Journal of Intelligent Manufacturing (2020) 31:1751–1767
https://doi.org/10.1007/s10845-018-1401-3

A knowledge based intelligent process planning method for controller of computer numerical control machine tools

Yingxin Ye[1,2] · Tianliang Hu[1,2] · Yan Yang[1,2] · Wendan Zhu[1,2] · Chengrui Zhang[1,2]

Received: 5 December 2017 / Accepted: 14 February 2018 / Published online: 28 February 2018

Abstract
The development of computer, internet and information technology puts forward higher demands for Computer Numerical Control (CNC) machine tools to improve the intelligence in many aspects. Among these aspects, intelligent process planning plays an important role in current changeable market and customized product promotion by shortening production cycle and providing more stable process planning ability. To realize intelligent process planning, a CNC controller with cloud knowledge base support is proposed with ability of making process planning autonomously based on workpiece design. Previous work of knowledge model and cloud knowledge base framework design is introduced, and then this paper focuses on the complete process planning method within the intelligent CNC controller. Both interactivity between knowledge base and CNC controller, and query/infer mechanism in knowledge base are illustrated in detail. A case study of two shafts process planning is shown to demonstrate the feasibility of the intelligent process planning method.

Keywords Process planning · Intelligent CNC controller · Knowledge base

Introduction

With the development of computer technology, Computer Numerical Control (CNC) machine tool has more ability to do efficient and precise manufacturing than its ancestor Numerical Control (NC) machine tool which appeared since 1950s (Xu et al. 2005). Currently, development of internet and information technology brings new concept such as Internet of Things (IoT) and cloud computing into manufacturing field. IoT solutions (Tao et al. 2014b) enrich the use of sensors in manufacturing to collect real-time data, while cloud computer (Tao et al. 2017a, 2014a) enables network based data storage, transmission, analysis. These new concept contributes to the volume growth of data collection in manufacturing, which offers an opportunity in the transformation of today's manufacturing paradigm to intelligent manufacturing (Tao et al. 2017b). As the important part of manufacturing, CNC machine tools could and should take advantage of this large-scale data to adapt itself in intelligent manufacturing environment. In this situation, data-driven strategy is adopted to exploit the data from manufacturing to refine manufacturing process, improving the flexibility and intelligent level of manufacturing (Tao et al. 2018). For raw data (e.g. sensory data), data mining (Wang 2007) and related technology like machine learning (Pham and Afify 2005) and deep learning (Wang et al. 2018) are applied to analyze and extract useful information from discrete data. While for electric files (e.g. CAD design, program), because they already contain useful information, their management (e.g. sharing, storage, processing) becomes the main concerns.

Some researches that using data-driven strategy have been done to improve CNC intelligence in various aspects such as thermal error compensation (Lee et al. 2005) and control technological parameters adjustment (Yury and Abramov 2011). We can summarize that, the intelligence means CNC machine tool has ability to read and understand the external inputs and respond to the inputs through independent thinking.

In the current environment of changing market and customized product demands (Gui et al. 2002), process planning plays an important role among the manufacturing process. An effective and efficient process planning can shorten the

✉ Tianliang Hu
tlhu@sdu.edu.cn

1 School of Mechanical Engineering, Shandong University, Jinan, China

2 Key Laboratory of High Efficiency and Clean Mechanical Manufacture (Shandong University), Ministry of Education, Jinan, China

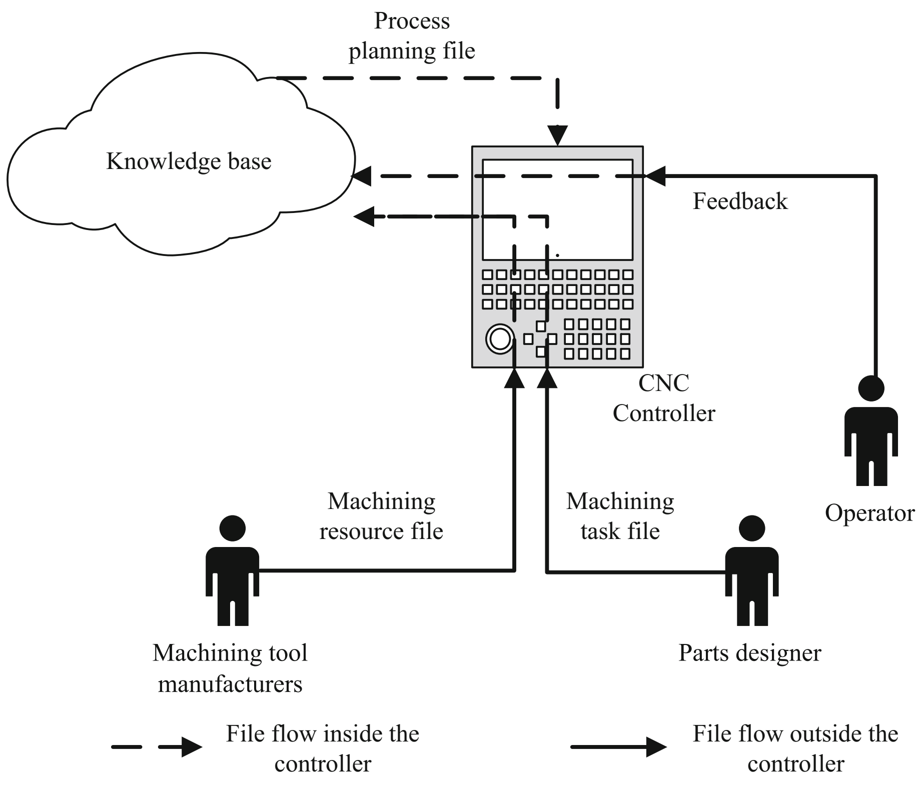


Fig. 1 CNC controller with cloud knowledge base support

production cycle and improve the utilization rate of CNC machine tools. Currently, process planning mostly depends on the experience and expertise of process planners, which means every shop floor level change or workpiece redesign would cost lots of additional preparing time before machining and the time is unstable because of planners' different level. Furthermore, due to the proliferation of terminology, organizations from similar business environments have trouble sharing and accumulating process knowledge (Jardim-Goncalves et al. 2011), which impedes the promotion of the whole industry. With the development of distributed manufacturing (Tao et al. 2010, 2012), if process knowledge could be unified in a understandable format and be collected through this distributed network, and then CNC controller could make process planning intelligently with the support of existed process knowledge. From this perspective, problems mentioned before would be solved simultaneously. To realize this assumption, three main challenges are listed as follows:

- A uniform process knowledge model needs to be designed, which is understandable by machine and convenient to transmit through internet.
- A suitable database needs to be found and designed properly to fulfill the large-scale process planning knowledge storage.
- An efficient way needs to be found to make process planning autonomously based on the large-scale process knowledge.

This paper proposed a knowledge-based process planning method for CNC controller to face these challenges. As shown in Fig. 1, the cloud knowledge base is considered as the brain of intelligent CNC controller, and takes main responsibility for knowledge storage and process planning. Compared with other knowledge based process planning method or expert system (Li et al. 2011; Liu et al. 2006), knowledge base in this paper mainly depends on instance knowledge rather than rule knowledge, which means its process planning ability can be developed with the instances accumulation. The utilization of cloud computing offers powerful storage and processing ability for collected process knowledge, which means that, the knowledge base has ability to deal with large scale knowledge accumulation and thus extendable enough for knowledge update.

This paper is organized as follows. "Related works" section introduces some efforts to realize the intelligent process planning. "Interactivity between intelligent CNC controller and knowledge base" section illustrates how intelligent CNC controller cooperates with cloud knowledge base. In "Knowledge acquisition flow of process planning" section, querying and inference mechanisms of cloud knowledge base are introduced to show the process planning generation flow. Then, a case study of shafts machining is given in "Case study" section to demonstrate the feasibility of the process planning method. Conclusion and future work is discussed in "Conclusion and Future work" section.

Related works

To realize the intelligence of CNC controller from process planning aspect, some researches have been done from two aspects. Besides reviewing these researches, previous work of our group about the realization of intelligent CNC controller is also introduced in this section.

Data interactivity aspect

STEP-NC (ISO 14649) was proposed to offer a consistent data model from product design stage to the manufacturing stage (Nassehi et al. 2006), which makes it possible for CNC to recognize the upstream design. Besides workpiece, operation, working step etc. already defined in STEP-NC, resource model is also defined based on STEP to improve the manufacturing information model (López-Ortega and Moramay 2005). Based on the STEP-NC data model, reformations have been carried out both to STEP-compliant process planning and STEP-compliant manufacturing (Xu et al. 2006).

STEP-compliant CAPP/CAM system was developed to generate STEP-NC file. Features were either manually created in a brand-new CAPP system (Yusof and Case 2008) or based on the secondary development of commercial software like CATIA (Xiao et al. 2015), which provided more intuitive graphical interface. Meanwhile, researches focused on STEP-compliant CNC controller were carried out to interpret and execute STEP-NC file (Zhang 2008; Calabrese and Celentano 2007; Huang 2010; Liang and Li 2013). Besides, there are also solutions for the integration of CAx (CAD/CAE/CAM/CAPP) and CNC system. An integrated and interoperable platform named INFELT STEP was proposed to overcome the communication problem between various CAx software and CNC system (Valilai and Houshmand 2010). Suh et al. developed a STEP-compliant CNC system for turning (Suh et al. 2006) to generate and interpret STEP-NC file using NPSG (nonlinear process sequence graph), which was proposed by their group to make CNC system more intelligent.

