow O_{11} \rightarrow O_{13} \rightarrow O_{14} \rightarrow O_6 \rightarrow O_1 \rightarrow O_7 \rightarrow O_9 \rightarrow O_{12} \rightarrow O_2 \rightarrow O_8 \rightarrow O_3 \rightarrow O_4 \rightarrow O_5$
$POSS_4$	$O_{13} \rightarrow O_{10} \rightarrow O_{11} \rightarrow O_{14} \rightarrow O_6 \rightarrow O_1 \rightarrow O_7 \rightarrow O_9 \rightarrow O_{12} \rightarrow O_5 \rightarrow O_3 \rightarrow O_4 \rightarrow O_2 \rightarrow O_8$
	$O_{13} \rightarrow O_{10} \rightarrow O_{11} \rightarrow O_{14} \rightarrow O_6 \rightarrow O_1 \rightarrow O_7 \rightarrow O_9 \rightarrow O_{12} \rightarrow O_5 \rightarrow O_8 \rightarrow O_3 \rightarrow O_4 \rightarrow O_2$
	$O_{13} \rightarrow O_{10} \rightarrow O_{11} \rightarrow O_{14} \rightarrow O_6 \rightarrow O_5 \rightarrow O_1 \rightarrow O_7 \rightarrow O_9 \rightarrow O_{12} \rightarrow O_3 \rightarrow O_4 \rightarrow O_2 \rightarrow O_8$
	$O_{13} \rightarrow O_{10} \rightarrow O_{11} \rightarrow O_{14} \rightarrow O_6 \rightarrow O_5 \rightarrow O_1 \rightarrow O_7 \rightarrow O_9 \rightarrow O_{12} \rightarrow O_8 \rightarrow O_3 \rightarrow O_4 \rightarrow O_2$
	$O_{13} \rightarrow O_{10} \rightarrow O_{11} \rightarrow O_{14} \rightarrow O_5 \rightarrow O_6 \rightarrow O_1 \rightarrow O_7 \rightarrow O_9 \rightarrow O_{12} \rightarrow O_3 \rightarrow O_4 \rightarrow O_2 \rightarrow O_8$
	$O_{13} \rightarrow O_{10} \rightarrow O_{11} \rightarrow O_{14} \rightarrow O_5 \rightarrow O_6 \rightarrow O_1 \rightarrow O_7 \rightarrow O_9 \rightarrow O_{12} \rightarrow O_8 \rightarrow O_3 \rightarrow O_4 \rightarrow O_2$
	$O_{13} \rightarrow O_{10} \rightarrow O_{11} \rightarrow O_{14} \rightarrow O_6 \rightarrow O_1 \rightarrow O_7 \rightarrow O_9 \rightarrow O_{12} \rightarrow O_5 \rightarrow O_2 \rightarrow O_3 \rightarrow O_4 \rightarrow O_8$
	$O_{13} \rightarrow O_{10} \rightarrow O_{11} \rightarrow O_{14} \rightarrow O_6 \rightarrow O_1 \rightarrow O_7 \rightarrow O_9 \rightarrow O_{12} \rightarrow O_5 \rightarrow O_8 \rightarrow O_2 \rightarrow O_3 \rightarrow O_4$
	$O_{13} \rightarrow O_{10} \rightarrow O_{11} \rightarrow O_{14} \rightarrow O_6 \rightarrow O_5 \rightarrow O_1 \rightarrow O_7 \rightarrow O_9 \rightarrow O_{12} \rightarrow O_2 \rightarrow O_3 \rightarrow O_4 \rightarrow O_8$
	$O_{13} \rightarrow O_{10} \rightarrow O_{11} \rightarrow O_{14} \rightarrow O_6 \rightarrow O_5 \rightarrow O_1 \rightarrow O_7 \rightarrow O_9 \rightarrow O_{12} \rightarrow O_8 \rightarrow O_2 \rightarrow O_3 \rightarrow O_4$
	$O_{13} \rightarrow O_{10} \rightarrow O_{11} \rightarrow O_{14} \rightarrow O_5 \rightarrow O_6 \rightarrow O_1 \rightarrow O_7 \rightarrow O_9 \rightarrow O_{12} \rightarrow O_2 \rightarrow O_3 \rightarrow O_4 \rightarrow O_8$
	$O_{13} \rightarrow O_{10} \rightarrow O_{11} \rightarrow O_{14} \rightarrow O_5 \rightarrow O_6 \rightarrow O_1 \rightarrow O_7 \rightarrow O_9 \rightarrow O_{12} \rightarrow O_8 \rightarrow O_2 \rightarrow O_3 \rightarrow O_4$
$POSS_6$	$O_3 \rightarrow O_4 \rightarrow O_{13} \rightarrow O_{10} \rightarrow O_{11} \rightarrow O_{14} \rightarrow O_6 \rightarrow O_7 \rightarrow O_9 \rightarrow O_{12} \rightarrow O_1 \rightarrow O_2 \rightarrow O_5 \rightarrow O_8$
	$O_{13} \rightarrow O_{10} \rightarrow O_{11} \rightarrow O_{14} \rightarrow O_3 \rightarrow O_4 \rightarrow O_6 \rightarrow O_7 \rightarrow O_9 \rightarrow O_{12} \rightarrow O_1 \rightarrow O_2 \rightarrow O_5 \rightarrow O_8$
	$O_{13} \rightarrow O_{10} \rightarrow O_{11} \rightarrow O_{14} \rightarrow O_6 \rightarrow O_7 \rightarrow O_9 \rightarrow O_{12} \rightarrow O_1 \rightarrow O_2 \rightarrow O_5 \rightarrow O_3 \rightarrow O_4 \rightarrow O_8$
	$O_{13} \rightarrow O_{10} \rightarrow O_{11} \rightarrow O_{14} \rightarrow O_6 \rightarrow O_7 \rightarrow O_9 \rightarrow O_{12} \rightarrow O_1 \rightarrow O_2 \rightarrow O_5 \rightarrow O_8 \rightarrow O_3 \rightarrow O_4$
	$O_3 \rightarrow O_4 \rightarrow O_{13} \rightarrow O_{10} \rightarrow O_{11} \rightarrow O_{14} \rightarrow O_6 \rightarrow O_7 \rightarrow O_9 \rightarrow O_{12} \rightarrow O_1 \rightarrow O_5 \rightarrow O_2 \rightarrow O_8$
	$O_{13} \rightarrow O_{10} \rightarrow O_{11} \rightarrow O_{14} \rightarrow O_3 \rightarrow O_4 \rightarrow O_6 \rightarrow O_7 \rightarrow O_9 \rightarrow O_{12} \rightarrow O_1 \rightarrow O_5 \rightarrow O_2 \rightarrow O_8$
	$O_{13} \rightarrow O_{10} \rightarrow O_{11} \rightarrow O_{14} \rightarrow O_6 \rightarrow O_7 \rightarrow O_9 \rightarrow O_{12} \rightarrow O_1 \rightarrow O_5 \rightarrow O_2 \rightarrow O_3 \rightarrow O_4 \rightarrow O_8$
	$O_{13} \rightarrow O_{10} \rightarrow O_{11} \rightarrow O_{14} \rightarrow O_6 \rightarrow O_7 \rightarrow O_9 \rightarrow O_{12} \rightarrow O_1 \rightarrow O_5 \rightarrow O_2 \rightarrow O_8 \rightarrow O_3 \rightarrow O_4$
	$O_3 \rightarrow O_4 \rightarrow O_{13} \rightarrow O_{10} \rightarrow O_{11} \rightarrow O_{14} \rightarrow O_6 \rightarrow O_5 \rightarrow O_7 \rightarrow O_9 \rightarrow O_{12} \rightarrow O_1 \rightarrow O_2 \rightarrow O_8$
	$O_{13} \rightarrow O_{10} \rightarrow O_{11} \rightarrow O_{14} \rightarrow O_3 \rightarrow O_4 \rightarrow O_6 \rightarrow O_5 \rightarrow O_7 \rightarrow O_9 \rightarrow O_{12} \rightarrow O_1 \rightarrow O_2 \rightarrow O_8$
	$O_{13} \rightarrow O_{10} \rightarrow O_{11} \rightarrow O_{14} \rightarrow O_6 \rightarrow O_5 \rightarrow O_7 \rightarrow O_9 \rightarrow O_{12} \rightarrow O_1 \rightarrow O_2 \rightarrow O_3 \rightarrow O_4 \rightarrow O_8$
	$O_{13} \rightarrow O_{10} \rightarrow O_{11} \rightarrow O_{14} \rightarrow O_6 \rightarrow O_5 \rightarrow O_7 \rightarrow O_9 \rightarrow O_{12} \rightarrow O_1 \rightarrow O_2 \rightarrow O_8 \rightarrow O_3 \rightarrow O_4$
	$O_3 \rightarrow O_4 \rightarrow O_{13} \rightarrow O_{10} \rightarrow O_{11} \rightarrow O_{14} \rightarrow O_5 \rightarrow O_6 \rightarrow O_7 \rightarrow O_9 \rightarrow O_{12} \rightarrow O_1 \rightarrow O_2 \rightarrow O_8$
	$O_{13} \rightarrow O_{10} \rightarrow O_{11} \rightarrow O_{14} \rightarrow O_3 \rightarrow O_4 \rightarrow O_5 \rightarrow O_6 \rightarrow O_7 \rightarrow O_9 \rightarrow O_{12} \rightarrow O_1 \rightarrow O_2 \rightarrow O_8$
	$O_{13} \rightarrow O_{10} \rightarrow O_{11} \rightarrow O_{14} \rightarrow O_5 \rightarrow O_6 \rightarrow O_7 \rightarrow O_9 \rightarrow O_{12} \rightarrow O_1 \rightarrow O_2 \rightarrow O_3 \rightarrow O_4 \rightarrow O_8$
	$O_{13} \rightarrow O_{10} \rightarrow O_{11} \rightarrow O_{14} \rightarrow O_5 \rightarrow O_6 \rightarrow O_7 \rightarrow O_9 \rightarrow O_{12} \rightarrow O_1 \rightarrow O_2 \rightarrow O_8 \rightarrow O_3 \rightarrow O_4$

