UNIVERSITI MALAYSIA PAHANG

BORANG PENGESAHAN STATUS TESIS ◆

JUDUL: OPTIMIZATION OF MILLING PARAMETERS USING ANT COLONY OPTIMIZATION

SESI PENGAJIAN: 2007/2008

Saya, **MOHD SAUPI MOHD SAUKI (860803-46-5793)**
(HURUF BESAR)

mengaku membenarkan tesis (Sarjana Muda / ~~Sarjana / Doktor Falsafah~~)* ini disimpan di perpustakaan dengan syarat-syarat kegunaan seperti berikut:

1. Tesis ini adalah hakmilik Universiti Malaysia Pahang (UMP).
2. Perpustakaan dibenarkan membuat salinan untuk tujuan pengajian sahaja.
3. Perpustakaan dibenarkan membuat salinan tesis ini sebagai bahan pertukaran antara institusi pengajian tinggi.
4. **Sila tandakan (√)

[ ]	**SULIT**	(Mengandungi maklumat yang berdarjah keselamatan atau kepentingan Malaysia seperti yang termaktub di dalam AKTA RAHSIA RASMI 1972)
[ ]	**TERHAD**	(Mengandungi maklumat TERHAD yang telah ditentukan oleh organisasi / badan di mana penyelidikan dijalankan)
[√]	**TIDAK TERHAD**	

Disahkan oleh:

(TANDATANGAN PENULIS)

(TANDATANGAN PENYELIA)

Alamat Tetap:

Lot 335, Kg Delong
23000 Dungun
Terengganu

MOHD FADZIL FAISAE AB RASHID
(Nama Penyelia)

Tarikh: **14 NOVEMBER 2008**

Tarikh: **14 NOVEMBER 2008**

CATATAN: * Potong yang tidak berkenaan.

** Jika tesis ini SULIT atau TERHAD, sila lampirkan surat daripada pihak berkuasa/organisasi berkenaan dengan menyatakan sekali tempoh tesis ini perlu dikelaskan sebagai SULIT atau TERHAD.

◆ Tesis dimaksudkan sebagai tesis bagi Ijazah Doktor Falsafah dan Sarjana secara Penyelidikan, atau disertasi bagi pengajian secara kerja kursus dan penyelidikan, atau Laporan Projek Sarjana Muda (PSM).

OPTIMIZATION OF MILLING PARAMETERS USING ANT COLONY OPTIMIZATION

MOHD SAUPI BIN MOHD SAUKI

A report submitted in partial fulfilment of the requirements
for the award of the degree of
Bachelor of Mechanical Engineering with Manufacturing Engineering

Faculty of Mechanical Engineering
UNIVERSITI MALAYSIA PAHANG

NOVEMBER 2008

SUPERVISOR'S DECLARATION

We hereby declare that we have checked this project and in our opinion this project is satisfactory in terms of scope and quality for the award of the degree of Bachelor of Mechanical Engineering with Manufacturing Engineering.

Signature:
Name of Supervisor: MR MOHD FADZIL FAISAE BIN AB RASHID
Position: LECTURER
Date: 14 NOVEMBER 2008

Signature:
Name of Panel: MADAM SITI HARYANI BINTI TOMADI
Position: LECTURER
Date: 14 NOVEMBER 2008

STUDENT'S DECLARATION

I hereby declare that the work in this thesis is my own except for quotations and summaries which have been duly acknowledged. The thesis has not been accepted for any degree and is not concurently submitted for award of other degree.

Signature:
Name: MOHD SAUPI BIN MOHD SAUKI
ID Number: ME05057
Date: 14 NOVEMBER 2008

Dedicated to my beloved

Mother and Father

For their endless support in term of motivation,

supportive and caring as well throughout the whole project

ACKNOWLEDGEMENTS

First of all I am grateful to ALLAH S.W.T for blessing me in finishing my final year project (PSM) with success in achieving my objectives to complete this project.

Secondly I want to thank my family for giving morale support and encouragement in completing my project and also throughout my study in UMP as they are my inspiration to success. I also would like to thank my supervisor Mr. Mohd Fadzil Faisae Ab. Rashid for guiding and supervising my final year project throughout these two semesters. He has been very helpful to me in finishing my project and I appreciate every advice that he gave me in correcting my mistake. I beg for the forgiveness to my supervisor for any mistakes and things that I done wrong while doing my project.

Lastly I want to thank all my friends that have given me advice and encouragement in completing my project. Thank you very much to all and May ALLAH S.W.T bless you.

ABSTRACT

In process planning of conventional milling, selecting reasonable milling parameters is necessary to satisfy requirements involving machining economics, quality and safety. This study is to develop optimization procedures based on the Ant Colony Optimization (ACO). This method was demonstrated for the optimization of machining parameters for milling operation. The machining parameters in milling operations consist of cutting speed, feed rate and depth of cut. These machining parameters significantly impact on the cost, productivity and quality of machining parts. The developed strategy based on the maximize production rate criterion. This study describes development and utilization of an optimization system, which determines optimum machining parameters for milling operations. The ACO simulation is develop to achieve the objective to optimize milling parameters to maximize the production rate in milling operation. The Matlab software will be use to develop the ACO simulation. All the references are taken from related articles, journals and books. An example to apply the Ant Colony Algorithm to the problem has been presented at the end of the paper to give clear picture from the application of the system and its efficiency. The result obtained from this simulation will compare with another method like Genetic Algorithm (GA) and Linear Programming Technique (LPT) to validation. The simulation based on ACO algorithm are successful develop and the optimization of parameters values is to maximize the production rate is obtain from the simulation.

ABSTRAK

Dalam proses perancangan untuk mengunakan mesin kisar konvensional, pemilihan parameter mesin kisar yang sesuai akan memenuhi segala keperluan dalam penjimatan ekonomi, kualiti dan keselamatan. Kajian ini adalah berdasarkan proses untuk mengoptimumkan parameter yang digunakan oleh mesin kisar mengunakan kaedah *Ant Colony Optimization (ACO).* Proses ini akan dijalankan untuk mengoptimumkan parameter untuk operasi mesin kisar. Parameter yang terlibat dalam proses mesin kisar ialah seperti kelajuan mata pemotong, kadar pemotongan dan kedalaman pemotongan. Semua parameter ini akan memberi kesan terhadap kos pemesinan, kadar penghasilan produk dan kualiti produk. Strategi ini digunakan untuk memaksimumkan kadar pengeluaran produk yang dihasilkan mengunakan mesin kisar. Kajian ini akan menerangkan secara terperinci tentang penghasilan dan pengunaan kaedah *ACO* untuk mengoptimumkan parameter ketika proses mengunakan mesin kisar. Simulasi untuk kaedah *ACO* akan dihasilkan untuk mencapai objektif kajian ini iaitu untuk mengoptimumkan parameter bagi memaksimumkan kadar pengeluaran produk. Perisian Matlab akan digunakan untuk menghasilkan simulasi untuk kaedah *ACO*. Semua rujukan adalah diambil dari artikel, jurnal dan buku yang berkaitan. Satu contoh masalah akan digunakan untuk menerangkan cara pengunaan kaedah *ACO* untuk memberi gambaran yang lebih jelas tentang kaedah ini. Keputusan yang diperolehi daripada simulasi ini akan dibandingkan dengan keputusan daripada kaedah lain seperti *Genetic Algorithm (GA)* dan *Linear Programming Technique (LPT)* yang diambil daripada sumber rujukan untuk pengesahan. Simulasi menggunakan kaedah *ACO* berjaya dibangunkan dan menghasilkan nilai parameter yang paling optimum untuk memaksimumkan kadar pengeluaran.

TABLE OF CONTENTS

LIST OF TABLE

LIST OF FIGURE

LIST OF APPENDICES

CHAPTER 1

INTRODUCTION

1.1 BACKGROUND

The advance of modern technology and a new generation of manufacturing equipment, particularly computer numerical control (CNC) machine, have brought enormous changes to the manufacturing sector. Generally, the handbook or human experience is used to select convenient machine parameters in manufacturing industry. In process planning of conventional milling, selecting reasonable milling parameters is necessary to satisfy requirements involving machining economics, quality and safety. M. Tolouei-Rad (1996).

The machining parameters in milling operations consists of cutting speed, depth of cut, feed rate and number of passes. These machining parameters significantly impact on the cost, productivity and quality of machining parts. The effective optimizations of these parameters affect dramatically the cost and production time of machined components as well as the quality of final products. M. Tolouei-Rad (1996).

1.2 PROBLEM STATEMENT

Establishment of efficient machining parameters has been a problem that has confronted manufacturing industries for nearly a century, and is still the subject of many studies.

Optimum machining parameters are of great concern in manufacturing environments, where economy of machining operation plays a key role in competitiveness in the market.

Although NC machines can reduce lead times considerably, the machining time is almost the same as in conventional machining when machining parameters are selected from machining databases or handbooks.

1.3 OBJECTIVES

(i) Develop Ant Colony Optimization (ACO) algorithm to optimize milling parameters.
(ii) Determine optimum milling parameters to maximize production rate.

1.4 SCOPE OF THE PROJECT

(i) The research will use ACO algorithm to optimize milling parameters to maximize production rate.
(ii) All the references are taken from related articles and journals.
(iii) All of this constant are taken from the references.
(iv) This study not involved any experiment.

1.5 IMPORTANT OF STUDY

This project will increase the knowledge about the way to optimize the machining parameter in order that to obtain minimum production time, maximum profit rate and minimum production cost. Optimization the machining parameter will avoid from doing any waste in production especially for material and time.

As a future engineer, must able to know about the way to optimize the parameter of milling or other machining processes in order that to increase the profit of the company. This study also giving a new knowledge about the method that using for optimize the machining operation like Ant Colony Optimization (ACO), Genetic Algorithm, Taguchi method, Tribes method etc.

1.6 SUMMARY

This chapter has been discussed generally about project background, problem statement, question which has been formulate from the problems, objective of the project and scope of the project in order to achieve the objective as mentioned.

CHAPTER 2

LITERATURE REVIEW

2.1 INTRODUCTION

Determination of the optimal cutting parameters (cutting conditions) such as the number of passes, depth of cut, speed, and feed is considered as a crucial stage of milling machining processes and especially in process planning. This is mainly due to the complex nature of optimization of machining operations that require the following:

(i) Knowledge of machining like drilling, turning or milling.
(ii) Empirical equations relating the tool life, forces, power, surface finish, material removal rate, and arbor deflection to develop realistic constraints.
(iii) Specification of machine tool capabilities like maximum power or maximum feed available from a machine tool.
(iv) Development of an effective optimization criterion like maximum production rate, minimum production cost, maximum profit or a combination of these.
(v) Knowledge of mathematical and numerical optimization techniques, such as the Simplex method, Search method, Geometric programming and dynamic programming.
(vi) Knowledge of stochastic optimization techniques, such as the Genetic Algorithms, Simulated Annealing, Scatter Search, Particle Swarm Optimization and Tribes.