These researches made contributions for the portability and interoperability between CAx system and CNC system, making workpiece design and process planning understandable by CNC system. However, due to the information lack of surface roughness and geometric tolerance in STEP-NC, it could not provide enough information for intelligent manufacturing.

Process planning optimization aspects

Some optimizations have been done to improve the performance of CAPP/CAM. Reasonable suggestions or machining estimations were provided by CAPP/CAM system and a new framework was developed to make process planning adapt to the changeable shop floor environment.

Safaieh et al. proposed a system named Cross-Technology CNC Interoperability System (XTSys) (Safaieh et al. 2013). XTSys took cutting tools into consideration to decide a feasible operation for its corresponding feature based on logic rules. Amaitik et al. developed an intelligent process planning system entitled ST-Feat CAPP which utilized various algorithms for different aspects of process planning (Amaitik and Kiliç 2007). Ridwan et al. developed a three-module system, which can optimize the feed-rate based on estimated cutting force (Ridwan et al. 2012). Liu et al. built typical process template based on rules, feature was the characteristic to classify various templates, then template would be selected according to similar feature (Liu et al. 2014). Turley proposed an approach to generate a process plan based on geometric reasoning (Turley et al. 2014). Deb et al. applied neutral network to generate an operation sequence based on the machining features (Deb et al. 2006). To make it convenient for process planners modifying the process planning for a certain optimization objective, roughness estimation (Zhang et al. 2011), energy consumption estimation (Borgia et al. 2013) and manufacturing cycle time and cost estimation (Allen et al. 2005) were added into CAPP system based on computational formula and empirical data.

Facing the uncertain shop floor environment, Wang proposed a two-layer system architecture, which included generic supervisory planning and machine-specific operation planning (Wang 2015). Through this architecture, adjustments on shop floor level can be realized according to the generic supervisory planning because it is machine-irrelevant. This framework is in accordance with adaptive process planning (APP), which has ability to communicate with CNC controller and does dynamic decision based on the seamless information flow. Other aspects of APP like setup planning (Wang et al. 2010) and process planning optimization based on industrial environment (Singh and Jebaraj 2005) were also researched.

These researches provide various directions to improve the intelligence of process planning, however, they are not designed for intelligent CNC controllers. Most of their outputs still are G-code, which cannot deliver workpiece design information to CNC controller. Besides, knowledge develops every day, no feedback or no update means no improvement. These systems are not open enough for knowledge collection, which would result in knowledge rigidify.

To realize intelligence of CNC controller from process planning aspect, an ideal scenario (shown in Fig. 1) is that, the machining task (hereinafter referred to as task), machining resource (hereinafter referred to as resource)and process planning solution are splatted in different files. Then CNC controller makes process planning on the shop floor level according to the task and its processing capacity to show the

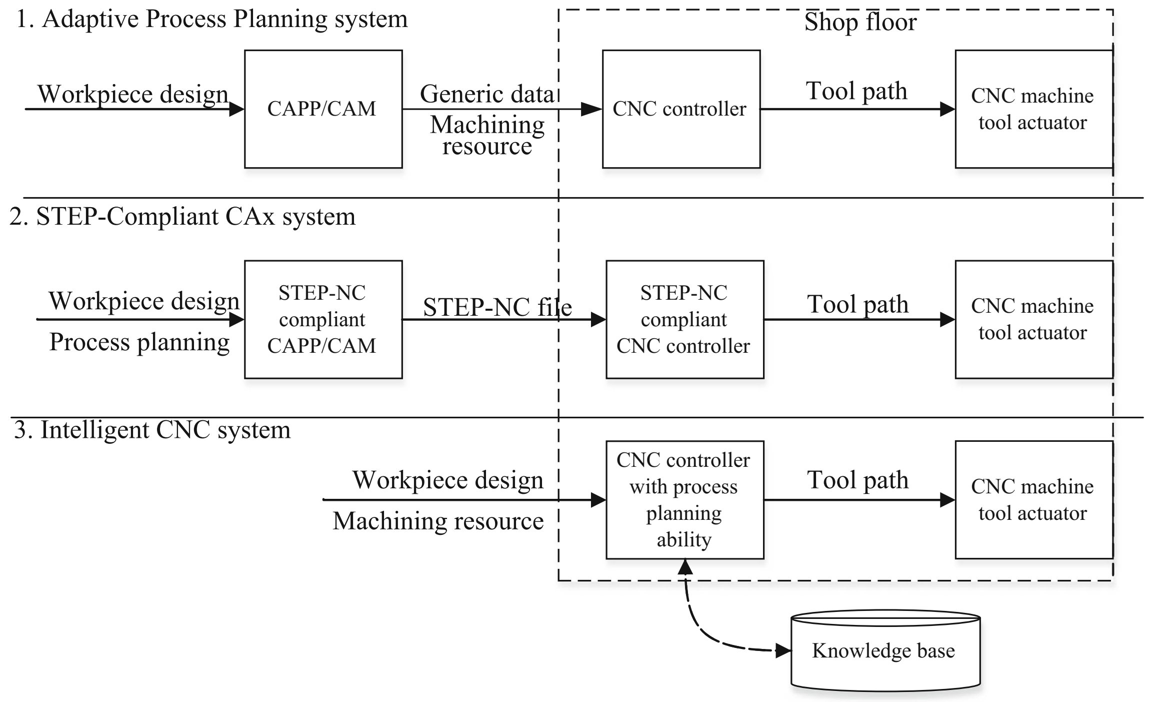

1. Adaptive Process Planning system
Shop floor
Workpiece design
CAPP/CAM
Generic data
Machining resource
CNC controller
Tool path
CNC machine tool actuator
2. STEP-Compliant CAx system
Workpiece design
Process planning
STEP-NC compliant CAPP/CAM
STEP-NC file
STEP-NC compliant CNC controller
Tool path
CNC machine tool actuator
3. Intelligent CNC system
Workpiece design
Machining resource
CNC controller with process planning ability
Tool path
CNC machine tool actuator
Knowledge base

Fig. 2 Differences between STEP-complaint CAx system, APP system and intelligent CNC system

entire intelligence of CNC system. The differences between STEP-complaint CAx System, APP system and intelligent CNC system are shown in Fig. 2.Compared with traditional CAPP/CAM, the two-layer structure of APP provides generic data (machining method, machining sequence, and machining strategy) rather than G-code to allow alternative CNC machine tools executing the task. While STEP-NC file not only contains the neutral process planning information but also includes the workpiece description to allow CNC controller knowing what it machines. However, both of them provide process planning information for CNC controller, which impedes CNC controller showing its intelligence in process planning.

Previous work

As shown in Fig. 2, intelligent CNC controller needs knowledge support to do intelligent process planning. In previous work, an ontology knowledge model was built and the framework of cloud process knowledge base was designed according to this model (Ye et al. 2018).

Knowledge model design

The ontology process knowledge model contains three important parts and their relationships are shown in Fig. 3. Feature and cutting tool are the overlap definition to make process planning connect the other two parts. Not all instances of these parts would be stored in the knowledge base, instances to be stored only contain workpiece design and its corresponding process planning, known as the project. While the resource instance is used as a filter to select out the process planning within the scope of its processing capacity. Both the concept and instance are described using Ontology Web Language (OWL), which is understandable for human and machine.

Knowledge base framework design

Based on this model, cloud knowledge base framework was designed from five modules as shown in Fig. 4. Considering the large amount of instances storage and processing, this knowledge base was built on Hadoop platform, which allows for distributed processing of large data set. HBase, which supports distributed storage and column dynamically added, is selected as data base.

Each module is designed as following:

- Input File Dispose Module: interprets input xml files and classifies them. If inputs are task and resource files, information in these files will be distilled to generate a task and added to the task queue waiting for query. If input is knowledge file, information in this file will be stored in the data base.
- Query Engine: inquiries similar workpieces to the one in task file and checks whether their corresponding solu-