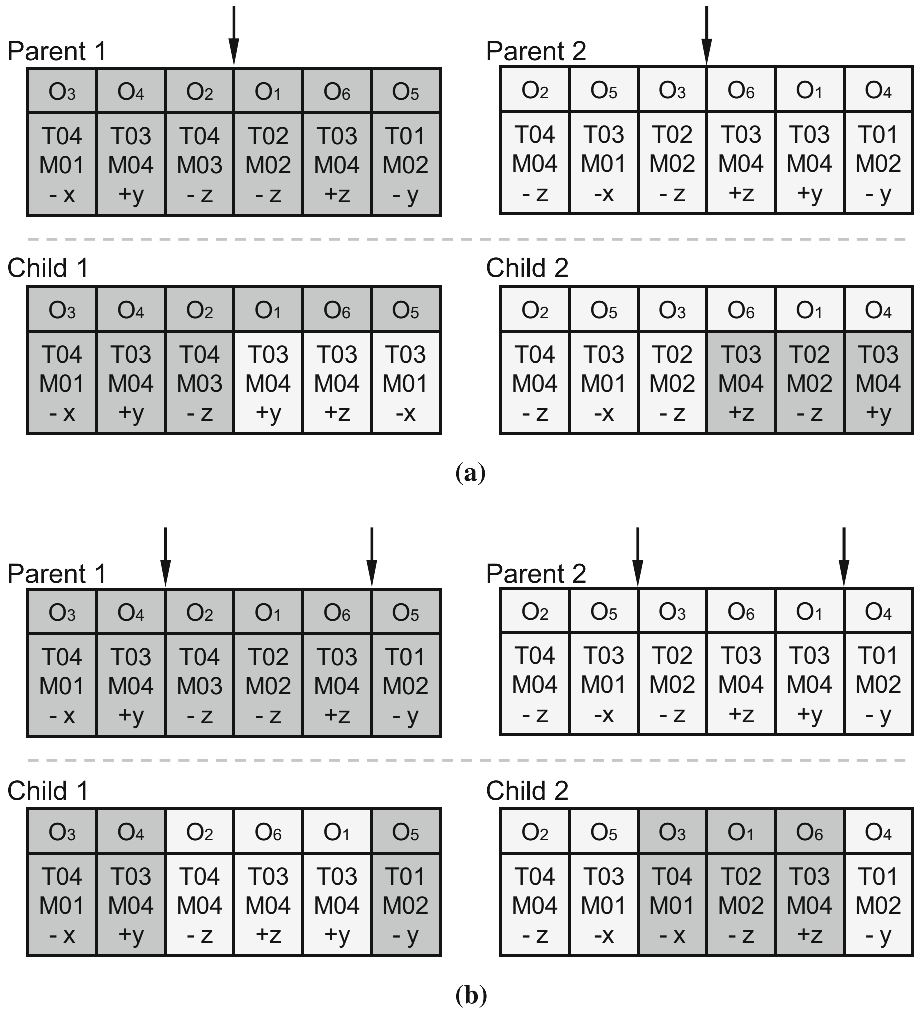
the T string, and TAD are generated randomly. The procedure of initialising the M, T, and TAD strings are as follows:

Step 1: For S_i, $(i = 1 \ldots, m)$, if $i > m$ stop.
Step 2: For O_i, $(i = 1 \ldots, n)$, if $i > n$ go to 1.
Step 3: Randomly select M from the operation $M[]$.
Step 4: Randomly select T from the operation $T[]$.
Step 5: Randomly select TAD from the operation $TAD[]$, and go to 1

Process plans evaluation

Total production cost and total processing time are the common evaluation criteria reported in the literature. In this research, a total weighted production cost (TWPC) that is comprised of six cost factors is used to evaluate the process plan. The cost factors are: the cost of machines (CM) (*obtained from the M string*), the cost of tools (CT) (*obtained from the T string*), the cost of machine changes (CMC) (*obtained from the M string*), the cost of tool changes (CTC)

Fig. 5 An example of: a 3SX crossover operator, b OX crossover operator



(*obtained from the M and T strings*), the cost of setup changes (CSC) (*obtained from the M and TAD strings*), and the cost of precedence constraints violation (CPV) (*obtained from the i string*). The computation of each cost factor is as follows:

1. Cost of machines (CM)

$$CM = \sum_{i=1}^{n} CIM_i \tag{8}$$

   Where n is the total number of operations, CIM is the cost index of using machine i, which is a constant value for a specific machine.

2. Cost of tools (CT)

$$CT = \sum_{i=1}^{n} CIT_i \tag{9}$$

   Where CIT is the cost index of using tool i, which is a constant value for a specific tool.

3. Cost of machine changes (CMC)

$$CMC = CIMC \times \sum_{i=1}^{n-1} \sigma_1(M_{i+1} - M_i) \tag{10}$$

   Where $CIMC$ is the cost index of machine changes, which is a constant value, M_i is the machine i used in operation i, σ_1 is a comparison factor and it is computed as follows:

$$\sigma 1(X - Y) = \begin{cases} 0\ if\ X \neq Y \\ 1\ if\ X \equiv Y \end{cases} \tag{11}$$

4. Cost of tool changes (CTC)

$$CTC = CITC \times \sum_{i=1}^{n-1} \sigma_2(\sigma_1(M_{i+1} - M_i)) - (\sigma_1(T_{i+1} - T_i)) \tag{12}$$