2.1.1 Milling Machine

Milling is the process of machining flat, curved, or irregular surfaces by feeding the work piece against a rotating cutter containing a number of cutting edges. The usual mill consists basically of a motor driven spindle, which mounts and revolves the milling cutter, and a reciprocating adjustable worktable, which mounts and feeds the work piece. (Wikipedia, milling machine)

Milling machines are basically classified as vertical or horizontal. These machines are also classified as knee-type, ram-type, manufacturing or bed type, and planer - type. Most milling machines have self-contained electric drive motors, coolant systems, variable spindle speeds, and power-operated table feeds. (Wikipedia, milling machine)

A milling machine is a machine tool that cuts metal with a multiple-tooth cutting tool called a milling cutter. The work piece is fastened to the milling machine table and is fed against the revolving milling cutter. The milling cutters can have cutting teeth on the periphery or sides or both. (Wikipedia, milling machine)

Milling machines can be classified under three main headings:

(i) General Purpose machines - these are mainly the column and knee type (horizontal & vertical machines).
(ii) High Production types with fixed beds- (horizontal types).
(iii) Special Purpose machines such as duplicating, profiling, rise and fall, rotary table, planetary and double end types.

Milling attachments can also be fitted to other machine tools including lathes planning machines and drill bench presses can be used with milling cutters. Milling machine is one of the most versatile conventional machine tools with a wide range of metal cutting capability. Many complicated operations such as indexing, gang milling, and straddle milling can be carried out on a milling machine. (Wikipedia, milling machine)

2.2 PREVIOUS METHOD

Many methods are used by related journal to optimizing milling parameters. M. Tolouei-Rad (1996), use Method of Feasible Direction for this purpose. This work does the optimization on unit cost, unit time and total profit rate resulting from all the machining operations required to produce the product. The parameter considered is depth of cut, feed rate and cutting speed. These values have been determined based on optimum machining parameters in comparison with those resulting from handbook recommendations. The result achieved when optimum machining parameters have been employed.

V. Tandon, H. El-Mounayri and H. Kishawy (2001) used Particle Swarm Optimization (PSO) for this optimization. This work has presented a new approach to optimizing the cutting conditions in end milling (feed and speed) subject to a near to comprehensive set of constraints. The original set of seventeen constraints was reduced to an equivalent set (of only three equations). Next, a production cost objective function was used to define the parameter to optimize (in this case, minimize). An algorithm for PSO was then developed and used to robustly and efficiently find the optimum cutting conditions. Both feed and speed were considered during optimization. The new technique has several advantages and benefits and is suitable for use with ANN based models.

G. Prabhaharan, N. Baskar, P. Asokan and R. Saravanan (2005), outlines the development of an optimization strategy to determine the optimum cutting parameters for multi-tool milling operations like face milling, corner milling, pocket milling and slot milling. The developed strategy based on the maximum profit rate criterion and incorporates five technological constraints. In this paper, optimization procedures based on the genetic algorithm, hill climbing algorithm and memetic algorithm were demonstrated for the optimization of machining parameters for milling operation. An objective function based on maximum profit in milling operation has been developed. Results obtained are used in NC machine. The results are compared and analyzed with method of feasible directions and handbook recommendations.

H.H Hasssan, J.A Ghani and I.A Choudhury (2003) used Taguchi optimization methodology, which is applied to optimize cutting parameters in end milling when machining hardened steel AISI H13 with TiN coated P10 carbide insert tool under semi-finishing and finishing conditions of high speed cutting. The milling parameters evaluated is cutting speed, feed rate and depth of cut. Using Taguchi method for design of experiment (DOE), other significant effects such as the interaction among milling parameters are also investigated. The paper shows that the Taguchi method is suitable to solve the stated problem with minimum number of trials as compared with a full factorial design.

J. Balic, M. Milfelner and F. Cus (2005), outline an approach for the systematic design of condition monitoring system for machine tool and machining operations. The research is based on utilising the genetic optimization method for the on-line optimization of the cutting parameters and to design a program for the signal processing and for the detection of fault conditions for milling processes. Cutting parameters and the measured cutting forces are selected in this work as an application of the proposed approach.

Godfrey C. Onwubolu (2005), proposes a new optimization technique based on Tribes for determination of the cutting parameters in multi-pass milling operations such as plain milling and face milling by simultaneously considering multi-pass rough machining and finish machining. The optimum milling parameters are determined by minimizing the maximum production rate criterion subject to several practical technological constraints. The cutting model formulated is a nonlinear, constrained programming problem. Experimental results show that the proposed Tribes-based approach is both effective and efficient.

Ali Riza Yildiz (2007), presents a new hybrid optimization approach based on immune algorithm and hill climbing local search algorithm. The purpose of the present research is to develop a new optimization approach for solving manufacturing optimization problems and to optimize machining parameters for milling operations. This research is the first application of immune algorithm to the optimization of machining parameters. In order to evaluate the proposed optimization

approach, single objective test problem, multi objective I-beam and machine-tool optimization problems taken from the literature are solved. The results of the hybrid approach for the case study are compared with those of genetic algorithm, the feasible direction method and handbook recommendation. Finally, the hybrid approach is applied to a case study for milling operations to show its effectiveness in machining operations.

J. Sun, Z.G. Wang, M. Rahman and Y.S. Wong (2005), presents 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 has been applied to many difficult combinatorial optimization problems with certain strengths and 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. For comparison, conventional parallel GA (PGA) is also chosen as another optimization method. An application example that has been solved previously using the geometric programming (GP) and dynamic programming (DP) method is presented. From the results, PGSA is shown to be more suitable and efficient for optimizing the cutting parameters for milling operation than GPCDP and PGA.

G.H. Qin, M. Wan, W.H. Zhang and G. Tan (2007), present the procedure integrates the cutting force module consisting of calculating the instantaneous uncut chip thickness (IUCT), calibrating the instantaneous cutting force coefficients (ICFC) and the cutting process module consisting of calculating the cutting configuration and static form errors. It used to check the process reasonability and to optimize the process parameters for high precision milling. Comparisons of the cutting forces and form errors obtained numerically and experimentally confirm the validity of the proposed simulation procedure.

M. Cengiz Kayacan, Oguz Çolak and Cahit Kurbanoglu (2005), used genetic expression programming method is used for predicting surface roughness of milling surface with related to cutting parameters. CNC milling has become one of the most competent, productive and flexible manufacturing methods, for complicated or sculptured surfaces. In order to design, optimize, built up to sophisticated, multi-axis milling centers, their expected manufacturing output is at least beneficial. Therefore data, such as the surface roughness, cutting parameters and dynamic cutting behavior are very helpful, especially when they are computationally produced, by artificial intelligent techniques. Cutting speed, feed and depth of cut of end milling operations are collected for predicting surface roughness.

2.2.1 Summarize from Literature Review

Use of many methods has been reported in the literature to solve optimization problem for machining parameters include Feasible Directions (M. Tolouei-Rad, 1996), Particle Swarm Optimization (V. Tandon, 2001), Memetic Algorithm (N. Baskar, 2005), Taguchi Method (J.A Ghani, 2003), Genetic Algorithm (M. Milfelner, 2005), Tribes Algorithm (Godfrey C. Onwubolu, 2005), Immune Algorithm (Ali Riza Yildiz, 2007), Simulated Annealing (Z.G Wang, 2005), Simulation Procedure (M.Wan, 2007), Genetic Expression Programming (Oguz Colak, 2005).

There are many parameters that can considered in optimize milling machining processes. The few parameters were reported include depth of cut, feed rate, chip depth of cut, work piece speed, cutting force, cutting speed, number of passes, tool diameter and tool length. The most three parameter that uses by literature is depth of cut, feed rate and cutting speed.

Table 2.1: Previous Method

Journal	Author	Year	Method	Parameter
1	• M. Tolouei-Rad • I.M. Bidhendi	1996	Feasible Directions	• Depth of Cut • Feed Rate • Cutting Speed
2	• V. Tandon • H. El-Mounayri • H. Kishawy	2001	Particle Swarm Optimization	• Depth of Cut • Feed Rate • Cutting Speed • Number of Passes
3	• N. Baskar • P. Asokan • R. Saravanan • G. Prabhaharan	2005	Memetic Algorithm	• Depth of Cut • Feed Rate • Cutting Speed • Cutting Force
4	• J.A Ghani • I.A Choudhury • H.H Hasssan	2003	Taguchi Method	• Depth of Cut • Feed Rate • Cutting Speed • Cutting Force
5	• M. Milfelner • F. Cus • J. Balic	2005	Genetic Algorithm	• Cutting Force
6	• Godfrey C. Onwubolu	2005	Tribes Algorithm	• Depth of Cut • Feed Rate • Cutting Speed • Chip depth of cut
7	• Ali Riza Yildiz	2007	Immune Algorithm	• Depth of Cut • Feed Rate • Cutting Speed
8	• Z.G. Wang • M. Rahman • Y.S. Wong • J. Sun	2005	Simulated Annealing	• Depth of Cut • Feed Rate • Cutting Speed • Number of Passes
9	• M. Wan • W.H. Zhang • G. Tan • G.H. Qin	2007	Simulation Procedure	• Depth of Cut • Feed Rate • Cutting Speed
10	• Oguz Çolak • Cahit Kurbanoglu • M. Cengiz Kayacan	2005	Genetic Expression Programming	• Depth of Cut • Feed Rate • Cutting Speed

2.3 METHOD

Use of many methods has been reported in the literature to solve optimization problems for machining parameters. All the method use the different procedure in optimize the machining parameter with the same objective including minimum production cost, minimum production time, maximum metal removal rate and maximum profit rate. These methods include:

(i) Feasible directions
(ii) Particle Swarm Optimization
(iii) Memetic Algorithm
(iv) Taguchi Method
(v) Genetic Algorithm
(vi) Tribes Algorithm
(vii) Immune Algorithm
(viii) Ant Colony Optimization
(ix) Simulated Annealing
(x) Simulation Procedure
(xi) Genetic Expression Programming

For this study, the milling parameters will optimize by Ant Colony Optimization (ACO) method. The research and analysis will be done to this method will be applied to this problem.

2.3.1 Ant Colony Optimization (ACO)

Ant Colony Optimization (ACO) is a multi-agent system (that iteratively searches for optimal solutions). Elements of optimal solutions are extracted according to the shortest path of ant tours. Ants deposit their searching reward, pheromone on their passed paths. These feedbacks may attract other ants to follow partially with probability called transition rules. State transition rules imply that shorter and more ant-experienced paths attract more ants to pass through. However, as with real ants, not all ants follow the most attractive paths, instead a few ants try to explore new paths. The process of taking the maximal probability path is called exploitation and the process of selecting the next path by probability is called exploration. Chien-Chang Chen (2007).

ACO can be applied to other optimization problems by modifying evolutionary procedures. However, the following elements need to be clearly defined. Chien-Chang Chen (2007).

(i) A problem graph that includes paths, cost of moving and the number of ants.
(ii) Initial states and moving rules of ants.
(iii) Termination conditions of ants.
(iv) Definitions of pheromone reward and natural evaporation.

The ACO mechanism has the following desirable properties:
Chien-Chang Chen (2007).

(i) ACO is versatile in that it can be applied to similar versions of the same problem, example of a similar version might be the travelling salesman problem.
(ii) ACO is robust and can be applied with only minimal changes to other combination optimization problems such as the quadratic assignment problem and the job shop scheduling problem.

2.4 PARAMETERS

Parameter sometimes loosely refers to an individual measured item. This usage isn't consistent, as sometimes the term channel refers to an individual measured item, with parameter referring to the setup information about that channel. Parameters are those combinations of the properties which suffice to determine the response of the system. Properties can have all sorts of dimensions, depending upon the system being considered; parameters are dimensionless, or have the dimension of time or its reciprocal.