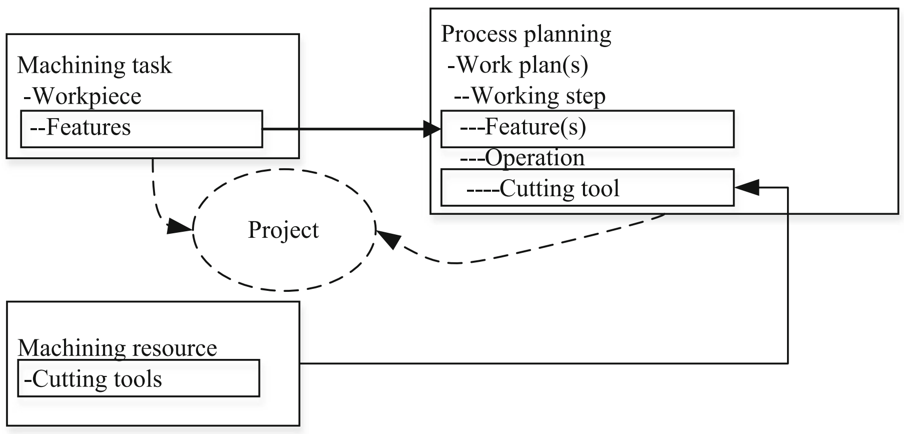


Fig. 3 Relationship between the three parts of model

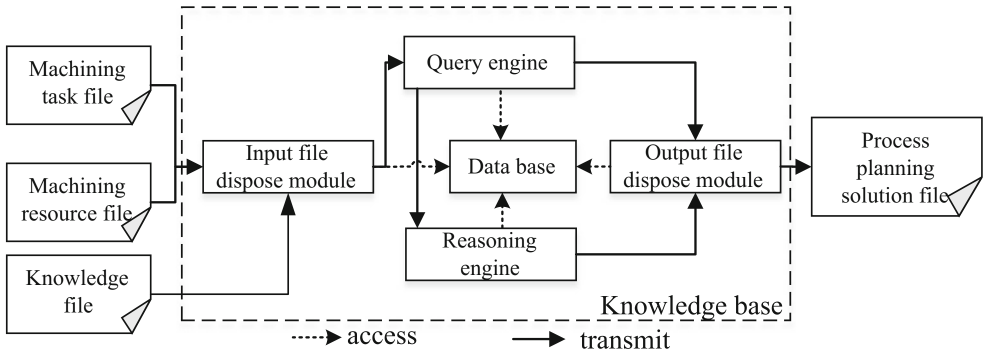


Fig. 4 Framework of knowledge base

tions can be directly applied. Besides, two mappings between global id (used for identifying instances in the scope of knowledge base) and local id (used for identifying instances in the scope of machine tools) are generated on both feature and cutting tool aspects.

- Reasoning Engine: undertakes two types of reasoning based on the query result and simple rules. One infers on the workpiece level, the other infers on the feature level.
- Data Base: stores project instances, two tables are designed separately for workpiece related storage and solution related storage. The ontology knowledge model is stored as its original ontology file in the HDFS, which is used for checking semantic consistency and showing relationship between classes.
- Output File Dispose Module: accepts the output from query engine or reasoning engine. This module accesses the database to acquire information and replace global id to local one, then construct the final solution file in the XML format.

Based on previous work of our group, this paper mainly focuses on the interactivity between CNC controller and knowledge base, and the inner process planning flow which depends on Query Engine and Reasoning Engine design.

Interactivity between intelligent CNC controller and knowledge base

The cloud process knowledge base is considered as the brain of this intelligent CNC controller, which aims at making process planning to meet the machining requirement. Files are transferred through internet as an information carrier between CNC controller and knowledge base. CNC controller submits the resource file and task file to the knowledge base, while the knowledge base outputs process planning solution file to CNC controller. These files can be considered as the instantiation of ontology knowledge model. After applying this solution, users can give feedback through CNC controller to assess the solution from several aspects, like efficiency, cost, quality, which is not the emphasis of this paper. The following sections describe the input and output file in detail.

Input of process knowledge base

As the brain of intelligent CNC controller, knowledge base is not exposed to users. All input files should go through CNC controller and then transport through Internet to the knowl-

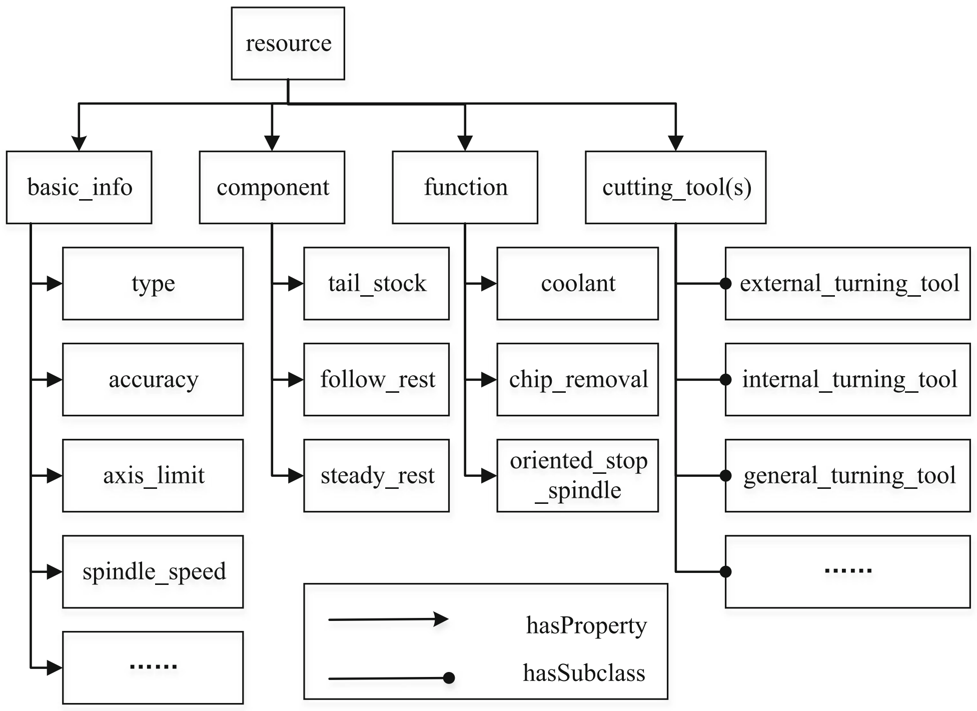


Fig. 5 Framework of resource

edge base. CNC controller not only takes the responsibility to offer a HMI but also transform the file format to make it suitable for network transmission. The file content is introduced following.

Machining resource file

In ideal conditions, the resource file is provided by the machine tool manufacture in a unified format, which exists exclusively in one machine tool.

The resource file aims at describing the machine tools' processing capacity both in hardware and software level. This file is used as a filter in the knowledge base to move out the process beyond resource's processing capacity. Four parts constitute the content of machining resource file, which are shown in Fig. 5.

Definitions of the four parts are listed following:

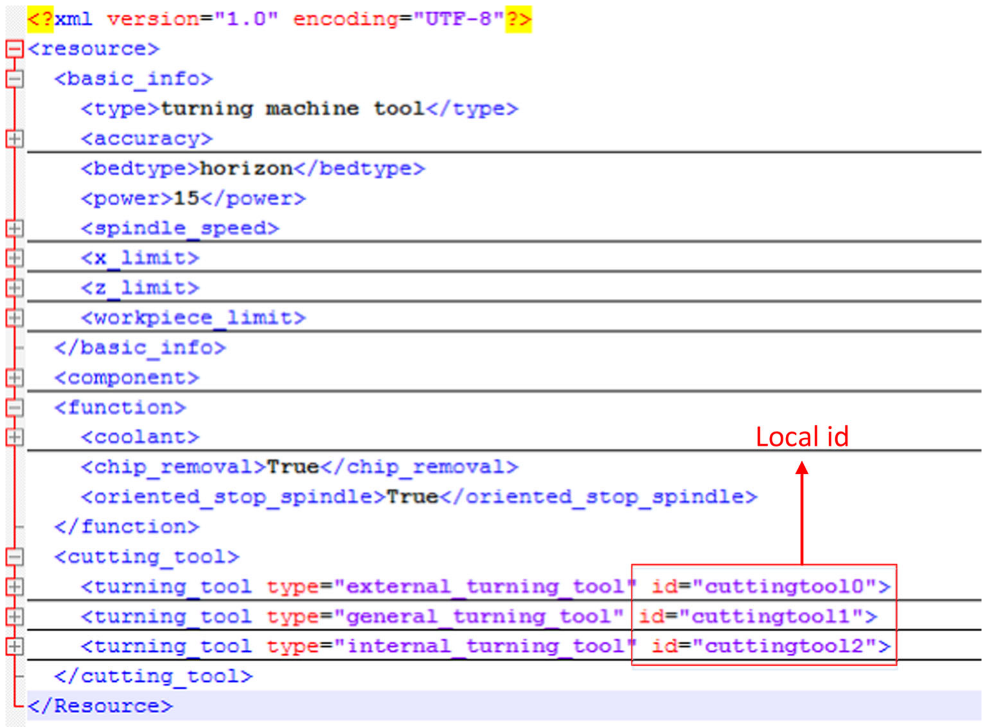
- basic_info: describes the machine tools' performance parameter. This part restricts the cutting parameters like feed-rate and spindle speed, and filters the workpiece, which are overweight or over length.
- component: indicates the fixture and the auxiliary tool installed on the machine tool. This part restricts some special workpiece's processing like slender shaft.
- function: indicates the functions that CNC controller owns. This part restricts some feature's processing as deep hole's machining.
- cutting tool: describes the cutting tool installed on the machine tool. This part restricts some feature's processing when parameter of cutting tool cannot meet the machining requirement.

The CNC controller communicates with knowledge base by using Internet, considering the transmission efficiency and readability, this file is written in the XML format and shown in Fig. 6. Only cutting tool has "id" to specify its usage in the final process planning solution.

Machining task file

As mentioned in "Related works" section, there are two methods to edit a feature-based workpiece and this research chooses creating features directly using new developed software. In this software, workblank is simplified to a bar stock and only turning features are supported. Milling related content will be added subsequently to improve the software. Features can be created in sequence with some basic parameter, and then workblank is defined to specify the material removal part. Because position tolerance may refer to other features' element as datum, this requirement needs to be edited after all features' creation.