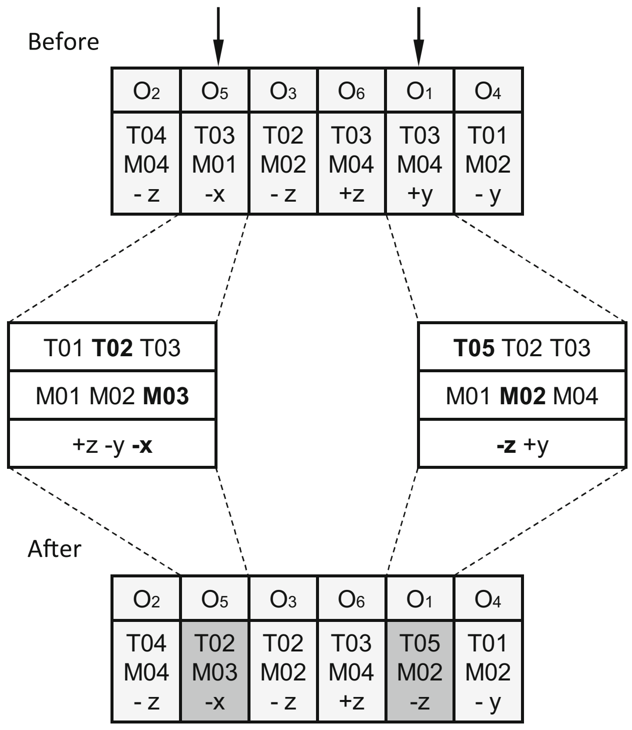

Before
O2 O5 O3 O6 O1 O4
T04 M04 - z | T03 M01 -x | T02 M02 - z | T03 M04 +z | T03 M04 +y | T01 M02 - y
T01 T02 T03
M01 M02 M03
+z -y -x
T05 T02 T03
M01 M02 M04
-z +y
After
O2 O5 O3 O6 O1 O4
T04 M04 - z | T02 M03 -x | T02 M02 - z | T03 M04 +z | T05 M02 -z | T01 M02 - y

Fig. 6 An example of the proposed mutation operator process

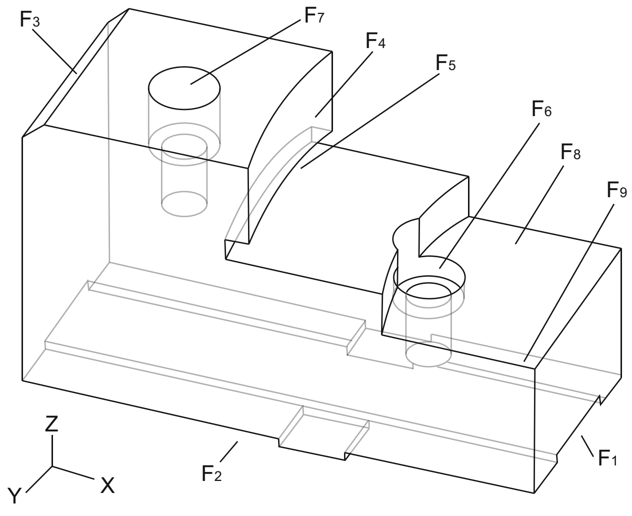


Fig. 7 Part 2 with 9 manufacturing features Ma et al. (2000)

Where $CITC$ is the cost index of tool changes, which is a constant value, T_i is the tool i used in operation i, σ_2 is a comparison factor and it is computed as follows:

$$\sigma_1(X-Y)=\begin{cases}0\, if\ X=Y=0\\ 1\ otherwise\end{cases} \tag{13}$$

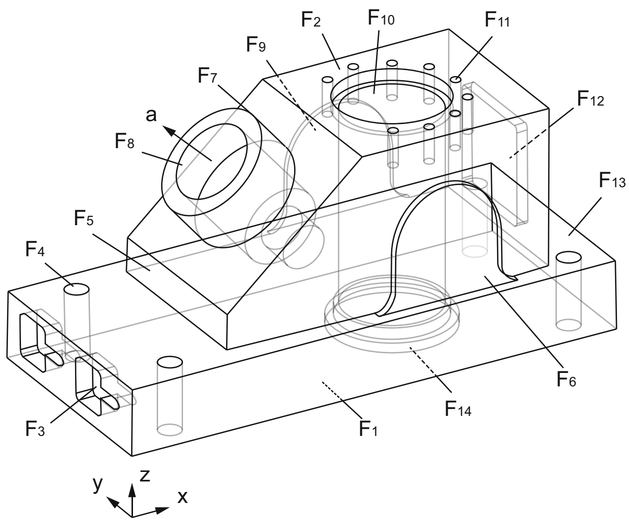


Fig. 8 Part 3 with 14 manufacturing features Li et al. (2004)

5. Cost of setup changes (CSC)

$$CSC=CISC \times\left(1+\sum_{i=1}^{n-1}\sigma_2(\sigma_1(M_{i+1}-M_i)) -(\sigma_1(TAD_{i+1}-TAD_i))\right) \tag{14}$$

Where $CISC$ is the cost index of setup change, which is a constant value, TAD_i is the TAD i used in operation i.

6. Cost of precedence constraints violation (CPV)

$$CPV=CIPV\times\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}\sigma_3(O_i-O_j) \tag{15}$$

Where $CIPV$ is the cost index of precedence constraints violation, σ_3 is a comparison factor and it is computed as follows:

$$\sigma_3(X-Y)=\begin{cases}1\ The\ sequence\ of\ x\ and\ y\ operations\\ violates\ constraints\\ 0\ The\ sequence\ of\ x\ and\ y\ operations\\ meets\ constraints\end{cases} \tag{16}$$

Finally, based on above cost factors, TWPC is computed as follow:

$$TWPC=w_1CM+w_2CT+w_3CMC+w_4CTC +w_5CSC+w_6CPV \tag{17}$$

Where w_1 to w_6 are weights.

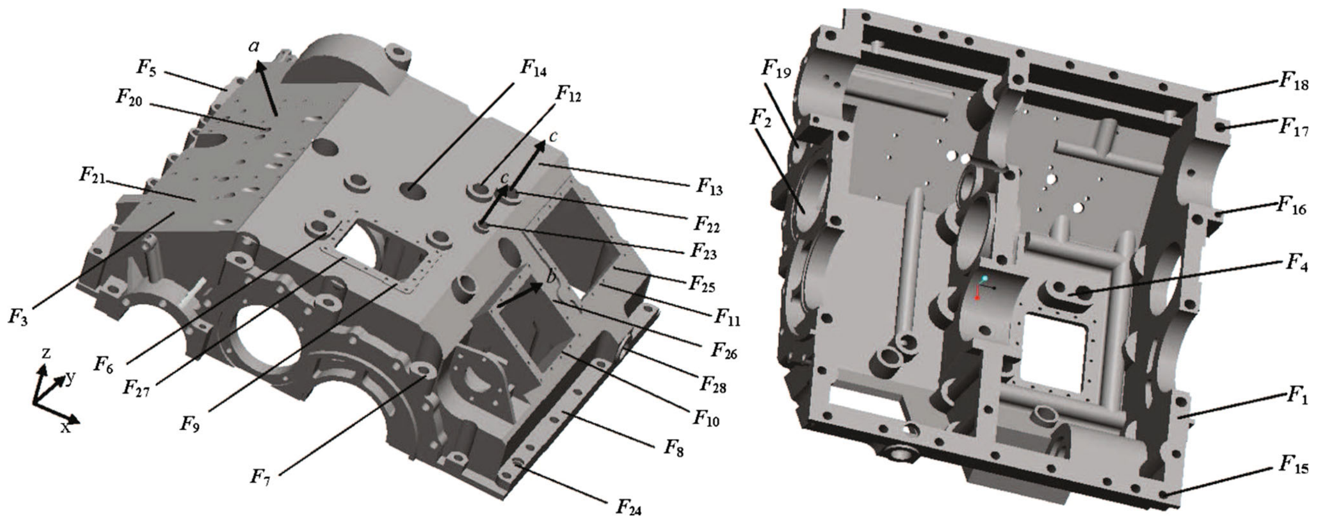


Fig. 9 Part 4 with 28 manufacturing features Huang et al. (2017)

Table 5 Features, operations, manufacturing resources flexibilities, and precedence constraints of Part 2 (Ma et al. 2000)

Features	O_i	TAD [ ]	M [ ]	T [ ]	Prior O_i
F1	Milling (O_1)	+ z	M01, M02	T01, T03	O_2, O_{13}
		+ z	M04, M05	T05, T015	
F2	Milling (O_2)	+ z	M01, M02	T01, T02, T03, T04	
		+ y, − y	M04, M05	T05	
F3	Milling (O_3)	− z, + x	M01, M02	T04	
		− z	M04, M05	T011	
F4	Milling (O_4)	− z	M01, M02	T01, T02, T04	
F5	Milling (O_5)	− z	M01, M02	T15	O_4
	Centre drilling (O_6)	− z	M01, M02, M03, M04, M05	T10	O_4
F6	Drilling (O_7)	− z	M01, M02, M03, M05	T14	O_1, O_4
	Milling (O_8)	− z	M01, M02	T03	O_1, O_4
	Centre drilling (O_9)	− z	M01, M02, M03, M04, M05	T10	O_1, O_4
F7	Drilling (O_{10})	− z	M01, M02, M04, M05	T14	
	Milling (O_{11})	− z	M01, M02	T03	
F8	Milling (O_{12})	− z	M01, M02	T01, T02, T03, T04	O_7, O_8, O_9
F9	Milling (O_{13})	+ z	M01, M02	T01, T02, T03, T04	
		+ z	M04, M05	T05	