There are few milling parameters that can be optimizing for achieve the objective function include:

(i) Depth of cut
(ii) Feed rate
(iii) Chip depth of cut
(iv) Work piece speed
(v) Cutting force
(vi) Cutting speed
(vii) Number of Passes
(viii) Tool diameter
(ix) Tool length

2.5 TOP PARAMETERS

There are many parameters that can be considered in optimize milling machining processes. Optimum machining parameters resulting from this work are intended for use by milling processes in order to improve machining efficiency, minimum production cost, and also minimum production time. The main three parameters that be used by related journals is depth of cut, feed rate and cutting speed.

2.6 SUMMARY

All the references in the literature review are taken from related journals, other websites and also from the book title, "*Manufacturing Optimization through Intelligent Techniques"* by R. Saravanan, 2006. This literatures describes about the milling machine and milling process, the previous methods that used like Feasible Direction, Particle Swarm Optimization, Memetic Algorithm, Taguchi Method, Genetic Algorithm, Tribes Algorithm, Immune Algorithm, Simulated Annealing, Simulation Procedure, Genetic Expression Programming and consider the cutting speed, feed rate and depth of cut as the parameters that give the greatest effect in optimize milling machine processes.

CHAPTER 3

METHODOLOGY

3.1 INTRODUCTION

There are many methods are reported in the literature like Genetic Algorithm, Tribes method, Particle Swarm Optimization and more. For this study, Ant Colony Optimization are being used as the method for optimize milling machining processes consider by cutting speed, feed rate and depth of cut as the chosen parameters. The entire step below will briefly detail in this chapter.

There are 5 basic steps in Ant Colony Optimization:

Step 1: Initialize
Step 2: Evaluation
Step 3: Reproduction
Step 4: Update Trail
Step 5: Termination

3.2 PROJECT FLOW CHART

For the diagram as shown in figure 3.0, the project starts with literature review and research about the title. This consist a review of the milling machine and milling process, problem and the objective project, the previous method uses and related ACO to the problem. These tasks have been done through research on the internet, books and others sources.

After gathering all the relevant information, the project undergoes with analysis the problem and the method were used. In this step, from the knowledge gather from the review is use to make a compared table to identified all the method used to optimize milling parameter and the top parameters were used.

Next, run the project scope to optimize milling parameters using Ant Colony Optimization (ACO). First, analysis what ACO and detail about the ACO methodology is by identify step of this algorithm and by flow chart.

After identified detail about ACO, identify how ACO will be applied to this problem and justify that ACO suitable for optimize the milling parameters. Finally, complete the report and submit on the submission date.

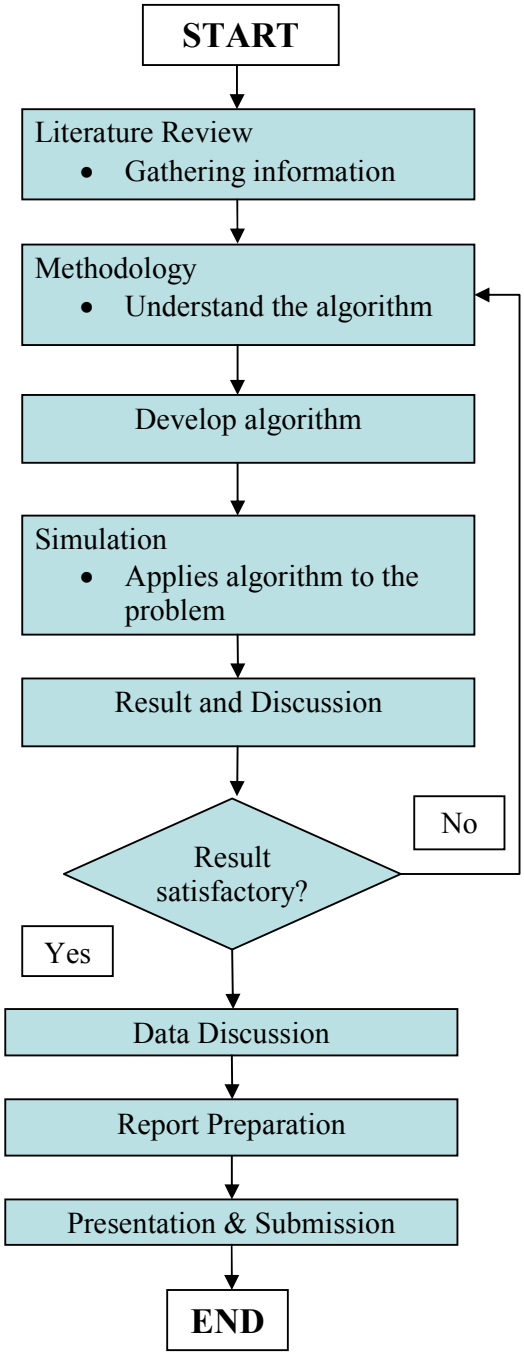


Figure 3.1: Project Flow Chart

3.3 ANT COLONY ALGORITHM

ACO is a multi-agent system that iteratively searches for optimal solutions. The ACO algorithm has been applied to combinatorial optimization problems such as the Traveling Salesman Problem (TSP), Quadratic Assignment Problem (QAP) and so on. The ACO algorithm can be applied to another problem by following the algorithm in flow chart on figure 3.2. The flow chart give the algorithm step how to apply the optimization process using Ant Colony Optimization.

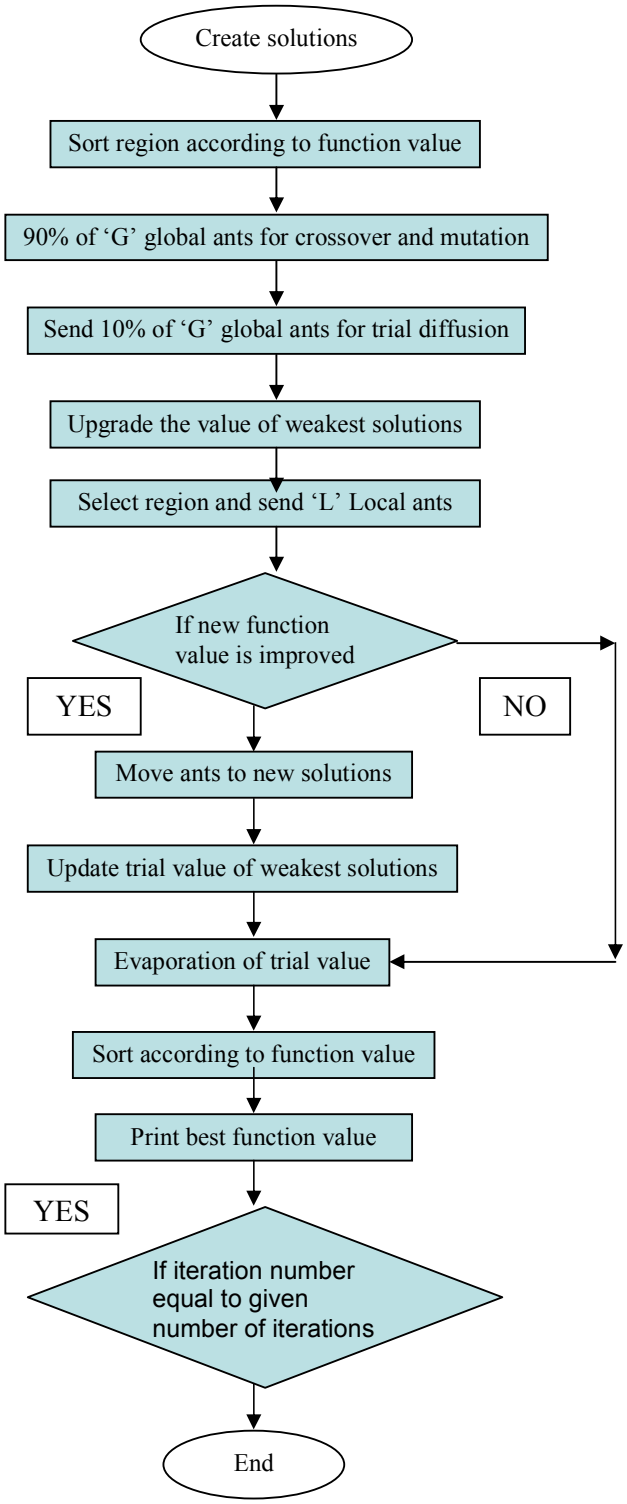


Figure 3.2: Flow Chart of Ant Colony Algorithm (R. Saravanan, 2006)

3.4 IMPLEMENTATION OF ANT COLONY OPTIMIZATION

Step 1: Initialize

For initialize step, the 20 solutions are randomly generated. At first, the solutions are randomly generated in binary number and then combine all the binary number for the three parameters to generate the solution. Generated the binary code for the range of parameter and then decode to decimal number. After get the binary code of the parameter, combined all the binary for each parameter to generate 20 solutions.

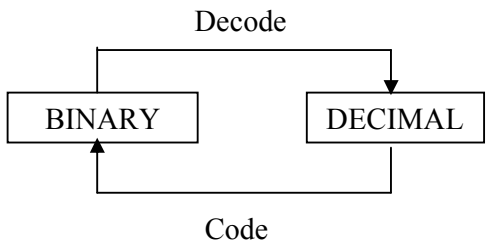


Table 3.1: The Value of Parameter in Binary and Decimal

Parameter	Range	Binary (1_2)	Decimal (1_{10})
Cutting Speed (CS) (4800-9060)mm/min	Min - 4800 ↓ Max - 9600	00000 ↓ 11111	0.00 ↓ 31.00
Feed Rate (F) (0.1 – 0.2)mm/tooth	Min - 0.100 ↓ Max - 0.200	00000 ↓ 11111	0.00 ↓ 31.00
Depth of Cut (DOC) (2 – 4)mm	Min - 2.0 ↓ Max - 4.0	00000 ↓ 11111	0.00 ↓ 31.00

Table 3.2: Arrangement the Binary Numbers

No.	Cutting Speed, V	Feed Rate, f	Depth of Cut, b
1	00000	00000	00000
2	00010	00010	00001
3	00110	00110	00011
4	01010	01010	01010
5	00100	00100	00100
6	01000	01000	01000
7	10000	10000	10000
8	11000	11000	11000
9	10100	10100	10100
10	10110	10110	10111
11	01110	01110	01111
12	01100	01100	01100
13	11010	11010	11010
14	10010	10010	10010
15	00100	01010	01001
16	11000	00110	00110
17	01010	10100	11001
18	00110	01100	00111
19	11100	11100	11100
20	11111	11111	11111

Table 3.3: Arrangement the Solution

Solution	Values
1	000000000000000
2	000100001000001
3	001100011000011
4	010100101001010
5	001000010000100
6	010000100001000
7	100001000010000
8	110001100011000
9	101001010010100
10	101101011010111
11	011100111001111
12	011000110001100
13	110101101011010
14	100101001010010
15	001000101001001
16	110000011000110
17	010101010011001
18	001100110000111
19	111001110011100
20	111111111111111

Step 2: Evaluation

The objective function will be finding in evaluation step. From the solution in initialize step, decode into decimal number. From the decimal number, the interpolation technique will be use to convert the solution into fitness values. The fitness values will be use for calculate the objective function. For example, the solution will use like shown. Firstly, the solution is separated to decode the binary number into decimal number.