The content of task file is shown in Fig. 7, which is in accordance with the task part of the ontology knowledge model.



```
<?xml version="1.0" encoding="UTF-8"?>
<resource>
  <basic_info>
    <type>turning machine tool</type>
    <accuracy>
    <bedtype>horizon</bedtype>
    <power>15</power>
    <spindle_speed>
    <x_limit>
    <z_limit>
    <workpiece_limit>
  </basic_info>
  <component>
  <function>
    <coolant>
    <chip_removal>True</chip_removal>
    <oriented_stop_spindle>True</oriented_stop_spindle>
  </function>
  <cutting_tool>
    <turning_tool type="external_turning_tool" id="cuttingtool0">
    <turning_tool type="general_turning_tool" id="cuttingtool1">
    <turning_tool type="internal_turning_tool" id="cuttingtool2">
  </cutting_tool>
</Resource>
```

Fig. 6 Resource file

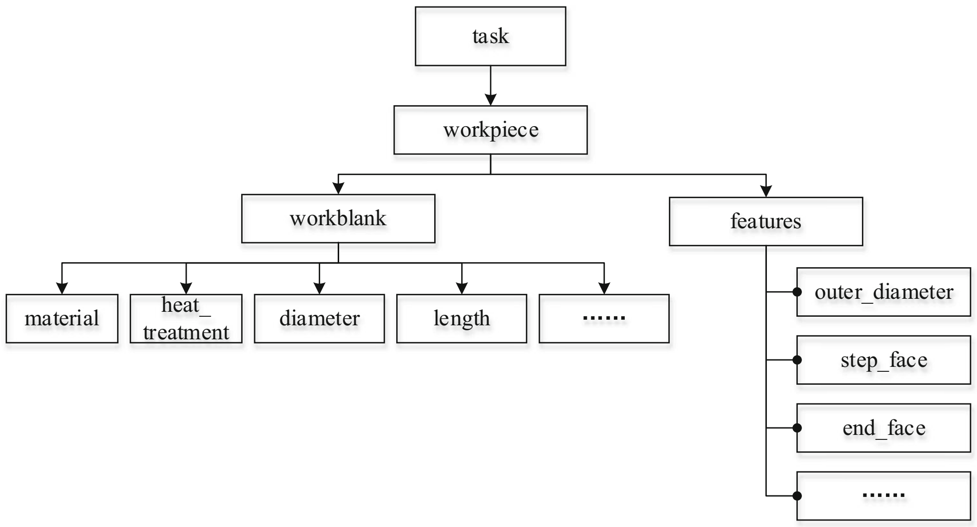
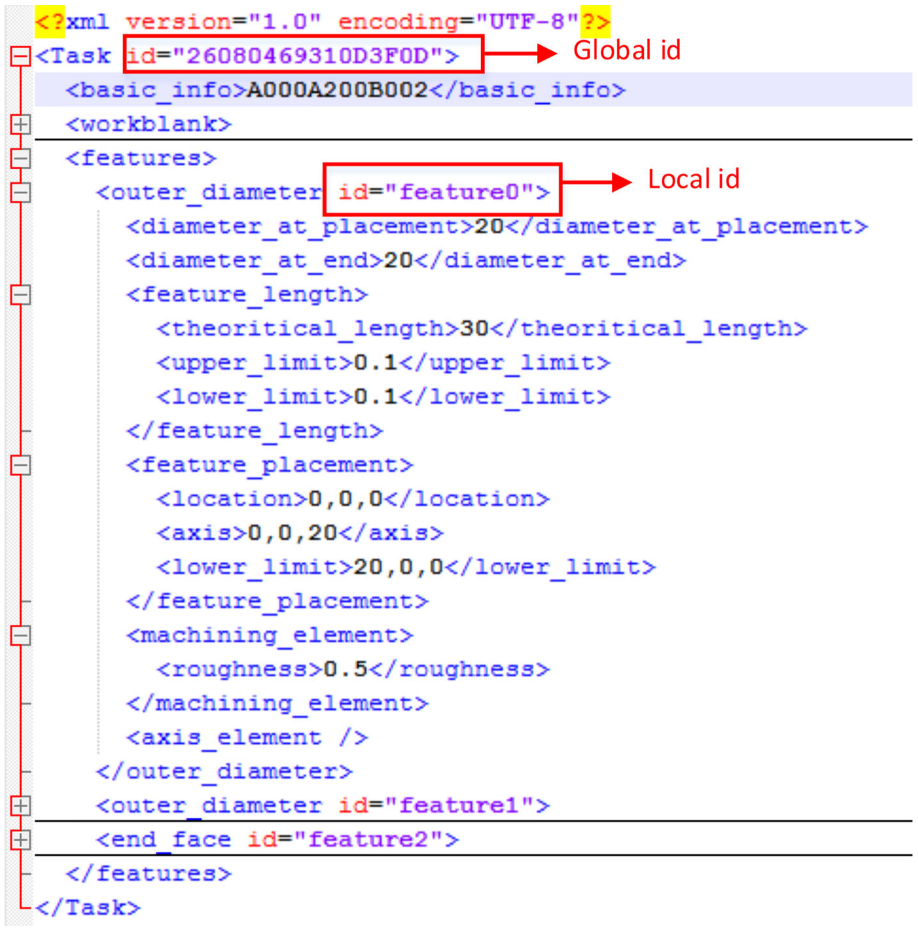


Fig. 7 Content of task

Considering the Internet transmission, this file is also in XML format and shown in Fig. 8. However, in terms of standard information flow from workpiece design to manufacturing, this file should represent the workpiece design compatibly to STEP-NC, which expressed in EXPRESS. As mentioned before, STEP-NC file cannot be utilized directly due to the lack of some machining requirement definition, and it includes the process planning which impedes showing the entire intelligence of CNC controller. Thus a new task model expressed in EXPRESS should be designed, then, a transfer should be carried out by CNC controller between part 21 file and XML file. Currently, related work is in process by our group member, and in this research, task file is submitted to the CNC controller in the XML format temporarily for method verification.

This file doesn't need to be stored in the knowledge base, so the feature id is still in local. But for the purpose to link the process planning solution to specified task, global id is applied in task with a MD5 code. The four high digits indi-



```
<?xml version="1.0" encoding="UTF-8"?>
<Task id="26080469310D3F0D">
  <basic_info>A000A200B002</basic_info>
  <workblank>
  <features>
    <outer_diameter id="feature0">
      <diameter_at_placement>20</diameter_at_placement>
      <diameter_at_end>20</diameter_at_end>
      <feature_length>
        <theoritical_length>30</theoritical_length>
        <upper_limit>0.1</upper_limit>
        <lower_limit>0.1</lower_limit>
      </feature_length>
      <feature_placement>
        <location>0,0,0</location>
        <axis>0,0,20</axis>
        <lower_limit>20,0,0</lower_limit>
      </feature_placement>
      <machining_element>
        <roughness>0.5</roughness>
      </machining_element>
      <axis_element />
    </outer_diameter>
    <outer_diameter id="feature1">
    <end_face id="feature2">
  </features>
</Task>
```

Fig. 8 Task file

cate class type, here is "task", while others are generated according to system time and a random number.

The basic_info aims at describing the general shape of the workpiece to inquire a similar workpiece instance in the knowledge base and obtain its corresponding process planning. It consists of several four-digit code in sequence according to feature position's z value. Each feature uses this code to reflect its character; six turning features are taken for example and shown in Table 1.

Output of process knowledge base

The process planning file only describes the process parameters which ensure machining quality. Other process parameters like start point of operation and approach/retract strategy can be decided by CNC controller and thus is not included in this file. The content of process planning file is shown in Fig. 9, while files are shown in Figs. 15 and 16.

Because features and cutting tools have been defined in detail in the task and resource file, only their ids are referred in process planning solution file to point out which features or cutting tools are included. The local cutting tool id comes from the resource file stored in the CNC controller, while the feature id comes from a specific task file whose global task id is referred in the solution file.

If there was a workpiece stored in knowledge base is very similar to the one in task file, and the resource can support its corresponding process planning solution, then the "ref_solution" would be the referred solution's id. Otherwise, the solution would be composed by inferred working steps and thus the "ref_solution" is null. This definition is used to guarantee user's feedback acting on the original solution stored in the knowledge base. When "ref_solution" exists, users can evaluate this solution from three aspects: efficiency, cost and quality. Then knowledge base inquiries "ref_solution" id from solution table and stores these evaluation to corresponding cells. New solutions are also welcomed, users can improve the solution provided by the knowledge base and submit it as a new one.