Process plans optimization

Selection, crossover, and mutation operators are the key characteristics of reproducing the next generation in GA. During selection, two chromosomes are selected in order to perform the crossover process between them. A stochastic selection operator, which is an extension of a roulette wheel, is used in this research. The difference is that instead of selecting one chromosome in each wheel spinning, two chromosomes are selected in one-wheel spinning, which, in turn, increases the probability of selecting better chromosomes. On the other hand, the crossover operator performs the information exchanging process between the two selected chromosomes, with the objective of obtaining better offspring. The most used crossover operator in the operation sequencing problem is the order crossover (OX) since it maintains the precedence constraints unviolated (Dou et al. 2018b; Salehi and Tavakkoli-Moghaddam 2009; Dou et al. 2018a; Petrović et al. 2016). In this research, a mixed crossover operator involving three-string crossover (3SX) and OX is used to enhance and explore the solution space. The new developed 3SX aim to exchange the resources strings information of different CO while maintaining the operations sequence string of the chromosome as it is. Whereas, the aim of OX is to exchange the

Table 6 Features, operations, manufacturing resources flexibilities, and precedence constraints of Part 3 (Li et al. 2004)

Features	Feature describtions	O_i	TAD []	M []	T []	Prior O_i
F1	A planar surface	Milling (O_1)	+ z	M02, M03	T06, T07, T08	
F2	A planar surface	Milling (O_2)	− z	M02, M03	T06, T07, T08	O_1
F3	Two pockets arranged as a replicated feature	Milling (O_3)	+ x	M02, M03	T06, T07, T08	O_1
F4	Four holes arranged as a replicated feature	Drilling (O_4)	+ z, − z	M01, M02, M03	T02	O_1, O_5, O_{18}
F5	A step	Milling (O_5)	+ x, − z	M02, M03	T06, T07	O_1
F6	A protrusion (rib)	Milling (O_6)	+ y, − z	M02, M03	T07, T08	O_1
F7	A boss	Milling (O_7)	− a	M02, M03	T07, T08	O_1, O_5
F8	A compound hole	Drilling (O_8)	− a	M01, M02, M03	T02, T03, T04	O_1, O_7
		Reaming (O_9)	− a	M01, M02, M03	T09	O_1, O_7, O_8
		Boring (O_{10})	− a	M03, M04	T10	O_1, O_7, O_8, O_9
F9	A protrusion (rib)	Milling (O_{11})	− y, − z	M02, M03	T07, T08	O_1
F10	A compound hole	Drilling (O_{12})	− z	M01, M02, M03	T02, T03, T04	O_1, O_2, O_6, O_{11}
		Reaming (O_{13})	− z	M01, M02, M03	T09	O_1, O_2, O_6, O_{11}, O_{12}
		Boring (O_{14})	− z	M03, M04	T010	O_1, O_2, O_6, O_{11}, O_{12}, O_{13}
F11	Nine holes arranged in a replicated feature	Drilling (O_{15})	− z	M01, M02, M03	T01	O_1, O_2, O_{12}, O_{13}, O_{14}
		Tapping (O_{16})	− z	M01, M02, M03	T05	O_1, O_2, O_{12}, O_{13}, O_{14}, O_{15}
F12	A pocket	Milling (O_{17})	− x	M02, M03	T07, T08	O_1, O_{18}
F13	A step	Milling (O_{18})	− x, − z	M02, M03	T06, T07	O_1
F14	A compound hole	Reaming (O_{19})	+ z	M01, M02, M03	T09	O_1, O_{12}
		Boring (O_{20})	+ z	M03, M04	T10	O_1, O_{12}, O_{19}

information of both the operations sequence string and the resources strings to generate chromosomes that have new operations sequence. Figure 5 is an illustration of the two crossover operators process of a 6-bit chromosome length. The mixed crossover procedure is as follows:

Step 1: Select two parents' chromosomes according to stochastic selection operator.
Step 2: If the generation number is $\leq n/2$ (n is the total number of generations).

- Step 2.1: Generate two child's chromosomes with same operation sequences and operation resources of the parents.
- Step 2.2: Randomly generate a cutting point.
- Step 2.3: Replace M, T, and TAD of the operations in the right side of child 1 by M, T, and TAD of the same operations from parent 2.
- Step 2.4: Replace M, T, and TAD of the operations in the right side of child 2 by M, T, and TAD of the same operations from parent 1.

Step 3: Else

- Step 3.1: Generate two child's chromosomes with same operation sequences and operation resources of the parents.
- Step 3.2: Randomly generate two cutting points.
- Step 3.3: Reorder the gens between the two crossover points of child 1 according to the same gens order of parent 2.
- Step 3.4: Replace M, T, and TAD of the operations between the two crossover points of child 1 by M, T, and TAD of the same operations from parent 2.
- Step 3.5: Reorder the gens between the two crossover points of child 2 according to the same gens order of parent 1.
- Step 3.6: Replace M, T, and TAD of the operations between the two crossover points of child 2 by M, T, and TAD of the same operations from parent 1.

Step 4: End.

Table 7 Features, operations, manufacturing resources flexibilities, and precedence constraints of Part 4 (Huang et al. 2017)