Solution: 10010**01110**00101

Table 3.4: Decode Binary into Decimal Number.

Binary	Decimal
10010	17
01110	14
00101	5

From the decimal number, find the parameter value for each parameter. The interpolation technique will use to convert the parameters values from the decimal number. The objective function equation uses the parameter values to obtain the value of production rate.

Table 3.5: Convert Binary Number to Decimal Number

Cutting Speed, V	Feed Rate, f	Depth of Cut, a
0	0	0
2	2	1
6	6	3
10	10	10
4	4	4
8	8	8
16	16	16
24	24	24
20	20	20
22	22	23
14	14	15
12	12	12
26	26	26
18	18	18
4	10	9
24	6	6
10	20	25
6	12	7
28	28	28
31	31	31

Table 3.6: Convert Decimal Number to Fitness Values

Cutting Speed, V	Feed Rate, f	Depth of Cut, a
4800	0.1000	2.0000
5109	0.1065	2.0645
5729	0.1194	2.1935
6348	0.1322	2.6452
5419	0.1129	2.2581
6038	0.1258	2.5161
7277	0.1516	2.5806
8516	0.1774	3.5484
7896	0.1645	3.2903
8206	0.1709	3.4839
6967	0.1452	2.9677
6658	0.1387	2.7742
8825	0.1839	3.6774
7587	0.1581	3.1613
5419	0.1323	2.5806
8516	0.1194	2.3871
6348	0.1645	3.6129
5729	0.1387	2.4516
9135	0.1903	3.8064
9600	0.2000	4.0000

3.5 OBJECTIVE FUNCTION

Production rate is considered to be an objective function (R. Saravanan, 2006). The production rate can be expressed by the total processing time for one component (T_{ed}):

$$Q = 1/T_{ed} \quad \textbf{(Eq. 3.1)}$$

Where the processing time per component is determined as:

$$T_{ed} = T_m + T_{cm} + T_r \quad \textbf{(Eq. 3.2)}$$

Where T_m is the processing time, T_{cm} is the time for the tool changing and setting up and T_r is the time for manual operations. All the times are given in minutes.

For the face milling process, the production time is defined as:

$$T_m = \pi DL(W/a)/(1000ZVf) \quad \textbf{(Eq. 3.3)}$$

Where:

$$L = L_b + L_n + L_a \quad \textbf{(Eq. 3.4)}$$

3.5.1 Data of the Problem

All the value of process variable is taken from the literature (R. Saravanan, 2006), Diameter of the mill, D = 160 mm, Number of cutting edges in a tip, Z = 4, width = 30mm, Length of movement before cutting, L_b =10 mm, Length of movement after cutting, L_a = 90 mm, Time of manual operations, T_r = 0.12 min and Time for tool changing and setting, T_{cm} = 1 min. The values of parameter, cutting speed, minimum = 60 rev/min and maximum = 120 rev/min, feed rate, minimum = 0.1 mm/tooth and maximum = 0.2 mm/tooth, depth of cut minimum = 2mm and maximum = 4mm.

The cutting speed must convert from unit RPM (Revolution per Minutes) into unit (mm/min) before doing the calculation. The range of cutting speed is between 60 RPM and 120 RPM.

$V = r\omega$

Where, $r = D/2$

$= 160/2$

$= 80$ mm

$\omega = 60, 75, 90, 105, 120$ (rev/min)

Table 3.7: Range of Parameters

PARAMETER	RANGE
Cutting Speed, V	4800 – 9600 (mm/minute)
Feed Rate, f	0.1 – 0.2 (mm/tooth)
Depth of Cut, a	2.0 – 4.0 (mm)

The machining time is defined as:

$T_m = \pi DL(W/a)/(1000ZVf)$

Where:

$L = L_b + L_n + L_a$

$L_a = 90$ mm

$L_b = 10$mm

$L_n = 200$mm

$L = 10\text{mm} + 200\text{mm} + 90\text{mm}$

$= 300$mm

The value of **D** and **Z** were taken from the data given table and the value of **V**, **f** and **a** is taken from table. Substitute all the values into equation.

L = 300mm

D = 160mm

V = 4800mm/min

f = 0.1mm/tooth

a = 2mm

Apply the machining time equation:

$$T_m = \pi DL(W/a)/(1000ZVf)$$

$$= \frac{\pi\,(160mm)(300mm)(30mm)}{1000\,(4)(4800mm/min)(0.1mm/tooth)(2mm)}$$

$$= 1.1781 \text{ minute}$$

The values of T_{cm} and T_r are constant and the data are taken from the table.

T_{cm} = 1 minute

T_r = 0.12 minute

Substitute all the T_m, T_{cm} and T_r into equation:

$$T_{ed} = T_m + T_{cm} + T_r$$

$$= 1.1781 \text{ min} + 1 \text{ min} + 0.12 \text{ min}$$

$$= 2.2981 \text{ minute}$$

Substitute T_{ed} = 2.29 min into objective function equation:

$$Q = 1/T_{ed}$$

$$= 1/\ 2.2981\text{min}$$

$$= 0.4351 \text{ unit/minute}$$

After obtained the 20 values of the production rate, sort the values in descending order to set the value in the region. The first 12 solutions are taken as superior solutions and next 8 solutions are taken as inferior solutions. Then sort the solution again corresponding to the new arrange of the objective function.

Table 3.8: Objective Function Values

Solution	Production Rate (unit/mm)
1	0.4351
2	0.4702
3	0.5337
4	0.6137
5	0.5158
6	0.5842
7	0.6539
8	0.7513
9	0.7222
10	0.7399
11	0.6681
12	0.6404
13	0.7636
14	0.7051
15	0.5776
16	0.6305
17	0.7051
18	0.5880
19	0.7746
20	0.7891

Table 3.9: Arrange Objective Function Values in Region

Solution	Solution	Objective Function Values (unit/mm)
Superior	1	0.7891
	2	0.7746
	3	0.7636
	4	0.7513
	5	0.7399
	6	0.7222
	7	0.7051
	8	0.7043
	9	0.6681
	10	0.6539
	11	0.6404
	12	0.6305
Inferior	13	0.6137
	14	0.5880
	15	0.5842
	16	0.5776
	17	0.5337
	18	0.5158
	19	0.4702
	20	0.4351

Table 3.10: New Solution after Arrange in Region

Solution	Values
1	111111111111111
2	111001110011100
3	110101101011010
4	110001100011000
5	101101011010111
6	101001010010100
7	100101001010010
8	010101010011001
9	011100111001111
10	100001000010000
11	011000110001100
12	110000011000110
13	010100101001010
14	001100110000111
15	010000100001000
16	001000101001001
17	001100011000011
18	001000010000100
19	000100001000001
20	000000000000000

Step 3: Reproduction

Reproduction step is use to improve the all 20 solution. This step must undergo 4 steps. First step is Crossover or also known as Random Walk. This step will improve the solution in inferior step from solution 13 until solution 18. For second step is calling Mutation. This step will repair the improve solution from Crossover step. Trail Diffusion is the third step in reproduction step. This step function is to improve the remained solution in inferior solution, solution 19 and solution 20. And the last step is known as Local Search step. This step will improve the superior solution, solution 1 until 12.

Four sub-step in reproduction step:

(i) Crossover
(ii) Mutation
(iii) Trail diffusion
(iv) Local search

i) Crossover or Random Walk

Crossover or Random Walk is a replacement strategy, usually done to modify the solutions by replacing the inferior solutions. The following step explains this procedure. A random number is generated between 1 and 12. The corresponding solution in the superior region replaces the inferior solution. The solution for solution 10 is known as parent 1 and solution 13 is known as parent 2. For example, solution number 10 is randomly generated for replacement solution 13.

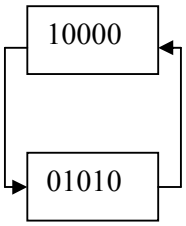
Solution 10: 100001000010000
Solution 13: 010100101001010

Parent 1 = 100001000010000
Parent 2 = 110101101011010

Generate another random number between 1 and15. This number is representing as the binary digit for a solution. The function of this number is to cut the binary solution. For example, number 10 randomly generated again. From now, the solution 10 and solution 13 is know as child 1 and child 2, respectively. The solution 10 and solution 13 is cutting corresponding to 10 digit and the remained digit of solution left is change to the others like shown below.

Parent 1= 1000010000 | **10000**

Parent 2= 0101001010 | **01010**



Child 1= 100001000011010

Child 2=010100101010000

Evaluate child to find the objective function value. After get the function value, compared the value of child 1 and child 2 and take the best. Take the best value and replace into inferior solution 13.

Child 1 = 0.6633 unit/minute

Child 2 = 0.6392 unit/minute

From the fitness number above, the solution of child 1 is obtained 0.6633 unit/minute better than the solutions for child 2 obtain 0.6392 unit/minute. The solution for child 1 is taking as the best values for replace in solution 13. Similar procedure is carried out for the other inferior solutions from 14-18. Thus the random walk procedure is done. Selected solutions in the superior region should be excluded so that they will not be selected again for replacement.

New solution 13 = 100001000011010

ii) Mutation

The mutation step used for repair the solution 13 until solution 18. First, set the level of mutation between 0 and 1. For example the level of mutation is set as 0.45. Then, generate random number between 0 and 1. For applying mutation, the random number, r must greater or equal to level of mutation, m and if the random number generate is less than level of mutation, ignore the mutation step.

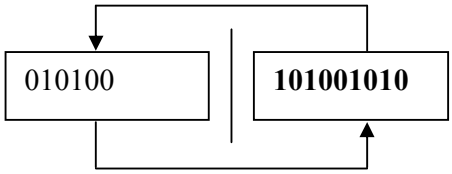
If $\boxed{r < m}$ Example: $r = 0.40 < m = 0.45$ (Ignore mutation)

If $\boxed{r \geq m}$ Example: $r = 0.60 < m = 0.45$ (Apply mutation)

First solution that needs to repair is solution 13. For the example, the 0.45 is set as the level of mutation. After generated the random number, 0.60 is obtained. The solution will through the mutation step. The random number is generated again represent as binary digit for the solution to cut the solution. For example, number 6 randomly generated. Change the position of binary digit based on number 6. Then the solution will replace as the new solution 13. Apply the mutation step to solutions from 14-18. The mutation procedure is done.

Solution 13: 010100101001010

Solution 14: 010100 | **101001010**



New solution 13 = **101001010**010100

iii) Trail Diffusion

The remaining two solutions 19 and 20 at the tail end are improved by the trail diffusion technique. Two parent variables are selected from the superior region, for which two dissimilar random number are generated between 1 and 12 and named as parent 1 and parent 2. For example, the number 8 and number 9 is generated. The number generated is representing as the solution in superior region. The solution 8 and solution 9 is chosen and know as parent 1 and parent 2. Decode Parent 1 and Parent 2 into decimal number.

Parent 1 = 010101010011001 (solution 8)
Parent 2 = 011100111001111 (solution 9)

A random number α is generated between 0 and 1 for every region. If α is less than equal to 0.5, apply the trail diffusion equation and if the random number generated is more than 0.5, child solution is equal parent 2 solution.