Knowledge acquisition flow of process planning

After the task file and resource file are submitted to the knowledge base and processed by the input file dispose module, a

Table 1 Feature code

Feature type	Basic	Chamfer location			Chamfer type		
		None	Left	Right	None	Slope	Round
outer_cylinder	A0	0	1	2	0	1	2
outer_cone							
$75 \leq$ taper < 90	A1	0	1	2	0	1	2
$45 \leq$ taper < 75	A2	0	1	2	0	1	2
$0 <$ taper < 45	A3	0	1	2	0	1	2
Feature type	**Basic**	**Transition location**			**Transition type**		
step_face	C0	0	1	2	0	1	2

Feature type	Basic	Location		Not defined
		Left	Right	
end_face	B0	1	2	0

Feature type	Basic	Axial	Radial	Not defined
Groove	D0	1	2	0

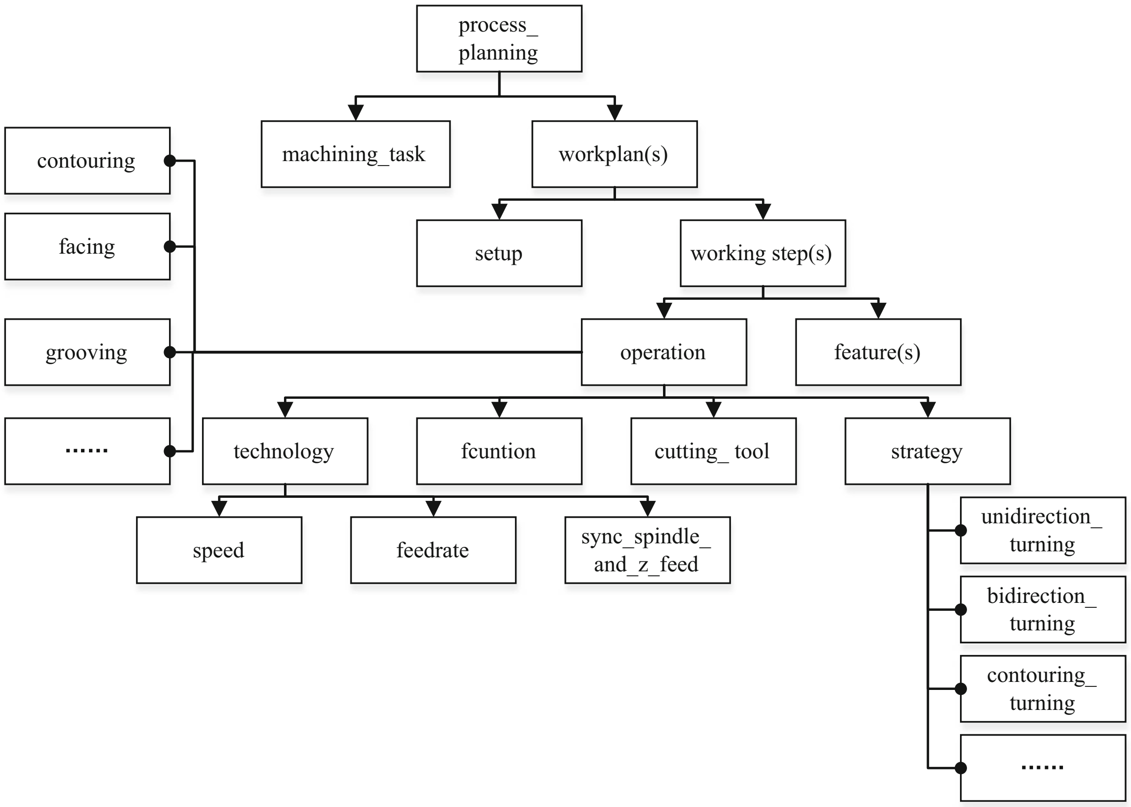
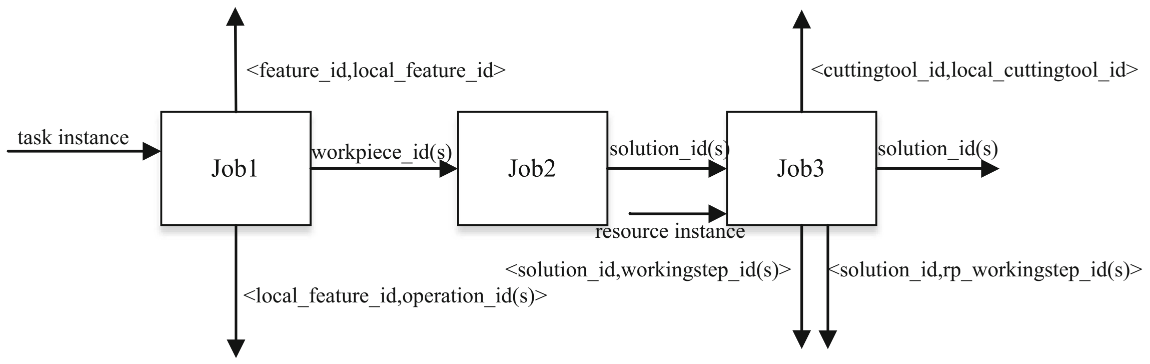


Fig. 9 Content of process planning

knowledge acquisition task would be pushed to a queue and be executed in sequence. This section mainly introduces the algorithms in the query engine and reasoning engine, and the relationship between them.

Workflow of query engine

There are three MapReduce Jobs executed in sequence in the query engine. The input and output are shown in Fig. 10.



<feature_id,local_feature_id>
<cuttingtool_id,local_cuttingtool_id>
task instance
Job1
workpiece_id(s)
Job2
solution_id(s)
Job3
solution_id(s)
resource instance
<solution_id,workingstep_id(s)>
<solution_id,rp_workingstep_id(s)>
<local_feature_id,operation_id(s)>

Fig. 10 Job flow in query engine

Table 2 Two mappers of Job1

Instancetype	Mapper	Property	Requirement
Workblank	M1	Length	1. Difference of length is no more than 50 mm
		Radius	2. Length and diameter ratio is similar (difference is no more than 0.5)
		Material	Exactly the same
Feature	M2	Type	Exactly the same
		Roughness	Equal or less than it in the task
		tolerance_type	Exactly the same
		tolerance_value	Equal or less than it in the task
		Code	Exactly the same
		Material	Exactly the same

Workpiece table in the data base is set as the data source of *Job1* and *Job2*, while solution table is the data source of *Job3*. The detailed query flow is introduced as following.

- Job1

To make it convenient for acquiring workpiece id from feature and workblank, qualifier “belonged_workpiece” is added to the workpiece table which stored the workpiece id. Two mappers are set separately to determine the similarity of workpiece from different aspects as shown in Table 2.

Each mapper outputs the workpiece ids which fulfill the requirement. Then in the reduce phase, through counting the workpiece id, only the workpiece id with count value 2 is written in the output file. That means, the final workpieces are the intersection of former map outputs and fulfill requirements of two aspects.

We can see from Fig. 10 that three files are output by*Job1*, the other two are generated in the M2 process without going through reducer. As each feature has a property “its_operation”, when a feature in the data base fulfills the requirement, outputting its corresponding information as the key-value $<$ global feature id, local feature id $>$ to one file, and $<$ local feature id, operation id(s) $>$ to the other file.

- Job2

This job aims at restricting the workpiece from the whole shape, which means the value of “basic_info” should be exactly the same with the task one. Only filtered workpiece of*Job1* will be accessed and when their basic information fulfills the requirement, acquiring the data in the “its_solution” and outputting available solution ids.

However, at the end of*Job2*, there may be a limit case that no workpiece in the data base is similar with the one in task, and thus no solution is provided to the *Job3*. Reasoning engine is utilized in this situation, and this is illustrated in the 4.2.2.

- Job3

As shown in Fig. 9, one workingstep links one operation, and one operation links one cutting tool, that means workingstep and cutting tool has one-to-one relationship. For the purpose to link cutting tool to workingstep, “its_workingstep” is added to the cutting tool in the database.

In actual, *Job3* consists of several MapReduce Jobs which are executed in sequence without reduce phase. After taking solutions provided by *Job2* as input, the MapReduce Job outputs key-values files in the sequence of $<$

Table 3 Two mappers of Job3

Instancetype	Mapper	Property	Requirement
cutting_tool	M1	Type	Exactly the same
		Material	Exactly the same
		side_cutting_edge_angle	Difference value is no more than 3°
		end_cutting_edge_angle	Difference value is no more than 3°
		dual_direction	Exactly the same
		nose_radius	Exactly the same
		edge_inclination_angle	Same as the positive or negative
Technology	M2	Spindle speed	No more than the max spindle speed
		Feedrate	No more than the max feedrate

Table 4 Assumed solutions

	Solution1	Solution2
Workingstep1	**face_rough(F1, T1)**	**face_rough(F1, T1)**
Workingstep2	contour_rough(F2, F3, T4)	**contour_rough(F2, F3, T2)**
Workingstep3	**contour_finish(F2, F3, T3)**	contour_finish(F2, F3, T5)

solution_id, workplan_id(s) $>$, $<$ solution_id, workingstep _id(s) $>$, $<$ solution_id, operation_id(s) $>$, $<$ solution_id , cuttingtool_id(s)> and $<$ solution_id, (spindle,feedrate) $>$.

Two mappers are set in final Job which takes $<$ solution_id, cuttingtool_id(s) $>$ and $<$ solution_id, (spindle, feedrate) $>$ as input. The former one is used to check cutting tool, and the other is used for technology information check. Both of them output solution ids, which can fulfill the requirement of resource. Table 3 lists the requirements of these two aspects.

For $M1$, if cutting tools used in the solution cannot be all mapped to the cutting tools of resource, this solution cannot be utilized directly and thus not output by this mapper. The workingstep,which is linked to the unmapped cutting tool is output as key-value $<$ solution_id, rp_workingstep_id(s) $>$. While workingstep which is linked to the mapped cutting tool is also output as $<$ solution_id, workingstep_id(s) $>$. Besides, In the process of determining the similarity between global cutting tool and local one, as long as they matches each other, a key-value $<$ cuttingtool_id, local_cuttingtool_id $>$ is output in a file for the subsequent use.

The reduce phase is similar to the *Job1*, final solutions are intersection of these two mapper outputs. The two files include solutions and their corresponding working steps are also filtered by the output of M2 to make sure solutions in this file meets the technology requirement. At the end of *Job3*, either there are available solutions can be directly applied, or no available solution is output by this job. In the first situation, available solution ids, and the map of features and cutting tools are transmitted to the output file dispose module. While the second situation needs some reasoning which is introduced in the 4.2.1.