Features	Feature describtions	O_i	TAD []	M []	T []	Prior O_i
F1	A flat surface	(O_1)	+ z	M01, M02	T01, T02, T03	
		(O_2)	+ z	M02	T01, T02, T03	O_1,O_{22}
		(O_3)	+ z	M02	T01, T02, T03	O_1,O_{22},O_2
F2	Bearing hole	(O_4)	− y,+ y	M07, M08	T04	O_1,O_{22},O_2,O_3,O_{27}
		(O_5)	− y,+ y	M06, M07, M08	T05	O_1,O_{22},O_2,O_3,O_{27},O_4
		(O_6)	− y,+ y	M06, M07, M08	T05	O_1,O_{22},O_2,O_3,O_{27},O_4,O_5
F3	Angular surface	(O_7)	− a	M05, M07, M08	T07, T08, T09	O_1,O_{22},O_2,O_3,O_{27},O_4
		(O_6)	− y,+ y	M06, M07, M08	T05	O_1,O_{22},O_2,O_3,O_{28}
		(O_8)	− a	M03, M04, M06	T07, T08, T09	O_1,O_{22},O_2,O_3,O_{28},O_{31},O_{32},O_{33}
F4	Four bosses	(O_9)	+ z	M03, M04, M05	T08, T09	O_1,O_{22}
F5	Plane 1	(O_{10})	− z	M03, M04, M05	T08, T09	O_1
F6	Top boss	(O_{11})	− z	M03, M04, M05	T08, T09	O_1
		(O_{12})	− z	M04, M05	T08, T09	O_1,O_{22},O_2,O_3,O_{11}
F7	Six bosses	(O_{13})	− z	M03, M04, M05	T08, T09	O_1
F8	Plane 2	(O_{14})	− z	M03, M04, M05	T08, T09	O_1
F9	Top window surface	(O_{15})	− z	M03, M04, M05	T07, T08	O_1,O_{22},O_2,O_3
		(O_{16})	− z	M03, M04, M05	T07, T08	O_1,O_{22},O_2,O_3, O_{15}
F10	Inclined plane 1	(O_{17})	− b	M03, M04, M05, M07	T07, T08	O_1,O_{22},O_2,O_3
		(O_{18})	− b	M03, M04, M05, M07	T07, T08	O_1,O_{22},O_2,O_3, O_{17}
F11	Inclined plane 2	(O_{19})	− b	M03, M04, M05, M07	T07, T08	O_1,O_{22},O_2,O_3
		(O_{20})	− b	M03, M04, M05, M07	T07, T08	O_1,O_{22},O_2,O_3, O_{19}
F12	The top holes	(O_{21})	− z, + z	M09, M10	T10	O_1,O_{22},O_2,O_3, O_{12}, O_9
F13	Top plane	(O_{22})	− z	M04, M5	T07, T08, T09	O_1
F14	Counter-bore hole	(O_{23})	− z	M09, M10	T20	O_1,O_{22},O_2,O_3, O_{27}
F15	18H7 hole	(O_{24})	+ z	M09, M10	T11	O_1,O_{22},O_2,O_3
		(O_{25})	+ z	M09, M10	T22	O_1,O_{22},O_2,O_3, O_{34}
F16	2-12.5 hole	(O_{26})	+ z	M09, M10	T12	O_1,O_{22},O_2,O_3
F17	12-21 hole	(O_{27})	+ z	M09, M10	T13	O_1,O_{22},O_2,O_3
F18	18-17 hole	(O_{28})	+ z	M09, M10	T14	O_1,O_{22},O_2,O_3
F19	Side hole	(O_{29})	− y	M07, M08	T06	O_1,O_{22},O_2,O_3
		(O_{30})	− y	M06,M07, M08	T06	O_1,O_{22},O_2,O_3, O_{29}
F20	15-20 holes	(O_{31})	− a	M07, M08, M09, M10	T15	O_1,O_{22},O_2,O_3, O_7, O_{28}
F21	24-8 holes	(O_{32})	− a	M07, M08, M09, M10	T16	O_1,O_{22},O_2,O_3, O_7, O_{28}
		(O_{33})	− a	M07, M08, M09, M10	T23	O_1,O_{22},O_2,O_3, O_7, O_{28}, O_{33}
F22	Oil passage hole 1	(O_{34})	− c	M06, M07, M08	T28	O_1,O_{22}
		(O_{35})	− c	M06, M07, M08	T24	O_1,O_{22}, O_{34}
F23	Oil passage hole 2	(O_{36})	− c	M06, M07, M08	T17	O_1,O_{22}
		(O_{37})	− c	M06, M07, M08	T25	O_1,O_{22}, O_{36}
F24	Counter-bore	(O_{38})	− z	M03, M04, M09, M10	T21	O_1,O_{22},O_2,O_3, O_{24}, O_{25}
F25	Holes in inclined plane 1	(O_{39})	− b	M09, M10	T18	O_1,O_{22},O_2,O_3, O_{17}, O_{18} ,O_{27}, O_4
		(O_{40})	− b	M09, M10	T26	O_1,O_{22},O_2,O_3, O_{17}, O_{18} ,O_{27}, O_4, O_{39}
F26	Holes in inclined plane 2	(O_{41})	− b	M09, M10	T18	O_1,O_{22},O_2,O_3, O_{29}, O_{19} ,O_{20}
		(O_{42})	− b	M09, M10	T26	O_1,O_{22},O_2,O_3, O_{29}, O_{19} ,O_{20}, O_{41}
F27	Top holes	(O_{43})	− z	M03, M04, M09, M10	T18	O_1,O_{22},O_2,O_3, O_{15}, O_{16}
		(O_{44})	− z	M03, M04, M09, M10	T26	O_1,O_{22},O_2,O_3, O_{15}, O_{16}, O_{43}
F28	Oil passage hole 3	(O_{45})	− x	M06, M07, M080	T19	O_1,O_{22},O_2,O_3, O_{28}
		(O_{46})	− x	M06, M07, M080	T27	O_1,O_{22},O_2,O_3, O_{28}, O_{45}

Table 8 The cost indices of Part 1, Part 2, Part 3, and Part 4 (Li et al. 2004; Ma et al. 2000; Huang et al. 2017)

Part.	M	CIM	T	CIT	Indices	Indices cost
1	M01	10	T01	3	CIMC	300
	M02	35	T02	3	CISC	120
	M03	60	T03	8	CITC	15
			T04	15	CIPV	100
			T05	10		
			T06	15		
			T07	10		
			T08	10		
2	M01	70	T01	10	CIMC	150
	M02	35	T02	10	CISC	90
	M03	10	T03	10	CITC	20
	M04	40	T04	12		
	M05	85	T05	8		
			T10	2		
			T11	10		
			T14	3		
			T15	6		
3	M01	10	T01	7	CIMC	160
	M02	40	T02	5	CISC	100
	M03	100	T03	3	CITC	20
	M04	60	T04	8	CIPV	100
			T05	7		
			T06	10		
			T07	15		
			T08	30		
			T09	15		
			T10	20		
4	M01	20	T01	5	CIMC	120
	M02	45	T02	6	CISC	90
	M03	50	T03	7	CITC	15
	M04	55	T04	12		
	M05	20	T05	13		
	M06	80	T06	9		
	M07	45	T07	8		
	M08	48	T08	9		
	M09	16	T09	10		
	M10	18	T10	4		
			T11	4		
			T12	3		
			T13	4		
			T14	3		
			T15	4		
			T16	3		
			T17	4		
			T18	2		
			T19	2		
			T20	4		
			T21	3		
			T22	5		
			T23	3		
			T24	4		
			T25	4		
			T26	3		
			T27	4		
			T28	3		

Mutation process aims to increase the population diversity with a view of avoiding premature convergence. In this research, the mutation operator is designed to replace the operation resources randomly according to the given operation resources flexibilities. Figure 6 is an illustration of a mutation process of a 6-bit chromosome length. The proposed mutation procedure is as follows:

Step 1: Select a chromosome randomly.
Step 2: Select two operations randomly.
Step 3: Randomly select M from the operation M [].
Step 4: Randomly select T from the operation T [].
Step 5: Randomly select TAD from the operation TAD [].
Step 6: End.

Comparative case studies

To appraise the performance of CPAGA, four comparative case studies are carried out through Part 1 (Fig. 3), Part 2 (Fig. 7), Part 3 (Fig. 8), and Part 4 (Fig. 9). The relevant manufacturing information, including features, operations, manufacturing resource flexibilities, and precedence constraints, are placed in Tables 2, 5, 6, 7 respectively, while the cost indices of the four parts are placed in Table 8. The main GA parameters are population size, generation number, crossover probability (PC), and mutation probability (PM). The population size is predetermined by the number of operation sequences generated for the given part by CPA. A comprehensive test for the four parts is carried out to obtain the best value of the remaining parameters, and the best performance of GA in the CPAGA is gained by setting PC to 0.7 and PM to 0.3. Since the convergence of all tests occurs in the early stages, the number of generations is set as 100 generations. The CPAGA was programmed in MATLAB and executed on a Windows 10 platform by a 2.6 GHz Core i7 CPU with a 16 GB RAM computer.

In Part 1, which is reported by Li et al. (2004), CPA generated 360 sequences of operations for the four possibilities of applying the soft constraints. GA was executed 20 times

Table 9 The best obtained process plans of Part 1

Condition (1)														
O	6	9	1	12	7	3	5	4	8	10	13	11	2	14
M	2	2	2	2	2	2	2	2	2	2	2	2	2	2
T	2	1	1	1	1	5	5	5	5	5	5	5	8	1
TAD	−z	−z	−z	−z	−z	+y	+y	+y	+x	−y	−y	−y	−y	−y
CM: 490, CT: 98, CMC: 0, CTC: 60, CSC: 480, PC:200, TWPC: 1328														
Condition (2)														
O	8	10	13	11	14	6	7	1	12	9	2	5	3	4
M	2	2	2	2	2	2	2	2	2	2	2	2	2	2
T	5	6	6	5	1	2	1	1	1	1	8	5	6	5
TAD	+x	−y	−y	−y	−y	−z	−z	−z	−z	−z	+y	+y	+y	+y
CM: 490, CT: 0, CMC: 0, CTC: 0, CSC: 480, PC:200, TWPC: 1170														

Table 10 A comparison of the three parts results obtained by CPAGA with those of other approaches