$\boxed{\alpha \leq 0.5}$ $$X_{child} = \alpha X_{parent} + (1-\alpha)\, X_{parent2}$$ **(Eq. 3.5)**

$\boxed{\alpha > 0.5}$ $$X_{child} = X_{parent2}$$ **(Eq. 3.6)**

For example, after decode Parent 1= 1000 and Parent 2 = 2000. The random generated number for α is 0.3. Trail diffusion equation is applied. From the equation below, the solution for child is obtained. After applying the trail diffusion equation, the decimal values for child is obtained is 1700. The decimal value for child is converting into binary number to make the solution. The solution obtain is 111010101010101. The child solution obtains then replace into the solution 19. The same procedure is done for the solution 20.

iv) Local Search

The local searches improve 12 solutions in the superior region only. The following steps explain the local search. Initially the pheromone value (ph) for every region is set to 1.0 and the age for every region is taken as 10.

Table 3.11: The Value of Pheromone and Age

Solution	Ph_i (i =1,2..12)	Age
1	1	10
2	1	10
3	1	10
4	1	10
5	1	10
6	1	10
7	1	10
8	1	10
9	1	10
10	1	10
11	1	10
12	1	10

The average pheromone value (Ave ph) is calculated using:

$$\text{Ave ph} = \frac{\sum ph}{\text{Number of superior solution}} \qquad \textbf{(Eq. 3.7)}$$

A random number is then generated in the range 0-1. If the random number is less than ave ph, the search is further pursued; else the ant quits and then leaves the solution without any alteration. Now the limiting step is calculated for the region using the constant k_1 and . Then k_1 and k_2 values are dependent upon the natural problem. For this case, the k_1 and k_2 values are taken as k_1=0.1 and k_2 =0.001. Here k_1 is always greater than k_2.

Limiting step: $(1s) = k_1 - (age_i \times k_2)$ **(Eq. 3.8)**

Again, a random number is generated and based on this random number, the following operation is performed. As the random number is greater than 0.5, the limiting step is added; else the limiting step is subtracted from the X_i values. X is representing as the solution.

If $\boxed{r > 0.5}$ $X_{i\ new} = X_{i\ old} + \text{Limiting step}$ **(Eq. 3.9)**

If $\boxed{r < 0.5}$ $X_{i\ new} = X_{i\ old} - \text{Limiting step}$ **(Eq. 3.10)**

Step 4: Update Trail

The average pheromone value is calculated as follows. If the current solution is less than the previous solution, the age is incremented by one. Otherwise, it is decremented by one. Calculate fitness values for the new solution and old solution. The new pheromone value is calculated by using following expression:

$F(X_{i\ new})$ = fitness values for new solution

$F(X_{i\ old})$ = fitness values for old solution

If $F(X_{i\ new}) < F(X_{i\ old})$:

$$\boxed{Age_{new} = Age_{old} + 1} \qquad \textbf{(Eq. 3.11)}$$

If $F(X_{i\ new}) < F(X_{i\ old})$:

$$\boxed{Age_{new} = Age_{old} - 1} \qquad \textbf{(Eq. 3.12)}$$

$$Ph_{new} = \frac{F(X_{i\ new}) - F(X_{i\ old})}{F(X_{i\ old})} (Ph_{old}) \qquad \textbf{(Eq. 3.13)}$$

The pheromone average (ave ph) now is calculated using the new values. The above step are perform are remaining 11 solutions and then 19 more such iterations are performed to improve the solution in the superior region. In the case of solution 1, the values are taken as k_1=0.1 and k_2 =0.001. The limiting step is calculated to be for example 0.09. Also, now a random number is generated (for example, 0.6). The new value of X_1 is calculated. Similarly, the other parameters are also modified and the new solution is obtained. Since the new solution is inferior to the old solution, the average pheromone value of this solution is modified.

Step 5: Termination

There were two main methods for termination step. First method for termination is with set the number of generation. For example, the number of generation is set for 100. The computer will run the simulation until 100 times. For the second method for termination is set the require fitness values. Example, the fitness values set is 0.7555 unit/minutes. The computer will run the simulation until the set up fitness value is obtained.

For this study, the simulation will optimize the parameters to obtain the maximize production rate. The exact maximize value for production rate is don't know at initialize project and this method will waste the time cause the will run the simulation until get the approaching set up fitness value. The first method is choosing for termination step in this study with set the number of generation.

3.6 SUMMARY

The step in Ant Colony Optimization will start with initialize step followed by evaluation step, reproduction step, update trail step and finally last with termination step. The entire step will followed and complete the optimization using Ant Colony Optimization. The objective functions are being calculated to maximize the production rate. All the values used are taken from literature and the optimization process will run by using Matlab software.

CHAPTER 4

RESULT AND DISCUSSION

4.1 INTRODUCTION

The Ant Colony Optimization simulation is done with follow all the step in this method. Using ACO simulation, we can obtain the best value of cutting speed, feed rate and depth of cut. The best production rate as the objective function in this optimization process also obtains from the simulation. The entire values can be use in real machine to get the maximum production of product.

4.2 SIMULATION SETUP AND ASSUMPTION

In the simulation, production rate as the objective function will be optimize. The value of the parameters is randomly generated based on the range for each parameter. The range of parameter, type of material, all the constant values and the case study is taking from **"***Manufacturing Optimization through Intelligent Techniques*" by R. Saravanan, (2006). The objective function equation is taking from *"NC End Milling Optimization Using Evolutionary Computation"* by V. Tandon et. al. (2001).

During program the simulation, the system of computer is very important to make sure all program can running clearly and faster. Acer model, TravelMate 4150 Laptop is used to make the program. The system of this computer is Microsoft Window XP, Professional, and Version 2002. The computer used Intel(R) Pentium(R) M, Processor 1.73GHz and 504 MB of RAM.

This computer has a little problem during simulate if using the latest Matlab version 7.1. This software needs very high capacity 1.84 GB from computer memory compare with version 6.5 only need 700 MB cause this simulation running slowly and take almost 7 minute for running 100 generation. Mablab version 6.5 is choosing to make sure the simulation run faster to decrease the run time of simulation. The simulation of 100 generation can be complete below than 1 minutes.

4.3 SIMULATION OBJECTIVE

There are two main objectives in simulation ACO. First, to find the best parameter values of cutting speed, feed rate and depth of cut. And second is to find the best production rate. The objective function in ACO simulation will compare with objective function from LPT and GA simulation to validate the value of production rate obtains from ACO simulation.

4.3.1 The Best Parameters Value

There are three main parameters that be used by related journals is depth of cut, feed rate and cutting speed. These parameters give the great effect for milling machining. The range of each parameter is taking from references (R. Saravanan, 2006). The best parameter can be obtained through ACO simulation. Optimum machining parameters resulting from this work are intended for use by milling processes in order to improve machining efficiency, minimum production cost, and also minimum production time. For this study, the maximum production rate is choosed as the objective function.

4.3.2 The Best Production Rate

The production rate is the objective function for this study. For maximize the production rate, only the best parameter is choosed. The best parameter is the main factor the objective function. The production rate equation is taking from "*NC End Milling Optimization Using Evolutionary Computation*" by V. Tandon et. al. The function of machining time as the related equation use to find the production rate considers entire three parameters. The best production rate value can be obtained from the ACO simulation.

4.4 RESULT

The ACO simulation involves 5 steps. First step is Initialize, follow by Evaluation step, Reproduction step, Update Trail step and finally Termination step. To complete the step, the several procedures and condition must be followed. All the result from entire step is explained.

Step1: Initialize

For the first step called initialize, the parameter value for cutting speed, feed rate and dept of cut is randomly generated for 20 values. All the parameter values then convert to decimal number and binary number. Five digit of binary number is set for each parameter values and then all three parameter in binary number combined to produce the solution in 15 digit of binary number. The last result for step initialize like shown below:

The 20 solution for Initialize
111010000011011
100001100000111
000011000000110
100010111110001
110001001100000
110000001011000
001000111101011
110001101111000
011100101000111
110010111011110
100100110010000
010100011000001
100110111101111
101101101011110
010110011101000
011011101001010
001001111011100
010101100000110
100110111100110
110001101001011

Step2: Evaluation

For this step, the objective function (Eq. 3.1) is applied. The parameter values from initialize step are use as the variable in the objective function and all the constant values are taking from the references. After apply the objective function, the production rate value is obtained as below:

The Production Rate (unit/minute)
0.6972
0.6852
0.5790
0.6923
0.6573
0.6852
0.6179
0.7619
0.6212
0.7444
0.6853
0.5545
0.6929
0.7658
0.5974
0.6887
0.7195
0.6566
0.6598
0.7257

The production rate value then arrange in descending order and then made the new chromosome that equivalent with the arrangement like shown below:

The New Chromosome for Evaluation
101101101011110
110001101111000
110010111011110
110001101001011
001001111011100
111010000011011
100110111101111
100010111110001
011011101001010
100100110010000
100001100000111
110000001011000
100110111100110
110001001100000
010101100000110
011100101000111
001000111101011
010110011101000
000011000000110
010100011000001

Step3: Reproduction

1) Crossover

There are three sub-steps in the reproduction, crossover, mutation, trail diffusion and local search. For the crossover, the solution is modifying by replacing inferior solution for solution 13-18. The solution is taking from the step evaluation to modify. After get the new inferior solution, the old solution for solution 1-12 and 19-20 are take from evalution step and then combine with the new solution 13-18. The new chromosome is produce like shown in the result below.

The New Chromosome for Crossover
101101101011110
110001101111000
110010111011110
110001101001011
001001111011100
111010000011011
100110111101111
100010111110001
011011101001010
100100110010000
100001100000111
110000001011000
110001101001011
110001101111001
100110111110001
110011101011110
011011101011110
110001100000111
000011000000110
010100011000001

2) Mutation

This step will update the solution 13-18 again. The solution is taking from crossover step. Each solution will divide into two based on random number generated rename as parent1 and parent 2. The mutation is done when the randomly generated number of mutation is large than number of mutation set. Then, the new solutions 13-18 know as child obtains. Take the old solution 1-12 and 19-20 from crossover step then combine with the new solution 13-18 to obtain the new chromosome like the result below.