Workflow in the reasoning engine

As mentioned in the 4.1, there are two types of solution need reasoning engine. One is that workpiece is similar with the task one, but due to the cutting tool limitation, the corresponding solution cannot be directly applied. The other is that no similar workpiece stored in the data base, then operations should be inferred separately according to the feature and constitute the solution based on rules for working step sequence.

Reasoning from workpiece

In this situation, solution can be constituted from working step level without considering the sequence planning. Provided that, resource offers three cutting tools of T1, T2, T3 while the workpiece of task has three features F1, F2, F3. Two solutions are listed in Table 4.

Both of the solutions cannot be applied directly because cutting tools cannot be all mapped to the local ones. However, through combining their working steps (as bold in the table), a new solution can be generated under the support of resource. As solution already provides a suitable machining sequence, this type of reasoning aims at replacing the unavailable working step without disorder the sequence. Only solution table is set as the data source of this reasoning type.

The input and output of this type of reasoning are shown in Fig. 11, and detailed query flow is introduced as following.

- Job1

As shown in Fig. 11, all the input files are generated by the query engine. Input solutions are those that fulfill the

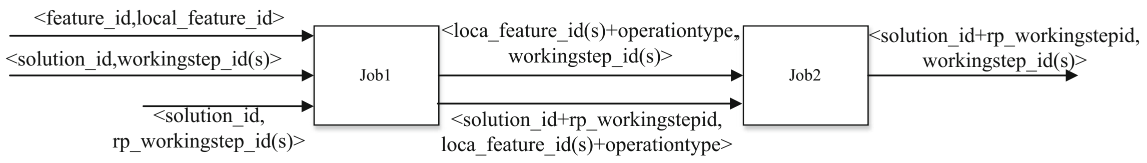


Fig. 11 Job flow in reasoning engine of type1

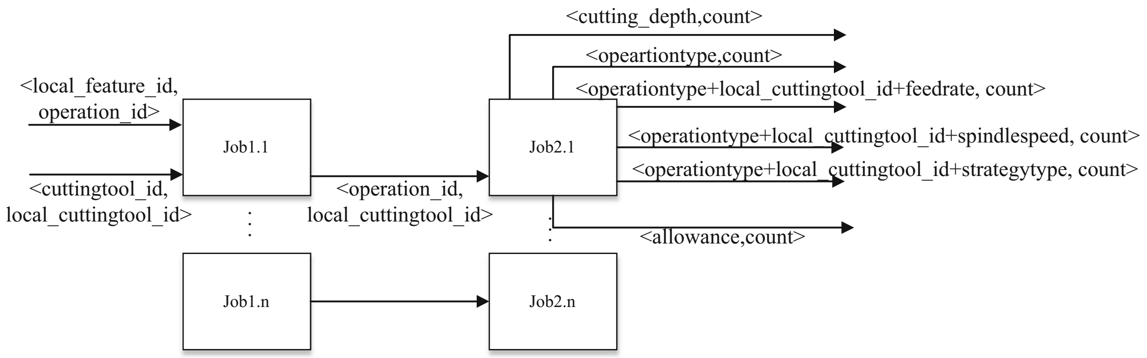

<local_feature_id, operation_id>
<cuttingtool_id, local_cuttingtool_id>
Job1.1
Job1.n
<operation_id, local_cuttingtool_id>
Job2.1
Job2.n
<cutting_depth,count>
<opeartiontype,count>
<operationtype+local_cuttingtool_id+feedrate, count>
<operationtype+local_cuttingtool_id+spindlespeed, count>
<operationtype+local_cuttingtool_id+strategytype, count>
<allowance,count>

Fig. 12 Job flow in reasoning engine of type2

technology requirement and output by *Job3* of query engine. Two mappers exists, one is for available working steps, the other is for unavailable working steps. $M1$ links local feature and operation type to the working steps that can be used for replacement. Through accessing available working steps, $M1$ acquires the corresponding features and operations. Then based on the feature map, output key-value in the format like $<$ F1F2_facingrough, ABD3A3F672B068F7 $>$. While in the reduce phase, working steps with the same key will be gathered and output key-value like $<$ F1F2_facingrough, ABD3A3F672B068F7ABD35FB9488F3D06 $>$. In $M2$, unavailable working steps are also accessed and based on the feature map to output key-value in the format like $<$ ABD3C8BAE8B791A1_ABD33778784175FD, F1F2_facingrough $>$.

- Job2

We can see in *Job1* that, the key of $M1$ output is the value of $M2$ output. Therefore, *Job2* can be considered as a join operation of two files that output by *Job1*, and thus there is no need for data base access. Finally, as long as the key has value, it means unavailable working step can find replacement to make its corresponding solution available.

At the end of *Job2*, available solutions with its working steps to be replaced and optional working steps, and the map of features and cutting tools are transmitted to the output file dispose module.

Reasoning from feature

In this situation, there is no similar workpiece in the data base, which means operations should be inferred from features. Each feature of workpiece in task should start-up one job to infer available operations, which is executed parallel. The input and output of this type of reasoning are shown in Fig. 12, and detailed query flow is introduced as following.

The input operations in *Job1* are acquired from the stored features which are similar to the those defined in task file, without checking availability of cutting tools. Through comparing the cutting tool map, *Job1* outputs operations that can be executed by the local resource. Then *Job2* counts up several important parameters (shown in Fig. 12, output of Job2) through accessing these operation instances. Parameters like allowance, cutting depth and other numerical related value is not restricted to a number but a suitable range. While for the parameters like operation type and strategy type, the two most common types would be selected (rough and finish operation type counts separately). Then, at the end of *Job2*, available operations for each local feature can be determined and corresponding class in Java program instantiate several instances.

In order to organize features that can be processed in one working step and sort working steps in a sequence, some simple rules are listed in Table 6 to introduce how to construct working step and work plan.

All these rules are utilized in the knowledge base programming, a class named FeatureForReasoning is designed for organize this rule-related information. Then, these feature instances are stored in a list in sequence according to

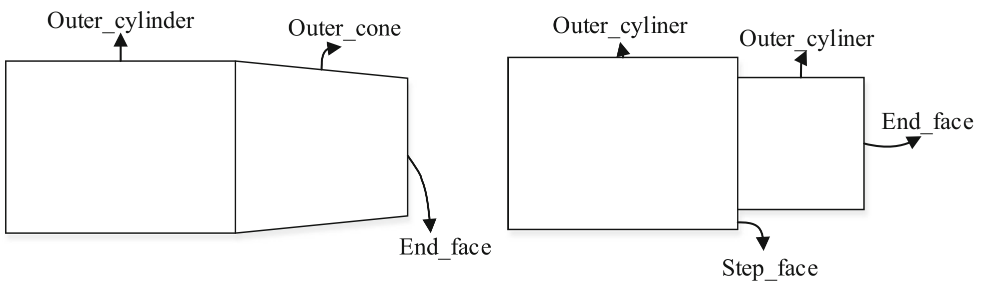


Fig. 13 Two types of workpiece stored in the knowledge base

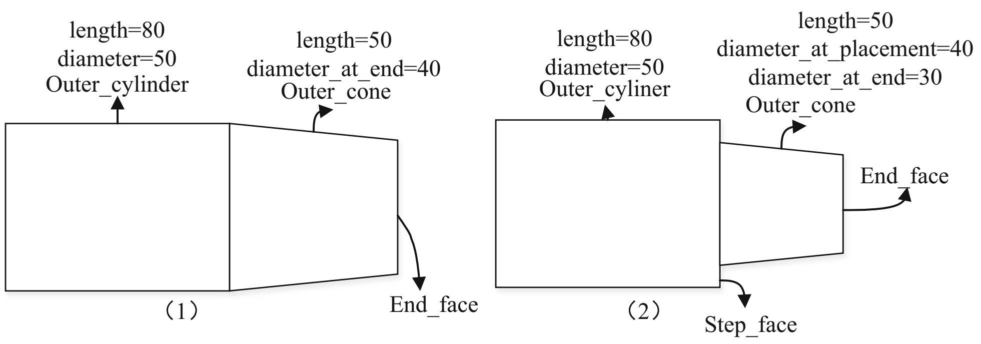


Fig. 14 Two pieces of workpiece defined in the task file

their position's z value. Through traversing this feature list, information can be retrieved to match conditions in the rules and organized as sequenced working steps in a work plan instance. Then, work plan instance is transmitted to the file output dispose module to generate the final solution file.

Case study

To verify the feasibility of the algorithm, knowledge collection is carried out using the project model defined in this research. Considering the efficiency of inputting workpiece and process parameters manually, it is unavailable to generate sufficient entirely different knowledge files that meet the amount of experiment requirement. So, two typical workpieces are introduced in this research with a little parameter adjustment (shown in Fig. 13). Property values like material, feature length or radius are modified in different knowledge files. Besides, process related information like the cutting parameter, strategy, cutting tool are also modified within limits in different files.

In this situation, 20 different entirely different knowledge files of each type are designed. All class instances in knowledge file use class name, system time and a random number as seed to generate a MD5 code as its global id when the instance is written to file. That means, even though parameters of one project do not change, the ids in different files are not same because of the change of system time and the random number.

