OPERATION SEQUENCING & MACHINING PARAMETERS SELECTION FOR ROTATIONAL COMPONENTS USING GENETIC ALGORITHM & EXPERT SYSTEM

A Thesis submitted to

THAPAR INSTITUTE OF ENGINEERING & TECHNOLOGY,

PATIALA

(DEEMED UNIVERSITY)

In partial fulfillment of the requirements for the award of the degree of

MASTER OF ENGINEERING

IN

(CAD/CAM & ROBOTICS)

Submitted By

GURPREET SINGH

Roll No. 8048106

Under the supervision of

MR. GAURAV BARTARYA

Lecturer, Department of Mechanical Engineering

Thapar Institute of Engineering & Technology, Patiala

DEPARTMENT OF MECHANICAL & INDUSTRIAL ENGINEERING

THAPAR INSTITUTE OF ENGINEERING AND TECHNOLOGY

(DEEMED UNIVERSITY)

PATIALA-147004, (INDIA)

June 2006

Dedicated to Almighty

AKAL PURKH

CERTIFICATE

This is to certify that the thesis entitled **“OPERATION SEQUENCING & MACHINING PARAMETER SELECTION FOR ROTATIONAL COMPONENTS USING GENETIC ALGORITHM & EXPERT SYSTEM”** being submitted by **Mr. Gurpreet Singh** in partial fulfillment of the requirements for the award of the degree of **MASTER OF ENGINEERING** in **CAD/CAM & ROBOTICS** to **THAPAR INSTITUTE OF ENGINEERING & TECHNOLOGY (Deemed University), Patiala (India).** The matter embodied in this report has not been submitted in part or full to any other university or institute for the award of any degree.

GAURAV BARTARYA

Lecturer
Mechanical Engineering Department
Thapar Institute of Engineering & Technology
Patiala-147004 (INDIA)

Countersigned by:

(Dr. S.K. MOHAPATRA)

Prof. & Head, MIED
Thapar Institute of Engg. & Tech.
Patiala-147004 (INDIA)

(Dr. T.P. SINGH)

Dean, Academic Affairs
Thapar Institute of Engg. & Tech.
Patiala-147004 (INDIA)

Dated:

ACKNOWLEDGEMENT

I attribute my honors in showing my feelings of indebtedness and thankfulness to Shri Gaurav Bartarya, lecturer, Mechanical Engineering Department, Thapar Institute of Engineering and Technology, Patiala, for his valuable guidance, proper advice and constant encouragement during the course of my work on this thesis.

I am deeply indebted to my parents for their inspiration and ever-encouraging moral support, which enabled me to purse my studies and always teach me that: -

Success Is Becoming Who You
Are Capable Of Being.

And Progressing Toward
Worthwhile Goals.

Each Of Us Defines For Ourselves
What It Means
To Be Successful

I am also very thankful to all the faculty and staff members of Mechanical Engineering Department for their direct and indirect help and cooperation.

(GURPREET SINGH)

ABSTRACT

.

Manufacturing a part, as per the technical specifications in the part drawing and selection of machining operations sequence and machining parameters, is one of most critical activity, in any CAPP system. Operation sequencing in process planning is concerned with selection of machining operations in steps that can produce each form feature of the part by satisfying the relevant technological constraints specified in the part drawing. A single sequence of the operations may not be the best for all situations in a changing production environment with the multiple objectives such as minimizing, number of machine changes, set-up changes, tool changes, and parameter changes. Most of the work in this area however was carried out in determining the operation sequence on a single machine.

Economy of machining operation plays a key role in the competitive market so the selection of efficient machining parameters is of great concern in manufacturing industries. An attempt is made to automate the selection of machining parameters. The work reported here is concerned with application of Genetic Algorithm & Expert System. Genetic Algorithm is used to find the optimal operation sequence because it generates search space of feasible sequences and selects the best ones. Expert System is used for selecting the machining parameters for facing, turning and boring operations for three types of materials. A program is developed in C++ based on proposed algorithm, to check its validity and the results are compared with the previous work.

INDEX

CONTENTS **Page No.**

LIST OF FIGURES

LIST OF TABLES

NOMENCLATURE

Abbreviation	Meaning
AI	Artificial Intelligence
ANN	Artificial Neural Network
CAD	Computer Aided Design
CAM	Computer Aided Manufacturing
CAPP	Computer Aided Process Planning
CIM	Computer Integrated Manufacturing
CNC	Computer Numeric Control
DOS	Disk Operating System
DBM	Data Base Module
E_i	Expected Count
EEM	Empirical Equation Module
ES	Expert System
FMS	Flexible Manufacturing System
FPG	Feature Precedence Graph
GA	Genetic Algorithm
GD	Gang Drill
GT	Group Technology
KB	Knowledge Base
M	Maximum Cost of String
NC	Numeric Control
OPPS	Optimized Process Planning System
OX	Order Crossover
PCM	Precedence Cost Matrix
PD	Pillar Drill
PGSA	Parallel Genetic Simulated Annealing
PMX	Partially Mapped Crossover
SA	Simulated Annealing
U_{avg}	Average Cost of All Strings
$(U_{min})_i$	Total cost if i^{th} string
UIM	User Interface Module
VMA	Vertical Machining Center

CHAPTER 1

INTRODUCTION

1.1 Introduction

Process planning is the function that establish the machining processes, machines and tooling capable of performing these processes, and the machining parameters to be used to convert a piece part from its initial from to a predetermined shape, as per the engineering drawing. [22]

In the new era manufacturing firms have much interest in automating the task of process planning using computer-aided process planning (CAPP) system. The shop-trained people, who are familiar with details of machining and other processes, are gradually retiring, and these people will be unavailable in the future for process planning purposes. Traditionally, a planner who has accumulated practical experience carries out the process planning. By this, if different planners are asked to develop a process plan for the same part, they all should probably come up with a different plan. Obviously, these entire plans cannot reflect the most economic manufacturing method, and in fact there is no guarantee that many one of them will constitute the optimum method for manufacturing this part. In process planning, a planner must manage and retrieves reams of standards and documents, such as old process plan, established standards, facilities data, machine tool specification, tooling inventory and stock availability.

Because of the problem encountered with manual process planning, attempts have been made in recent years to capture the logic, judgment and experience required for this important function and to incorporates them in to the computer programs. Based on the characteristics of the given part, the program automatically generates the manufacturing operation sequence. A Computer-Aided Process Planning (CAPP) system offers the potential for reducing the routine clerical work of manufacturing engineers. At the same time, it provides the opportunity to generate production routings, which are rotational, consistent and perhaps even optimal. The manual process planning highly relies on the expertise of a process planner. Though the CAPP follows almost the same steps taken in the manual process planning, but it requires very short time compared to manual process planning. CAPP optimizes the process planning by using different optimization techniques.

CAPP is usually considered to be part of Computer-Aided-Manufacturing (CAM). However, this tends to imply that CAM is stand-alone system. In fact, a synergy results when CAM is combined with Computer- Aided Design to create a CAD/CAM system. In such a system, a CAPP becomes the direct connection between design and manufacturing.

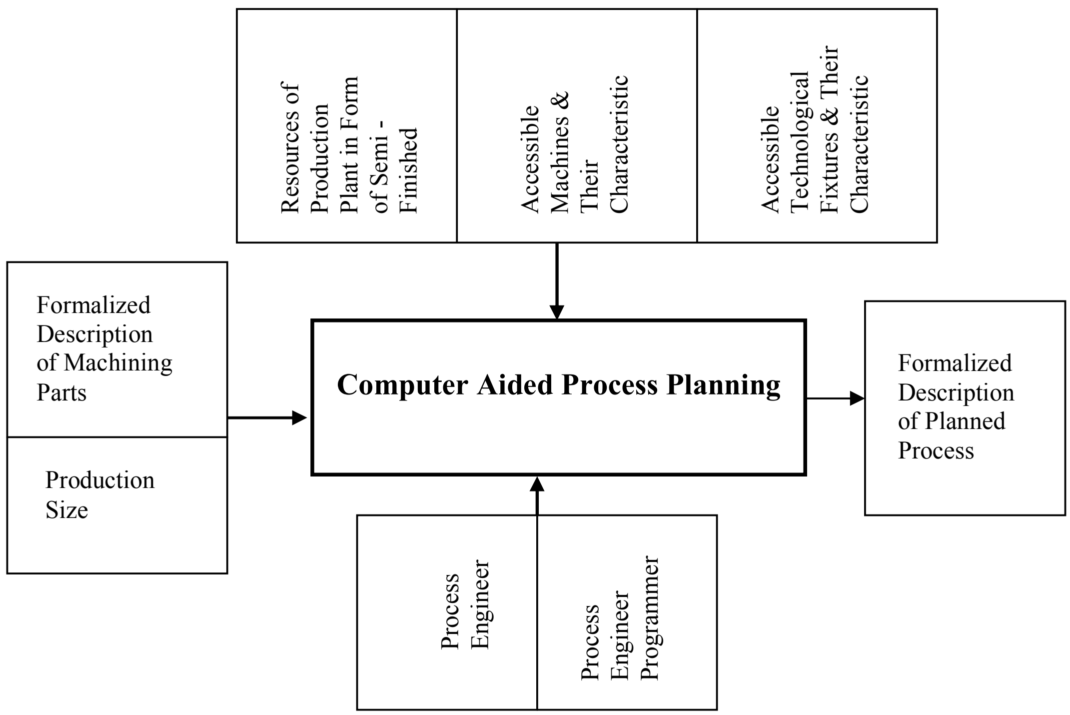


Figure1.1 Components taken in to consideration in Computer Aided Process Planning [9]

Input data information for computer-aided process planning system will be formalized part description (half-finished product description and finished part description) and production size, whereas output data information is a formalized description of planned process. First of all this description should contain [9]:

- General information about: part name, part class, drawing of part and its number (code symbol).
- Technological process structure including its elements (technological operations, set-ups, positions, cuts).
- Information for every operation: operation name (string description), operation number (code symbol), production department name (code symbol), work (machining) station name and type (code symbol of work

station), part drawing before and after carrying of operation, specification of part fixtures, specification of tool fixtures, kinds and sequence of technological cuts, technical standard operation time, control program.

- Information for every technological cut: word description of cut, its number, kind and type of tool and its characteristic (code symbol), machining parameters for example: speed of cutting– v, rotational speed – n, feed – p), main time of cut (machine time). Components taken into consideration in computer aided process planning are shown on Fig. 1.1.

1.2 Functions of Process Planning

The functions of process planning are enlisted below:-

- Selection of machining operations by viewing the engineering drawing of the part.
- Generating the optimum sequence of machining operations.
- Selection of suitable machine tool.
- Selection of suitable cutting tool.
- Determine the setup requirements.
- Calculation of cutting parameters.
- Tool path planning and generation of NC part program.
- Design of Jig and Fixture.

1.3 Modules in CAPP

A complete computer aided process planning system is assumed to be consisting of the following modules.

- *Part Modeling*: - This incorporates the creation of the wire frame or solid model. This modeler provides a custom build interface to capture information such as tolerance, surface finish, etc. if the CAPP system uses a feature based design then there is no need for a separate feature recognition module and feature information can be easily extracted using an interpreting mechanism.
- *Feature Recognition*: - A feature has a special design or manufacturing characteristic and is a sub set of geometry or an engineering part. We only deal with geometric features and not with the functional features, in CAPP. There could be a mechanism which using rules or conditions infer the shapes of features. There will be a program for translating the CAD data in to some

suitable form. The feature information is very useful for the next modules as the features and their relations are used to decide the machining operations, their sequences, and tools required etc.

- *Determination of Machining Methods*: - Manufacturing operations required to produce the part are determined in this module. The selection of the methods depends on feature type. Milling operation is selected for a slot and drilling for holes, etc.
- *Machine Tool Selection* -: The machine required for operation is selected by this module. It depends on the availability of the equipment and capability of the machine tools. Capability may be of tolerances, maximum dimensions that can be produced etc. The feature type as already mentioned decides machining methods and this decide machine tools.
- *Selection of Cutting Tools*: - The tool selected depends on the feature and its selection decides the fixturing elements. The tool and tool holder not interfere with the clamps or other elements of the fixture.
- *Cutting Parameters / Conditions*: - The parameters such as feed, depth of cut, speed and others like cutting fluids are selected depending upon the process and material of the workpiece and feature from a data base which maintains theses data.
- *Setup Planning*: - The decisions taken in this module include determination of sequence of machining operations and the orientation of workpiece with respect to tool and machine in a particular setup with operations in that particular setup.
- *Fixture Design*: - This module uses a setup data and determines the fixture (its elements) also using data from machine selection and tool selection. Principles of fixturing are used to restrict the workpiece. The elements of fixture are determined using a dynamic database, which has listing of available elements. Strength considerations also effect the selection of particular element.

1.4 Benefits of Computer Aided Process Planning

- It can reduce the skill required of a planner.
- It can reduce the process planning time.
- It can reduce both process planning and manufacturing cost.
- It can create more consistent plans.

- It can produce more accurate plans.
- It can increase productivity.

1.5 Operation Sequencing

In any CAPP system, selection of the machining operations sequence in one of the most critical activity for manufacturing a part as per the technical specification and the part drawing. Any fixed sequence of the operations that is generated in a process plan can not be the best possible sequence for all the production periods or for the criteria such as quality and machine utilization. Thus the aim should be to generate feasible operation sequences, which have to be pruned by the criteria in order to determine the best sequence of operations for the prevailing production environment.

Manned methods, mathematical programming method as well as computer based method have been used to determine the process plan. Regardless of which method is used, the operation sequence problem is inescapable. This problem arises due to type of operations involved, part features to be produced and blueprint tolerances to be satisfied. The order in which the operations are performed will influence setup depends cost and non- cutting time.

1.6 Constraints in Operation Sequencing [22][26]

The constraints, as shown in Table 1, which affect sequencing, can be divided in those, which address either the feasibility or optimality of a sequence. A feasible sequence is deemed to be one, which does not violate any of the feasibility constraints listed in Table 1.1 These constraints are processed sequentially by the system with the results of each application being the generation of either *precedence* or *relation* statements. A *precedence* statement takes the form, f_a (f_b), meaning that feature, f_a, cannot be machined until feature, f_b, has been cut. A *relation* statement of the form, $(f_a, f_b, \ldots..)$, pertains to multiple features indicating that these features must be machined in the same setup.

Feasibility constraints	Location reference Accessibility Non-destruction Geometric –tolerance Strict precedence
Optimality criteria	Number of setups Continuity of motion Loose precedence

Table 1.1 Sequencing constraints [26]

The location constraint is concerned with an examination of the defined part to determine what reference face is used to locate each feature. This reference identifies the necessity that the locating surface be machined prior to the associated feature.

The accessibility constraint evaluates each feature's accessibility based on the feature type and its location relative to the features. Features are defined as either primary or secondary. With the primary features defining the basic shape of the part (diameters, tapers, etc.) and secondary features defining the detailed shape aspects (grooves, threads, etc.).

The non-destruction constraint is concerned with ensuring that a subsequent operation does not destroy the properties of features machined in prior operation.

The strict precedence whereby order is determined based on feature type and properties. One example would be an internal cylinder whose properties require the use of a reaming operation. However, before reaming can be performed, there is a need to first drill, and possibly bore, the cylinder. The need for those preparatory operations is actually determined during operations selection. Therefore, the result of this constraint will not actually influence the plan until the operations are considered when writing on the sequence.

The geometric tolerances defined for the part. The types of the tolerances that affect sequencing include: profile of a line and surface, perpendicularity, angularity, parallelism, total and circular runout, position, symmetry and concentricity. Sequences, which satisfy all the feasibility constraint, can then be judged and ranked using the optimality criteria.

Number of Setups should be as less as possible. So that the maximum number of operations can be performed in minimum setups in order to reduce the cutting cost and to reduce the non cutting time.

Continuity of Motion means maximum number of operations should be performed in continuity without changing the setups, parameter change, tool change etc.

1.7 GA and Other Search Algorithms [19]

Many works have so far been done to optimize these parameters by using different optimization techniques like goal programming, multistage dynamic programming, linear programming, geometric programming, branch and bound algorithm etc. But all of them face great difficulties when the number of variables increases, because the problem becomes combinatorial explosive and hence computationally complex. Different researchers used different techniques to optimize process parameters but all of those techniques have their own limitations. Direct search methods include function evaluation and comparisons only. Gradient search methods need values of function and its derivatives, and their computerizations are also problematic. They are more difficult than the direct search methods, but they can yield more accurate for some computational efforts. Derivative-based mathematical optimizations are not manageable for optimizing functions of discrete variables. Dynamic programming that may be applied to problems whose solution involves a multistage decision process, can handle both continuous and discrete variables. Contrary to many other optimization methods it can yield a global optimum solution. However if the optimization problem involves a large number of independent parameters with a wide range of values (as in the case of optimization of cutting parameters), the use of dynamic programming is limited. As the numbers of variables and constraints increases, the optimum has a tendency to grow flatter with less probability that the realizable optimum will be a mathematical optimum, and hence computational effort increases considerably. Geometric programming is a useful method that can be used for solving nonlinear problems subject to nonlinear constraints, especially if the objective function to be optimized is a polynomial with fractional and negative exponents, while the constraints may be incorporated in the solution techniques. It is more powerful than other mathematical optimization techniques when the problem is restricted by one or two constraints. However, if the degree of difficulty increases, the formulated problem might be more complicated than the original problem. Geometric programming can only handle continuous variables.

The solution to the optimization problems, which includes real value variables, can be obtained using numerous methods. There is no efficient all-purpose optimization method available for nonlinear programming problems like process parameter optimization. The computational time and cost involved in the determination of optimal parameters commonly depends on the complexity or simplicity of the model. Some models can produce accurate solutions by making rigorous computation, which is not economic in terms of computation time and cost. Sometimes the solution from these models may not be optimal. Some other models may develop solutions far from the optimum in a fast manner. Therefore a compromise between the high accuracy of a rigorous solution and low accuracy of an oversimplified solution should be made. Genetic Algorithms (GAs) are robust search algorithms that are based on the mechanics of natural selection and natural genetics. They combine the idea of "survival of the fittest" with some of the mechanics of genetics to form a highly effective search algorithm. Genetic algorithms belong to a class of stochastic optimization techniques known as evolutionary algorithms. Among the three major types of evolutionary algorithms genetic algorithms, evolutionary programming, and evolution strategies) genetic algorithms are the mostly widely used. GA's are most often used for optimization of various systems, especially complex problems such as those involving manufacturing systems analysis.

1.8 Introduction to Genetic Algorithm [1]

Genetic Algorithms (GA) have been developed by John Holland, his colleagues, and his students at the University of Michigan. The goals of their research have been two fold: (1) to abstract and rigorously explain the adaptive processes of natural system, and (2) to design the artificial systems software that retains the important mechanism of natural system. This research has led to important discoveries in both natural and artificial system science.

Genetic algorithms are a family of computational models inspired by evolution. These algorithms encode a potential solution to a specific problem on a simple chromosome–like data structure and apply recombination operators to these structures so as to preserve critical information. Genetic algorithms are often viewed as function optimizers, although the range of problems to which genetic algorithms have been applied is quite broad. An implementation of a genetic algorithm begins with a population of (typically random) chromosomes. One then evaluates these structures and allocates reproductive opportunities in such a way that those chromosomes which represent a better solution to the target problem are given more chances to ‘‘reproduce’’ than those chromosomes which are poorer solutions. The ‘‘goodness’’ of a solution is typically defined with respect to the current population.

In a broader usage of the term, a genetic algorithm is any population–based model that uses election and recombination operators to generate new sample points in a search space. Many genetic algorithm models have been introduced by researchers largely working from an experimental perspective. Many of these researchers are application oriented and are typically interested in genetic algorithms as optimization tools. Machining sequencing, selection of cutting parameters and monitoring often has to deal with the problem of optimization.

The GA mimics the process of natural evolution by combining the survival of the fittest among solution structures with a structured, yet randomized, information exchange and creates offspring. The offspring displaces weak solutions during each generation. GA exploits the idea of the survival of the fittest and an interbreeding population to create a novel and innovative search strategy. A population of strings, representing solutions to a specified problem, is maintained by the GA. The GA then iteratively creates new populations from the old by ranking the strings and interbreeding the fittest to create new strings, which are (hopefully) closer to the optimum solution to the problem at hand. So in each generation, the GA creates a set of strings from the bits and pieces of the previous strings, occasionally adding random new data to keep the population from stagnating. GA’s are a form of randomized search, in that the way in which strings are chosen and combined is a stochastic process.

The idea of survival of the fittest is of great importance to genetic algorithms. GA’s use a fitness function in order to select the fittest string that will be used to create new, and conceivably better, populations of strings. The fitness function takes a string and assigns a relative fitness value to the string. The method by which it does this and the nature of the fitness value does not matter. The only thing that the fitness function must do is to rank the strings in some way by producing the fitness value. These values are then used to select the fittest strings.

The population can be simply viewed as a collection of interacting creatures. As each generation of creatures comes and goes, the weaker ones tend to die away without producing children, while the stronger mate, combining attributes of both parents, to produce new, and perhaps unique children to continue the cycle. Occasionally, a mutation creeps into one of the creatures, diversifying the population even more. Remember that in nature, a diverse population within a species tends to allow the species to adapt to its environment with more ease. The same holds true for genetic algorithms.

There are three basic operators found in every genetic algorithm: reproduction, crossover and mutation.