The New Chromosome for Mutation
101101101011110
110001101111000
110010111011110
110001101001011
001001111011100
111010000011011
100110111101111
100010111110001
011011101001010
100100110010000
100001100000111
110000001011000
101001011110001
111001110001101
100110111110001
110110011101011
011011101011110
000011111000110
000011000000110
010100011000001

3) Trail Diffusion

This step will update the solution 19 and 20. The old solution 19-20 is taking from mutation step and replacing with the solution in superior solution. The new solution is obtained know as child. The old solution 1-18 is taking from mutation step and combine with the solution 19-20. The new choromosome is obtain like the result shown:

The New Chromosome for Trail Diffusion
101101101011110
110001101111000
110010111011110
110001101001011
001001111011100
111010000011011
100110111101111
100010111110001
011011101001010
100100110010000
100001100000111
110000001011000
101001011110001
111001110001101
100110111110001
110110011101011
011011101011110
000011111000110
101011010001111
100100110010000

4) Local Search

Local search step will improve the superior solution (solution 1-12). The old solution 1-12 is taking from trail diffusion step. Limiting step will apply to each solution to improve the solution. Take the old solution 13-20 from trail diffusion step then combine with the new superior solution to obtain the new chromosome like the result below:

The New Chromosome for Local Search
101101101010111
110001101110001
110010111010111
110001101000100
001001111010101
111010000010100
100110111101000
100010111101010
011011101000011
100100110001001
100001100000000
110000001010001
101001011110001
111001110001101
100110111110001
110110011101011
011011101011110
000011111000110
101011010001111
100100110010000

Step4: Update Trial

The step update trail is to obtain the new age and new pheromone value for all solution in superior region. For the initial, the ages are set as 10 and the pheromone value is set as 1. The superior solution is update from local search step. The age and pheromone value are update for each generation. The average of pheromone value is use to update the old pheromone value like shown in result below:

The New Age
11
11
11
11
11
11
11
11
11
11
11
11

The New Pheromone Value
-0.0185
-0.0216
-0.0215
-0.0370
-0.0261
-0.0300
-0.0394
-0.0375
-0.0459
-0.0404
-0.0518
-0.0337

The New Pheromone Average
-0.0336

Step5: Termination

For step termination, the simulations are running for 100 generation. The best of production rate, the best solution, the best cutting speed, the best feed rate and the best depth of cut is recorded as the final result for this simulation. The result for 100 generation as below:

The Best Production Rate (unit/minute)
0.7797

The best Cutting Speed (mm/minute)
9290.3

The Best Feed Rate (mm/tooth)
0.1968

The Best Depth of Cut (mm)
3.8065

4.5 RESULT AND DISCUSSION

The graph best solution versus generation obtain from simulation of Ant Colony Optimization for find the best parameters, cutting speed, feed rate, and depth of cut that give the greatest effect to milling machine process.

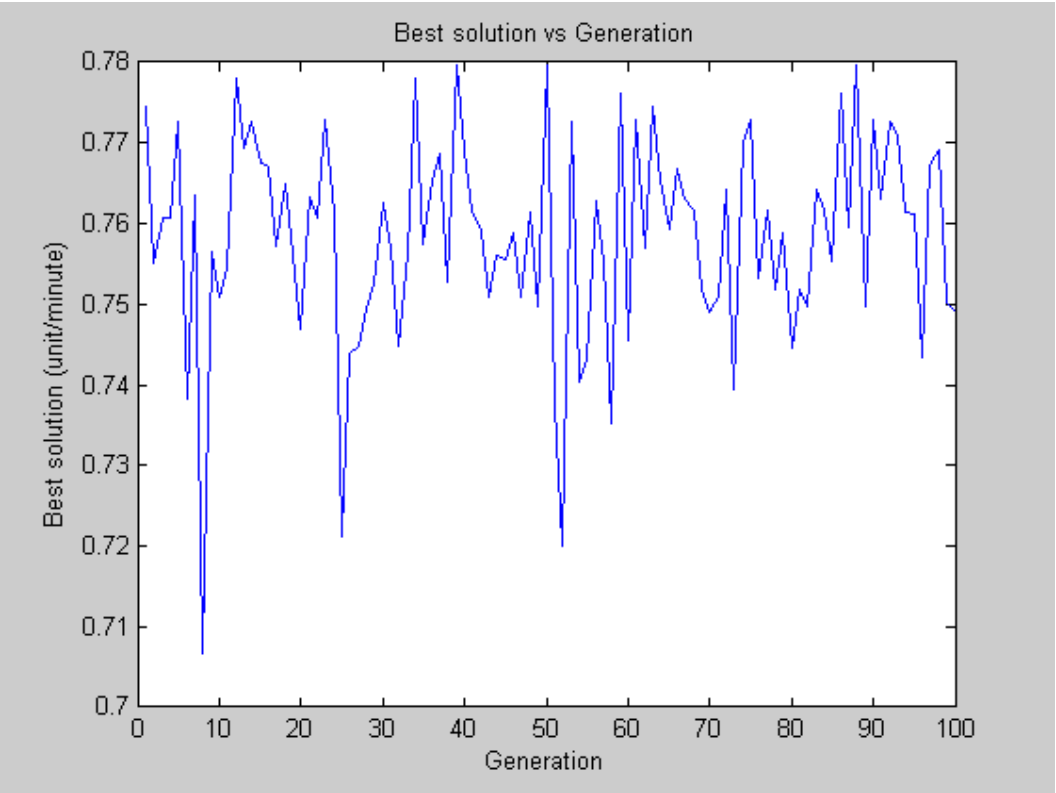


Figure 4.1: Best Solution versus Generation

The graph is resulted from the best production rate obtained versus 100 generation. The random plot of graph obtains because the values of production rate is not uniform at all. The values of production rate are depending from the values of three parameters that randomly generated between the ranges. From this graph, the minimize production rate is obtain at generation 8 with the values 0.7066 unit/minute and the maximum or the best production rate is obtain at generation 87 with the values 0.7797 unit/minute.

The graph above is modified to get the optimal value from this graph. The graph best overall versus generation below show the optimal or best values of production rate are obtain from the generation 87.

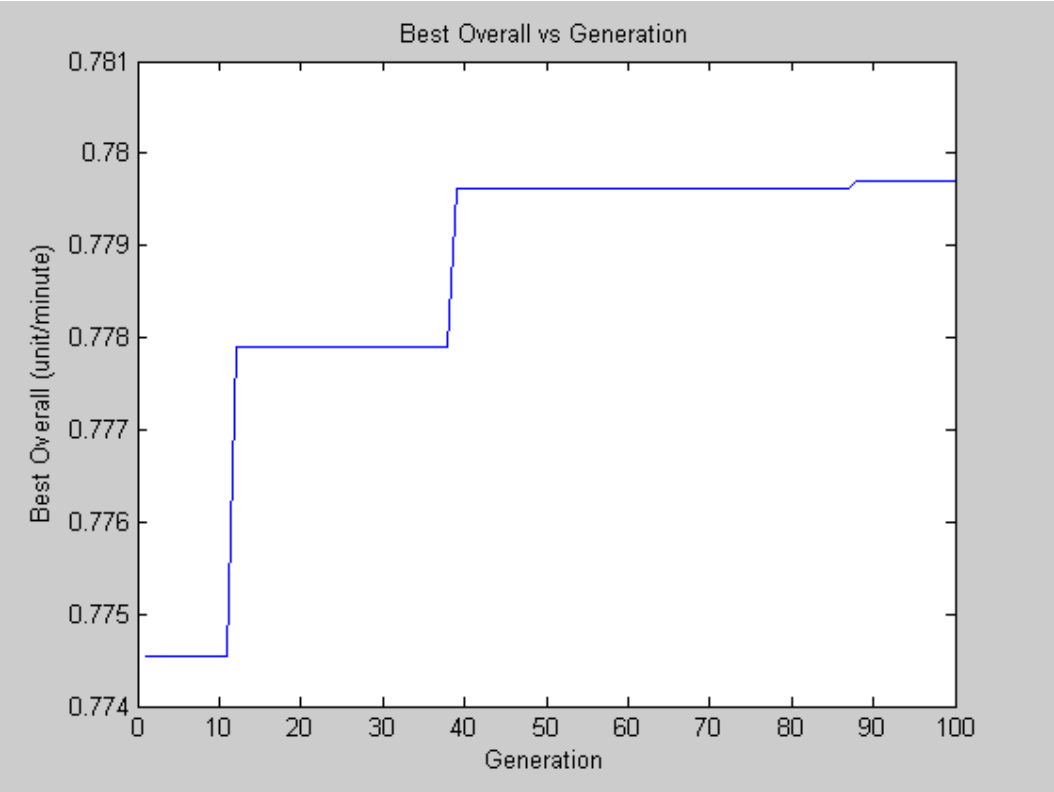


Figure 2: Graph Best Overall versus Generation

The graph is resulted from the best overall production rate obtained versus 100 generation. From the graph, the minimum and maximum value is clearly than the first graph. The graph show that the maximize value of the production rate start obtain uniformly at generation 87 with the value 0.7797 unit/minute. The best value can obtained from the simulation before the end of simulation for 100 generation. The simulation no needs to run until 100 generation at the next simulation to get the best value using Ant Colony Optimization simulation. This metaheuristic have the ability to decrease the simulation run time.

Table 4.1: Selected Result from 100 Generation

No. of generation	Best solution	Best production rate(unit/min)	Best cutting speed, V(mm/min)^10	Best feed rate, f(mm/tooth)	Best depth of cut, a(mm)
1	110101110011110	0.7745	8.8258	0.1903	3.9355
5	110011110111101	0.7727	8.6710	0.1935	3.8710
15	110011101111100	0.7674	8.6710	0.1871	3.8065
23	111001101111100	0.7729	9.1355	0.1871	3.8065
36	110001101011101	0.7654	8.5161	0.1839	3.8710
50	110101111011111	0.7795	8.8258	0.1968	4.0000
65	101101110011010	0.7594	8.2065	0.1903	3.6774
70	110001010111010	0.7489	8.5161	0.1677	3.6774
87	**111011111011100**	**0.7797**	**9.2903**	**0.1968**	**3.8065**
100	101101101010111	0.7490	8.2065	0.1839	3.4839

In Table 4.1, we have also computed the best of parameters values, the best solution and the best production rate achieved for 10 randomly selected from the generation with the corresponding graph being shown in Figure 2. From these tables, we are able to summarize ACO simulation as follows:

(i) The simulation gets the best result on generation 87 with the production rate values 0.7797 unit/minutes.

(ii) The metaheuristic has the ability to reduce the simulation run time.

4.5.1 Verification

For test the efficiency of this programmed, the calculation is done to prove the values obtain from the simulation. We run this simulation until 100 generation and take the best objective function. For this simulation, the best objective functions are obtained from generation 87 with the value 0.7797 unit/minute.

Calculation:

The best solution: 11101**11110**11100

Separate the solution:

Cutting Speed	Feed Rate	Depth of Cut
11101	**11110**	11100

The first separate value is represent for cutting speed, **V**, second for feed rate, **f** and third for depth of cut, **a**. Decode all the binary number into decimal values. Using the interpolation, convert the decimal values to parameter values.

Convert binary values into decimal values:

Binary	**Decimal**
11101	29
11110	30
11100	28

Convert decimal values into parameter values:

Cutting Speed, V = 4800 x (29/31) + 4800

= 9290.3 mm/min

Feed Rate, f = 0.1 x (30/31) + 0.1

= 0.1968 mm/tooth

Depth of Cut, a = 2 x (28/31) + 2

= 3.8065mm

Apply the objective function equation:

L_a = 90 mm

L_b = 10mm

L_n = 200mm

Formula to find the total length:

L = 10mm + 200mm + 90mm

= 300mm

L = 300mm

D = 160mm

V = 9445.2 mm/minute

f = 0.1935 mm/tooth

a = 3.9355 mm

Z = 4 tooth

W=30mm

Formula to find the machining time:

$T_m = \pi DL(W/a)/(1000ZVf)$

$$= \frac{\pi\,(160mm)(300mm)(30mm)}{1000\,(4)(\,9290.3\ mm/min)(\,0.1968\ mm/tooth)(\,3.8065mm)}$$

= 0.1625 minute

T_{cm} = 1 minute

T_r = 0.12 minute

Formula to find the total time:

$T_{ed} = T_m + T_{cm} + T_r$

= 0.1625 min + 1 min + 0.12 min

= 1.2825 minute

Formula to find the production rate:

$Q = 1/T_{ed}$

= 1/ 1.2825min

= 0.7797unit/minute **(proven)**

4.6 SUMMARY RESULT

All the result needed including best production rate and the best parameters values will be recorded in the simulation output. The output recorded the 100 values of the production rate, the values of each parameters and solution for each generation. The best values of the result choosing from the 100 generation to satisfy the objective of ACO simulation, find the optimum parameter value to maximize production rate.