Approach	Condition (1) Minimum	Maximum	Mean	Condition (2) Minimum	Maximum	Mean	Condition (3) Minimum	Maximum	Mean
Part 1									
CPAGA	**1328**	**1328**	**1328**	**1170**	**1170**	**1170**	–	–	–
TSACO (Wang et al. 2015)	1328	1348	1329	1170	1170	1170	–	–	–
ACO (Liu et al. 2013)	1328	1343	1329.5	1170	1170	1170	–	–	–
TS (Li et al. 2004)	1328	1378	1342	1170	1290	1194	–	–	–
SA (Li et al. 2004)	1328	1518	1373.5	1170	1345	1217	–	–	–
GA (Li et al. 2004)	1478	1778	1611	1410	1650	1482	–	–	–
Part 2									
CPAGA	**743**	**743**	**743**	**1198**	**1198**	**1198**	–	–	–
SA (Ma et al. 2000)	853	–	–	1288	1310	1295	–	–	–
GA (Ma et al. 2000)	853	–	835.9	–	–	–	–	–	–
Part 3									
CPAGA	**2530**	**2535**	**2530.5**	**2090**	**2090**	**2090**	**2500**	**2500**	**2500**
FSDPSO (Dou et al. 2018a)	2530	–	2532.5	2090	–	2090	–	–	–
DDPSO (Dou et al. 2018a)	2535	–	2575.7	2090	–	2115	–	–	–
IGA (Dou et al. 2018b)	2530	2547	2538.7	2090	2120	2111	–	–	–
ACO (Hu et al. 2017)	2530	–	2666	2090	–	2115	–	–	–
ESGA (Su et al. 2018)	2530	2562	2539.1	2090	–	–	2590	–	–
TSGA (Su et al. 2018)	2582	2669	2595.7	–	–	–	–	–	–
PSO (Guo et al. 2006)	2535	–	2680.5	–	–	–	–	–	–
HGGA (Huang et al. 2012)	2527	–	–	2120	–	–	2590	–	–
TSACO (Wang et al. 2015)	2525	2557	2552	2090	2380	2120.5	2590	2740	2600.8
TS (Li et al. 2004)	2527	2690	2609	2120	2390	2208	2580	2740	2630
SA (Li et al. 2004)	2535	2829	2668.5	2120	2380	2287	2590	2740	2630
GA (Li et al. 2004)	2667	2885	2796	2220	2580	2370	2600	2840	2705
Part 4									
CPAGA	**4299**	**4315**	**4302**	**4503**	**4503**	**4503**	–	–	–
GA-SA (Huang et al. 2017)	4683	–	–	5024	–	–	–	–	–

for the two conditions: (1) All resources are available, and $w_1 - w_6$ in Eq. (17) are set equal to 1. (2) All resources are available, and $w_2 = w_4 = 0$, $w_1 = w_3 = w_5 = w_6 = 1$. The best obtained process plans for the two conditions are shown in Table 9. Results were compared against GA, SA, and TS based approaches by Li et al. (2004), the TSACO by

Table 11 The best obtained process plans of Part 2

Condition (1)													
O	3	2	13	1	4	5	6	9	10	7	11	8	12
M	2	2	2	2	2	2	2	2	2	2	2	2	2
T	4	1	1	1	1	15	10	10	14	14	3	3	3
TAD	+ z	+ z	+ z	+ z	− z	− z	− z	− z	− z	− z	− z	− z	− z
CM: 455, CT: 98, CMC: 0, CTC: 100, CSC: 90 PC:0, TWPC: 743													
Condition (2)													
O	3	2	13	1	4	5	9	6	7	10	11	8	12
M	1	1	1	1	1	1	1	1	1	1	1	1	1
T	4	1	1	1	1	15	10	10	14	14	3	3	3
TAD	+ z	+ z	+ z	+ z	− z	− z	− z	− z	− z	− z	− z	− z	− z
CM: 910, CT: 98, CMC: 0, CTC: 100, CSC: 90, PC:0, TWPC: 1198													

Table 12 The best obtained process plans of Part 3

Condition (1)																				
O	1	3	5	2	18	11	6	12	13	19	17	7	8	9	14	10	20	15	4	16
M	2	2	2	2	2	2	2	2	2	2	2	2	2	2	4	4	4	1	1	1
T	6	6	6	6	6	7	7	3	9	9	7	7	3	9	10	10	10	1	2	5
TAD	+ z	+ x	− z	− z	− z	− z	− z	− z	− z	+ z	− x	− a	− a	− a	− z	− a	+ z	− z	− z	− z
CM: 770 CT: 240 CMC: 320 CTC: 200 CSC: 1000 PC:0, TWPC: 2530																				
Condition (2)																				
O	1	18	17	11	6	2	12	13	5	3	7	8	9	19	20	14	10	15	16	4
M	2	2	2	2	2	2	2	2	2	2	2	2	2	2	4	4	4	1	1	1
T	6	6	8	7	7	8	4	9	7	7	7	4	9	9	10	10	10	1	5	2
TAD	+ z	− x	− x	− z	− z	− z	− z	− z	+ x	+ x	− a	− a	− a	+ z	+ z	− z	− a	− z	− z	− z
CM: 770 CT: 0 CMC: 320 CTC: 0 CSC: 1000, PC:0, TWPC: 2090																				
Condition (3)																				
O	1	19	20	3	5	7	8	9	10	18	6	2	11	4	12	13	14	15	16	17
M	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
T	8	9	10	8	6	8	4	9	10	6	8	6	8	2	4	9	10	1	5	8
TAD	+ z	+ z	+ z	+ x	+ x	− a	− a	− a	− a	− z	− z	− z	− z	− z	− z	− z	− z	− z	− z	− x
CM: 2000 CT: 0 CMC: 0 CTC: 0 CSC: 500, PC:0, TWPC: 2500																				

Wang et al. (2015), and the ACO by Liu et al. (2013), and are presented in Table 10. Under condition (1), among 20 trials, the result TWPC (1328) occurs 20 times. The mean TWPC (1328) is the best result among all the six algorithms. Under condition (2), the TWPC (1170.0) occurs 20 times in 20 trials, which is better than the performances of TS, SA, and GA and is the same as the performance of ACO and TSACO. In Part 2, which is reported by Ma et al. (2000), CPA generated 208 sequences of operations, and GA was executed 60 times for the two conditions: (1) All resources are available, and $w_1 - w_6$ in Eq. (17) are set equal to 1. (2) M02 is down, and $w_1 - w_6$ are set equal to 1. The best obtained process plans for the two conditions are shown in Table 11. Comparison of the results with those of GA and SA based approaches by Ma et al. (2000) are presented in Table 9. For this part, Eq. (14) modified to be the same as Ma evaluation model, as follows: (1) All resources are available, and $w_1 - w_6$ in Eq. (17) are set equal to 1. (2) M02 is down, and and $w_1 - w_6$ in Eq. (17) are set equal to 1.

$$CSC = CISC \times \sum_{i=1}^{n-1} \sigma_2(\sigma_1(M_{i+1} - M_i)) - (\sigma_1(TAD_{i+1} - TAD_i)) \tag{18}$$

Under condition (1), among 60 trial results, TWPC (743) occurred 60 times. The mean and minimum TWPC (743) was better than the other two algorithms. Under condition (2), TWPC (1198) occurred 60 times in 60 trials, which was better than the performances of SA for both mean and minimum TWPC values.

Table 13 The best obtained process plans of Part 4 by CPAGA and GA-SA(Huang et al. 2017)