The software generates 100 copies of the same project design with totally unique global ids. Then 4000 files are submitted to the knowledge base to be stored as the experiment data.

Two tasks are used to test the query and reasoning algorithm separately, their design is shown in Fig. 14. The collected knowledge can make sure that, there are similar workpieces to the first task and similar features to the second task.

Cutting tool resource in this experiment is shown in Table 5.

After submitting the first task and resource, knowledge base provides 400 solutions (actually, there are 4 similar workpieces with available resources stored in knowledge base, but with 100 copies). As there is no feedback currently, evaluation does not work in this experiment to further filter solutions. Fig. 15 is fragment of one file, which focuses on showing one working step planning.

The global solution id identifies the project instance referred from the knowledge base, and the global task id identifies the task submitted through the CNC controller. Four working steps are included and with the help of task and resource files designed in this experiment, we can know that the first working step is rough facing of end face (feature3) using external turning tool (cuttingtool1).

After submitting the second task and resource, knowledge base provides 1 solution. Fig. 16 is fragment of this file, which focuses on showing the working step sequence.

As there is no similar workpiece that can be referred from knowledge base, the "ref_solution" is null, which means all working steps are generated from similar features. Four working steps are clearly listed in sequence with the operation type and features to be processing, which are in consist with the rules defined in Table 6.

Table 5 Cutting tool resource defined in this experiment

Type	hand_of_tool	Material	tool_body _type	side_cuuting_edge_angle	end_cuuting_edge_angle	edge_inclination_angle	dual_direction
external_turning_tool	Left	Medium-carbon steel	Side	95	5	2	True
external_turning_tool	Left	Medium-carbon steel	Side	90	30	0	False
face_turning_tool	Left	Medium-carbon steel	Side	90	30	0	False

Table 6 Rules on working step and work plan design

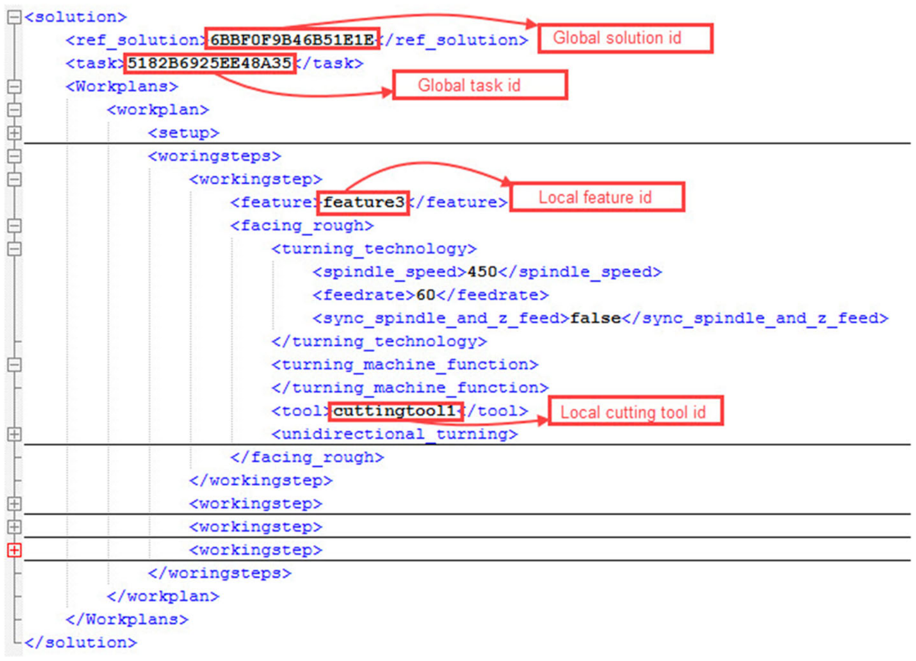
Features in one working step should fulfill:	Working step in one work plan should fulfill (priority decrease)
1. Location is adjacent	1. If features in working step had location tolerance requirement, they should be processed in one work plan
2. Corresponding operation has the same local cutting tool	2. If there were two end faces existed, two work plans are made
3. Corresponding operation has the same type	3. If two work plans are made, features near the end should be separated in two work plans
4. Range of feedrate/ spindle speed/ cutting depth/ allowance of corresponding operation has intersection	4. Do rough machining firstly
	5. “End_face”feature should be processed firstly
	6. “outer_diameter_cylinder” or “outer_diameter_cone” feature with larger diameter should be processed firstly
	7. “Groove” feature machining is after all other features except “cut_in” feature

At last, CNC controller can interpret these process planning solution files with resource file and task file to generate the final tool path and execute it, which is not the emphasis of this paper.

This case study shows that, this knowledge base is open enough to collect knowledge and provide enough space for knowledge storage. Based on the consistent knowledge model, CNC controller interacts with knowledge base through submitting task and resource file, and then knowledge base can generate the process planning solution to CNC controller for subsequent interpreting and executing.

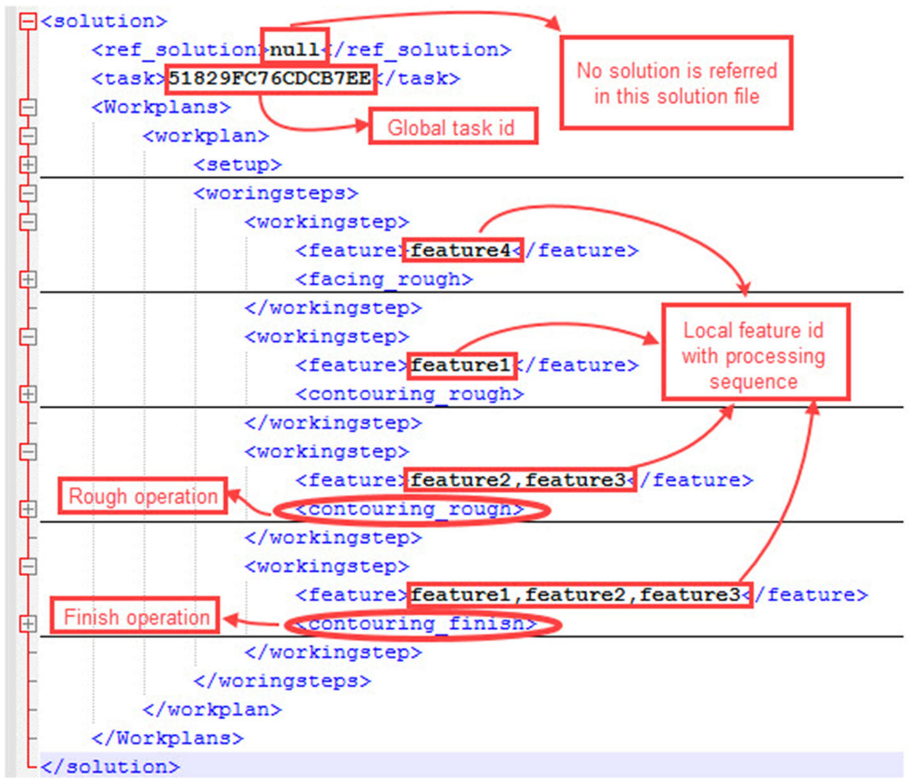
Conclusion and future work

This paper proposed a method for intelligence realization of CNC controller in process planning aspect. The method is based on a knowledge base, which takes advantage of distributed structure to provide theoretical infinite storage space for knowledge storage and parallel processing ability for knowledge query and reasoning. Meanwhile, it is open enough to collect knowledge, which means every sharing knowledge would contribute to the improvement of process planning ability of knowledge base. Then with the support of knowledge base, CNC controller can be considered as having capacity to make process planning autonomously based on the interpretation of machining task and its own processing resource. As this process planning procedure takes place in the CNC controller, users can focus on workpiece design without considering process planning related problem like



```
<solution>
    <ref_solution>6BBF0F9B46B51E1E</ref_solution>
    <task>5182B6925EE48A35</task>
    <Workplans>
        <workplan>
            <setup>
            <woringsteps>
                <workingstep>
                    <feature>feature3</feature>
                    <facing_rough>
                        <turning_technology>
                            <spindle_speed>450</spindle_speed>
                            <feedrate>60</feedrate>
                            <sync_spindle_and_z_feed>false</sync_spindle_and_z_feed>
                        </turning_technology>
                        <turning_machine_function>
                        </turning_machine_function>
                        <tool>cuttingtool1</tool>
                        <unidirectional_turning>
                    </facing_rough>
                </workingstep>
                <workingstep>
                <workingstep>
                <workingstep>
            </woringsteps>
        </workplan>
    </Workplans>
</solution>
```

Fig. 15 Solution file of first task



```
<solution>
    <ref_solution>null</ref_solution>
    <task>51829FC76CDCB7EE</task>
    <Workplans>
        <workplan>
            <setup>
            <woringsteps>
                <workingstep>
                    <feature>feature4</feature>
                    <facing_rough>
                </workingstep>
                <workingstep>
                    <feature>feature1</feature>
                    <contouring_rough>
                </workingstep>
                <workingstep>
                    <feature>feature2,feature3</feature>
                    <contouring_rough>
                </workingstep>
                <workingstep>
                    <feature>feature1,feature2,feature3</feature>
                    <contouring_finish>
                </workingstep>
            </woringsteps>
        </workplan>
    </Workplans>
</solution>
```

No solution is referred in this solution file
Global task id
Local feature id with processing sequence
Rough operation
Finish operation

Fig. 16 Solution file of second task

suitable cutting parameter or resource availability, which will reduce the preparing time under current changeable market environment.