1.8.1. Reproduction [13]

The reproduction operator allows individual strings to be copied for possible inclusion in the next generation. The chance that a string will be copied is based on the string's fitness value, calculated from a fitness function. For each generation, the reproduction operator chooses strings that are placed into a mating pool, which is used as the basis for creating the next generation. There are many different types of reproduction operators:

- Proportional selection: This method only work with fitness values above zero (non- negative) and scaling may sometimes be necessary. It has been shown that proportional selection performs poorly compared with other selection schemes in many GA problems.
- Tournament selection: Choose (t) individuals at random from the population and copy the best individual from this group into the new population. Repeat N times.
- Truncation selection: With truncation selection that has a threshold of T between 0 and 1, only the fraction T best individuals can be selected. They all have the same selection probability.
- Linear ranking selection: The individuals are sorted according to their fitness values and the rank N is assigned to the best individual, the rank 1 assigned to the worst. The selection probability is linearly assigned to the individuals according to their rank and a selection equation.
- Exponential ranking selection: This follows the same methodology of linear ranking selection, the only difference being that the probabilities of the ranked

individuals are exponentially weighted. One always selects the fittest and discards the worst, statistically selecting the rest of the mating pool from the remainder of the population. There are hundreds of variants of this scheme. None are right or wrong. In fact, some will perform better than others depending on the problem domain being explored.

1.8.2. Crossover

Once the mating pool is created, the next operator in the GAs arsenal comes into play. Remember that crossover in biological terms refers to the blending of chromosomes from the parents to produce new chromosomes for the offspring. The analogy carries over to crossover in GAs. The GA selects two strings at random from the mating pool. The strings selected may be different or identical, it does not matter. The GA then calculates whether crossover should take place using a parameter called the crossover probability. If the GA decides not to perform crossover, the two selected strings are simply copied to the new population. If crossover does take place, then a random splicing point is chosen in a string, the two strings are spliced and the spliced regions are mixed to create two (potentially) new strings. These child strings are then placed in the new population.

As an example, say that the strings 10000 and 01110 are selected for crossover (Fig. 1.2) and the GA decides to mate them. The GA selects a splicing point of 3. The following then occurs: The newly created strings are 10010 and 01100. Crossover is performed until the new population is created. Then the cycle starts again with selection. This iterative process continues until any user specified criteria are met.

Parent 1 =	**1**	**0**	**0**	**0**	**0**
Parent 2 =	**0**	**1**	**1**	**1**	**0**
Child 1 =	**1**	**0**	**0**	**1**	**0**
Child 2 =	**0**	**1**	**1**	**0**	**0**

Fig 1.2 Crossover operation [13]

1.8.3. Mutation [1][13]

Selection and crossover alone can obviously generate a staggering amount of differing strings. However, depending on the initial population chosen, there may not be enough variety of strings to ensure the GA sees the entire problem space. Or the GA may find itself converging on strings that are not quite close to the optimum it seeks, due to a bad initial population. Some of these problems are overcome by introducing a mutation operator into the GA. The GA has a mutation probability, m, which dictates the frequency at which mutation occurs. Mutation can be performed either during selection or crossover (though crossover is more usual). For each string element in each string in the mating pool, the GA checks to see if it should perform a mutation. If it should, it randomly changes the element value to a new one. In our binary strings, 1s are changed to 0s and 0s to 1s. For example, the GA decides to mutate bit position 4 in the string 10000 (Fig. 1.3).

Parent 1 =	**1**	**0**	**0**	**0**	**0**
Parent 1' =	**0**	**1**	**1**	**0**	**0**

Figure 1.3 Mutation operation [13]

The resulting string is 10010 as the fourth bit in the string is flipped. The mutation probability should be kept very low (usually about 0.01%) as a high mutation rate will destroy fit strings and degenerate the GA algorithm into a random walk, with all the associated problems. But mutation will help prevent the population from stagnating, adding ''fresh blood'', as it were, to a population. Remember that much of the power of a GA comes from the fact that it contains a rich set of strings of great diversity. Mutation helps to maintain that diversity throughout the GAs iterations.

1.9 Machining Parameters [31]

The advancement in the manufacturing engineering and development in the areas of computer-aided manufacturing is high level of automation in all the manufacturing activities. In accordance with this, attempts have also been made to automate the selection of machining parameters. These traditional methods have been replaced by reliable computer aided procedures in the selection of machining parameters. So far there have been four approaches suggested for the automated selection of the machining parameters. These are given below:-

- Data storage and retrieval approach.
- Empirical equation approach.
- Expert system approach.
- Mathematical model approach.

1.9.1 Data Storage and Retrieval System [5]

This approach consists of two steps as shown in Figure-1.4. In the first step, recommended parameters for various combinations of work material, tool material and machining operations are collected from the shop floor, cutting tool and machine tool manufacture's catalogues and handbooks. Then in second step these parameters are stored in a database in a structured format as shown in figure 1.4 (a).

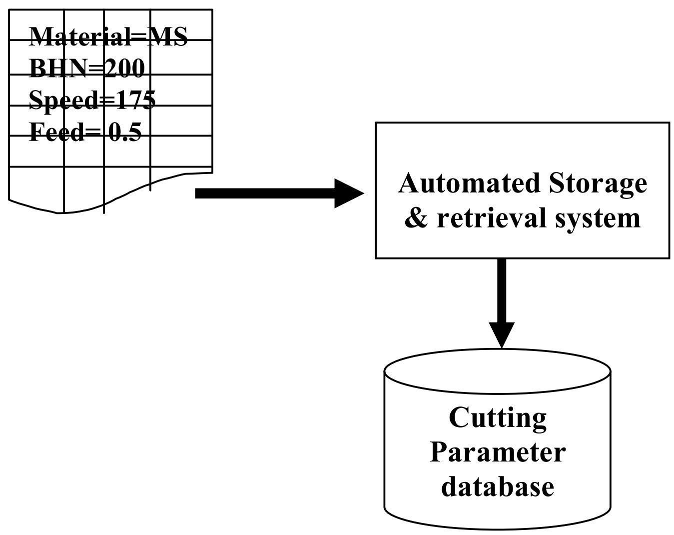


Fig 1.4 (a) Data storage approach [5]

Based on the user inputs the proper sets of cutting conditions are retrieved from the database as shown in figure 1.4(b) and present it to the user. This is in-effect a computerized look-up table approach. Though this approach aids in the fast and consistent selection of parameters the parameters are general in nature and lot of computer storage is required. Also they are often characterized by lack of adequate maintenance and difficulties in updating the current machining recommendations.

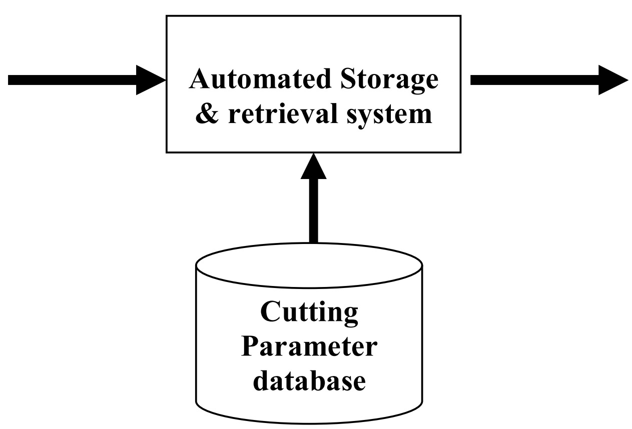


Fig 1.4 (b) Data retrieval approach [5]

1.9.2 Empirical Equation Approach [5]

In this approach generalized empirical equations based upon extended Taylor's tool life equations are derived from the data obtained from extensive tests or handbooks for different combinations of work material, tool material and machining operation. These equations along with their constants are stored in computer data files as shown in figure 1.5 (a) based on the user inputs these constants are first retrieved from the computer data files and then the values of cutting speed, feed and depth of cut are computed from the proper equation as shown in figure 1.5 (b).

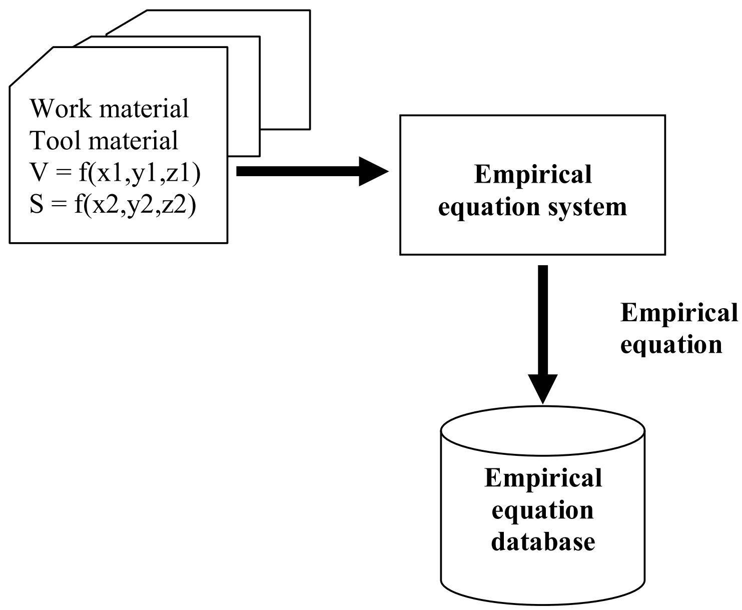


Figure 1.5 (a) Storage of empirical equations [5]

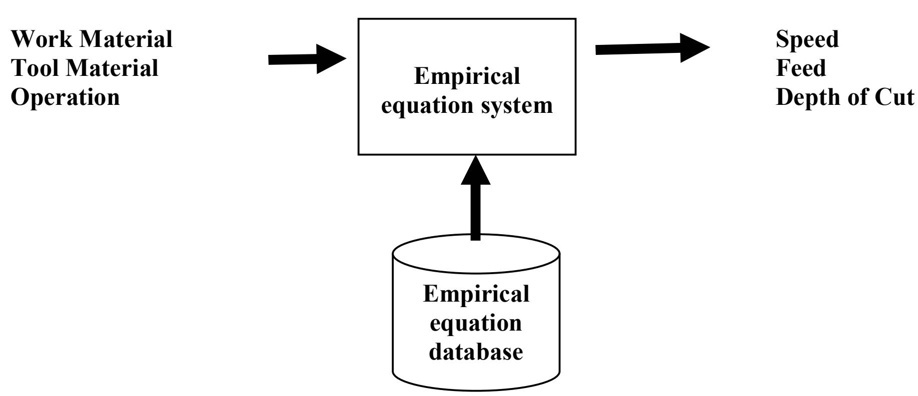


Figure 1.5 (b) Computations of cutting conditions [5]

In comparison with previous approach the following advantages are observed in this approach:

- The requirements for storage space are relatively small, as the parameters are not being stored.

- Unlike the previous approach, the parameters calculated from the equations are applicable over a range of situations.

1.9.3 Expert System Approach [5] [31]

The term "Expert System" do researchers in-to artificial intelligence coin one and broadly speaking refers to a system, which seeks to emulate the reasoning capacity of an expert in a particular field of expertise. Expert systems differ from algorithmic system, in that they are rule driven rather than the code driven. The rules which describe the decisions logic of a problem are extended to the main system which is essentially an "Inference Engine", capable of interpreting the rules and inferring decision from them.

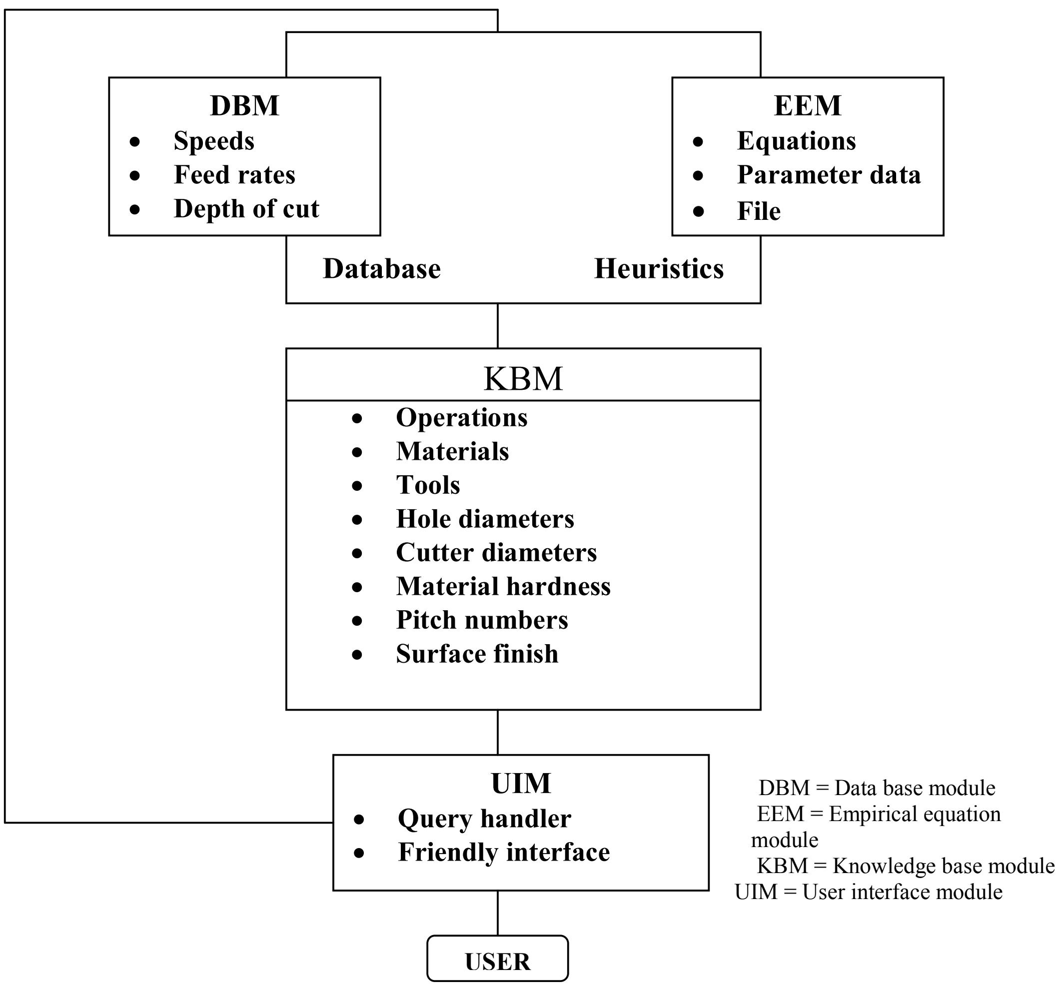


Fig. 1.6 Expert system approach

Expert system provides a user-friendly, flexible, intelligence solution to the kind of problems involving complex logic, which occurs frequently in the engineering such as process planning. Attempts were made to apply these techniques in selecting machining parameters also. In this approach the machining parameters are stored in a database in either one or a combination of first two approaches. Based on the user inputs the system input the proper machining rules form the knowledge base and retrieves the parameters from the machining parameter selection as shown in figure 1.6.

The advantage in approach lies in the ability to these systems to provide what-if analysis and in their better data handling capability.

1.9.4 Mathematical model approach [5]

In this approach the problem is converted in to mathematical models based on criteria. The criterion differs for different problems. In case of automated machining parameters selection, mathematical models are formulated to determine that machining parameters for a particular objective such as minimization of unit cost and maximization of production rate.

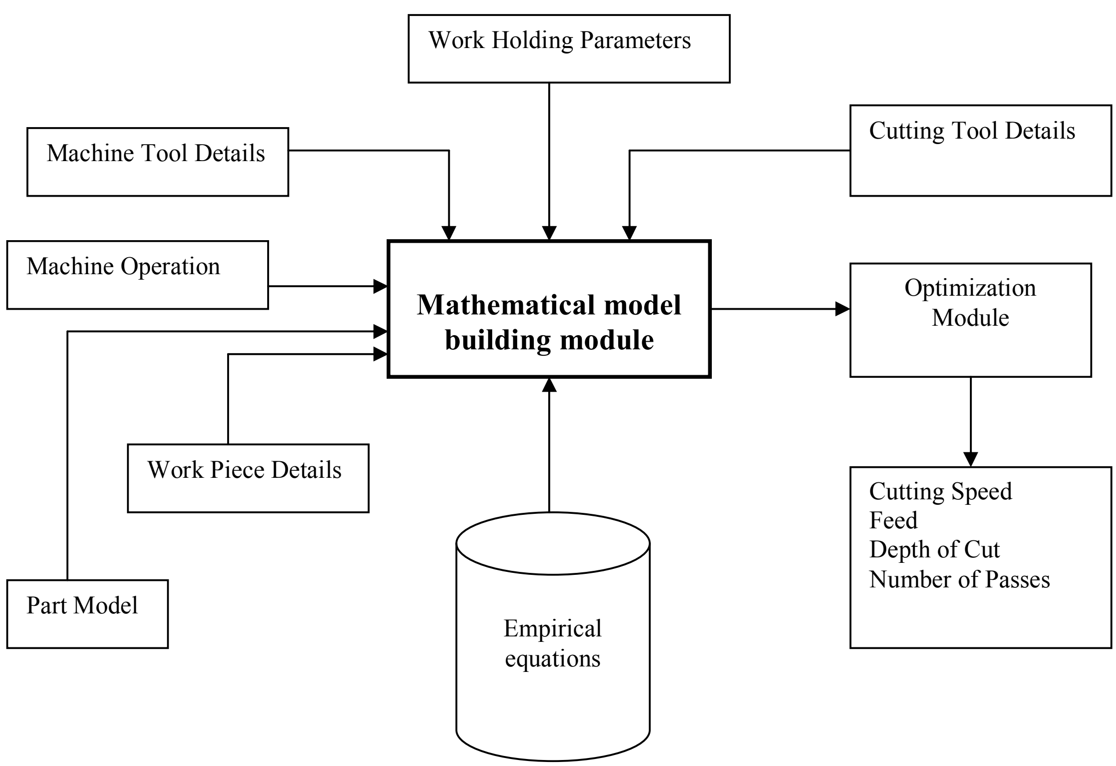


Fig. 1.7 Mathematical model approach

In this approach also, empirical equation formulated based on the experimental data for tool life (usually Taylor's tool life equation) and other practical restrictions and machining variables are used in the model development. The resultant function from this mathematical modeling is called objective function, which is subjected to practical constraints. The developed model is then solved using an optimization technique to yield optimal parameters that reflect the shop floor capabilities. This procedure is illustrated in figure 1.7. The success of this approach depends upon the reliability of the empirical equation used in the model formulation.

From the above approaches the following important points are observed.

- The first three approaches either retrieve the recommended machining parameters from the data files or calculate from the empirical equations depending on the user inputs.
- In this the first three approaches the machining economics and practical constraints on the machining variables are usually considered. However, in the last approach the machining process and the constraints acting on it are modeled mathematically and solved for optimal parameters.

In this chapter the definition, need, functions and modules of CAPP are explained along with the need & importance of operation sequencing & machining parameters. The various constraints to operation sequencing are mentioned, which must be satisfied to generate a feasible sequence. Introduction to Genetic Algorithm and the various techniques used for giving the machining parameters are explained. Our aim in this research project is to generate the operation sequencing of rotational parts using Genetic Algorithm & then give the machining parameters using Expert system. The next chapter includes the contributions made by the different researchers in the field of operation sequencing & machining parameters by Artificial Intelligence base techniques.

CHAPTER 2

LITERATURE SURVEY

Large number of CAPP systems has been reported in the literature. Different researchers adopted different advanced techniques and approaches such as feature or solid model based design, object oriented programming, technological databases, and utilized advanced computing methods including expert system and artificial intelligence.

Recently some research areas such as feature recognition or feature extraction from CAD file, application of AI techniques like Genetic Algorithms (GAs), Artificial Neural Network (ANN), are gaining more attention among the researchers.

In this chapter, views of various researchers are discussed regarding the use & capabilities above mentioned techniques. An attempt is made to explore various techniques used for operation sequencing and selection of machining parameters.

Sankha Deb, Kalyan Ghosh & S.R. Deb [8] proposed a computer aided process planning (CAPP) methodology based on neural network. Different Artificial Intelligence (AI) based approaches have been explored in computer-aided process planning for machining. These approaches may be broadly classified into the following categories: expert system, fuzzy reasoning and neural networks. A survey on the application of various Artificial Intelligence approaches in computer-aided process planning and their potential advantages and their shortcomings are discussed The objective of the proposed methodology is automated selection of all feasible machining operation sequences and the preferred operation sequence after taking cost factor into consideration by using a neural network based approach. The first step in development of the neural network architecture is deciding on the output process planning decision variables, followed by all related process planning input variables that could affect the selection of the output variables. It constitutes an important step as the optimal performance of the neural network depends on the chosen input patterns and output variables describing the problem. The learning process or knowledge acquisition takes place by presenting the network with a set of training examples and the neural network through the learning algorithm implicitly derives the rules. All the neural networks developed for the study use a fully connected feedforward multilayer perceptron. The standard back propagation algorithm is used as the learning mechanism for the neural network. Some advantageous features of the neural network based CAPP methodology as against the knowledge based approach are: its efficient knowledge acquisition capability owing to its ability to implicitly derive the rules from sample machining cases presented to the neural network. Its capability to generalise beyond the original machining cases to which it is exposed during the training and face intermediate situations with reasonably good accuracy with respect to those proposed during the training. High processing speed once the neural network is trained.