4.6.1 Validation

The best results obtains in simulation is compare with the metaheuristics that we have proposed and the best results in literature for the benchmark problem in literature. The comparison is between Ant Colony Optimization, (ACO) with Linear Programming, (LPT) and Genetic Algorithm, (GA). It can be seen that ACO is able to obtain a solution close to the best solution obtained by GA.

Table 4.2: Comparison between ACO with LPT and GA

Method	Production Rate (unit/minute)
Linear Programming Technique	0.3800
Ant Colony Optimization	0.7797
Genetic Algorithm	0.9380

Comparison of results of competing metaheuristics of the problems, a total of 100 runs is performed and the best solution, the best parameters values and the best production rate are recorded. Table 4.2 shows the comparison of the best production rate obtained by each metaheuristic for the benchmark problems.

Based on the result of production rate, the quality of the solutions obtained by ACO is better than LPT and not as good as that of the GA heuristics. LPT get the result for the best production rate 0.3000 unit/minute while ACO simulation obtained the best production rate 0.7797 unit/minute and GA simulation by references obtained the best production rate 0.9380 unit/minute.

The LPT and GA simulation in reference (R. Saravanan, 2006) only considers two parameters, cutting speed and feed rate in the objective function. The reference not considers the depth of cut as the parameter that gives the greatest effect to the machining process. The function from other references journal, "*NC End Milling Optimization Using Evolutionary Computation*" by V. Tandon et. al (2001).

The function above add the depth of cut, a as the variable parameter. However, the quality of the solutions obtained by LPT is not as good as that of the other heuristics, such as GA and ACO. Hence, ACO and GA can be preferred if solutions of good quality were to be obtained within reasonable computation result. The computational results of ACO and GA almost same although use the different function of machining time.

CHAPTER 5

CONCLUSION & RECOMMENDATION

5.1 INTRODUCTION

In this study, the methodology is presented to maximize production rate for milling machining processes. The main objective of this study is to develop ACO algorithms to give a good solution for the parameter that give the greatest effect to the machining process to maximize the production rate. The metaheuristics ACO are developed and applied to solve the problem.

ACO is designed according to the multistation nature of the problem by calculating the probability functions. Computational results are compared to those published in the literature (R. Saravanan, 2006) for the base problem. A simulation is also carried out on 100 generated at random to further compare the performance of the metaheuristics in terms of computational efficiency and solution quality.

From the Ant Colony Optimization simulation, all the result needed including best production rate and the best parameters values are obtain. Hence, the ACO heuristic is not as good as GA heuristic, ACO can be preferred if solutions of good quality were to be obtained within reasonable computation result.

5.2 CONCLUSION

For ACO, the encouraging results indicate that there is potential for further improvement in their procedures, such as using the 3-optimize procedure to optimize each parameter formed when applying ACO, so that ACO performance could be as good as that of GA. The development simulation for ACO for optimize milling parameters is successful. The best production rate with the best parameters values is obtained from the simulation.

5.3 RECOMMENDATION

The Matlab software versions 7.1 have more advantage than Matlab version 6.5. For run this software using low system of computer cause the computer become slow and not responding. For using Matlab version 7.1, the recommended to upgrade the systems of this computer especially RAM from 512 MB to 1GB memory.

For test the simulation efficient, the experiment will provide to test the parameter values obtain is satisfy the value of production rate. The experiment must use the same problem and the constant value taken from the literature.

Besides only using the provide metaheuristic by references, we must exploring on how the design of metaheuristics can be further improved, it is also possible to consider modifying the objective function and constraints of the problem to consider more scenarios in future research work. Other extensions include considering more parameters like Number of Passes, Cutting Force and another parameters to get more efficient result for objective function.

REFERENCES

M. Tolouei-Rad and I. M. Bidhendi (1996). *On the optimization of machining parameters for milling operations.* Elsevier Science Ltd.

V. Tandon, H. El-Mounayri and H. Kishawy (2001). *NC end milling optimization using evolutionary computation*. International Journal of Machine Tools & Manufacture 42 (2002) 595–605.

N. Baskar, P. Asokan , R. Saravanan and G. Prabhaharan(2005). *Selection of optimal machining parameters for multi-tool milling operations using a memetic algorithm.* Journal of Materials Processing Technology 174 (2006) 239–249.

J.A. Ghani, I.A. Choudhury and H.H. Hassan (2002). *Application of Taguchi method in the optimization of end milling parameters*. Journal of Materials Processing Technology 145 (2004) 84–92.

M. Milfelner, F. Cus and J. Balic. *An overview of data acquisition system for cutting force measuring and optimization in milling*. Journal of Materials Processing Technology 164–165 (2005).

Godfrey C. Onwubolu (2005). *Performance-based optimization of multi-pass face milling operations using Tribes.* International Journal of Machine Tools & Manufacture 46 (2006) 717–727.

Ali Rıza Yıldız (2007). *A novel hybrid immune algorithm for optimization of machining parameters in milling operations.* Robotics and Computer-Integrated Manufacturing.

Z.G. Wang, M. Rahman, Y.S. Wong and J. Sun (2005). *Optimization of multi-pass milling using parallel genetic algorithm and parallel genetic simulated annealing.* International Journal of Machine Tools & Manufacture 45 (2005) 1726–1734.

M. Wan, W.H. Zhang, G. Tan and G.H. Qin (2007). *Systematic simulation procedure of peripheral milling process of thin-walled workpiece*. Journal of materials processing technology 1 9 7 (2 0 0 8) 122–131.

Oğuz Çolak, Cahit Kurbanoğlu and M. Cengiz Kayacan (2005). *Milling surface roughness prediction using evolutionary programming methods*. Materials & Design Volume 28, Issue 2, 2007, Pages 657-666.

Duan Hai-bin, Wang Dao-bo and Yu Xiu-fen (2006). *Novel Approach to Nonlinear PID Parameter Optimization Using Ant Colony Optimization Algorithm*. Journal of Bionic Engineering 3 (2006) 073-078.

De-Sian Lu, Chien-Chang Chen (2007), *Edge Detection Improvement by Ant Colony Optimization*. Pattern Recognition Letters.

R. Saravanan (2006). *Manufacturing Optimization through Intelligent Techniques.*

APPENDICES

APPENDIX A-1: GANT CHART

GANT CHART FOR PSM 1

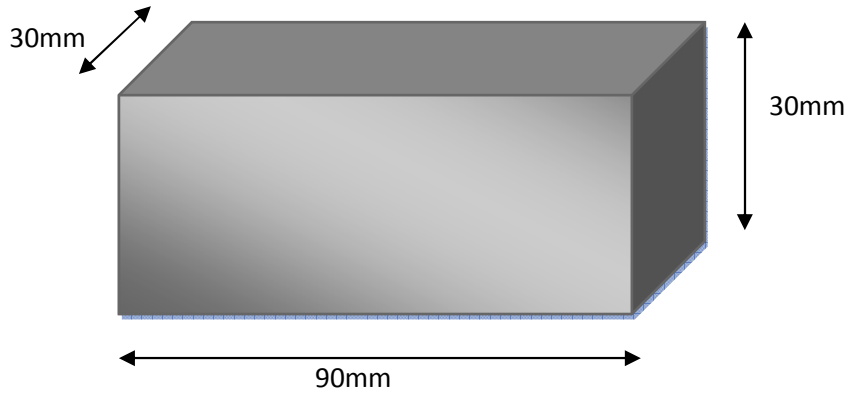
Task	Weeks														
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Literature Review	■	■	■												
Gathering Information			■	■											
Indentify Objective and Problem Statement				■											
Identify Related Parameter and Previous Method				■	■										
Identify Objective Function					■	■									
Methodology							■	■	■						
Understand the Algorithm										■	■				
Apply Algorithm to the Problem										■	■				
Report Writing												■	■	■	■
Project Presentation															■
Final Report Check and Submit															■

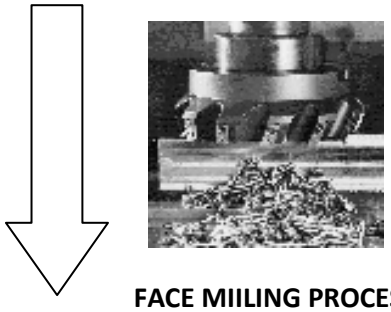
GANT CHART FOR PSM 2

Tasks	Weeks														
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Develop Algorithm	■	■	■	■	■	■	■	■							
Perform Optimization Simulation								■	■						
Confirm Result									■	■	■				
Report Writing									■	■	■	■	■		
Project Presentation													■		
Final Report Check and Submit														■	■

APPENDIX B-1: SAMPLE PRODUCT

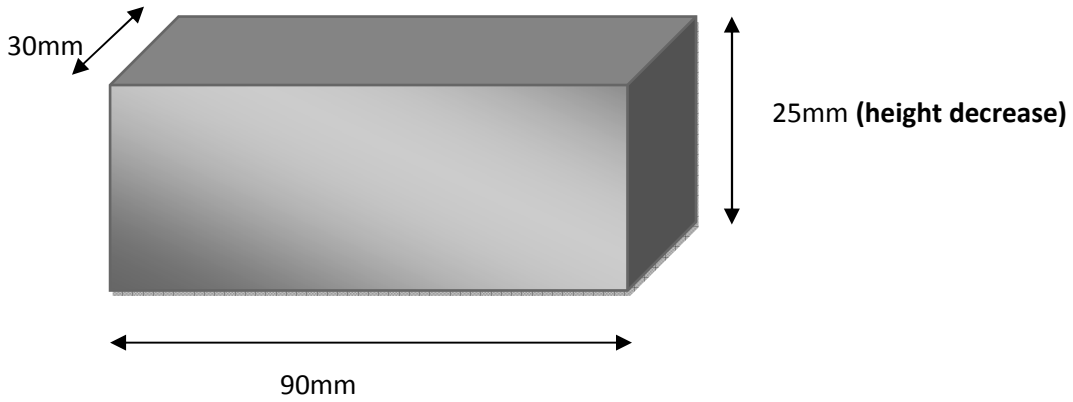
ORIGINAL WORKPIECE (Carbon Steel)





FACE MIILING PROCESS

PRODUCT



MATLAB PROGRAM FOR ANT COLONY OPTIMIZATION SIMULATION

APPENDIX C-1: INITIALIZE STEP

```
%START INITIALIZE STEP

disp('Step1: Initialize')

%start with randomly generate 20 parameter values for cutting
speed,V,feed rate,f and depth of cut,a

V = randint(20,1,[4800,9600]);
f = 0.1*rand(20,1)+0.1;
a = 2*rand(20,1)+2;

disp('The random values for cutting speed is:');
V
disp('The random values for feed rate is:');
f
disp('The random values for deep of cut is:');
a

%convert the parameter values into a decimal number

for i = 1:1:20;
V(i,1);

%formula to convert from parameter value to decimal

dV = 31*(V-4800)/4800;
df = 31*(f-0.1)/0.1;
da = 31*(a-2)/2;
end

disp('The decimal values for V is:');
dV
disp('The decimal values for f is:');
df
disp('The decimal values for a is:');
da

%convert the decimal value to binary number

bV = dec2bin(dV);
bf = dec2bin(df);
ba = dec2bin(da);

disp('The binary number for cutting speed is:');
bV
disp('The binary number for feed rate is:');
bf
disp('The binary number for depth of cut is:');
ba
```

```
%combine the binary number for each parameter

for nchrom=1:1:20;
    num=[bV(nchrom,:) bf(nchrom,:) ba(nchrom,:)];
    chrom(nchrom,:)=[num];
end

disp('The 20 solution is:');
chrom

%END OF INITIALIZE STEP
```