Future work will focus on the multi-CNC controller intelligent process planning. This research assumes one machine tool can carry out the machining task. However, workpiece usually goes through several machine tools from workblank to product in actual. How to consider several resources from the knowledge base aspect, and how to control several machine tools executed collaboratively from the CNC controller aspect would be the long-term future work.

Acknowledgements The work is supported by National Natural Science Foundation of China (Grant No. 51405270).

References

Allen, R. D., Harding, J. A., & Newman, S. T. (2005). The application of STEP-NC using agent-based process planning. *International Journal of Production Research*, *43*(4), 655–670.

Amaitik, S. M., & Kiliç, S. E. (2007). An intelligent process planning system for prismatic parts using STEP features. *International Journal of Advanced Manufacturing Technology*, *31*(9–10), 978–993.

Borgia, S., Leonesio, M., Pellegrinelli, S., & Valente, A. (2013). Energy driven process planning and machine tool dynamic behavior assessment. In *Cirp global web conference: Interdisciplinary research in production engineering* (pp. 91–96).

Calabrese, F., & Celentano, G. (2007). Design and realization of a STEP-NC compliant CNC embedded controller. In *IEEE conference on emerging technologies and factory automation* (pp. 1010–1017).

Deb, S., Ghosh, K., & Paul, S. (2006). A neural network based methodology for machining operations selection in computer-aided process planning for rotationally symmetrical parts. *Journal of Intelligent Manufacturing*, *17*(5), 557–569.

Gui, Y. T., Yin, G., & Taylor, D. (2002). Internet-based manufacturing: A review and a new infrastructure for distributed intelligent manufacturing. *Journal of Intelligent Manufacturing*, *13*(5), 323–338.

Huang, X. (2010). Enhancing STEP-NC compliant CNC controller for distributed and reconfigurable environment in production line. In *International Conference on computer, mechatronics, control and electronic engineering* (pp. 106–109).

Jardim-Goncalves, R., Sarraipa, J., Agostinho, C., & Panetto, H. (2011). Knowledge framework for intelligent manufacturing systems. *Journal of Intelligent Manufacturing*, *22*(5), 725–735.

Lee, S. W., Dong, H. K., Song, J. Y., & Lee, H. K. (2005). Agent-based decision support system for realizing intelligent machine tools. In *Wseas international conference on dynamical systems and control* (pp. 85–90).

Li, P., Hu, T., & Zhang, C. (2011). STEP-NC compliant intelligent process planning module: Architecture and knowledge base. *Procedia Engineering*, *15*, 834–839.

Liang, H., & Li, X. (2013). Five-axis STEP-NC controller for machining of surfaces. *International Journal of Advanced Manufacturing Technology*, *68*(9–12), 2791–2800.

Liu, C., Li, Y., & Shen, W. (2014). Integrated manufacturing process planning and control based on intelligent agents and multi-dimension features. *International Journal of Advanced Manufacturing Technology*, *75*(9–12), 1457–1471.

Liu, R., Nassehi, A., Allen, R., & Newman, S. T. (2006). Feature-Based process planning for interoperable STEP-NC manufacture. *Information Control Problems in Manufacturing*, *39*(3), 781–786.

López-Ortega, O., & Moramay, R. (2005). A STEP-based manufacturing information system to share flexible manufacturing resources data. *Journal of Intelligent Manufacturing*, *16*(3), 287–301.

Nassehi, A., Newman, S. T., & Allen, R. D. (2006). STEP-NC compliant process planning as an enabler for adaptive global manufacturing. *Robotics and Computer-Integrated Manufacturing*, *22*(5–6), 456–467.

Pham, D. T., & Afify, A. A. (2005). Machine-learning techniques and their applications in manufacturing. *Proceeding of the Institute of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, *219*(5), 395–412.

Ridwan, F., Xu, X., & Liu, G. (2012). A framework for machining optimisation based on STEP-NC. *Journal of Intelligent Manufacturing*, *23*(3), 423–441.

Safaieh, M., Nassehi, A., & Newman, S. T. (2013). A novel methodology for cross-technology interoperability in CNC machining. *Robotics and Computer-Integrated Manufacturing*, *29*(3), 79–87.

Singh, D. K. J., & Jebaraj, C. (2005). Feature-based design for process planning of machining processes with optimization using genetic algorithms. *International Journal of Production Research*, *43*(18), 3855–3887.

Suh, S. H., Chung, D. H., Lee, B. E., Shin, S., Choi, I., & Kim, K. M. (2006). STEP-compliant CNC system for turning: Data model, architecture, and implementation. *Computer-Aided Design*, *38*(6), 677–688.

Tao, F., Cheng, J., Cheng, Y., Gu, S., Zheng, T., & Yang, H. (2017a). SDMSim: A manufacturing service supply-demand matching simulator under cloud environment. *Robotics & Computer Integrated Manufacturing*, *45*(6), 34–46.

Tao, F., Cheng, J., Qi, Q., Tao, F., Cheng, J., & Qi, Q. (2017b). IIHub: An industrial internet-of-things hub towards smart manufacturing based on cyber-physical system. *IEEE Transactions on Industrial Informatics*, *PP*(99), 1-1.

Tao, F., Cheng, Y., Xu, L. D., Zhang, L., & Li, B. H. (2014a). CCIoT-CMfg: Cloud computing and internet of things-based cloud manufacturing service system. *IEEE Transactions on Industrial Informatics*, *10*(2), 1435–1442.

Tao, F., Guo, H., Zhang, L., & Cheng, Y. (2012). Modelling of combinable relationship-based composition service network and the theoretical proof of its scale-free characteristics. *Enterprise Information Systems*, *6*(4), 373–404.

Tao, F., Hu, Y., & Zhou, Z. (2010). Correlation-aware resource service composition and optimal-selection in manufacturing grid. *European Journal of Operational Research*, *201*(1), 129–143.

Tao, F., Qi, Q., Liu, A., & Kusiak, A. (2018). Data-driven smart manufacturing. *Journal of Manufacturing Systems*.

Tao, F., Zuo, Y., Xu, L. D., & Zhang, L. (2014b). IoT-based intelligent perception and access of manufacturing resource toward cloud manufacturing. *IEEE Transactions on Industrial Informatics*, *10*(2), 1547–1557.

Turley, S. P., Diederich, D. M., Jayanthi, B. K., Datar, A., Ligetti, C. B., Finke, D. A., et al. (2014). Automated process planning and CNC-code generation. In *Industrial and systems engineering research conference*.

Valilai, O. F., & Houshmand, M. (2010). *INFELT STEP: An integrated and interoperable platform for collaborative CAD/CAPP/CAM/CNC machining systems based on STEP standard*. London: Taylor & Francis, Inc.

Wang, J., Ma, Y., Zhang, L., Gao, R. X., & Wu, D. (2018). Deep learning for smart manufacturing: Methods and applications. *Journal of Manufacturing Systems*.

Wang, K. (2007). Applying data mining to manufacturing: The nature and implications. *Journal of Intelligent Manufacturing*, *18*(4), 487–495.

Wang, L. (2015). An overview of function block enabled adaptive process planning for machining. *Journal of Manufacturing Systems*, *35*, 10–25.

Wang, L., Cai, N., Feng, H. Y., & Ma, J. (2010). ASP: An adaptive setup planning approach for dynamic machine assignments. *IEEE Transactions on Automation Science and Engineering*, *7*(1), 2–14. https://doi.org/10.1109/TASE.2008.2011919.

Xiao, W., Zheng, L., Ji, H., & Lei, P. (2015). A complete CAD/CAM/CNC solution for STEP-compliant manufacturing. *Robotics & Computer Integrated Manufacturing*, *31*(C), 1–10.

Xu, X., Klemm, P., Proctor, F., & Suh, S. H. (2006). STEP-compliant process planning and manufacturing. *International Journal of Computer Integrated Manufacturing*, *19*(6), 491–494.

Xu, X. W., Wang, H., Mao, J., Newman, S. T., Kramer, T. R., Proctor, F. M., et al. (2005). STEP-compliant NC research: The search for intelligent CAD/CAPP/CAM/CNC integration. *International Journal of Production Research*, *43*(17), 3703–3743.

Ye, Y., Hu, T., Zhang, C., & Luo, W. (2018). Design and development of a CNC machining process knowledge base using cloud technology.

International Journal of Advanced Manufacturing Technology, *94*(9–12), 3413–3425. https://doi.org/10.1007/s00170-016-9338-1.

Yury, K., & Abramov, I. V. (2011). Building a knowledge base for intelligent control system of mechatronic machining center. In *Mechatronika, 2011 international symposium* (pp. 93–94).

Yusof, Y., & Case, K. (2008). STEP compliant CAD/CAPP/CAM system for turning operations. In *Proceedings of the World Congress on Engineering and Computer Science* (pp. 22–24).

Zhang, C. (2008). A multi-agent-based intelligent STEP-NC controller for CNC machine tools. *International Journal of Production Research*, *46*(14), 3887–3907.

Zhang, X., Liu, R., Nassehi, A., & Newman, S. T. (2011). A STEP-compliant process planning system for CNC turning operations. *Robotics and Computer-Integrated Manufacturing*, *27*(2), 349–356.