Edward Gawlik [9] explains the two methods (Variant, Generative) for carry out the computer aided process planning and also recognized the third category, which is the combination of both the two methods and is known as semi-generative method. Variant method of computer aided process planning is comparable with traditional manual method where a process plan for a new part is created by identifying and retrieving an existing plan for a similar part (sometimes called a master part) and making the necessary modifications for the new part. One of the disadvantages of variant method is that the quality of the process plan still depends on the knowledge background of a process planer. The computer is just a tool to assist in manual process planning activities. Generative method process plans are generated by means of decision logic, formulae, technology algorithms and geometry based data. Generally, format of input to computer aided process planning systems can be divided in two categories: as a text input, where the user answers a number of questions (defined as interactive input), or as graphic input, where the part data is gathered from a CAD module (defined as interface input). The semi-generative method can be characterized as an advanced application of variant technology employing generative type features. Systems utilized semi-generative method to process planning should co-operate with planner, who posses technological knowledge. Planner’s responsibility is the interpretation of decision data and/or working drawing. The advantages of computer aided process planning are : reduced clerical effort, fewer calculations, fewer oversights in logic, immediate access to up-todate information, consistent information, faster response to engineering or production changes, use of latest revisions, more detailed and uniform planning, more efficient use of resources.

Z.G. Wang, M. Rahman, Y.S. Wong, J. Sun [10] proposed an approach to select the optimal machining parameters for multi-pass milling. It is based on two recent approaches, genetic algorithm (GA) and simulated annealing (SA), which have been applied to many difficult combinatorial optimization problems with certain strengths and weaknesses. In this paper, a hybrid of GA and SA (GSA) is presented to use the strengths of GA and SA and overcome their weaknesses. In order to improve, the performance of GSA further, the parallel genetic simulated annealing (PGSA) has been developed and used to optimize the cutting parameters for multi-pass milling process. An application example that has been solved previously using the geometric programming (GP) and dynamic programming (DP) method is presented. a parallelized version of GSA is presented to optimize the cutting conditions in plain milling (feed rate, speed and depth of cut) subject to a set of constraints. For comparison, conventional PGA was also attempted in the study. In the proposed PGSA algorithm and PGA, the whole population is divided into several sub-groups according to the depth of cut. In each sub-group, the coarse-grained parallel GSA, combined with a master–slave model, is implemented. Based on the given results, it can be seen that PGSA is more effective for optimizing the cutting parameters for milling operation than conventional GPC DP. The numerical results show that PGSA has faster convergence to the global optimum solution than PGA, and thus is superior to the conventional PGA. In PGSA, at the initial stage, higher mutation rate is helpful for maintaining population diversity. The local selection of SA is involved after crossover and mutation, which can ensure that at the later stage good candidates still exist in the next generation. Therefore, by maintaining more diverse subpopulations at the initial stage, the PGSA mitigates the premature convergence of the standard GA. On the other hand, at the later stage, more and more good candidates exist in the next generation because of the local selection of SA, and therefore, it can narrow the search space so that fast convergence can be achieved.

A. Vidal, M. Alberti, J. Ciurana, M. Casadesu [11] focused on the problem of choosing the manufacturing route in metal removal processes, a highly important element in computer aided process planning (CAPP) systems. At present, decisions concerning the cutting parameters used in metal removal operations are based on experience. The object of this work was to design a system to help in selecting the parameters in the cutting process in the case of milling. The algorithm is based on optimizing the cost of the operation. The result is the selection of parameters that take into account all the existing restrictive factors (material, geometry, roughness, machine and tool). A wide range of parts have been evaluated, using group technology in order to choose representative cases. The results of this system were compared with the values proposed in the reference manuals. The experiment had served to determine the different relationships between the factors that had an influence on milling operations and the features which allow us to take into account the wide range of geometrical possibilities existing for all the pieces we studied. Applying this method helps to make the right decisions on the optimal parameters in mill operations as applied to the machining processes.

Zhang Wei BO, Lu Zhen Hua, Zhu Guang Yu [12] states process route sequencing as a process of analyzing the internal relations and affections of many factors in machining, and is difficult to be solved exactly with mathematical expression, so the enumeration method or analytical method cannot be used in this sequencing work. In practice, process route sequencing can be regarded as a process where a series of constraints are imposed repeatedly one by one on the aggregation composed of many machining feature cells. The constraints include geometry shape, technique request, technologic method, optimization index and so on. The final machining sequence which can fully or farthest satisfy all the constraints is considered to be the optimal process route. In this paper the natural number is adopted in coding strategy and the method of path representation is used for coding vector representation, and the value of element in the string is the allocated sequence number of machining cell. Any change of the order of the sequence number will bring meaningful vectors. Suppose that after machining method selecting and the operator string breaking down, there are 'n' machining cells in a part; apparently the total number of possible process sequence is n!.

In order to guarantee the astringency of GAs, the optimal individual in one generation must be kept to the next generation. Other individuals in population are selected by using the ‘‘tournament selection’’ operator. Suppose there are W individuals to be selected, selecting two individuals randomly from the population and keeping the better one for the next generation. Repeating this process W_1 times, then at last all individuals in the next generation will be obtained. There are three crossover operators for path representation: partially mapped crossover (PMX), order crossover

(OX) and cycle crossover (CX). The character of PMX crossover is keeping the important similarities of parent and child generation. The OX crossover emphasizes that the sequence order is very important. The CX reserves the absolute position of the elements in the parent generation.

In order to establish the fitness function, the constraints of process sequencing must be analyzed firstly. There are three kinds of quantitative constraints: Order constraint, Clustering constraint Optimization constraint. Among the above-mentioned constraints, the order constraint and the clustering constraint are the main constraints in process sequencing, so they are compulsive constraints, and the optimization constraint is often considered as an additive constraint. So the operators of selection, crossover and mutation are executed on the population repeatedly to search the optimal process route in the global feasible space. At last the optimal or near-optimal process route can be obtained.

Franci Cus, Joze Balic [13] proposed a new optimization technique based on genetic algorithms (GA) for the determination of the cutting parameters in machining operations. In metal cutting processes, cutting conditions have an influence on reducing the production cost and time and deciding the quality of a final product. The main objective of the present paper is to determine the optimal machining parameters that minimize the unit production cost without violating any imposed cutting constraints. This paper presents a new methodology for continual improvement of cutting conditions with genetic algorithm. It performs the following: the modification of recommended cutting conditions obtained from a machining data, learning of obtained cutting conditions using neural networks and the substitution of better cutting conditions for those learned previously by a proposed genetic algorithm. Experimental results show that the proposed genetic algorithm-based procedure for solving the optimization problem is both effective and efficient, and can be integrated into an intelligent manufacturing system for solving complex machining optimization problems.

K. Vijayakumar, G. Prabhaharan, P. Asokan, R. Saravanan [14] proposed a new optimization technique based on the ant colony algorithm for solving multi-pass turning optimization problems. The cutting process has roughing and finishing stages. The objective of this optimization cutting model is to determine the optimal machining parameters including cutting speed, feed rate, depth of cut and number of rough cuts in order to minimize the unit production cost. The unit production cost, (UC), for multi-pass turning operations can be divided into four basic cost elements. 1. Cutting cost by actual time in cut, C_M. 2. Machine idle cost due to loading and unloading operations and idling tool motion, C_I. 3. Tool replacement cost, C_R; and 4. Tool cost, C_T. The unit production cost is defined as the sum of cutting cost C_M, machine idling cost C_I, tool replacement cost C_R, and tool cost C_T. $U_C = C_M + C_I + C_R + C_T$ The constraints which are considered during the roughing and finishing operations include 1. Parameter bounds 2. Tool-life constraints 3. Operating constraints consisting of surface finish constraint, force and power constraint 4. Stable cutting region constraint. 5. Chip–tool interface temperature constraint. 6. Roughing and finishing parameter relations. The objective function as mentioned previously is minimized subject to practical constraints. In the first step, 200 solutions are generated randomly within parameter bounds satisfying the constraints. Then the global and local search is performed.

N. Ben Yahia, F. Fnaiech [16] developed a feedforward Neural Network based intelligent system for computer aided process planning. This system suggests the sequence of manufacturing operations to be used, based on the attributes of a feature of the component. The first part of this paper presents a new methodology for the description of the profile aimed job. This method is called job kind in dynamic study. This analyst is capable to identify the different real problem types well often complex that pose. The workshop producing in unit or very small batch size of various pieces, that are composed by several machining features is considered as the case study. In the second part a method of process organization of manufacturing by using Neural Networks system, where the learning mechanism is the backpropagation algorithm. The advantage of this new method of process planning is the ability to generate an optimal sequence which is difficult in real manufacturing environment, the sequence generated is near optimal when it is successful in minimizing the number of set-ups and minimizing the number of manufacturing tool changes.

Dong-Ho Lee, D. Kiritsis, P. Xirouchakis [17] focused on the operation sequencing problem in process planning, which is the problem of determining the sequence of operations required for producing a part with the objective of minimizing the sum of machine, set-up and tool change costs. In general, operation sequencing in process planning has been a difficult optimization problem, since it has combinatorial characteristics and complex precedence relations. This paper suggested a precedence-directed graph to describe the precedence relations among operations with a compact form. Then, using the directed graph, the authors develop optimal and heuristic branch and fathoming algorithms that can give optimal and near optimal solutions respectively. Application of the two algorithms is illustrated using an example part. Also, to show the performance of the suggested algorithms, computational experiments were done on randomly generated problems and the results show that the heuristic algorithm gives optimal solutions for all the generated problems with much shorter computation time.

Nafis Ahmad and A.F.M. Anwarul Haque [18] optimize process planning parameters for machining rotational components by a Genetic Algorithm Optimization Toolbox developed in Matlab environment. Determination of optimum parameters is one of the vital stages of process planning since the economy of machining operation plays the most important role in increasing productivity and competitiveness. Genetic algorithm is one of the most efficient tools for optimization of such problems.

This paper presents the application of GA in process planning parameters optimization. The solution to the optimization problems, which includes real value variables, can be obtained using numerous methods. Therefore a compromise between the high accuracy of a rigorous solution and low accuracy of an oversimplified solution should be made. Genetic Algorithms (GAs) are robust search algorithms that are based on the mechanics of natural selection and natural genetics.

GAs start with an initial set of random solutions called the population. There is no strict rule to determine the population size. Population sizes of 100- 200 are common in GA research. Once the chromosomes are coded as bit strings, the genetic algorithm manipulates these strings using three genetic operators - reproduction, crossover, and mutation. The easiest way is to create a biased roulette wheel slot sized in proportion to each current string in the population. After the selection, the strings are paired up, and a percentage of the pairs trade parts of their strings. This is known as crossover. These newly formed strings by crossover operator are then subjected to a random screening, where random bits in random strings are picked and modified.

This is known as mutation. This combined process of exploiting knowledge about a search space (reproduction) and exploring a search space (crossover, mutation) is what drives the performance of a genetic algorithm. Over time, bad bit strings disappear from a population, while good bit strings live on and reproduce with other good strings to form even better strings.
In this work machining time is considered as the objective function and constraints are machine capacity, limits of feed rate, depth of cut, cutting speed etc. Machining time is minimized through a series of generations while some genetic operators are applied at each generation. The result of the work shows how a complex optimization problem is handled by a genetic algorithm and converges very quickly.

Nafis Ahmad, A.F.M. Anwarul Haque [19] introduced computer aided process planning system as a key to integrate design and manufacturing or assembly systems properly considering the available resources and design constraints. New systems with many advanced techniques and approaches such as feature-based modeling, object oriented programming, effective graphical user interfaces, technological databases and utilize advanced computing methods including expert system and artificial intelligence are explained. Application of various search techniques such as breadth-first, direct search, gradient search etc. with derivative based mathematical optimization, geometric programming, and genetic algorithm is explored.

T. Dereli, I. H. Filiz, A. Baykasoglu [20] introduced a system software' developed to optimize the cutting parameters for prismatic parts. The determination of optimal cutting parameters, such as the number of passes, depth of cut for each pass, cutting speed and feed, which are applicable for assigned cutting tools, is one of the vital modules in process planning of metal parts, since the economy of machining operations plays an important role in increasing productivity and competitiveness. It is implemented using C programming language and on a PC. It can be used as standalone system or as the integrated module of a process planning system called OPPS-PRI (optimized process planning system for prismatic parts) that was also developed for prismatic parts and implemented on a vertical machining centre (VMC). With the use of Genetic Algorithm, the impact and power of artificial intelligence techniques have been rejected on the performance of the optimization system.

G.Dinin, F. Failli, B. Lazzerini, F. Marcelloni [21] developed the optimized assembly sequence method by using the Genetic Algorithm, which drastically reduces the high computational time, usually necessary to evaluate the best assembly sequences. The input data, mainly represented by the CAD models of the product components, are pre-processed in order to generate four types of data: geometrical relationships among components, connections (threaded connections, drive fits, etc,) subassemblies and the list of grippers available for grasping and malfunctioning of the objects. The final output is represented by one or more assembly sequences optimized according to the minimization of object orientations, minimization of gripper changes, presence of similar and consecutive operations. The fitness of each individual is evaluated on the basis of the feasibility of the sequence, object orientations, gripper changes, and presence of similar and consecutive assembly operations. Each aspect is numerically evaluated by scores calculated as a function of the gene positions within the chromosome sections. Different crossover and mutation operators have been adopted in each section of the chromosome. A modified partially matched crossover is used. So this demonstrated how a genetic operator can be applied to solve the problem of sequencing in assembly planning.

S. V. Bhaskara Reddy, M.S. Shunmugam & T.T. Narendran [22] demonstrates the application of genetic algorithms as a global search technique for a quick identification of optimal or near optimal operation sequences in a dynamic planning environment. A careful study of this paper reveals that the operation sequencing in process planning is an AND-OR graph search involving various constraints. Genetic algorithms, being heuristic in nature, are quite promising in arriving at optimal operations sequences. The genetic string representation scheme should be based on the precedence constraints and the design of appropriate crossover and mutation operators also plays a major role for the successful implementation. This paper uses a precedence graph based on various manufacturing constraints, generates a precedence cost matrix using production rules and applies Genetic algorithms to arrive at optimal or near optimal operation sequence.

This paper comprises the following steps. First, the part drawing is analysed to identify the form features to be machined, with details of geometric specifications and technological information such as surface finish and tolerance requirements. Then this is followed by the selection of machining operations, machine tools, cutting tools and cutting parameters required to generate each of the form features. Considering the various feasibility constraints, the precedence relationship among the machining operations is obtained. A relative cost matrix is generated using the feature precedence and their attribute values, such as machine change, tool change, set-up change and machining parameter change based on the knowledge of various manufacturing processes and capabilities of machine tools in the shop floor. In the last the sequences of operations is represented as a path of least cost that visit the entire feature once and only once. The generation of feasible plans has the structure of the well known Traveling Salesman Problem in the graph theory literature. A feasible sequence is one that does not violate any of the constraints. The manufacturing constraints considered in this study are: accessibility, non-destruction together called Pre-conditions, location and geometric tolerance. The manufacturing constraints mentioned above provide a basis for the generation of Feature Precedence Graph (FPG) which is used to identify the feasible sequences. A precedence cost matrix (PCM) is generated for any pairs of features based on the approximate relative cost corresponding to the number of tasks that need to be performed in each category of attribute as mentioned above.

The proposed method obtains quick solutions, requiring about 10 to 40 seconds depending on the number of generations. Since the computation time taken to generate the feasible sequence is low, the software can be run several time to facilitate the process planner in obtaining the alternative operation sequence and there by generating alternative process plans.

G. Van Zeir, J.P. Kruth & J. Detand [23] described the information that reside on the blackboard and how this information is instantiated: the blackboard approach provides the human process planner with assisting ‘expert modules’, each capable of performing a specific automated process planning task. In each process planning step that is performed by such expert module, the human operator has full control over the generation of process plan and can always change the solution the proposed by the expert modules. The computer aided process planning system described in this paper has a very diverse manufacturing knowledge. It relates the object classes: workpiece families, workpiece features, relations, processes, manufacturing direction entities and various knowledge sources in to one common object-oriented model. The developed CAPP system is based on blackboard architecture: several expert modules can be triggered in an arbitrary order to perform a specific planning task, ensuring a very flexible way of performing the CAPP activities. Solutions generated by the assisting expert module result in objects on the blackboard. Theses solutions can be adjusted or overruled by operator, since each blackboard object has an interface to the human expert for manually adding or changing the information residing in the objects.

M. Tolouei-Rad, I. M. Bidhendi [24] described development and utilization of an optimization system which determines optimum machining parameters for milling operations. These parameters are intended for use by numeric control (NC) machines, however, they can also be used by conventional machines. The paper discusses both single-tool and multi-tool milling operations where emphasis has been placed on the latter. Owing to the significant role that machining parameters play in performing successful and efficient machining operations, determination of the best or optimum machining parameters is still the subject of many studies. The need to use optimum machining parameters to improve machining efficiency is of greater importance when NC machines with high capital cost are employed. Efficient utilization of machine tools has been a problem for manufacturing firms for a long time. The problem becomes more important when numeric control machines are employed owing to the high capital cost of these machines. Quite a number of attempts have been made to improve the efficiency of machining operations by using optimum machining parameters. However, most of them focused only on turning operations, leaving room for further investigation on other machining operations, including milling. The work described in this paper has made a significant improvement in machining efficiency over the handbook recommendations. This work considers the optimization of machining parameters for a single set-up. If machining of a product requires more set-ups, then the optimization procedure for each set-up should be performed individually. In such cases, the material cost for the second and subsequent optimization procedures becomes zero, since it has already been considered for the first set-up.

Jung-Ug Kim & Yeong-Dae Kim [25] proposed that many manufacturing systems can be considered to be composed of two basic processing units machining and assembly, each of which processes various operations for various products. These two units are run or managed relatively separately once a production plan for each unit is determined. A short term production scheduling problem for the production plans of the two units in such manufacturing system is considered. In general, products that are manufactured in these manufacturing systems have multi level product structure i.e. those products are composed of several subassemblies and component and each subassembly may also be composed of subassemblies and components. In this study, operations are aggregated into two basic operations, machining and assembly and processing unit in the manufacturing systems are also aggregated into a machining shop and assembly shop. Since the processing capacities of the system must be considered as well as precedence relationship among the items that result from the product structure, it is not easy to solve the problem optimally in a reasonable amount of time. Two search techniques, simulated annealing and genetic algorithms, which are often used to find near optimal solutions in various optimization problems.

A short term production scheduling problem in a manufacturing system producing products with multi-level product structure, and their components and subassemblies is considered. The problem is to schedule production of components, subassemblies and final products with objective of minimizing the weighted sum of tardiness and earliness of the items, due dates of final products, precedence relationship among the items, and processing capacity of the manufacturing system. The scheduling problem is solved with simulated annealing and genetic algorithms, which are search heuristic often used for global optimization in complex search space

John M. Usher &Royce O. Bowden [26] presents an approach to operation sequence coding that permits the application of genetic algorithm (GA) for quickly determining optimal, or near-optimal, operation sequences for parts of varying complexity. This approach improves on existing techniques by utilizing common sequencing constraints to guide the coding process resulting in a further reduction in the size of the solution search space. These improvements permit the determination of near-optimal operation sequences for complex parts within a time frame necessary for real-time dynamic planning. The overall goal of this research project is the development of a distributed process planning system where planning is performed in two stages: resource-independent planning followed by resource dependent planning. A feasible sequence is deemed to be one which does not violate any of the feasibility constraints (Location reference, Accessibility, Non-destruction, Geometric-tolerance, Strict precedence). Application of the feasibility constraints provide the system with the capability to define a set of precedence's between the features of a part resulting in the construction of a feature precedence graph (FPG) to represent these precedence relationships.

This paper presents a strategy for operation sequence representation and coding that can be used within an evolutionary search technique. A solution is represented using a string where each element represents a feature and the order of those features within the string defines the operation sequence. Therefore, the size of the solution is defined by the number of features which comprise the part. Given that the fitness of each member has been determined, the system makes use of an elitist strategy by identifying the member with the best fitness and placing it in the next generation unchanged. Next, members from the population are selected for reproduction and the remainder of the new generation is created using the genetic operators: crossover and mutation. The probability (Prob) that a member is selected for reproduction is based on its fitness relative to the fitness of the other members in the population (e.g. Prob [selecting Member (i)] = Fitness Member (i)/ ∑Fitness of all members). Members are selected two at a time and the application of the standard single point crossover operator is considered based on a predefined probability. Mutation of each element within the sequence string is also determined based on a predefined probability.

The fitness calculations include the number of setups, continuity of motion, and loose precedence. For each criterion, a scoring method was devised to rate the fitness of each alternative on a scale from 0 (representing the worst case) to 1 (the best). The square of the weighted sum of these scores is used to determine the overall fitness of each population member.

Application of this strategy is illustrated using three parts of varying complexity as well as comparing the genetic algorithm's performance using the improved constrained coding strategy with that of an unconstrained strategy. The results of this study also show that improvements in the search for optimal sequences for process plans are possible through the incorporation of constraints within the representation and coding of the operation sequences. This use of constraints in the decoding of operation sequences was found to simplify the application of the genetic operators and enhance the performance of the genetic algorithm as compared to the use of a genetic algorithm without considering constraints. The performance improvements result from the ability of the constrained approach to find a better (and often optimal) solution more often than the unconstrained approach, and to find this solution in fewer generations (meaning less time). Therefore, this technique for planning operation sequencing has proven beneficial for its use as a module within the development of a computer-aided process planning system. Also, based on these results, the use of genetic algorithms shows promise in supporting the notion of dynamic process planning.