CPAGA: Condition (1)																							
O	1	13	10	11	22	9	2	3	28	27	15	12	16	14	7	19	20	17	18	29	30	4	5
M	1	5	5	5	5	5	2	2	9	9	5	5	5	5	5	7	7	7	7	7	7	7	7
T	1	8	8	8	8	8	1	1	14	13	8	8	8	8	8	7	7	7	7	6	6	4	5
TAD	+ z	− z	− z	− z	− z	+ z	+ z	+ z	+ z	+ z	− z	− z	− z	− z	− a	− b	− b	− b	− b	− y	− y	− y	− y
O cont.	6	45	46	36	37	34	35	26	24	25	21	23	38	43	44	31	32	33	39	41	40	42	8
M cont.	7	7	7	7	7	7	7	9	9	9	9	9	9	9	9	9	9	9	9	9	9	9	3
T cont.	5	19	27	17	25	28	24	12	11	22	10	20	21	18	26	15	16	23	18	18	26	26	7
TAD cont.	− y	− x	− x	− c	− c	− c	− c	+ z	+ z	+ z	+ z	− z	− z	− z	− z	− a	− a	− a	− b	− b	− b	− b	− a
CM: 1307 CT: 277 CMC: 840 CTC: 435 CSC: 1440 PC:0, TWPC: **4299**																							
GA-SA(Huang et al. 2017): Condition (1)																							
O	1	22	11	2	3	19	20	17	18	14	10	13	15	12	16	28	27	26	24	25	9	7	29
M	1	5	5	2	2	3	3	3	3	3	3	3	3	3	3	9	9	9	9	9	5	5	7
T	1	8	8	1	1	8	8	8	8	8	8	8	8	8	8	14	12	13	11	22	8	8	6
TAD	+ z	− z	− z	+ z	+ z	− b	− b	− b	− b	− z	− z	− z	− z	− z	− z	+ z	+ z	+ z	+ z	+ z	+ z	+ z	− y
O cont.	4	45	46	36	37	34	35	8	5	6	30	21	23	38	43	44	31	32	33	39	41	40	42
M cont.	7	7	7	7	7	7	7	6	6	6	6	9	9	9	9	9	9	9	9	9	9	9	9
T cont.	7	19	27	17	15	28	24	7	5	5	6	10	20	21	18	26	15	16	23	18	18	26	26
TAD cont.	− y	− x	− x	− c	− c	− c	− c	− a	− y	− y	− y	− z	− z	− z	− z	− z	− a	− a	− a	− b	− b	− b	− b
CM: 1642 CT: 281 CMC: 960 CTC: 450 CSC: 1350 PC:0, TWPC: 4683																							
CPAGA: Condition (2)																							
O	1	11	22	9	2	3	28	27	15	16	13	10	12	14	7	29	30	4	5	6	45	46	36
M	1	5	5	5	2	2	9	9	5	5	5	5	5	5	5	8	8	8	8	8	8	8	8
T	1	9	9	9	1	1	14	13	7	7	9	9	9	9	9	6	6	4	5	5	19	27	17
TAD	+ z	− z	− z	+ z	+ z	+ z	+ z	+ z	− z	− z	− z	− z	− z	− z	− a	− y	− y	− y	− y	− y	− x	− x	− c
O cont.	37	34	35	26	24	25	21	23	38	43	44	31	32	33	19	20	17	18	8	39	41	40	42
M cont.	8	8	8	9	9	9	9	9	9	9	9	9	9	9	4	4	4	4	4	9	9	9	9
T cont.	25	28	24	12	11	22	10	20	21	18	26	15	16	23	7	7	7	7	7	18	18	26	26
TAD cont.	− c	− c	− c	+ z	+ z	+ z	− z	− z	− z	− z	− z	− a	− a	− a	− b	− b	− b	− b	− a	− b	− b	− b	− b
CM: 1385 CT: 283 CMC: 960 CTC: 435 CSC: 1440 PC:0, TWPC: **4503**																							
GA-SA(Huang et al. 2017): Condition (2)																							
O	1	11	22	2	3	24	25	26	28	27	7	9	17	18	19	20	15	16	13	10	14	12	4
M	1	5	5	2	2	9	9	9	9	9	5	5	4	4	4	4	4	4	4	4	4	4	8
T	1	9	9	1	1	11	22	13	14	12	9	9	7	7	7	7	7	7	9	9	9	9	6
TAD	+ z	− z	− z	+ z	+ z	+ z	+ z	+ z	+ z	+ z	− a	+ z	− b	− b	− b	− b	− z	− z	− z	− z	− z	− z	− y
O cont.	29	45	46	34	35	36	37	38	23	21	43	44	39	41	40	42	31	32	33	30	5	6	8
M cont.	8	8	8	8	8	8	8	9	9	9	9	9	9	9	9	9	9	9	9	6	6	6	6
T cont.	4	19	27	28	24	17	15	21	20	10	18	26	18	18	26	26	15	16	23	6	5	5	7
TAD cont.	− y	− x	− x	− c	− c	− c	− c	− a	− a	− a	− b	− b	− b	− b	− z	− z	− z	− z	− z	− y	− y	− y	− a
CM: 1876 CT: 283 CMC: 960 CTC: 465 CSC: 1440 PC:0, TWPC: 5024																							

Table 14 CPU time of the best results obtained by CPAGA, ESGA, and TSGA

Approach	Part 1 CPU time(s)		Part 2 CPU time(s)		Part 3 CPU time(s)			Part 4 CPU time(s)	
	Cond.(1)	Cond.(2)	Cond.(1)	Cond.(2)	Cond.(1)	Cond.(2)	Cond.(3)	Cond.(1)	Cond.(2)
CPAGA	**1.32**	**0.84**	**0.96**	**0.63**	**1.94**	**1.24**	**1.3**	**3.26**	**5.41**
ESGA (Su et al. 2018)	–	–	–	–	3.46	–	–	–	–
TSGA (Su et al. 2018)	–	–	–	–	2.54	–	–	–	–

In addition to the two conditions of Part 1, third condition is used in Part 3 that is reported the first time by Li et al. (2004) to test CPAGA, which is: (3) M02 and T07 are down, and $w_2 = w_4 = 0$, $w_1 = w_3 = w_5 = w_6 = 1$. CPA generated 470 sequence of operations, and GA was executed 20 times for the three conditions. The best obtained process plans for the three conditions are shown in Table 12. Comparison of the results was carried out against FSDPSO and DDPSO by Dou et al. (2018a), IGA by Dou et al. (2018b), ACO by Hu et al. (2017), ESGA and TSGA by Su et al. (2018), PSO by Guo et al. (2006), HGGA by Huang et al. (2012), TSACO by Wang et al. (2015), and the TS, SA, and GA by Li et al. (2004), as shown in Table 9. Under condition (1), among 20 trial results, the TWPC (2530) occurred 18 times and the TWPC (2535) occurred two times. The minimum TWPC (2530) was better than the TWPC of DPPSO, TSGA, PSO, SA, and GA, equal to the TWPC of FSDPSO, IGA, ACO, and ESGA, and less than the TWPC of HGGA, TSACO, and TS. The reason that those approaches obtained less TWPC value than CPAGA TWPC value is due to the additional T06 in T [], which gives different optimal solutions. (*The reader may be referred to Table 11 of* Li et al. (2004), *Table 9 of* Wang et al. (2015), *and Table 7 of* Huang et al. (2012).) The mean TWPC (2530.5) is the best result among all the 13 approaches. Under condition (2), the TWPC (2090) occurred 20 times in 20 trials. The minimum TWPC (2090) is better than those of some approaches and the same as others, while the mean TWPC (2090) was the same as the mean of FSDPSO and better than all other approaches. Under condition (3), the TWPC (2500) occurred 20 times in 20 trials, which is the best result obtained among the 13 approaches in terms of minimum and mean TWPC values.

Finally, Part 4 that is reported by Huang et al. (2017) consist of 28 features that comprise of 46 operations with 213 precedence constraints among the operations. For this complex part, CPA generates 1618 sequences of operations. GA was executed 20 times for the following two conditions: (1) All resources are available, and $w_1 - w_6$ in Eq. (17) are set equal to 1. (2) M03, M07, and T08 are down, and $w_1 - w_6$ in Eq. (17) are set equal to 1. The best obtained process plans under the two conditions accompanied by those that are obtained by GA-SA (Huang et al. 2017) are placed in Table 13. Note that the process plan generated by GA-SA under condition (1) is unfeasible because some of the precedence constraints are violated such as O_8 came before O_{30}, O_{31}, and O_{32}. Furthermore, O_{12} assigned to M03 which it is not given in the O_{12}' M[]. Although the process plans generated by GA-SA are unfeasible, CPAGA generated better process plans under the two conditions. Where under condition (1) the TWPC (4299) occurred 16 times, the TWPC (4314) occurred 3 times, and the TWPC (4315) occurred 1 time, which are all better than the best TWPC value that is obtained by GA-SA. Under condition (2) the TWPC (4503) is obtained 20 times, which is better than the TWPC value that is obtained by GA-SA.