John M. Usher & Kiran J. Fernandes [27] presented an architecture for a computer-aided process planning system that is meant to be integrated with a scheduling system in which the dynamic process planning chore is to divide the planning tasks into the two phases: static and dynamic planning. The system is termed, PARIS (Process Planning Architecture for Integration with Scheduling). Overall approach to planning used in PARIS is divided into two phases: static planning and dynamic planning. The dynamic planning phase takes place when a job is released for production to the shop-floor. At this time, the macro-level plans are retrieved and planning is completed taking into consideration the availability of the shop-floor resources and the objectives specified by the scheduler. The primary goal of this first planning phase is to perform as many of the planning functions as possible that do not rely on information concerning the current status of the shop-floor resources. The secondary goal is to generate a list of alternative plans and archive the results for subsequent evaluation in the second phase. The steps performed during the static planning phase include: process selection, process sequencing, and machine group selection. Dynamic phase is the detailed planning phase which considers the operational status of the shop-floor resources. Remaining tasks to be performed include: tool selection, machine parameter determination, selection of jigs and fixtures, final machine selection, operation planning and sequencing, as well as, calculation of estimates for both machining cost and time. Using the cost and time estimates determined by the system for the alternative plans, the best plan will be selected based on one estimate or some weighted combination of the two as specified by the user.

S.A. Irani, H.Y. Koo, & S. Raman [28] explored in depth the integration of a machinist's concept of manufacturing precedence among part feature with a complete and explicit graph representation for alternative process plans. A precedence graph that captures the implicit predecessor-successor cost for any directed pair of features observed in the part replaces the state space representation adopted by AI-based search strategies.

The feature –based approach explored in this research combines the variant and coding generative computer aided process planning concepts using Graph Theory. The classification and coding system developed for group technology (GT) application in the 60's list the basic feature in to which any part can be decomposed. For each of these features, any experienced machinist can identify the machine tools, tools, fixtures, tool-workpiece orientations and relative motions, intermediate feature transitions, etc. immediately at the outset when a part drawing is studied. Given that the process plan is a sequence that links these part features, it is equivalent to the least cost Hamiltonian path extracted from the feature precedence cost matrix. The attribute vectors of the two features can be compared to evaluated precedence, if any, between them. The relative costs can be approximated as the number of tasks that need to be performed in each category of physical effort such as machining parameter changes, tool changes, setup changes, machine changes (arranged in order of increasing complexity). A process planner usually selects a process which can produce as many features as possible on one machine so that the total number of machine changes can be minimized. Analysis of the setup for producing any simple feature gives its physical attributes such as change of machine, orientation of the workpiece and tools in setup on the machine, relative motion between the tool and workpiece. The reason for ignoring the machining parameters is because higher quality surface finishes and dimensional tolerances either require adjusting the controls on the same machine or changing to different machine.

This study employed the Latin multiplication method (LMM) and Lin Kernighan (LKM) to generate alternative process plans and to rank them in order of increasing cost. A precedence matrix was used to represent the precedence among all pairs of features observed in a part. The problem with enumerating all the alternative sequences by using exhaustive enumeration is that the computational time and storage complexity increases exponentially.

LMM is a symbolic operation method for precedence matrices equivalent to breadth first search in graph theory and AI search. This method is practically only for a small precedence matrix(<8 nodes) or a high degree of sparseness in the case of large precedence matrix. LKM is a local search algorithm. Users of LKM may limit the complexity of the computation by constraining the search of solution space. An optimal solution can not be guaranteed unless the complete solution space is searched.

So the thrust of this research was bidirectional-incorporating the machinist's heuristic knowledge of manufacturing precedence and implicit costs to make generative CAPP software viable in practice, and systematically generating and ranking several feasible process plans which the planner may not have even considered in the first plan.

Cheng –Jung & Hsu-Pin [29] divide the operation planning in to two phases: phase 1 being the determination for each part scheduled for production for the next planning horizon; phase 2 being the sequencing of those machining steps defined in process plan in order to completely remove machinable volumes from blank piece. In a modern manufacturing system, we often schedule a number of parts for production for each planning horizon (production cycle). Each part may have a couple of feasible process plans. From cycle to cycle, the part mix constantly changes. It is the process planner's job to find the optimal process plan for each of those parts scheduled for production for each planning horizon.

In the 2nd phase, the process planner selects appropriate tools from among a group of feasible ones for those process plans. The major concern is to minimize the tool to this end, a two-stage approach is proposed. In stage 1, 0-1 integer programming model for the purpose of identifying feasible tool changeover combination for each one of the selected parts for production. In stage 2, another 0-1 integer programming model is developed for determining the optimal tool changeover sequence for each part. The enumeration begins from phase 2, as one has to know the total cost of each feasible process plan through the two- stage integer-programming models. The cost of information can be used, in phase 1, to determine the optimal configuration of process plan. So machining steps are then sequenced in an effort to minimize the total machining cost and tooling cost.

C. Koulamas [30] developed solutions techniques for the combined machining economics- operations sequencing problems. The problem of determining the sequence and cutting speeds for a number of operations performed using the same tool, where the independently calculated optimal cutting speeds differ among operations. The essence of the proposed solution method is that it may be beneficial to perform some operations at non-optimal speeds in order to reduce the cost associated with speed changes. The problem is initially formulated as a continuous nonlinear optimization problem combined with a discrete combinatorial scheduling problem. After the problem is descretized, an efficient heuristic technique, the overlapping speed ranges (OSR), is proposed for finding good solutions to the problem quickly. The heuristic determines the cutting speeds and the sequencing of operations utilizing the indifference speed ranges which are based on the sensitivity of operation's unit machining costs to the speed change cost. A branch and bound algorithm performs extremely well because it utilize as a strong upper bound the complete solution provide by the OSR heuristic, and because tight lower bounds are computed for all nodes throughout the branching procedure. Finally, for large problems a heuristic variant of the branch and bound procedure is suggested.

Rajiv Singh & Shivkumar Raman [31] provides a comprehensive survey of existing literature on machinability parameters election systems, followed by a literature- based analysis of the anomalies of machining and the process and material effects in machining. The different existing computerized machinability database system such as data storage & retrieval system, generalized empirical equation systems, mathematical model systems, machinability data selection and expert systems are discussed. A justification of expert system for qualitative modeling is provided.

A prototype system (METEX) for machining parameter selection is then discussed. This system uses production (IF-THEN) rules to qualify machining at various cutting conditions. It provides the user with nominal parameters, and then it gives the user a choice to change these parameters. The system then provides the user with the information about the process at these conditions, in terms of tool behaviour and active wear mechanisms. So the user can make decisions based on the situation at hand to select cutting parameters. Use of such a system will obviate the need to do real time experiments before the selection of final machining parameters. Currently the system is restricted to four combinations of tool and work materials, and it can be further extendable for other new tool and work materials, which increases it application in industry.

Present Work

The following observations are made from the literature survey and chapter 1.

1. Various optimization techniques are required for the formulation of computer aided process planning because as the part complexity increases the number of potential solutions increases exponentially, so it becomes very critical activity.
2. Operation sequencing is one of the most important activity of computer aided process planning to manufacture the part as per the details mention in the engineering drawing. Any fixed sequence of operations can not be the best possible sequence for all production periods for the criteria such as quality and machine utilization.
3. An efficient search methodology is required for finding the valid operation sequence from large solution space under various sequencing constraints as mentioned earlier.
4. Traditional techniques have very limited scope because of the complexity of the optimization problem. So attempts have been made to include the potential of artificial intelligence in process planning applications and to optimize all events in an integrated (computer aided design) CAD/ (computer aided manufacturing) CAM environment.

Objective and Scope

The objective of the proposed methodology is automated selection of all feasible machining operation sequences and the preferred operation sequence after taking cost factor into consideration by using a genetic algorithm based approach & then assigning the optimum machining parameters.

The scope of the present methodology is restricted to features commonly encountered in symmetrical axis, rotational parts such as holes, external steps, etc. and the selection of operations is done for machining each feature of the part independently.

To satisfy the above criteria, the present work should be explored by following steps mention below:-

1. The part drawing is analysed to identify the form features to be machined, with details of geometric specifications and technological information such as surface finish and tolerance requirements.
2. Then various feasibility constraints are considered for generating the precedence relationship among machining operations.
3. Find the best sequence of operations for prevailing dynamic production environment by the application of Genetic Algorithm.

So an attempt is made to develop an efficient algorithm, which would give the optimum operation sequence in terms of cost and time effectiveness.

CHAPTER 3

METHODOLOGY & ALGORITHM

3.1 Introduction

The essence of operation sequencing involves determining in what order to perform a set of selected operations such that the resulting order satisfies the precedence constraint established by both parts and operations. The overall goal of this research project is the development of a process panning system where operation sequencing is independent of the availability of resources.

On the basis of the manufacturing constraints discussed earlier, each operation in the process plan is mapped to a specific shape feature in the part drawing. A precedence cost matrix (PCM) is generated for any pair of features based on the approximate relative cost corresponding to the number of tasks that need to be performed in each category of attribute.

The development of various feasible plans and identification of the best among these constitutes a combinatorial optimization problem, for which a heuristic search using a genetic algorithm is chosen for finding the optimal or near optimal sequence of operations. After sequencing the operations, range of depth of cut, feed & speed is assigned to each operations using expert system.

3.2 Identification of Form Features

According to the process planning perspective, a feature must be identified as an enhancement of shape, surface finish or size produced by a countable set of specific physical actions. These actions can be classified as changes in machining parameters, tool, set-ups or machines. Almost all the standard handbooks on machine shop practice describe these features with photograph of complete setup for the operation on the machine tool. The example in case of rotational parts are facing, step turning, rough turning, finish turning, drilling, boring, counter boring, reaming & chamfering etc as shown in Figure 3.1.

Figure 3.1: Feature library of lathe operations [28]

In this work we include 10 features which are facing, step turning, taper turning, rough turning, finish turning, drilling, boring, counter boring, reaming & chamfering. In the first 'Form Feature' group only those feature are included which are performed on the external surface of the workpiece, has no precedence between them, can be performed on the same machine tool. The features which satisfy these conditions are facing, step turning, taper turning and chamfering.

The next 'Form Feature' includes that feature which can be performed on same machine tool by machining the external surface of workpiece but they must have precedence between them. So rough turning and finish turning are included in this group.

The last and the third 'Form feature' group includes those feature which are performed by machining the workpiece internally, can be performed on more than one machine, and have precedence between them. Drilling, boring, counter boring, reaming are included in this group. Table 3.1 shows the details of feature in a particular group.

Form Feature Group Code No.	Features
0	• Facing • Step Turning • Taper Turning • Chamfering
I	• Rough Turning • Finish Turning
II	• Drilling • Boring • Counter Boring • Reaming

Table 3.1 Form Feature Group

If any feature mention in the 'Form Group' I or II repeats then 'Form Group' number goes on increasing.

3.2 Feature Precedence Analysis

After converting the operation to be performed on workpiece in to 'Form Feature" group, now is the time to check their precedence like boring can not performed before drilling and counter-boring can not be performed before boring. Likewise finish turning is performed after rough turning. Generally for 'n' feature part there should be 'n!' sequences but it does not happen due to the manufacturing constraints. So a feature from any 'Form Group' is selected randomly based on the precedence of operations, shown in table 3.2

<table>
<tr><th>Form Feature Group No.</th><th>Features</th><th colspan="3">Precedence of Features</th></tr>
<tr><td>0</td><td>• Facing
• Step Turning
• Taper Turning
• Chamfering</td><td colspan="3">Perform Operations in ANY order</td></tr>
<tr><td>I</td><td>• Rough Turning
• Finish Turning</td><td colspan="3">1. Rough Turning
2. Finish Turning</td></tr>
<tr><td rowspan="4">II</td><td rowspan="4">• Drilling (D)
• Boring (B)
• Counter Boring (CB)
• Reaming (R)</td><td>1. D</td><td>1.D</td><td>1.D</td></tr>
<tr><td>2. B</td><td>2. B</td><td>2. R</td></tr>
<tr><td>3. CB</td><td>3. R</td><td></td></tr>
<tr><td>4. R</td><td></td><td></td></tr>
</table>

Table 3.2 Feature Precedence

3.3 Generation of Precedence Cost Matrix [28]

A preceding cost matrix (PCM) is generated for any pair of features based on the approximate relative cost corresponding to the number of tasks that need to be performed in each category of attributes such as machining parameter change, tool change and set-up change, machine tool change & the type of constraint one feature has with the other, such as pre-condition, location, datum holding & bi-directional.

A process planner usually selects a process sequence which can produce as many feature as possible on one machine so that the total number of machine changes can be minimized. Thus to reflect the operational complexity of such change between the attribute of a pair of feature, the machine tool change attribute is given the highest cost among all attributes categories and the value of this attribute category will be larger than the sum of values of all preceding categories.

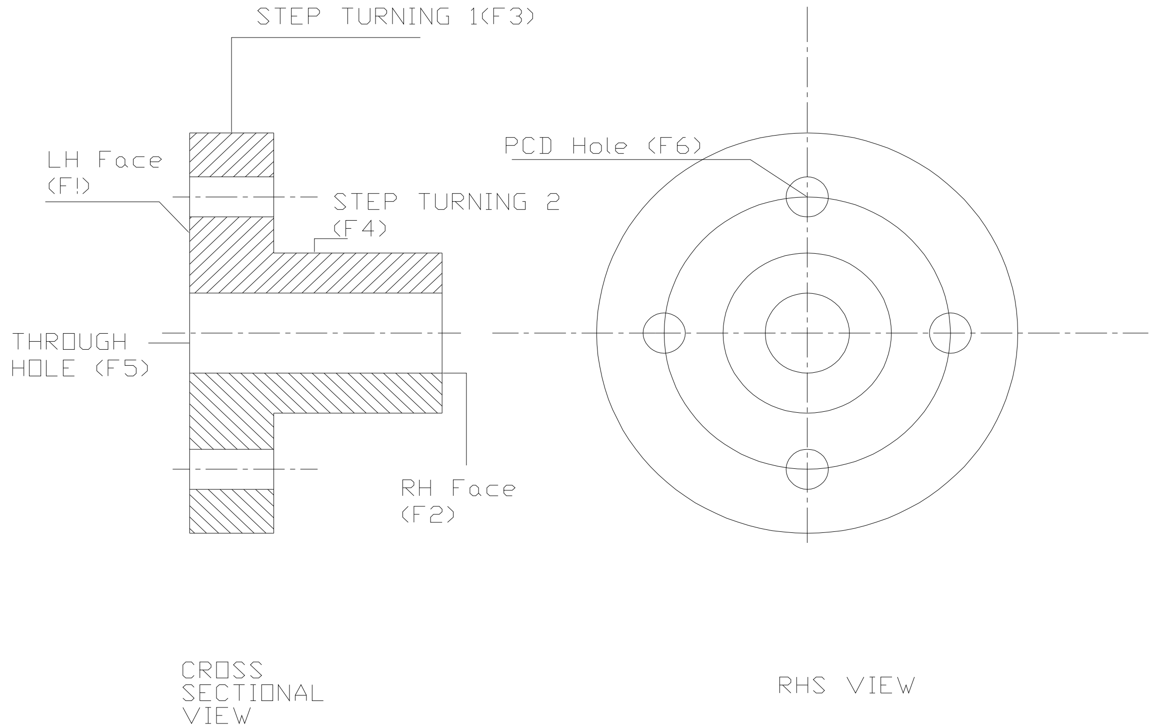


Figure 3.2 Feature Decomposition of a Part [28]

The part shown in figure 3.2 is rotational, stepped to one side, has a through hole at center and holes drilled on a pitch circle diameter (PCD). Table 3.3 shows the feature attribute vectors for each of these features. Table 3.3 shows how to get the precedence cost for any pair of features by observing the value of the attributes a shown in table 3.3 Features F1-F4 constitutes one cluster of operation in the process plan, regardless of the intra –cluster machining sequence for them because they can all be produced on the lathe. Feature F5 could be produced on Pillar Drill. However, this will change the machine & orientation of the particular tool to make the feature. The machinist would automatically include it in the F1 to F4 feature cluster. This leaves feature F6 which has the maximum complexity because it could be on either of the Pillar Drill or Gang Drill.

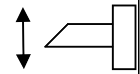
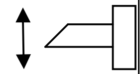
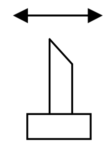
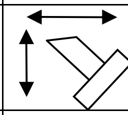
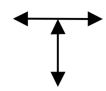
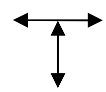
Feature No.	Machine	Tool Type	Machine Setup		Work Piece Motion	Tool Motion
			Location of Job	Orientation of Axis		
F1	Lathe	Straight Turning	Left End	Co-Linear	A.C.	
F2	Lathe	Straight Turning	Right End	Co-Linear	A.C.	
F3	Lathe	Straight Turning	Left End	Co-Linear	A.C.	
F4	Lathe	Straight Turning	Right End	Co-Linear	A.C.	
F5	Lathe, Pillar Drill	Drill	Right or Left End	Co-Linear	A.C.	
F6	Pillar Drill, Gang Drill	Drill	Right End	Parallel	Stationary	

Table 3.3 Feature Precedence Table [28]

If Feature 'I' Precedes Feature 'J' Then

Begin

If Parameter Change Is Required	**C [I, J] = P**
If Tool Change Is Required	**C [I, J] = PT**
If Set-Up Change Is Required	**C [I, J] = PTS**
If Machine Change Is Required	**C [I, J] = M**

End

Else If 'I' Does Not Proceed 'J' **Then C [I, J] = X**

A machine change must involve tool change & a tool change must involve setup change cost so that the cost value of each operational change is then the sum of all preceding categories. The cost assigned for each attribute included in this research work shown in table 3.4: -

Attribute	Cost	Total Cost
Parameter Change (P)	P=0	P=0
Tool Change (T)	T=1	PT=1
Setup Change (S)	S=10	PTS=11
Machine Change (M)	M=100	M=100
Not Preceding Operation (X)	X=999	X=999

Table 3.4 Precedence Matrix

Now apply all instruction mention above to the figure 3.2, then observations made are as follows and are shown in table 3.5 & table 3.6: -

- F1-F6 : The attribute vectors have different values for the machine attribute. The cost from F1 to F6 will include a machine change cost. Because Lathe machine is required to generate F1 and Pillar Drill or Gang Drill is needed for F6
- F3- F4 : The attribute vectors have different values for the location on workpiece which will require the workpiece to be flipped end to end. The cost for the change from F to F4 will include a setup change cost.
- F2- F3 : The attribute vectors have different value for the tool motion. The cost for the operation from F2 to F3 will include a control parameter change cost.

	F1	F2	F3	F4	F5	F6
F1	X	PT	P	PT	PTS	M
F2	PT	X	PT	P	PTS	M
F3	P	PT	X	P	PTS	M
F4	PT	P	X	X	PTS	M
F5	PTS	X	PTS	PTS	X	M
F6	X	X	X	X	X	X

Table 3.5 Cost of Each Attribute [28]

So the Precedence Cost Matrix (PCM) for part shown in figure 3.2 is: -

	F1	**F2**	**F3**	**F4**	**F5**	**F6**
F1	999	1	0	1	11	100
F2	1	999	1	0	11	100
F3	1	1	999	0	11	100
F4	1	0	999	999	11	100
F5	11	999	11	11	999	100
F6	999	999	999	999	999	999

Table 3.6 Precedence Cost Matrix

3.4 Genetic Algorithm for Operation Sequencing [22] [26]

In GA the candidate solution is represented by a sequence of numbers known as chromosome or string. A chromosome's potential as a solution is determined by its fitness function, which evaluates a chromosome with respect to the objective function of the optimization problem under consideration. A judiciously selected set of chromosomes is called a population & population at a given time is a generation. The population size remains fixed for generation to generation and has a significant effect on performance of GA. GA's operates on a generation and consist of three main operations (as shown in figure 3.3): -

- *Initialization*: - Randomly generate a population, which satisfies all the (manufacturing) constraints.
- *Fitness Evaluation*: - Calculate the fitness value for each string from precedence cost matrix (PCM).
- *Reproduction*: - Selection of copies of chromosome proportional to their fitness value.
- *Crossover*: - An exchange of sections of chromosomes.
- *Mutation*: - A random modification of chromosome.

The chromosome resulting from these operations, often known as offspring or children, from the next generation's population. The process is repeated for a desired number of generations, usually up to a point where the system converges to a significant well performing sequence.

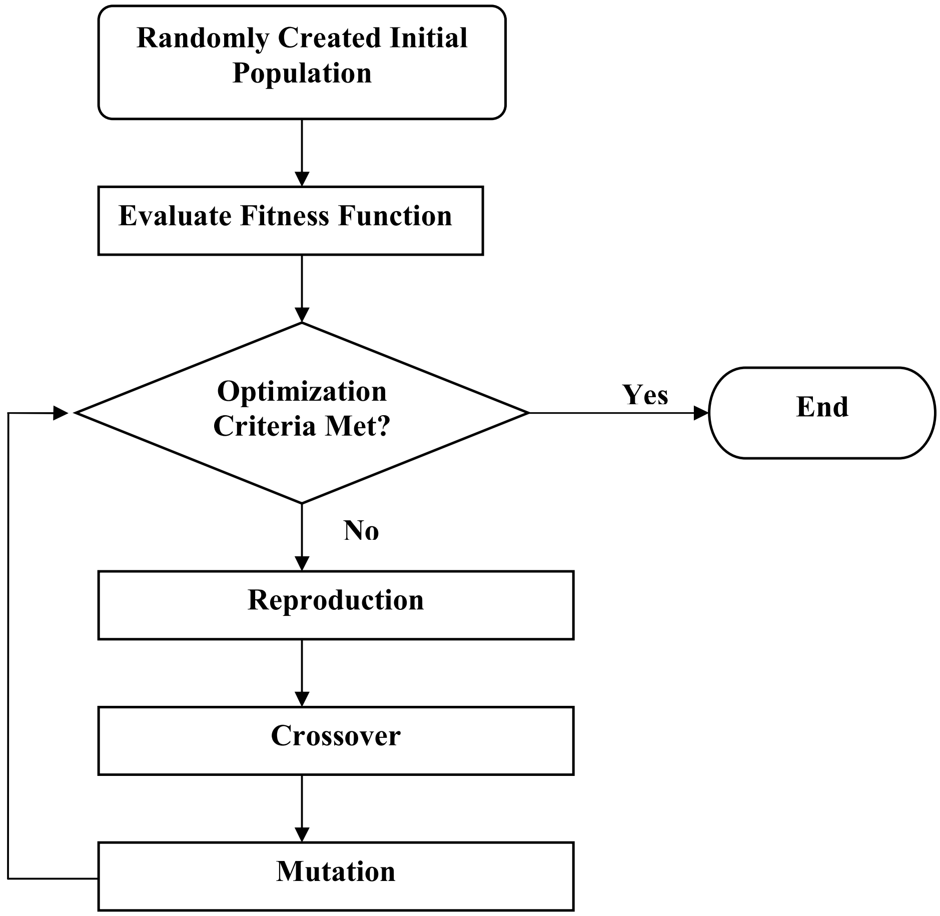


Figure 3.3 Flowchart of Genetic Algorithm [13]

3.4.1 Creation of Initial Population

The initial population can not consist of simple random generated strings as the local precedence of operations or features for each ‘Form Feature’ cannot be guaranteed. To create a valid initial string, an element of the string is generated randomly, from the first operations of each ‘Form Feature’ group so as to follow the precedence order of the operations and the procedure is repeated by selecting elements from the remaining operations groups until all the operations are represented in the string. Let ‘n’ be the number of features to be performed then the total number of strings generated initially are equal to 2n.

3.4.2 Evaluation of Fitness Function [22][26]

The objective of the sequencing problem is to get an optimal sequence that results in minimum production cost from the given precedence cost matrix (PCM).

The objective function is calculated for each string in the production as the sum of relative cost between the features or operations. The relative costs will correspond to the number of tasks that need to be performed in each category of attribute such as machine change, tool change, parameter change, setup change and the type of constraint one feature has with respect to the other. The fitness value for the each string is calculated and the expected count of each string for the next generation is obtained on the basis of the string weightage (survival of fittest) so that the total count becomes the population size, because as mention above in the whole cycle of GA the size of the population should remains same.

3.4.3 Reproduction [1]

This genetic operator is used to generate a new population which has better strings than the old population. The selection of the better strings is based on the actual count, if the actual count is more than the population size than the strings with poor fitness functions are removed and if actual count is less than the population size then string from population are added with high fitness function such that the size of total population remains unchanged. The reproduced population is used for the next GA operator that is crossover.

3.4.4 Crossover [1] [7] [13]

Crossover is a Genetic operator that generates new individuals based upon combination and possibly permutation of genetic material of ancestors. The crossover is carried out between the Parent 1 and Parent 2 by using bits that represent the alternative operation sequence that can be used. The two children are than reproduced, the crossover site can also be selected randomly.

In solving operation sequencing problem, the single point crossover for each pair of the individuals can only occur between the genes representing exactly operation sequences. If two parents to be used for generating new chromosomes are:

Parent A =	**1**	**2**	**3**	**4**	**5**
Parent B =	**5**	**3**	**1**	**4**	**2**

Let the crossover site is 2 (generated randomly)

Child A' =	**1**	**2**	**1**	**4**	**2**
Child B' =	**5**	**3**	**3**	**4**	**5**

From the example above it is clear that single point crossover produce undesirable results, i.e. few feature are repeating and few are missing in the child string. If the application of Partially Matched Crossover (PMX), which is a two point crossover is considered as shown below gives the undesirable results. Under PMX, two strings are taken randomly and two crossing sites are picked uniformly at random along the strings. These two points define a matching section that is used to effect a cross through position-by-position exchange operation.

To see this consider two strings [1]

Parent A =	**9**	**8**	**4**	**5**	**6**	**7**	**1**	**3**	**2**	**10**
Parent B =	**8**	**7**	**1**	**2**	**3**	**10**	**9**	**5**	**4**	**6**

Let the two crossover sites are 3 & 6 (generated randomly).

PMX proceeds by positionwise exchange. First, mapping string B to string A, the 5 with 2, 3 with 6, and the 10 with 7 exchanges. Similarly mapping string A to string B, the 5 and the 2, the 6 and the 3, and the 7 and the 10 exchange places. Following PMX we are left with two offspring i.e. Child A' and Child B'.

Child A' =	**9**	**8**	**4**	**2**	**3**	**10**	**1**	**6**	**5**	**7**
Child B' =	**8**	**10**	**1**	**5**	**6**	**7**	**9**	**2**	**4**	**3**

PMX operator as shown above does not ensure the local precedence of operations while generating offspring. So a new crossover operator is designed by modifying the PMX operator given by Goldberg & Lingle [1]. To produce a feasible offspring, two parents are randomly selected from the population. Based on the string length, two crossover sites are randomly generated to select a segment in one parent between these sites. The offspring, child 1, is generated by arranging the elements of the selected segments in this parent according to the order in which they appear in the other parent with the order of the remaining elements being the same in the as in the first parent, so as to obey the precedence of operation or features as mentioned earlier. The role of these parents will then exchange in order to generate another offspring, child 2. The crossover operator can be illustrated as follows.

Parent A =	**9**	**8**	**4**	**5**	**6**	**7**	**1**	**3**	**2**	**10**
Parent B =	**8**	**7**	**1**	**2**	**3**	**10**	**9**	**5**	**4**	**6**
Child A' =	**9**	**8**	**4**	**7**	**5**	**6**	**1**	**3**	**2**	**10**
Child B' =	**8**	**7**	**1**	**3**	**2**	**10**	**9**	**5**	**4**	**6**

3.4.5 Mutation [1]

Mutation is done to protect the loss of some potentially useful strings and to avoid being struck at the local optimum. The mutation operator makes random changes to one or more elements of the string. Mutation is done with a small probability (generally 0.001).The mutation operator proposed here randomly modifies the two elements to obtain the resulting population. However, there is possibility of the string becoming infeasible by violating the local precedence of operations for the 'Form Feature' groups.

So to avoid this the feasibility of the string element is checked, and if the string is infeasible it's total cost is given a very high value so that it would not come in next generation. The process of mutation is shown below. Let the string is:

Parent A = 9 8 ⋮ 4 5 ⋮ 6 7 1 3 2 10

And the mutation sites are 2 & 4. Then after mutation Parent A becomes Parent A'

Parent A' = 9 8 5 4 6 7 1 3 2 10

3.5 Machinability Data Selection Expert Systems [31]

As we know that the machining process exhibits piecewise behaviour and cannot be linearly extrapolated in a wide range. It also cannot be defined in a short range, and cannot be modeled effectively using theories and equations. Expert systems have emerged as a major tool for decision-making in such complicated situations. The need for an expert system arises because of the inherent weakness of process models to yield a logical solution to the data selection problem. An Expert system approach has the capability to take care of the numerous heuristics, exceptions, and anomalies associated with metal cutting process. It is felt that knowledge of metal cutting physics if properly complied can be coded symbolically IF-THEN rules and can from the basis of an expert system.

The knowledge base (KB) contains the actual expertise of the system. It may be represented in the form of IF-THEN rules, frames, networks, or objects. The inference mechanism is the control mechanism, which drives the system to reach a conclusion.

A series of recommended cutting speeds, feeds and other related information is stored in computer, which can be retrieved through a user-friendly (if-else) interface program. This information comes from machining data, design data handbooks. In this work three materials (Cast Iron, Carbon steel, Free cutting steel) with different range of hardness are included. The user has to select the material first and then enter the value of Brinell hardness number from the given range. The user enters the type of tool used for machining here the system provides two (HSS Toll, Carbide Tool) type of tools in the library of tools. The system gives the range of depth of cut for that particular material with specific BHN and tool used and allows the user to enter the value of depth of cut from that range and then gives the range for feed. Now user enters the value of feed from the given range and then the system in turn given the range for speed to perform the operation. Present work provides data (speed, feed depth of cut) for materials subjected turning, facing, boring, which can be extended as per requirement.

3.6 Assumptions

The following assumptions are framed before implementing the GA to the present work as described above.

- This is used only for sequencing the features or operations of rotational components.
- The material of the workpiece should not be composite.
- Maximum number of features should not more than 20, otherwise computer program will give the error.
- The datum location should always be taken from left hand side (LHS) of the part drawing.
- If any of two or more feature has the same specification i.e. same machine tool, same tool, same control parameters, same set-up then these operations are considered as single operation. For example, drilling of two same diameter holes with same specification on the same machine tool, then these features are considered as single feature or operation.
- Probability for crossover operator and mutation operator is taken as 1 in order to reduce the computation time of the program.
- This work provides data only for facing, turning, & boring operations.
- Tool life is 90 to 120 minutes, considered as constant.

3.7 Summery

The methodology described above can be summarized as follows:-

1. Making the standard material library.
2. Entering the material code from standard material library. (It contains three different materials)
3. If the entered code is out of library, go the step 2.
4. Making the Brinell Hardness Number (HB) library for each material mention in material library.
5. Entering the Brinell Hardness Number of material
6. If the BH number is not exist in the library go step 5.
7. Making the standard feature library. (It contains 10 different types features)
8. Entering the number of features to be produced on part (n).
9. If the numbers of features are more than 20 i.e. n>20, go to step 6.
10. Entering the feature code from the standard feature library.
11. Penalty cost matrix is generated for all the features entered by the user.
12. Arrange the entire feature in increasing order of their feature code.
13. Identify the form features and group them according to the 'Form Feature' group instructions.
14. Arrange the features from 'Form Feature' groups to 'Precedence' groups according to their precedence relationship as mention in the precedence relationship analysis
15. Generation of feasible strings that satisfy all the manufacturing constraints, here number of strings are double the number of features entered in step 8 i.e. 2n.
16. Calculation of each feasible string from penalty cost matrix generated in step 11.
17. Find the string with maximum cost (M).
18. Calculate the fitness value for each string.

    Fitness Value=Maximum cost (M) - Cost of each string
19. Calculate the average cost (Avg Cost).

    Avg Cost =Σ Fitness value/ Total number of strings

    If Avg cost =0, optimal sequence is reached, Go to Step 33.
20. Calculate the Expected Count for each feasible string.

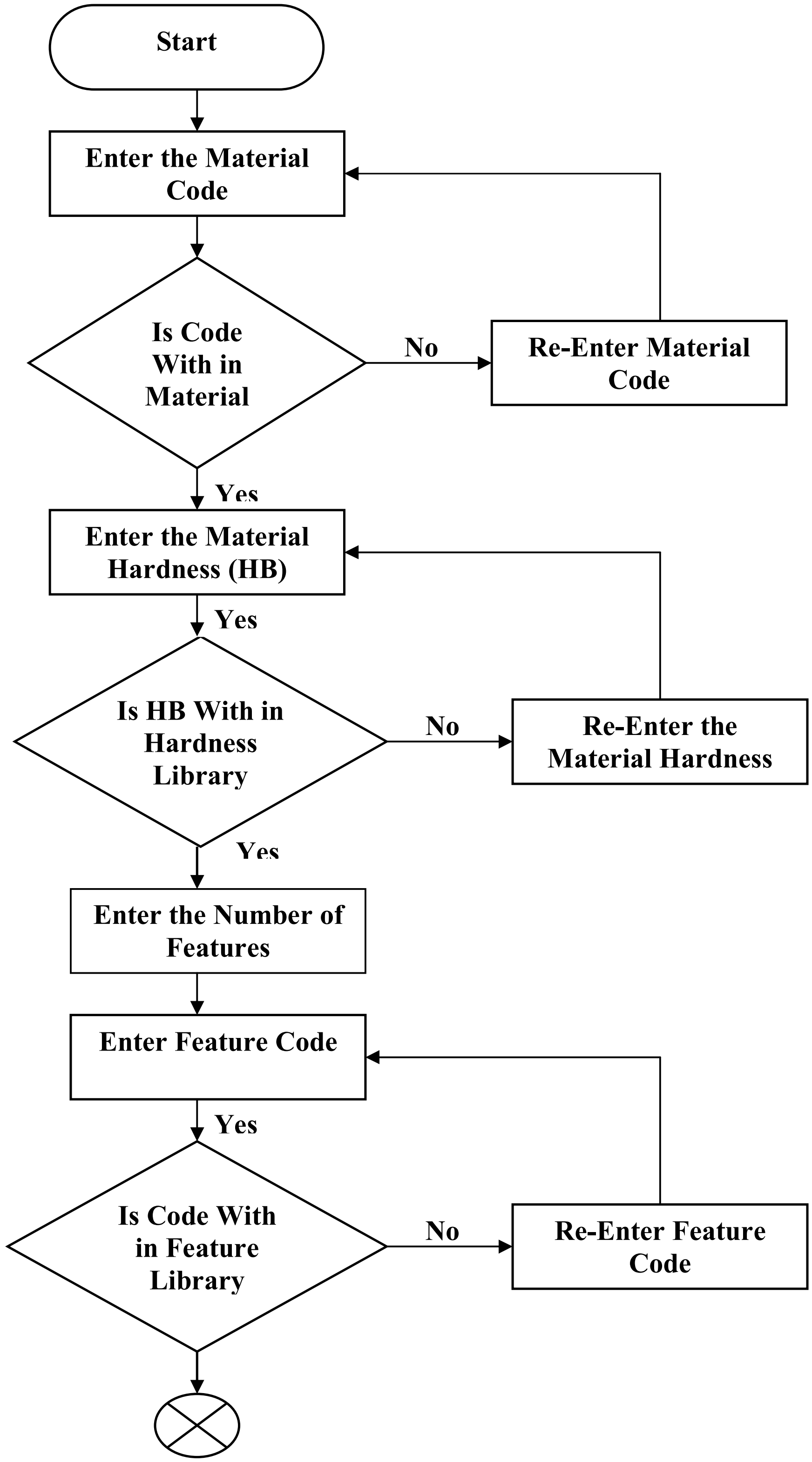
    Expected Count = Fitness Value / Average Value
21. Calculate the Actual count (AC) for each feasible string (by rounding of the float value of expected count in to integer value and term it as Actual count).

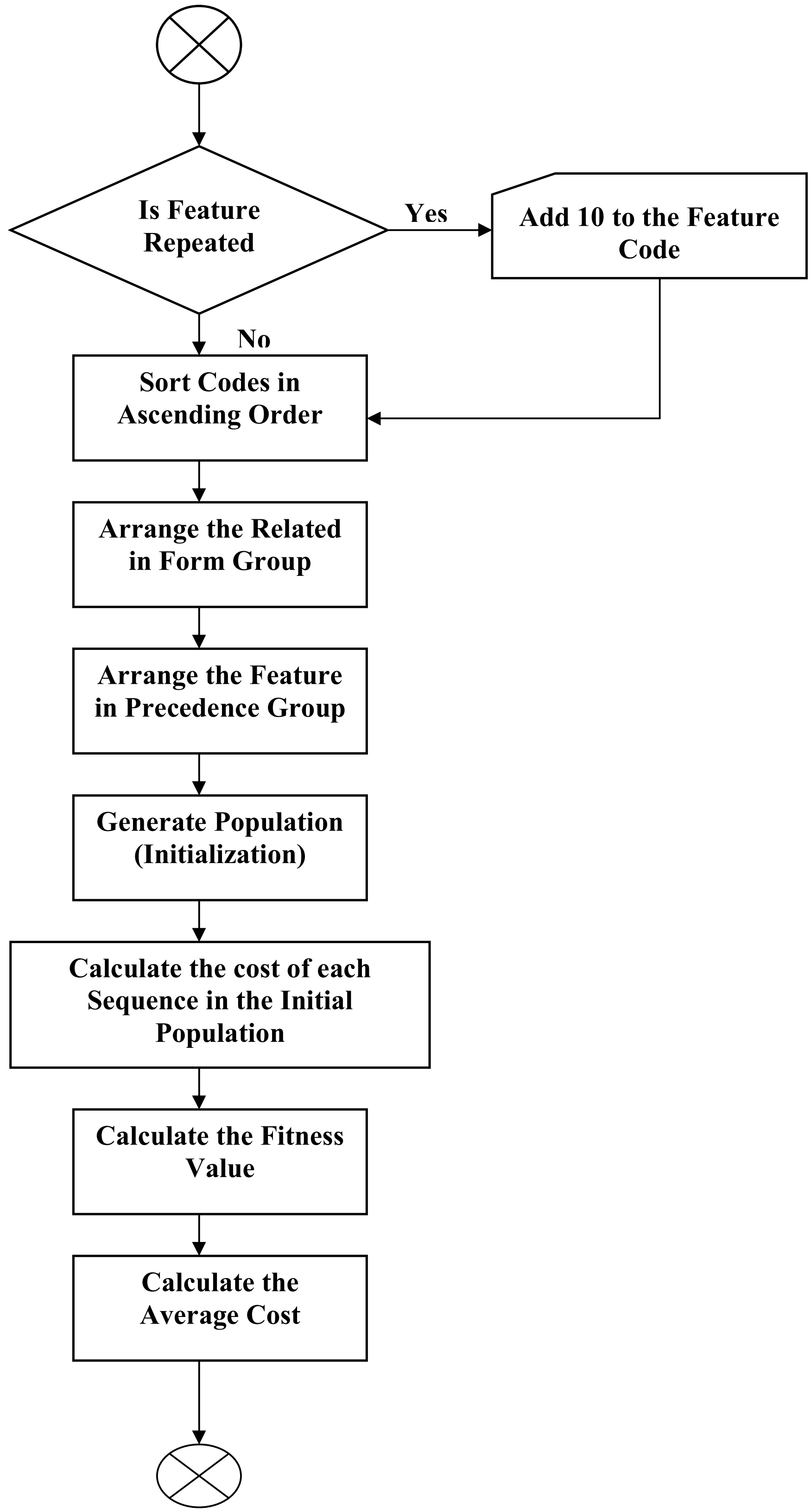
22. Calculate the Total Actual Count.