Unfortunately, all researchers mentioned in the above comparisons except Su et al. (2018) have not been taken CPU time into consideration. Therefore, the efficiency verification of the proposed approach is carried out against ESGA, and TSGA by Su et al. (2018). Table 14 shows the CPU time of the best obtained TWPC values of CPAGA for the four parts under different conditions, as well as the ESGA and TSGA CPU time when they are performed under condition (1) of Part 3. Generally, CPAGA is able to generate optimal or near-optimal solutions with reasonable CPU time. Compared with ESGA and TSGA, the proposed CPAGA is able to find a competitive solution with a lower CPU time. where, with the same TWPC values CPAGA dominates ESGA in term of the CPU time, on the other hand, CPAGA dominates TSGA in term of TWPC value and CPU time.

Conclusions

A CPAGA approach was proposed to solve the process planning problem. The process planning problem was represented as a graph of OSS, OTS, and O nodes and the CPA was used to generate an initial set of optimised operation sequences based on the principles of minimising the number of setup changes, tool changes, and machine changes, and supplying it to GA. GA, in turn, was used to obtain an optimal or near optimal process plan. Mixed OX and 3SX crossover were developed to generate better offspring to subsequently enhance the performance of GA. 3SX was used first to search within the operations sequences generated by CPA, then OX was used to explore alternative operation sequences to be evaluated. In comparing the CPAGA approach to different approaches reported in the literature for the four parts, CPAGA proved its efficiency and effectiveness. From a generated solutions quality perspective, CPAGA obtained better TWPC values for the two conditions of Part 2, the third condition of Part 3, and the two conditions of Part 4. Since the solution space was reduced by the CPA to be limited to only the optimised operation sequences. This, in turn, gives the advantage to GA in searching for an optimal or near optimal process plan. From the perspective of the repeatability of good quality of generated solutions, CPAGA obtained better TWPC means for all the conditions of the four parts, due to the good quality of operations sequences obtained by CPA, and the effectiveness of the proposed crossover operator. From the perspective of the CPAGA efficiency, a better CPU time has been accomplished against ESGA and TSGA for the first condition of Part 3. Although the proposed approach has demonstrated a good performance compared to other approaches, some issues need to be considered in future studies: first, incorporating more knowledge-based production rules to maintain different cases that may occur in OSS and OTS nodes construction; second, investigating more complex parts with difficult precedence constraints among their features, to show the effectiveness of the proposed approach; and, finally, incorporating CPA with different meta-heuristic algorithms instead of GA.

References

Al-wswasi, M., Ivanov, A., & Makatsoris, H. (2018). A survey on smart automated computer-aided process planning (ACAPP) techniques. *International Journal of Advanced Manufacturing Technology*, *97*(1–4), 809–832.

Ding, L., Yue, Y., Ahmet, K., Jackson, M., & Parkin, R. (2005). Global optimization of a feature-based process sequence using GA and ANN techniques. *International Journal of Production Research*, *43*(15), 3247–3272.

Dou, J., Li, J., & Su, C. (2018a). A discrete particle swarm optimisation for operation sequencing in CAPP. *International Journal of Production Research*, *56*(11), 3795–3814.

Dou, J., Zhao, X., & Su, C. (2018b). An improved genetic algorithm for optimization of operation sequencing. In *Proceedings of 2018 IEEE International Conference on Mechatronics and Automation, ICMA 2018* (pp. 695–700).

Guo, Y. W., Mileham, A. R., Owen, G. W., & Li, W. D. (2006). Operation sequencing optimization using a particle swarm optimization approach. *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, *220*(12), 1945–1958.

Hu, Q., Qiao, L., & Peng, G. (2017). An ant colony approach to operation sequencing optimization in process planning. *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, *231*(3), 470–489.

Huang, W., Hu, Y., & Cai, L. (2012). An effective hybrid graph and genetic algorithm approach to process planning optimization for prismatic parts. *International Journal of Advanced Manufacturing Technology*, *62*(9–12), 1219–1232.

Huang, W., Lin, W., & Xu, S. (2017). Application of graph theory and hybrid GA-SA for operation sequencing in a dynamic workshop environment. *Computer-Aided Design and Applications*, *14*(2), 148–159.

Krishna, A. G., & Mallikarjuna Rao, K. (2006). Optimisation of operations sequence in CAPP using an ant colony algorithm. *International Journal of Advanced Manufacturing Technology*, *29*(1–2), 159–164.

Li, W. D., O, S. K., & N, 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* 1899.

Li, W. D., Ong, S. K., & Nee, A. Y. (2004). Optimization of process plans using a constraint-based tabu search approach. *International Journal of Production Research*, *42*(10), 1955–1985.

Li, X., Gao, L., & Wen, X. (2013). Application of an efficient modified particle swarm optimization algorithm for process planning. *International Journal of Advanced Manufacturing Technology*, *67*(5–8), 1355–1369.

Lian, K., Zhang, C., Shao, X., & Gao, L. (2012). Optimization of process planning with various flexibilities using an imperialist competitive algorithm. *International Journal of Advanced Manufacturing Technology*, *59*(5–8), 815–828.

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.

Ma, G. H., Zhang, Y. F., & Nee, A. Y. (2000). A simulated annealing-based optimization algorithm for process planning. *International Journal of Production Research*, *38*(12), 2671–2687.

Moon, C., Kim, J., Choi, G., & Seo, Y. (2002). An efficient genetic algorithm for the traveling salesman problem with precedence constraints. *European Journal of Operational Research*, *140*(3), 606–617.

Musharavati, F., & Hamouda, A. S. M. (2011). Modified genetic algorithms for manufacturing process planning in multiple parts manufacturing lines. *Expert Systems with Applications*, *38*(9), 10770–10779.

Nallakumarasamy, G., Srinivasan, P. S., Venkatesh Raja, K., & Malayalamurthi, R. (2011). Optimization of operation sequencing in CAPP using simulated annealing technique (SAT). *International Journal of Advanced Manufacturing Technology*, *54*(5–8), 721–728.

Petrović, M., Mitić, M., Vuković, N., & Miljković, Z. (2016). Chaotic particle swarm optimization algorithm for flexible process planning. *International Journal of Advanced Manufacturing Technology*, *85*(9–12), 2535–2555.

Reddy, S. V., Shunmugam, M. S., & Narendran, T. T. (1999). Operation sequencing in CAPP using genetic algorithms. *International Journal of Production Research*, *37*(5), 1063–1074.

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.

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.

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*(2), 313–332.

Su, Y., Chu, X., Zhang, Z., & Chen, D. (2015). Process planning optimization on turning machine tool using a hybrid genetic algorithm with local search approach. *Advances in Mechanical Engineering*, *7*(4), 1–14.

Wang, J. F., Wu, X., & Fan, X. (2015). A two-stage ant colony optimization approach based on a directed graph for process planning. *International Journal of Advanced Manufacturing Technology*, *80*(5–8), 839–850.

Wang, Y. F., Zhang, Y. F., & Fuh, J. Y. (2012). A hybrid particle swarm based method for process planning optimisation. *International Journal of Production Research*, *50*(1), 277–292.

Wen, X. Y., Li, X. Y., Gao, L., & Sang, H. Y. (2014). Honey bees mating optimization algorithm for process planning problem. *Journal of Intelligent Manufacturing*, *25*(3), 459–472.

Xu, L., Deng, W., Liu, W., Ma, S., Li, A., & Matta, A. (2014). Optimization of process planning for cylinder block based on feature machining elements. In *Conference Proceedings–IEEE International Conference on Systems, Man and Cybernetics* (vol. 2014, pp. 2575–2580).

Yun, Y., & Moon, C. (2011). Genetic algorithm approach for precedence-constrained sequencing problems. *Journal of Intelligent Manufacturing*, *22*, 379–388.

Zacharia, P. T., Tsirkas, S. A., Kabouridis, G., & Giannopoulos, G. I. (2015). Planning the construction process of a robotic arm using a genetic algorithm. *The International Journal of Advanced Manufacturing Technology*, *79*(5–8), 1293–1302.

Zacharia, P. T., Tsirkas, S. A., Kabouridis, G., Yiannopoulos, A. C., & Giannopoulos, G. I. (2018). Genetic-Based Optimization of the Manufacturing Process of a Robotic Arm under Fuzziness. *Mathematical Problems in Engineering*, *2018*, 1–12.

Zhang, F., Zhang, Y., & Nee, A. (1997). Using genetic algorithms in process planning for job shop machining. *IEEE Transactions on Evolutionary Computation*, *1*(4), 278–289.