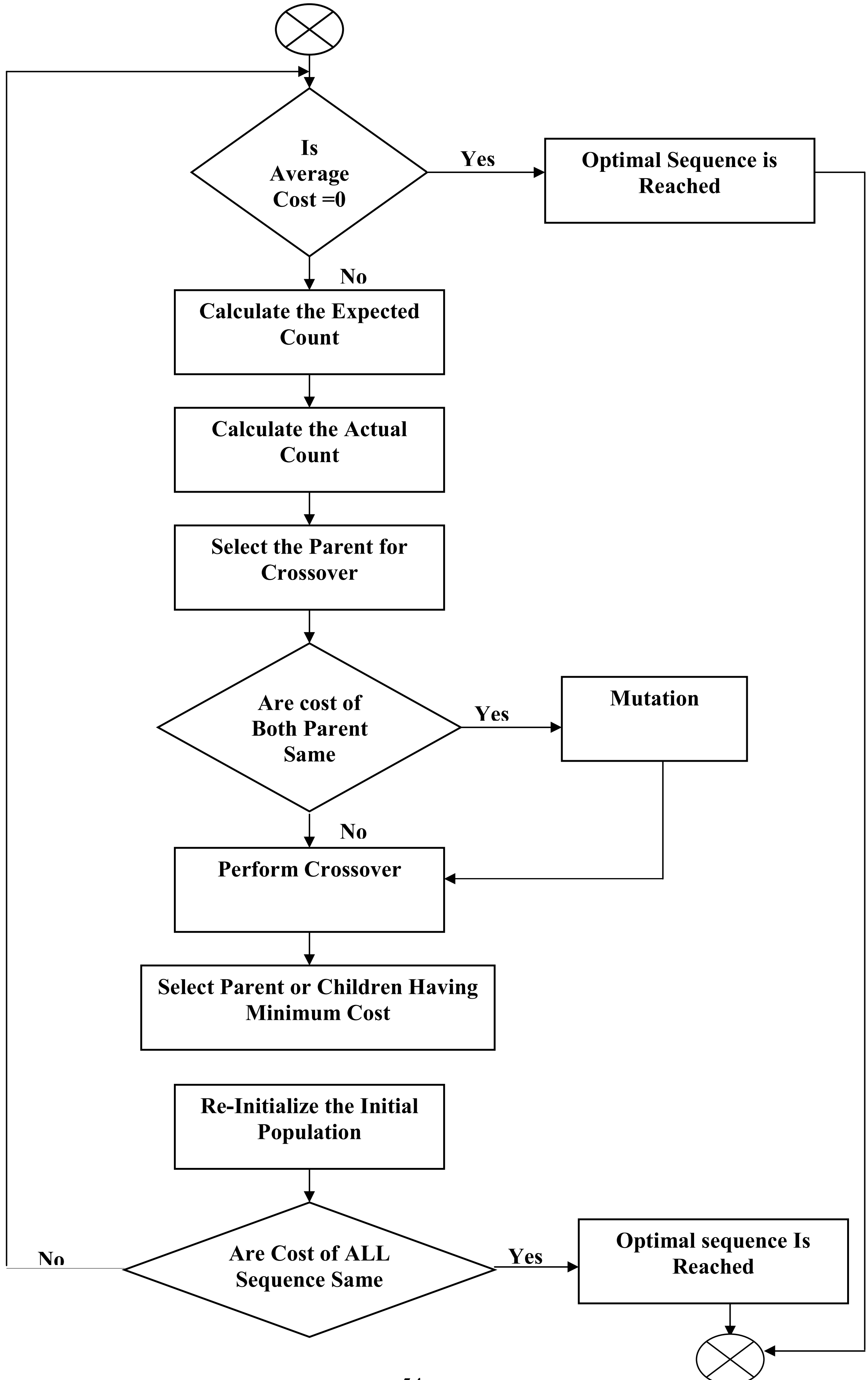
    Total Actual Count = Σ Actual Count

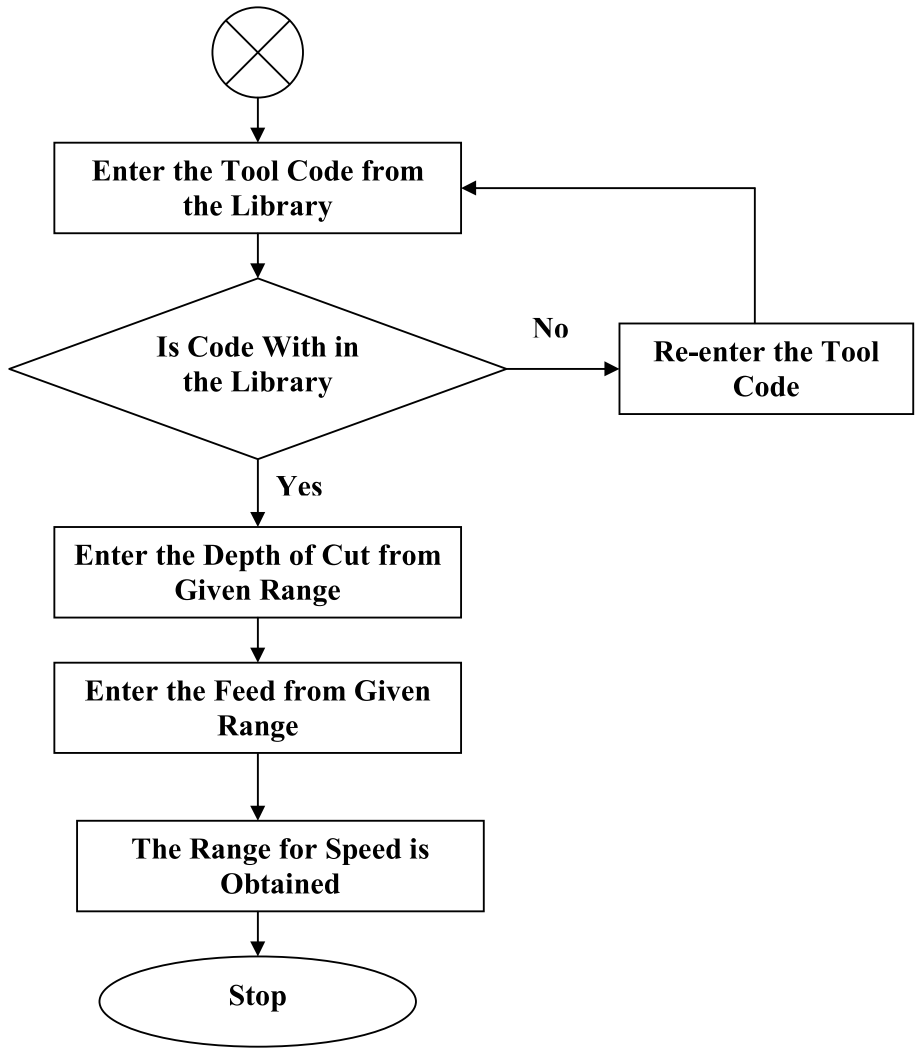
23. If Total Actual Count is more than number of strings generated in step 15, then arrange them with cost ascending downwards and after that sequence with lower cost replace the initial population.
24. If Total Actual Count is less than number of strings generated in step 15, then arrange them with cost descending downwards and after that sequence with higher cost replaced the initial population with lower cost.
25. Randomly selects two strings (Parent 1 & Parent 2) from feasible strings.
26. If two parents i.e. parent 1 & parent 2 has same cost then go to step 27 else go to step 28.
27. Mutate.
28. Crossover.
29. Calculate the cost of all 4 strings i.e. parent 1, parent 2, child 1 & child 2.
30. Select one with minimum cost among 4 for the new population.
31. Repeat step 16 to 30 for a specified number of generations. At the end the string(s) with minimum cost is taken as the optimal sequence.
32. Making the Tool library.
33. Enter the code for tool used to perform the operation.
34. If the code enter is wrong then go to step 33.
35. Enter the value of depth of cut from the given range.
36. Enter the value of the feed from the given range.
37. The range of speed is given as the output.

3.8 Flow Chart




Is Feature Repeated
Yes
Add 10 to the Feature Code
No
Sort Codes in Ascending Order
Arrange the Related in Form Group
Arrange the Feature in Precedence Group
Generate Population (Initialization)
Calculate the cost of each Sequence in the Initial Population
Calculate the Fitness Value
Calculate the Average Cost


Is
Average
Cost =0
Yes
Optimal Sequence is Reached
No
Calculate the Expected Count
Calculate the Actual Count
Select the Parent for Crossover
Are cost of Both Parent Same
Yes
Mutation
No
Perform Crossover
Select Parent or Children Having Minimum Cost
Re-Initialize the Initial Population
No
Are Cost of ALL Sequence Same
Yes
Optimal sequence Is Reached


Enter the Tool Code from the Library
Is Code With in the Library
No
Re-enter the Tool Code
Yes
Enter the Depth of Cut from Given Range
Enter the Feed from Given Range
The Range for Speed is Obtained
Stop

CHAPTER 4

COMPUTER PROGRAM DISCUSSIONS

To prove the validity of the algorithm, a program was made in Turbo C++ on a DOS platform which implemented this algorithm, this section is consist of several functions which are discussed below: -

4.1 main ( ): -

The execution of the program starts from this function. All other functions are called by this function in an order mentioned below, the details of each function are given in the following sections: -

- randomize ( )
- material_lib ( )
- ci_bhn ( )
- carbon_steel ( )
- f_cutting_steel_bhn ( )
- feature_lib ( )
- display_cost_mat ( )
- sort ( )
- group_coding ( )
- precedence_coding ( )
- initial_population ( )
- population_cost ( )
- find_max ( )
- fitness_fun ( )
- total_count ( )
- sel_of_parent ( )
- crossover_fun ( )
- mutation_fun ( )
- find_minimum ( )
- next_population ( )
- descending ( )
- ascending ( )

- terminate ( )
- result ( )
- parameter ( )
- tool_type ( )
- cast_iron ( )
- carbon_steel ( )
- f_cutting_steel ( )

4.2 randomize ( ): -

This is the in built function in the standard library of C++. This function initializes the random number generator with a random value. The randomly generated values by this function are used for selecting crossover site between any two selected strings. This function is implemented by including the TIME.H header file in the program.

4.3 material_lib ( ): -

This function incorporates the various types of material on which manufacturing operations are to be performed. The user has to enter code for material of his/ her choice. Name of material and their respective codes are enlisted below: -

CODE	MATERIAL
0	CAST IRON
1	CARBON STEEL
2	FREE CUTTING STEEL

4.4 ci_bhn ( ): -

This function includes the different types of cast iron materials on the basis of their Brinell Hardness number (BHN). The user has to enter the value of BHN of the material on which operations are to be performed.

TYPE OF CI	BHN RANGE
SOFT	110 to 160
MEDIUM	160 to 220
HARD	220 to 360

4.5 carbon_steel_bhn ( ): -

This function incorporates two type of carbon steel with different BHN and it allow user to choose one of them.

TYPE	BHN RANGE
C20, C45	85 to 125
20 Ni55CrMo2	125 to 175
	175 to 225
	225 to 275
	275 to 325
	325 to 375
	375 to 345

4.6 f_cutting_steel ( ): -

This function incorporates different type of free cutting steel with varying BHN ranges and it allow user to choose one of them.

TYPE	BHN RANGE
12S12	100 to 150
	150 to 200
	200 to 250

4.7 feature_lib ( ): -

This function incorporates various features (operations) require for manufacturing a rotational component and their respective codes. The various features and their respective codes used in this function are: -

CODE	FEATURE
0	Facing
1	Rough Turning
2	Finish Turning
3	Step Turning
4	Taper Turning
5	Drill
6	Boring
7	Counter Boring
8	Reaming
9	Chamfering

4.8 display_cost_mat ( ): -

The penalty cost matrix for the feature sequence entered by user is generated by this function. It picks the cost between each pair of feature from the globally declared matrix, which consist of cost of each feature enter by user before running the program based on precedence analysis (for both succeeding and proceeding feature). The size of this globally declared matrix is 20 X 20. So if number of features entered by user is more than 20, then this function will give the error. This function shows the penalty cost matrix only for the features entered by the user.

If any feature is repeated e.g. to drill three holes of different diameters then new code for the repeating features, is generated in the main function by adding 10 to its basic code. As mention earlier that code for drilling is 5 then new codes for the repeated features are 5, 15, and 25. This is necessary to differentiate the repeated features and provide the different code, if this will not happen the same cost will be assigned to the repeated feature & provide the wrong result.

If any feature is repeated and it needs the same machine tool, same cutting tool, no change in tool control parameter, & same setup then tat feature will be considered as single feature/ operation.

4.9 sort ( ): -

The features entered by user are arranged in ascending order by this function. So the user may enter feature code in any order, this function sort the feature code in proper sequence which is the foremost requirement in this program so as to make the 'Form Feature' group easily.

4.10 group_coding ( ): -

This function identified the form features required for the manufacturing a part and always gives first priority to this feature, as mentioned earlier. After identifying the form feature, from the feature entered by the user, it gives the unique code to each feature based on the type of 'Form Feature' group to which it belongs.

To create a valid initial string an element of string is generated randomly from the first operation of each 'Form Feature' group & the procedure is repeated by selecting element from the remaining 'Form Feature' group until all the operations are represented in the string.

Let to manufacture a part the features are facing (A), rough turning (B), finish turning (C), step turning (D), taper turning (E), drilling (F), boring (G), counter boring (H), reaming (I), chamfering (J), then this function generates the following code for these features: -

FEATURE	**CODE**
Facing (A)	0
Rough Turning (B)	1
Finish Turning (C)	1
Step Turning (D)	0
Taper Turning (E)	0
Drilling (F)	2
Boring (G)	2
Counter Boring (H)	2
Reaming (I)	2
Chamfering (J)	0

It results three form feature group containing the following features: - (A, D, E, J), (B, C), (F, G, H, I).

4.11 precedence_coding ( ): -

In this function, the entire 'Form Feature' groups identified in the previous function are used to generate the strings. In case of example considered above each string in the population should contain ten elements corresponding to ten operations. The first element of the string is generated randomly from the first element of three 'Form Feature' groups i.e. from A, B, F. if A is selected randomly from these three elements, then second element of the string has to be generated randomly from D, B, F. Now if B is selected as the second element, the third element of the string is generated randomly from D, C, F. This procedure is repeated until all the elements are not included in each string. The second element would not appear until the first feature of that group appears first. So this function again codes this coded group of features in to binary codes. The codes assigned by this function to the same example are: -

FEATURE	CODE
Facing	1
Rough Turning	1
Finish Turning	0
Step Turning	1
Taper Turning	1
Drilling	1
Boring	0
Counter Boring	0
Reaming	0
Chamfering	1

The features which are coded as 1 always comes first while generating feasible number of sequence and feature coded as 0 come afterwards.

4.12 initial_population ( ): -

The feasible strings are generated by this function. The total numbers of strings generated are twice the number of features. For example if the number of features entered by user are 10 then total number of feasible string becomes 20.

4.13 population_cost ( ): -

This function calculates the total cost of each feasible string. A feasible string is nothing but a string which must satisfies all the constraints.

4.14 find_max ( ): -

This function finds the string which has maximum cost among all generated feasible strings.

4.15 fitness_function ( ): -

This is very important function, which finds out the fitness value of each feasible string. [1] [22]

Fitness Value = Maximum Cost – Cost of Each String

i.e. $(U_{max})_i = M-(U_{min})_i$

Where $(U_{min})_i$ is the total cost of each feasible string and
M is the maximum value.

After calculating the fitness value of each string, this function then calculates the expected count for each string. The expected count can be calculated as [1] [22]:-

$$\text{Expected Count } (E_i) = (U_{max})_i / (U_{avg})$$

Where U_{avg} is the average of the cost of all feasible strings.

$$U_{avg} = \Sigma(U_{max})_i / \text{Total Number Of Feasible Strings}$$

If average cost is zero then optimal sequence is reached.

After calculating the expected count, this function calculates actual count for each string to be appeared in the crossover. The count should not be in decimal form so this function converts the float value of expected count in to integer value, i.e. actual count. For example if the expected count is 1.87 then its actual count will be 2 and if expected count is 1.19 then 1 will be its actual count.

4.16 total_count ( ): -

This function calculates the total number of actual count for all the strings as mentioned earlier.

4.17 sel_of_parent ( ): -

This function store the feasible strings according to the number of actual count i.e. if the actual count for a particular string is 2 then it appears twice in the population for mating with other strings and similarly if actual count for a particular string is 0, then it will not appeared in population for mating. The population generated by this function is called ‘parents’. The string which has the maximum cost has 0 actual count.

4.18 cross_over_fun ( ): -

This function randomly selects two parents from the population to produce a feasible offspring. Based on the string length, two crossover sites are randomly generated to select a segment in one parent between these crossover sites. The offspring, child 1, is generated by arranging the elements of the selected segment in this parent according to the order in which they appear in the other parent with the order of the remaining elements being the same as in the first parent. The role of these parents will then exchange in order to generate other offspring, child 2. The crossover operator can be illustrated as follows: -

Select two strings randomly from the population and denote them as Parent 1 and Parent 2, [22]

Parent 1 =	**1**	**2**	**3**	**4**	**5**	**8**	**6**	**7**
Parent 2 =	**8**	**2**	**1**	**4**	**5**	**6**	**7**	**3**

Consider two random crossover sites as X=2 and Y=6 and the segment from Parent 1 between crossover sites is 3, 4, 5, 8. Arranging the selected elements in order of Parent 2, results in 8, 4, 5, 3, then the offspring Child 1from Parent 1 is generated as: -

Child 1 =	**1**	**2**	**8**	**4**	**5**	**3**	**6**	**7**
Child 2 =	**8**	**2**	**1**	**4**	**5**	**6**	**7**	**3**

4.19 mutation_fun ( ): -

This function is called only, when the cost of two parents are same. This function makes the changes to one or more elements of the string. Mutation is done with a small probability. This is done to protect the loss of some potentially useful strings and to avoid being struck at the local optimum. The mutation operator proposed here randomly modifies the two elements to obtain the resulting population. However, there is possibility of the string becoming infeasible by violating the local precedence of operations for the 'Form Feature' groups. So to avoid this the feasibility of the string element is checked, & if the string is infeasible it's total cost is given a very high value so that it would not come in next generation.

4.20 find_minimum ( ): -

This function finds the minimum value of cost among each set of Parent 1, Parent 2, Child 1, Child 2 and then store the minimum value & corresponding sequence of string for the next population & delete the remaining three. This function is applied to all other parents and their off springs.

4.21 next_population ( ): -

This function regenerates the population for next cycle. If Total Actual Count is more than total number of strings generated as initial population, then arrange strings with cost ascending downwards and after that sequence with lower cost replace the initial population. If Total Actual Count is less than number of strings generated as initial population, then arrange strings with cost descending downwards and after that sequence with higher cost replaced the initial population with lower cost.

4.22 descending ( ): -

This function rearranged the new population generated by crossover & mutation in descending order of cost, when the initial population is greater then total number of actual counts.

4.23 ascending ( ): -

When the initial population is less then the total number of actual count, then this function is rearranged the initial population in ascending order of cost.

4.24 terminate ( ): -

The iterative procedure of the program is terminated by this function. It returns zero, when all the sequences of new population have the same cost & terminate the process otherwise it will return 1 & iterative process will continue.

4.25 result ( ): -

The strings with minimum cost obtained after termination of iterative process will be printed as result by this function.

4.26 parameter ( ): -

This function calls one function from three functions each carry the parameters for different material.

4.27 tool_type ( ): -

This function incorporates the different cutting tool material and allows the user to choose one from them. The code for different cutting tool as follows: -

CODE	TOOL MATERIAL
0	HSS
1	CARBIDE

4.28 cast_iron ( ): -

This function carries the parameters (depth of cut, feed and speed) for turning, facing & boring operations for cast iron with both type of cutting tools.

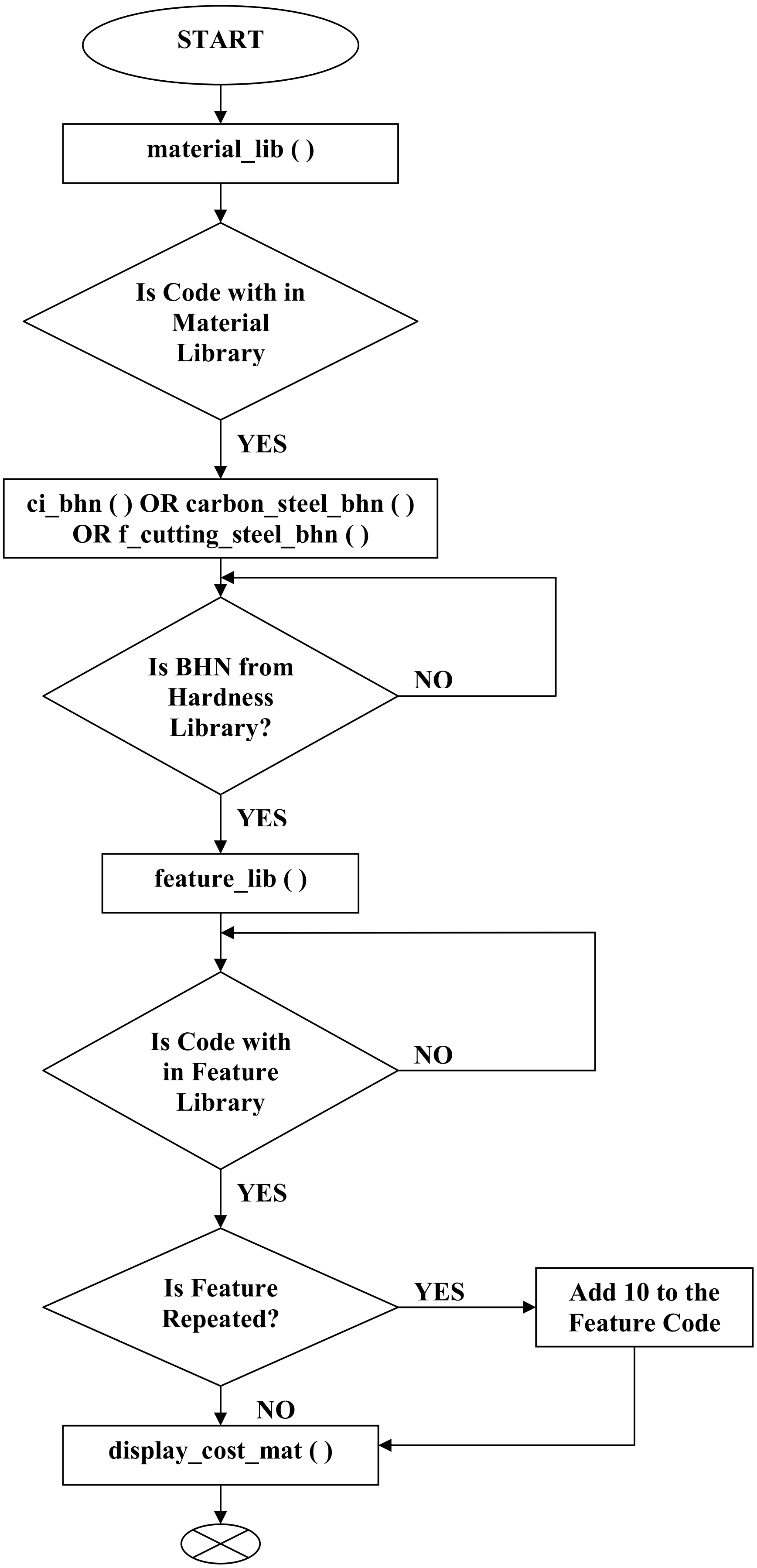
4.29 carbon_steel ( ): -

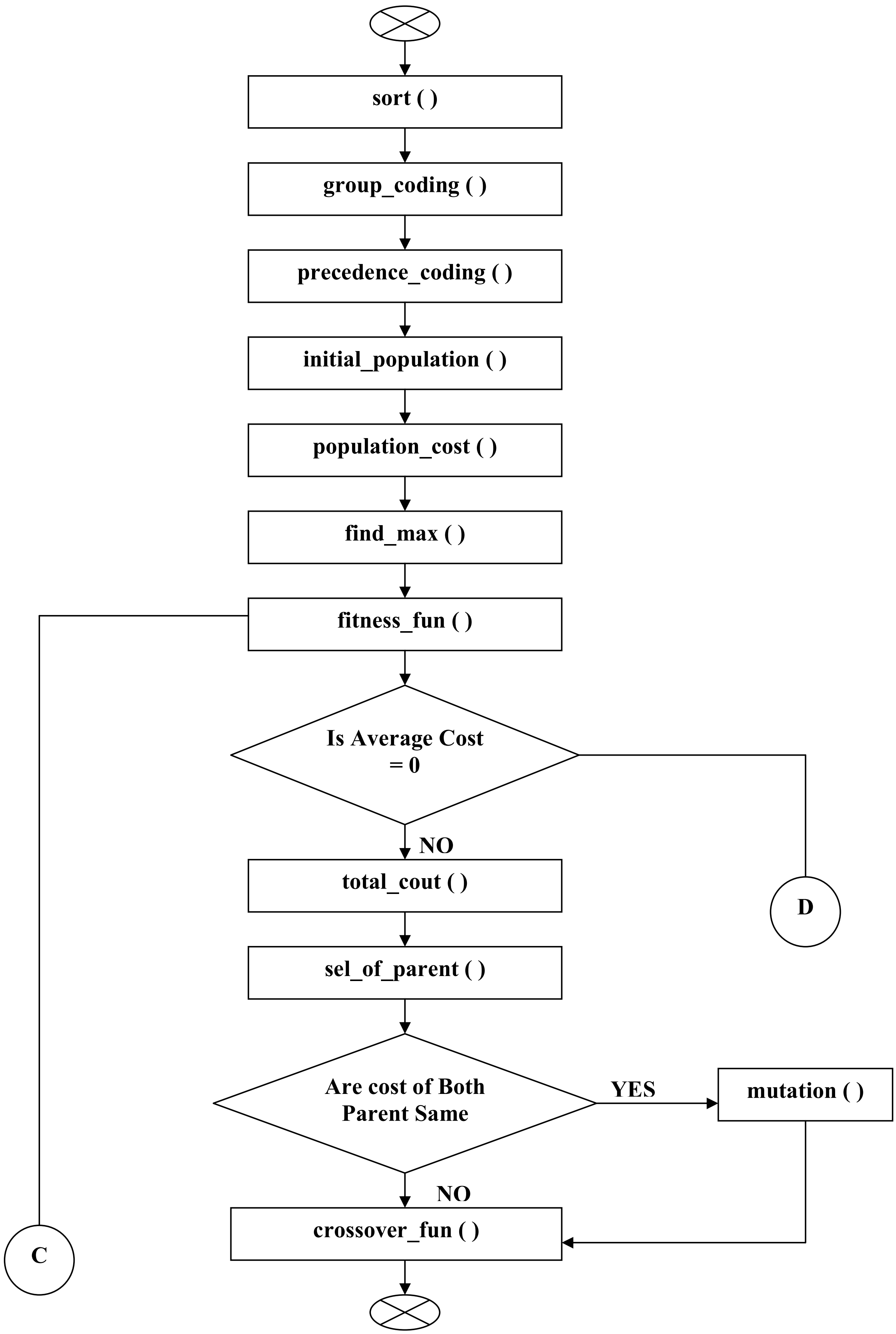
This function carries the parameters (depth of cut, feed and speed) for turning (rough, finish), facing & boring operations for carbon steels with both type of cutting tools.

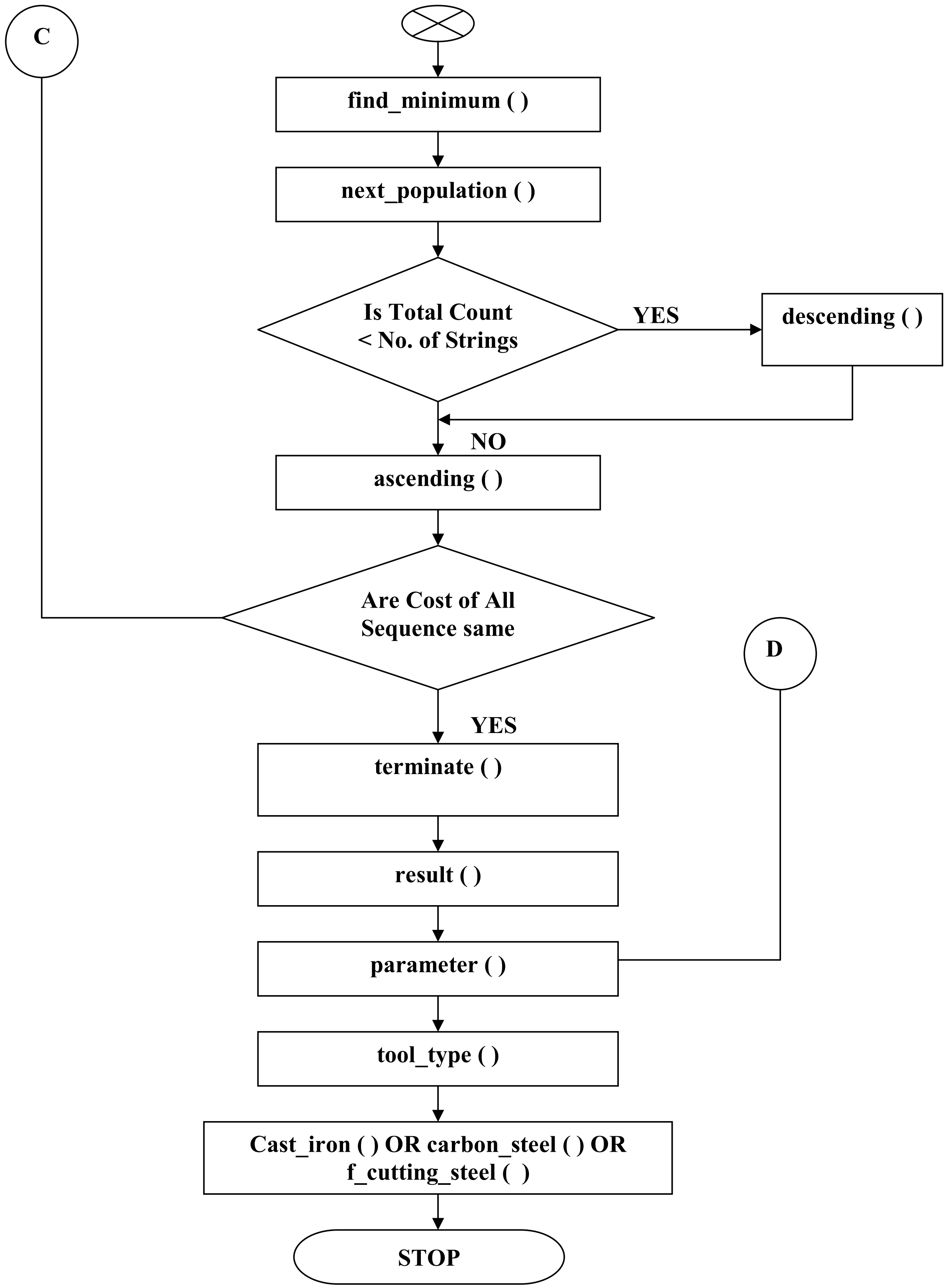
4.30 f_cutting_steel ( ): -

This function carries the parameters (depth of cut, feed and speed) for turning, facing & boring operations for free cutting steels with both type of cutting tools.

4.31 Flow Chart




sort ( )
group_coding ( )
precedence_coding ( )
initial_population ( )
population_cost ( )
find_max ( )
fitness_fun ( )
Is Average Cost = 0
NO
total_cout ( )
D
sel_of_parent ( )
Are cost of Both Parent Same
YES
mutation ( )
NO
crossover_fun ( )
C


C
find_minimum ( )
next_population ( )
Is Total Count
< No. of Strings
YES
descending ( )
NO
ascending ( )
Are Cost of All
Sequence same
D
YES
terminate ( )
result ( )
parameter ( )
tool_type ( )
Cast_iron ( ) OR carbon_steel ( ) OR
f_cutting_steel ( )
STOP

CHAPTER 5

RESULTS & DISCUSSIONS

A computer program has been developed based on the algorithms discussed in chapter 3. Details of the program are discussed in chapter 4. This program can be used for sequencing the operations (maximum up to 20) of any rotational component. After sequencing the features, this program gives the machining parameters for turning, boring and facing operations. Operation sequence generated by other researchers are also presented and discussed in this chapter.

5.1 Operation Sequencing for a Rotational Part

To validate the operation sequence generated from the Genetic Algorithm based program, a case study is taken from Weill et. al. (S. V. Bhashkara Reddy et. al.) [22] and suitably modified as shown in figure 5.1. Let the material of the part shown in figure 5.1 is Carbon steel with Brinell hardness of 200. The total numbers of features to be generated on component are 8, and are labeled as A1, B1, B2, C1, D1, D2, D3, and E1. These operations are coded as 5, 1, 2, 8, 5, 6, 7, and 9. The data enter by user is shown in the table 5.1.

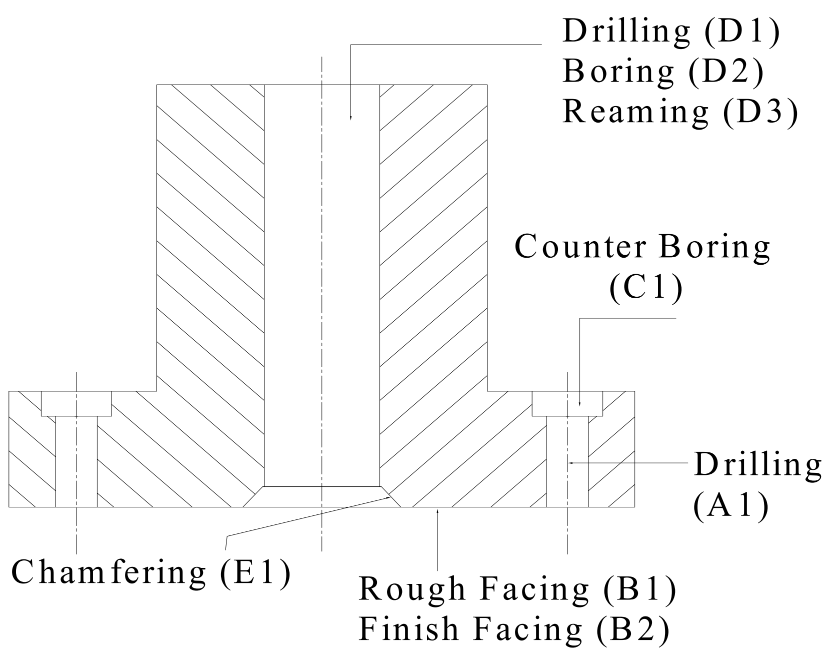


Figure 5.1 Example of case study of Weill et al. [22]

MATERIAL	CODE
Carbon Steel	1
MATERIAL	**BHN Value**
20 Ni55CrMo2	200
FEATURE	**CODE**
Drill (A1)	5
Rough Facing (B1)	1
Finish Facing (B2)	2
Counter Boring (C1)	8
Drill (D1)	5
Rough Boring (D2)	6
Finish Boring (D3)	7
Chamfering (E!)	9

Table 5.1 Input data

Here drill is repeating so its code becomes 15 (i.e. 10 is added in its basic code). The penalty cost matrix for the feature entered by user is generated by display_cost_mat ( ) function, which will be shown in table 5.2

Features	5	1	2	8	15	6	7	9
5	999	100	100	1	100	100	100	100
1	11	999	0	100	1	100	100	100
2	11	100	999	100	1	100	1	1
8	100	100	100	999	100	100	100	100
15	11	1	100	100	999	0	100	100
6	11	1	100	100	100	999	100	100
7	11	100	100	100	100	100	999	100
9	100	100	100	100	100	100	1	999

Table 5.2 Precedence cost matrix (PCM) for figure 5.1

The function group_coding ( ) identified the form features & group them in different 'Form Feature' groups as shown in Table 5.3

FEATURE CODE	FORM FEATURE GROUP CODE
1	1
2	1
5	2
6	2
7	2
8	2
9	0
15	3

Table 5.3 Form feature codes

The function precedence_coding ( ) code the features identified by the 'Form Feature' group on the basis of their precedence, as shown in table 5.4

FEATURE CODE	PRECEDENCE CODE
1	1
2	0
5	1
6	0
7	0
8	0
9	1
15	1

Table 5.4 Precedence codes

The feasible strings are generated by initial_population ( ) function. The total numbers of strings generated are twice the number of features. Here, numbers of features entered by user are 8 then total number of feasible string becomes 16. Next function population_cost ( ) calculates the total cost of each feasible string. The generated strings and their cost are shown in table 5.5

The function find_max ( ) finds the string which has maximum cost among all generated feasible strings. In this case, maximum cost is 700 units.

The function fitness_function ( ) calculate the fitness value for each string, and also calculate the expected count and actual count. Table 5.6 shows the values calculated by this function.

S. No.	STRING	COST
1	1-2-5-6-7-8-9-15	511
2	5-1-15-2-9-6-7-8	502
3	1-5-9-15-6-7-2-8	511
4	15-5-6-9-7-8-1-2	402
5	5-1-15-6-7-2-9-8	402
6	5-1-15-6-7-8-9-2	501
7	15-1-5-6-2-7-8-9	413
8	1-5-15-2-6-7-8-9	611
9	15-9-1-5-2-6-7-8	611
10	5-1-2-9-15-6-7-8	401
11	1-9-2-15-5-6-7-8	502
12	1-9-15-5-2-6-7-8	601
13	1-9-2-15-5-6-7-8	502
14	5-1-6-7-8-15-9-2	**700**
15	15-9-5-6-7-8-1-2	600
16	1-9-2-15-5-6-7-8	502

Table 5.5 Cost of feasible strings for figure 5.1

S. No.	STRINGS	COST	FITNESS VALUE $(U_{max})_i = M-(U_{min})_i$	EXPECTED COUNT $(E_i) = (U_{max})_i / (U_{avg})$	ACTUAL COUNT
1	1-2-5-6-7-8-9-15	511	189	1.032787	1
2	5-1-15-2-9-6-7-8	502	198	1.081967	1
3	1-5-9-15-6-7-2-8	511	189	1.032787	1
4	15-5-6-9-7-8-1-2	402	298	1.628415	2
5	5-1-15-6-7-2-9-8	402	298	1.628415	2
6	5-1-15-6-7-8-9-2	501	199	1.087432	1
7	15-1-5-6-2-7-8-9	413	287	1.568306	2
8	1-5-15-2-6-7-8-9	611	89	0.486339	0
9	15-9-1-5-2-6-7-8	611	89	0.486339	0
10	5-1-2-9-15-6-7-8	401	299	1.63388	2
11	1-9-2-15-5-6-7-8	502	198	1.081967	1
12	1-9-15-5-2-6-7-8	601	299	0.540984	1
13	1-9-2-15-5-6-7-8	502	198	1.081967	1
14	5-1-6-7-8-15-9-2	**700**	0	0	0
15	15-9-5-6-7-8-1-2	600	100	0.546448	1
16	1-9-2-15-5-6-7-8	502	198	1.081967	1

Table 5.6 Fitness values and count calculation for figure 5.1

The function total_count ( ) calculates the total number of actual count for all the strings. Here total count is 17. The function sel_of_parent ( ) store the feasible strings according to the number of actual count, then the function cross_over_fun ( ) randomly selects two parents from the population to produce a feasible offspring. If cost of two parents is same then this function calls another function for mutation. The process is shown in table 5.7

Parent 1	Parent 2	Cost 1	Cost 2	Mutation Site		Parent 1	Parent 2	Cross over site	Child1	child 2	Cost1	Cost 2
1-2-5-6-7-8-9-15	15-5-6-9-7-8-1-2	511	402	-	-	-	-	1, 7	1-5-6-9-7-8-2-15	15-1-5-6-7-8-9-2	413	512
5-1-15-2-9-6-7-8	5-1-15-6-7-8-9-2	502	501	-	-	-	-	2, 5	5-1-15-9-2-6-7-8	5-1-15-6-7-8-9-2	601	501
1-5-9-15-6-7-2-8	1-9-2-15-5-6-7-8	511	502	-	-	-	-	3, 6	1-5-9-15-6-7-2-8	1-9-2-5-15-6-7-8	511	511
15-5-6-9-7-8-1-2	1-2-5-6-7-8-9-15	402	511	-	-	-	-	0, 4	5-6-9-15-7-8-1-2	5-6-1-2-7-8-9-15	600	402
15-5-6-9-7-8-1-2	1-9-2-15-5-6-7-8	402	502	-	-	-	-	2, 7	15-5-1-9-6-7-8-2	1-9-15-5-6-7-2-8	601	601
5-1-15-6-7-2-9-8	1-5-9-15-6-7-2-8	402	511	-	-	-	-	3, 7	5-1-15-9-6-7-2-8	1-5-9-15-6-7-2-8	601	511
5-1-15-6-7-2-9-8	15-1-5-6-2-7-8-9	402	413	-	-	-	-	0, 5	15-1-5-6-7-2-9-8	5-1-15-6-2-7-8-9	413	402
5-1-15-4-7-8-9-2	1-5-9-15-6-7-2-8	501	511	-	-	-	-	0, 7	1-5-9-15-6-7-8-2	5-1-15-6-7-9-2-8	511	501
15-1-5-6-2-7-8-9	5-1-15-6-7-2-9-8	413	402	-	-	-	-	0, 5	5-1-15-6-2-7-8-9	15-1-5-6-7-2-9-8	402	413
15-1-5-6-2-7-8-9	1-9-2-15-5-6-7-8	413	502	-	-	-	-	2, 5	15-1-2-5-6-7-8-9	1-9-15-5-2-6-7-8	412	601
5-1-2-9-15-6-7-8	1-5-9-15-6-7-2-8	401	511	-	-	-	-	2, 5	5-1-9-15-2-6-7-8	1-5-9-15-6-7-2-8	700	571
5-1-2-9-15-6-7-8	1-2-5-6-7-8-9-15	401	511	-	-	-	-	2, 7	5-1-2-6-7-9-15-8	1-2-5-9-6-7-8-15	600	511
1-9-2-15-5-6-7-8	1-2-5-6-7-8-9-15	502	511	-	-	-	-	1, 5	1-2-5-9-15-6-7-8	1-2-5-6-7-8-9-15	411	571
1-9-15-5-2-6-7-8	1-9-15-5-2-6-7-8	601	601	[0, 5]	[1,5]	2-5-15-9-1-6-7-8	1-2-5-15-9-6-7-8	2, 5	2-5-1-15-9-6-7-8	1-2-5-15-9-6-7-8	512	511
1-9-2-15-5-6-7-8	15-5-6-9-7-8-1-2	502	402	-	-	-	-	0, 5	15-5-9-1-2-6-7-8	9-15-5-6-7-8-1-2	501	501
15-9-5-6-7-8-1-2	15-1-5-6-2-7-8-9	600	413	-	-	-	-	0, 3	15-5-9-6-7-8-1-2	15-5-1-6-2-7-8-9	501	502
5-15-2-9-1-6-7-8	1-9-2-15-5-6-7-8	601	502	-	-	-	-	2, 7	5-15-1-9-2-6-7-8	1-9-5-15-2-6-7-8	601	700

Table 5.7 Crossover and mutation for figure 5.1

The iterative process is going on i.e. the new population is generated, fitness is evaluated again crossover is performed and so on. The optimal operation sequence 15-6-1-2-9-7-5-8 i.e. D1-D2-B1-B2-E1-D3-A1-C1 is obtained after 90 iterations with a population size of 16 strings having total cost of 15 units each, where as the same sequence is obtained in 200 generations with a population of 40 string having total cost of 15 units each, by S. V. Bhaskara Reddy.

After sequencing the operations system ask the user about tool material for turning, let here tool is HSS. Code enters by user is as: -

CODE	**TOOL MATERIAL**
0	HSS

The range of depth of cut for this choice is as follows: -

DEPTH OF CUT	
Rough Turning	0.6 TO 4.0mm
Finish Turning	0 TO 0.6 mm

Let depth of cut entered by user is 2 mm. Then feed range for this choice is: -

FEED RANGE 0.04 TO 0.4mm/rev

Value for feed entered by user is 0.2 mm/rev. Then range of speed for this depth of cut and feed is 30 m/min to 20 m/min.

Now select tool for boring operation, let the tool for boring is carbide.

CODE	**TOOL MATERIAL**
1	CARBIDE

The range of depth of cut for this choice is as follows: -

DEPTH OF CUT	
Rough Boring	0.25 TO 2.5mm
Finish Boring	UP TO 0.25 mm

Let depth of cut entered by user is 2 mm. Then feed range for this choice is: -

FEED RANGE 0.375mm/rev TO 0.18 mm/rev

Value for feed entered by user is 0.3 mm/rev. Then speed for this depth of cut and feed is 93 m/min. Values for depth of cut, feed and speed, for particular work material and tool combination are stored in the program & are retrieved by if-then rules and are taken from design data book [7] directly.

5.2 Operation Sequencing for a Rotational Part Containing All Ten Features

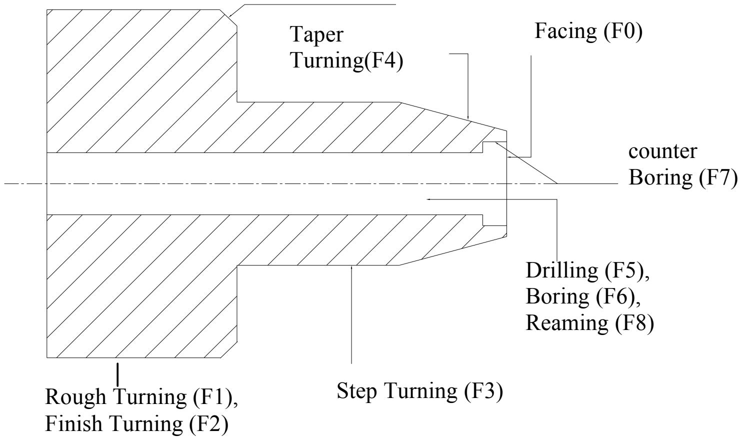


Figure 5.2 Example of a rotational part containing all features

Let the material of the part shown in figure is Carbon steel with Brinell hardness of 200. The total numbers of features are 10. The data enter by user is shown in the table 5.8.

MATERIAL	**CODE**
Free Cutting Steel	2
MATERIAL	**BHN VALUE**
12 S 12	175
FEATURE	**CODE**
Facing (F0)	0
Rough Turning (F1)	1
Finish Turning (F2)	2
Step Turning (F3)	3
Taper Turning (F4)	4
Drill (F5)	5
Boring (F6)	6
Counter Boring (F7)	7
Reaming (F8)	8
Chamfering (F9)	9

Table 5.8 Input data

The penalty cost matrix for the feature entered by user is generated by display_cost_mat ( ) function, which is shown in table 5.9

Feature Code	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
0	999	11	11	0	1	100	100	100	100	0
1	0	999	1	0	1	100	100	100	100	0
2	11	999	999	1	1	100	100	100	100	0
3	0	11	11	999	1	100	100	100	100	0
4	1	11	11	1	999	100	100	100	100	0
5	100	100	100	100	100	999	0	1	1	100
6	100	100	100	100	100	999	999	0	0	100
7	100	100	100	100	100	999	999	999	0	100
8	100	100	100	100	100	100	999	999	999	100
9	1	11	11	1	1	100	100	100	100	999

Table 5.9 Precedence cost matrix (PCM) for figure 5.2

The function group_coding ( ) identified the form features & group them in different 'Form Feature' groups as shown in table 5.10.

FEATURE CODE	**FORM FEATURE GROUP CODE**
0	0
1	1
2	1
3	0
4	0
5	2
6	2
7	2
8	2
9	0

Table 5.10 Form feature codes

The function precedence_coding ( ) code the features identified by the 'Form Feature' group on the basis of their precedence, as shown in table 5.11.

FEATURE CODE	**PRECEDENCE CODE**
0	1
1	1
2	0
3	1
4	1
5	1
6	0
7	0
8	0
9	1

Table 5.11 Precedence codes

The feasible strings are generated by initial_population ( ) function. The total numbers of strings generated are twice the number of features. Here, numbers of features entered by user are 10 then total number of feasible string becomes 20. Next function population_cost ( ) calculates the total cost of each feasible string. The generated strings and their cost is shown in table 5.12.

S. No.	STRINGS	COST
1	0-1-2-3-4-5-6-7-8-9	214
2	9-0-4-3-5-6-1-2-7-8	304
3	3-1-5-2-0-6-9-7-4-8	722
4	1-4-3-9-2-5-0-6-7-8	313
5	9-1-4-0-5-3-6-2-7-8	513
6	9-5-4-1-6-3-7-0-8-2	**811**
7	3-4-1-9-5-6-2-0-7-8	323
8	3-0-4-1-5-9-2-6-7-8	323
9	0-1-5-2-6-4-9-7-3-8	711
10	9-5-1-6-0-7-8-3-4-2	612
11	1-9-3-2-5-0-4-6-7-8	313
12	9-4-1-0-2-3-5-6-7-8	124
13	0-4-1-9-5-2-6-7-3-8	512
14	9-5-3-4-1-2-0-6-7-8	324
15	9-0-1-3-5-6-4-2-7-8	323
16	5-1-6-3-0-2-4-9-7-8	412
17	4-3-1-2-5-0-9-6-7-8	313
18	4-9-0-3-1-5-6-7-2-8	312
19	3-5-0-4-1-9-2-6-7-8	323
20	9-3-5-4-6-0-7-1-8-2	801

Table 5.12 Cost of feasible strings for figure 5.2

The function find_max ( ) finds the string which has maximum cost among all generated feasible strings. In this case, maximum cost is 811.The function fitness_function ( ) calculate the fitness value for each string, and also calculate the expected count and actual count. The values calculated by this function are shown in table 5.13.

S. No.	STRINGS	COST	FITNESS VALUE $(U_{max})_i = M-(U_{min})_i$	EXPECTED COUNT $(E_i) = (U_{max})_i / (U_{avg})$	ACTUAL COUNT
1	0-1-2-3-4-5-6-7-8-9	214	597	1.567546	2
2	9-0-4-3-5-6-1-2-7-8	304	507	1.331233	1
3	3-1-5-2-0-6-9-7-4-8	722	89	0.233688	0
4	1-4-3-9-2-5-0-6-7-8	313	498	1.307601	1
5	9-1-4-0-5-3-6-2-7-8	513	298	0.78246	1
6	9-5-4-1-6-3-7-0-8-2	**811**	0	0	0
7	3-4-1-9-5-6-2-0-7-8	323	488	1.281344	1
8	3-0-4-1-5-9-2-6-7-8	323	488	1.281344	1
9	0-1-5-2-6-4-9-7-3-8	711	100	0.262571	0
10	9-5-1-6-0-7-8-3-4-2	612	199	0.522515	1
11	1-9-3-2-5-0-4-6-7-8	313	498	1.307601	1
12	9-4-1-0-2-3-5-6-7-8	124	687	1.80386	2
13	0-4-1-9-5-2-6-7-3-8	512	299	0.785086	1
14	9-5-3-4-1-2-0-6-7-8	324	487	1.278719	1
15	9-0-1-3-5-6-4-2-7-8	323	488	1.281344	1
16	5-1-6-3-0-2-4-9-7-8	412	399	1.047657	1
17	4-3-1-2-5-0-9-6-7-8	313	498	1.307601	1
18	4-9-0-3-1-5-6-7-2-8	312	499	1.310227	1
19	3-5-0-4-1-9-2-6-7-8	323	488	1.281344	1
20	9-3-5-4-6-0-7-1-8-2	801	10	0.026257	0

Table 5.13 Fitness values and count calculation for figure 5.2

The function total_count ( ) calculates the total number of actual count for all the strings. Here total count is 18. The function sel_of_parent ( ) store the feasible strings according to the number of actual count, then the function cross_over_fun ( ) randomly selects two parents from the population to produce a feasible offspring. If cost of two parents is same then this function calls another function for mutation. The process is shown in table 5.14.

Parent 1	Parent 2	Cost 1	Cost 2	Mutation Site		Parent 1	Parent 2	Cross over site	Child1	child 2	Cost1	Cost 2
0-1-2-3-4-5-6-7-8-9	0-1-2-3-4-5-6-7-8-9	214	214	-	[1, 5]	-	0-4-3-2-1-5-6-7-8-9	0, 5	0-4-3-2-1-5-6-7-8-9	0-1-2-3-4-5-6-7-8-9	1212	214
0-1-2-3-4-5-6-7-8-9	9-0-4-3-5-6-1-2-7-8	214	304	-	-	-	-	4, 9	0-1-2-3-4-5-6-7-8-9	9-0-4-3-1-2-5-6-7-8	214	115
9-0-4-3-5-6-1-2-7-8	9-5-1-6-0-7-8-3--4-2	304	612	-	-	-	-	2, 7	9-0-5-1-6-3-4-2-7-8	9-5-0-6-1-7-8-3-4-2	513	612
1-4-3-9-2-5-0-6-7-8	9-0-4-3-5-6-1-2-7-8	313	304	-	-	-	-	4, 9	1-4-3-9-0-5-6-2-7-8	9-0-4-3-1-2-5-6-7-8	303	115
9-1-4-0-5-3-6-2-7-8	0-4-1-9-5-2-6-7-3-8	513	512	-	-	-	-	0, 4	0-4-1-9-5-3-6-2-7-8	9-1-4-0-5-2-6-7-3-8	512	513
3-4-1-9-5-6-2-0-7-8	0-1-2-3-4-5-6-7-8-9	323	214	-	-	-	-	5, 9	3-4-1-9-5-0-2-6-7-8	0-1-2-3-4-5-6-7-8-9	323	214
3-0-4-1-5-9-2-6-7-8	9-1-4-0-5-3-6-2-7-8	323	513	-	-	-	-	0, 5	1-4-0-5-3-9-2-6-7-8	0-4-1-5-9-3-6-2-7-8	313	513
9-5-1-6-0-7-8-3-4-2	9-4-1-0-2-3-5-6-7-8	612	124	-	-	-	-	4, 8	9-5-1-6-0-3-7-8-4-2	9-4-1-0-5-6-3-2-7-8	611	323
1-9-3-2-5-0-4-6-7-8	9-4-1-0-2-3-5-6-7-8	313	124	-	-	-	-	1, 7	1-9-4-0-2-3-5-6-7-8	9-1-3-2-5-0-4-6-7-8	114	323
9-4-1-0-2-3-5-6-7-8	4-9-0-3-1-5-6-7-2-8	124	312	-	-	-	-	0, 5	4-9-0-1-2-3-5-6-7-8	9-4-1-0-3-5-6-7-2-8	114	312
9-4-1-0-2-3-5-6-7-8	9-1-4-0-5-3-6-2-7-8	124	513	-	-	-	-	2, 5	9-4-1-0-2-3-5-6-7-8	9-1-4-0-5-3-6-2-7-8	124	513
0-4-1-9-5-2-6-7-3-8	3-4-1-9-5-6-2-0-7-8	512	323	-	-	-	-	2, 9	0-4-3-1-9-5-6-2-7-8	3-4-0-1-9-5-2-6-7-8	313	313
9-5-3-4-1-2-0-6-7-8	9-4-1-0-2-3-5-6-7-8	324	124	-	-	-	-	2, 8	9-5-4-1-0-2-3-6-7-8	9-4-5-3-1-2-0-6-7-8	323	324
9-0-1-3-5-6-4-2-7-8	1-4-3-9-2-5-0-6-7-8	323	313	-	-	-	-	1, 4	9-1-3-0-5-6-4-2-7-8	1-9-3-4-2-5-0-6-7-8	322	313
5-1-6-3-0-2-4-9-7-8	4-9-0-3-1-5-6-7-2-8	412	312	-	-	-	-	0, 9	4-9-0-3-1-5-6-7-2-8	5-1-6-3-0-2-4-9-7-8	312	412
4-3-1-2-5-0-9-6-7-8	4-3-7-6-9-0-5-2-1-8	313	2500	-	-	-	-	1, 5	4-3-5-2-1-0-9-6-7-8	4-3-9-6-7-0-5-2-1-8	1300	1500
4-9-0-3-1-5-6-7-2-8	4-3-1-2-5-0-9-6-7-8	312	313	-	-	-	-	1, 5	4-3-1-0-9-5-6-7-2-8	4-3-1-5-2-0-9-6-7-8	312	323
3-5-0-4-1-9-2-6-7-8	9-1-4-0-5-3-6-2-7-8	323	513	-	-	-	-	0, 5	1-4-0-5-3-9-2-6-7-8	5-0-4-1-9-3-6-2-7-8	313	413

Table 5.14 Crossover and mutation for figure 5.2

The iterative process is going on i.e. the new population is generated, fitness is evaluated again crossover is performed and so on. For this part there are three optimum solutions (1-2-4-9-3-5-6-7-8, 1-2-3-0-9-4-5-6-7-8 and 1-2-4-3-0-5-6-7-8) with a cost of 103 units obtained in 30 generations with population size of 20 strings.

After sequencing the operations system ask the user about tool material for turning, let here tool is carbide. As code entered by user system ask whether it brazed or disposable. Let here tool is brazed.

CODE	**TOOL MATERIAL**
1	Carbide
0	Brazed

The range of depth of cut for this choice is as follows: -

DEPTH OF CUT	
Rough Turning	0.6 TO 4.0mm
Finish Turning	0 TO 0.6 mm

Let depth of cut entered by user is 1.5 mm. Then feed range for this choice is: -

FEED RANGE 0.04 TO 0.4mm/rev

Let value for feed entered by user is 0.3 mm/rev. The speed for this depth of cut and feed is 185 m/min.

Now select tool for boring operation, let the tool for boring is carbide.

CODE	**TOOL MATERIAL**
0	HSS

The range of depth of cut for this choice is as follows: -

DEPTH OF CUT	
Rough Boring	0.25 TO 2.5mm
Finish Boring	UP TO 0.25 mm

Let depth of cut entered by user is 1.5 mm. Then feed range for this choice is: -

FEED RANGE 0.25mm/rev TO 0.20 mm/rev

Value for feed enter by user is 0.22 mm/rev. The speed for this depth of cut and feed is 44 m/min.

Values for depth of cut, feed and speed, for particular work material and tool combination are stored in the program & are retrieved by if-then rules and are taken from design data book [7] directly.

CHAPTER 6

CONCLUSION & SCOPE OF FUTURE WORK

In a computer aided process planning system, an efficient search is required to explore the large solution space of valid operation sequence under various interacting constraints. The present work has shown that a genetic algorithm is a viable means for searching the solution space of operation sequence providing a computational time on the order of few seconds. The advantage of this method of operation sequencing is the ability to generate an optimal sequence which is difficult in real manufacturing environment. The sequence generated is near optimal when it is successful in minimizing the cost i.e. minimizing the number of setups and minimizing the number of manufacturing tool changes. The development of powerful and flexible computer aided process planning system will be useful to the manufacturing engineering specialists working in the concurrent engineering teams. In metal cutting processes, cutting conditions have an influence on reducing the production cost, time & deciding the quality of a final product. One of the important aim of this work is to develop a prototype to demonstrate the feasibility of machining planning & accordingly, select cutting data. Generally, an optimum set of parameters refers to the condition which will offer the most economical tool life. Here an attempt is made to replace the manual handbook handling with computerized machinability data base system. However, optimum conditions can differ from user to user, based on his need, here the choice is left to the planner to choose the condition, he refers.

Integrating the operation sequencing by genetic algorithm & machining parameters by expert system will obviate the need to do the real-time experiments before the selection of the final sequence and machining parameters. The present work is only subjected to the components with rotational features and provides the parameters for turning, facing, boring operations, only for three types of work materials. The rule base can be easily expanded to handle other tool and work material combinations. The comparison shows that the results are better on the basis of cost and time then the previous work reported by various authors.

6.1 Scope of Future Work

The present work provides efficient results for sequencing and machining parameters for rotational components only and can be further enhanced by following ways: -

- The algorithm can be extended for prismatic parts.
- While deciding the parameters tool life can be considered as variable.
- Machining parameter should be optimized by some other AI technique rather than using expert system.
- Number of feature to be sequenced should be increased so that complex part can be sequenced.
- The algorithm can be integrated with an efficient CAD oriented software so that feature can be picked/ identified directly from the CAD model
- Cost of each feature should be calculated dynamically in real time system rather than deciding the fixed cost for each operation.
- The work can also extended for interfacing with the clamping sequence problem of rotational components.

REFERENCES

[1] David E. Goldberg, “Genetic algorithms- in Search, Optimization & Machine Learning”, Pearson Education, Inc., New Delhi 2005.

[2] Elaine Rich and Kevin Knight, “Artificial Intelligence” Tata McGraw-Hill, New Delhi 2005.

[3] Mikell P. Groover, “Automation, Production System, and Computer-Integrated Manufacturing” Pearson Education, Inc., New Delhi 2004.

[4] Mikell P. Groover and Emory W. Zimmer, Jr., “CAD/CAM: Computer-Aided Design and Manufacturing” Pearson Education, Inc., New Delhi 2003.

[5] M Sasikumar, S Ramani, S Muthu Raman, KSR Anjaneyulu and R Chandraseker, “Rule Based Expert Systems- A Practical Introduction ”, Narosa Publishing House, New Delhi, 1993.

[6] Robert I. Levine, Diane E. Drang and Barry Edelson, “A Comprehensive Guide to AI and Expert System”, McGraw-Hill International Editions, 1986.

[7] Faculty of Mechanical Engineering, “Design Data”, PSG College of Technology, Coimbatore.

[8] Deb Sankha, Ghosh Kalyan, and Deb S. R., “Machining Process Planning for Rotational Components by Using Neural Network Approach”, Dept. of Mathematics and Industrial Engineering, Ecole Polytchnique, P.O. Box 6079, Station Center-ville, Montreal, Quebec, Canada.

[9] Edward Gawlik, “The Analysis of Methods for Computer Aided Process Planning”, Carcow University of Technology, Carcow, Poland.

[10] Z.G. Wang, M. Rahman, Y.S. Wong, J. Sun, “Optimization of multi-pass milling using parallel genetic algorithm and parallel genetic simulated annealing”, International Journal of Machine Tools & Manufacture xx (2005) 1–9 (Available online at www.elsevier.com/locate/ijmactool)

[11] A. Vidal, M. Alberti, J. Ciurana, M. Casadesu, “A decision support system for optimising the selection of parameters when planning milling operations”, International Journal of Machine Tools & Manufacture 45 (2005) 201–210 (Available online at www.elsevier.com/locate/ijmactool)

[12] Zhang, W.B., Zhen, L.H., and Guang, Z.Y., “Optimization of Process Route by Genetic algorithm”, Robotics and Computer-Integrated Manufacturing 25 May 2004 (Available online at www.elsevier.com/locate/rcim)

[13] Franci Cus, Joze Balic, “Optimization of Cutting Process by GA approach”, Robotics and Computer-Integrated Manufacturing 19, (2003) (Available online at www. elsevier.com/locate/rcim)

[14] K. Vijayakumar, G. Prabhaharan, P. Asokan, R. Saravanan, “Optimization of multi-pass turning operations using ant colony system”, International Journal of Machine Tools & Manufacture 43 (2003) 1633–1639 (Available online at www. sciencedirect.com)

[15] Rana A.S, Bartarya G., Seminar Report Titled, “Evolution and Future Perspective of CAPP”, T.I.E.T., 2002.

[16] Yahia, B.N., Fnaiech, F. “ Manufacturing Process Planning Application using Artificial Neural Networks”, Ecole Superieure des Sciences et Technique de Tunis: 5 Av Taha Husein 1008 Tunis, 2002

[17] Dong-Ho Lee, D. Kiritsis, P. Xirouchakis,” Branch and fathoming algorithms for operation sequencing in process planning”, int. j. prod. res., 2001, vol. 39, no. 8, 1649 to1669. (Available online at Taylor & Francis Group- Article.htm)

[18] Ahmad Nafis, Anwarul Haque A.F.M., Hasin A.A., “Current Trend in Computer Aided Process Planning”, Proceeding of the 7th Annual Paper Meet and 2nd International Conference, The Institution of Engineers, Bangladesh. Mechanical Engineering Division. 25-27 October 2001, Paper No. 19 Pages 81-92.

[19] Ahmad Nafis and Anwarul Haque A.F.M., “Optimization Of Process Planning Parameters For Rotational Components By Genetic Algorithms”, 4th International Conference on Mechanical Engineering, December 26-28, 2001, Dhaka, Bangladesh/pp.VII 227-233.

[20] T. Dereli, I. H. Filiz, A. Baykasoglu, “Optimizing cutting parameters in process planning of prismatic parts by using genetic algorithms”, int. j. prod. res., 2001, vol. 39, no. 15, 3303, 3328. (Available online at Taylor & Francis Group- Article.htm)

[21] Dini, G., Falili, F., Lazzerini, B., Marcellon, F., “Generation of Optimized assembly sequence Using Genetic Algorithms”, Deptt. Of Mechanical, Nuclear and Production engineering, University of Pisa, Pisa ,Italy. Received on January 10, 1999. (Available online at Taylor & Francis Group- Article.htm)

[22] Bhaskara Reddy S. V., Shunmugam M. S., and Narendran T. T., "Operation Sequencing in CAPP Using Genetic Algorithms", International Journal of Production Research, 1999, Vol. 37, No. 5, 1063-1074. (Available online at Taylor & Francis Group- Article.htm)

[23] Van Zeir H.R., Kruith J.R. and Detand J., "A Conceptual Framework for Interactive and Blackboard Based CAPP", International Journal of Production Research, 1998, Vol. 36, No. 6, 1453-1473. (Available online at Taylor & Francis Group- Article.htm)

[24] M. Tolouei-Rad, I. M. Bidhendi, "On the optimization of machining parameters for Milling operations", Int. J. Mach. Tools Manufact. Vol. 37, No. I, pp. 1-16, 1997 (Available online at www.elsevier.com)

[25] Kim, J.U.,Kim, Y.D.,"Simulated Annealing And Genetic Algorithms For Scheduling Products With Multi-Level Product Structure", Computers Ops. Res. Vol. 23, No. 9,pp 857-868,1996.

[26] Usher, J. M. and Bowden, R.O., "The Application of Genetic Algorithms to Operation Sequencing for Use in Computer-Aided Process Planning", Computers Ind. Engg. Vol. No. 4, pp. 999-1013, 1996. (Available online at www.elsevier.com)

[27] Usher, J.M. And Fernandes, K.J., "A Two-Phased Approach to Dynamic Process Planning", Computers Ind. Eng VoL 31, No. 1/2, pp. 173 - 176, 1996. (Available online at www.elsevier.com)

[28] S.A. Irani, H.Y. Koo, S. Raman, "Feature Based Operation Sequence Generation in CAPP", International Journal of Production Research, 1995, Vol. 33, No. 1, 17-39.

[29] Cheng-Jung (TIM) Lin, Hsu-Pin (BEN) Wang, "Optimal Operation and Sequencing: Minimization of Tool Changeovers", International Journal of Production Research, 1993, Vol. 31, No. 2, 311-324. (Available online at Taylor & Francis Group- Article.htm)

[30] Koulamas, C. "Operation Sequencing and Machining Economics", International Journal of Production Research, 1993, Vol. 31, No. 4, 957-975. (Available online at Taylor & Francis Group- Article.htm)

[31] Singh Rajiv, Raman Shivkumar, "An Expert System for Machining Planning", International Journal of Production Research, 1992, Vol. 30, No. 7, 1501-1516. (Available online at Taylor & Francis Group- Article.htm)