APPENDIX C-2: EVALUATION STEP

```
%START EVALUATION STEP

%load file from initialize.m

initialize;

%evaluation step

disp('Step 2: Evaluation')

%constant values for calculate the objective function

Lb = 10;
La = 90;
Ln = 200;
D = 160;
Z = 4;
W = 30;
Tr = 0.12;
Tcm = 1;

%formula to find total length

L = Lb + Ln + La;

for i = 1:1:20;
V(i,1);

%formula to find machining time

for j=1:1:20;
Tm = (pi*D*L*(W/a(j,1)))/(1000*Z*V(j,1)*f(j,1));
MachTime(j,:)=[Tm];
end
end

for i=1:1:20;
    MachTime(i,:);

%formula to find the total time

for j=1:1:20;
Ted = MachTime(j,:) + Tcm + Tr;
TotTime(j,:)=[Ted];
end
end

for i = 1:1:20;
    TotTime(i,:);
```

```
%formula to find the production rate as the objective funtion

for j = 1:1:20;
Q = (1/TotTime(j,:));
ProRate(j,:)=[Q];
end
end

disp('The length,L is:');
L
disp('The machining time,Tm is:');
MachTime
disp('The total time,Ted is:')
TotTime
disp('The production rate,Q');
ProRate

%sort the production rate values in decesding order

y=sort(ProRate);

for i=20:-1:1;
    z=y(i,1);
    ProrateDec(21-i,:)=[z];

end

disp('The production rate in descending order is:');
ProrateDec

%Find new chrom based on new arrangement of objective function

for i=1:20
    for j=1:20
        if ProrateDec(i,1)== ProRate(j,1)
            Newchrom(i,:)=[chrom(j,1:15)];
            break
        end
    end
end

disp('The Newchrom is:');
Newchrom

%END OF EVALUATION STEP
```

APPENDIX C-3: REPRODUCTION STEP

1) CROSSOVER

```
%START CROSSOVER STEP

%load file from evaluation.m

evaluation;

%Step Reproduction

disp('Step3: Reproduction')

%Start with Crossover or Random Walk

disp('Step3-1: Crossover or Random Walk')

%this step done to modify the inferior solution
%solution 13-18

for e=1:6;

%Randomly generate solution number

R=randint(1,2,[1,12]);

disp('The random solution number is:')
R

%find the choromozone based on random number,R

SC=Newchrom(R,:);

disp('The solution generate is:')
SC

%randomly generated the number for separate the chromozone

cut=randint(1,1,[1,15]);

disp('The generate number for separate the solution is:')
cut

%separated parent1 based on rondom number,cut

P1a=SC(1,1:cut)
P1b=SC(1,cut+1:15)

%separated parent2 based on rondom number,cut

P2a=SC(2,1:cut)
P2b=SC(2,cut+1:15)
```

```
%get the child1 and child2 solution

Child1= [P1a P2b];
Child2= [P2a P1b];

disp('The new child one is:');
Child1

disp('The new child two is:');
Child2

%compared the value of child1 and child2 and take the best

%separate the child1 solution

for i=5;
Cu=[Child1(1,1:i); Child1(1,i+1:10); Child1(1,16-i:15)];
end

%convert binary number into decimal number

du=bin2dec(Cu);

%convert decimal number into parameter values

fV=(4800*(du(1,1))/31)+4800;
ff=(0.1*(du(2,1))/31)+0.1;
fa=(2*(du(3,1))/31)+2;

%constant values for calculate the objective function

Lb = 10;
La = 90;
Ln = 200;
D = 160;
Z = 4;
W = 30;
Tr = 0.12;
Tcm = 1;

%formula to find total length

L = Lb + Ln + La;

%formula to find machining time

MachTime = (pi*D*L*(W/fa))/(1000*Z*fV*ff);
Tm=[MachTime];

%formula to find the total time

TotTime = Tm + Tcm + Tr;
Ted=[TotTime];
```

```
%find the objective function

Q = (1/Ted);
ChildOne=[Q];

%separate the child2 solution

for i=5;
Ce=[Child2(1,1:i); Child2(1,i+1:10); Child2(1,16-i:15)];
end

%convert binary number into decimal number

de=bin2dec(Ce);

%convert decimal number into parameter values

fV2=(4800*(de(1,1))/31)+4800;
ff2=(0.1*(de(2,1))/31)+0.1;
fa2=(2*(de(3,1))/31)+2;

%constant values for calculate the objective function

Lb = 10;
La = 90;
Ln = 200;
D = 160;
Z = 4;
W = 30;
Tr = 0.12;
Tcm = 1;

%formula to find total length

L = Lb + Ln + La;

%formula to find machining time

MachTime = (pi*D*L*(W/fa2))/(1000*Z*fV2*ff2);
Tm=[MachTime];

%formula to find the total time

TotTime = Tm + Tcm + Tr;
Ted=[TotTime];

%find the objective function

Q = (1/Ted);
ChildTwo=[Q];
```

```
%choose the best child

if ChildOne > ChildTwo;
    X = Child1;
else ChildOne < ChildTwo;
    X = Child2;
end

disp('The best child is:');
X

NewX(e,:)=[X];
end

disp('The new solution for solution 13-18 is:');
NewX

%find the old inferior solution solution

%solution 1-12 represent as Rchrom

for i= 1:1:12;
    R=Newchrom(i,:);
    Rchrom(i,:)=[R];
end
Rchrom

%solution 19-20 represent as Hchrom

for i=19:20;
    H=Newchrom(i,:);
    Hchrom(i-18,:)=[H];
end
Hchrom

%combine the old solution(Rchrom & Hchrom) with the new
solution(NewX)

for j=1:1:20;
   New=[Rchrom;NewX;Hchrom];
   NewChrom1=[New];
end

disp('The new chrom is:');
NewChrom1

%END OF CROSSOVER STEP
```

2) MUTATION

```
%START THE MUTATION STEP

%load file from reproduction1.m

reproduction1;

disp('Step3-2: Mutation:')

%To repair solution 13-18

for j=1:6;

%Set the level of mutation from 0-1

M = 0.45;

disp('The level of mutation are set as:');
M

%Randomly generated the mutation level

r = rand(1,1);

disp('The generate level of mutation is:');
r

%for applying mutation,r must greater or equal to level of mutation

if r < M;
    disp('NO need for mutation procedure')
   for i=1:1:6;
       w=NewX(i,:);
       for i=j;
           z=NewX(j,:);
           ChildNew(i,:)=[z];
       end
   end
ChildNew

else r >= M;

%apply mutation

  for r=1:1:6;
       y=NewX(r,:);
       for r=j;
           z=NewX(j,:);
           SolChrom(r,:)=[z];
       end
  end
SolChrom
```

```
%generate randomly from 1-15 represent as the number of binary
number

CutChrom=randint(1,1,[1,15]);

disp('The random number to separate the solution is:');
CutChrom

%separate the child based on random number,CutChrom

Child1=SolChrom(j,1:CutChrom)
Child2=SolChrom(j,CutChrom+1:15)

%arrange child2 at the front side and child1 at the back of solution

ChildNew(j,:)=[Child2 Child1];

disp('The new Child is:');
ChildNew
end

ChildNewX(j,:)=ChildNew(j,:)
end

%find the old inferior solution
%the solution 1-12 represent as Pchrom

for i= 1:1:12;
    P=NewChrom1(i,:);
    Pchrom(i,:)=[P];
end
Pchrom

%the solution 19-20 represent as Bchrom

for i=19:20;
    B=NewChrom1(i,:);
    Bchrom(i-18,:)=[B];
end
Bchrom

%combine the old solution with the new solution

for j=1:1:20;
   New=[Pchrom;ChildNewX;Bchrom];
   NewChrom2=[New];
end

disp('The new chrom is:');
NewChrom2

%replace new child into solution 13
%do the mutation procedure for other solution(14-18)

%END OF MUTATION STEP
```

3) TRAIL DIFFUSION

```
%START TRAIL DIFFUSION STEP

%load file from mutation.m

mutation;

disp('Step3-3: Trail diffusion')

%improvement solution 19 & 20

for q=1:2;%loop for two solution

%generate solution in superior region

y = randint(1,2,[1,12]);

disp('The randomly generate solution is:');
y

%The solution are taken as parent1 and parent2

ChromSel=NewChrom2(y,:);

disp('The solution number generate based on mutation number is:');
ChromSel

%convert the solution into decimal values

P1 = bin2dec(ChromSel(1,:))
P2 = bin2dec(ChromSel(2,:))

%random number is generated from 0-1

 alpha = rand(1,1)

 %find the child

 if alpha <= 0.5;
     C = (alpha)*P1+(1-alpha)*P2;
 else  alpha > 0.5;
         C = P2;
 end

disp('Child,C is:')
C

%convert the child into decimal number

C1= dec2bin(C,15);%to generate 15 bit
```

```
disp('The new child solution is:')
C1

CX(q,:)=[C1]
end

%find the old inferior solution

%solution 1-18 represent as Dchrom

for i= 1:1:18;
    N=NewChrom2(i,:);
    Dchrom(i,:)=[N];
end
Dchrom

%combine the old inferior sol. (Dchrom) with the new superior
solution, (CX)

for j=1:1:20;
   d=[Dchrom;CX];
   NewChrom3=[d];
end

disp('The new chrom is:');
NewChrom3

%END OF TRIAL DIFFUSION STEP
```

4) LOCAL SEARCH

```
%START LOCAL SEARCH STEP

%load file from trail.m

trail;

disp('Step3-4: Local Search')

%to improve 12 solutions in superior region(solution 1-12)

ph = 1;
Age = 10;
n = 12;

%finds the old solution

for i=1:1:12;
Xold(i,:) = Newchrom(i,:);
end

disp('The old solution is:');
Xold

%change the solution into decimal number

for i=1:1:12;
    Xold(i,:);
    for j=1:1:12;
        D=bin2dec(Xold(j,:));
        XoldDec(j,:)=[D];
    end
end

disp('The old solution in decimal number is:');
XoldDec

%random number generated between 0-1 for k1 and k2

k1 = rand(1,1)
k2 = rand(1,1)

%apply limiting step,LS

LS = k1 - (Age*k2)

%random number generated between 0-1 again

r = rand(1,1)
```

```
%find the new solution

 if r > 0.5;
     Xnew = XoldDec + LS;
 else r < 0.5;
     Xnew = XoldDec - LS;
 end

disp('The new solution,Xnew is:');
Xnew

%convert the decimal number for new solution into binary number

Xnewchrom=dec2bin(Xnew);

disp('The new superior chromozon is:');
Xnewchrom

%find the old inferior solution

%solution 13-20 represent as Nchrom

for i= 13:1:20;
    N=NewChrom3(i,:);
    Nchrom(i-12,:)=[N];
end
Nchrom

%combine the old inferior sol.(Nchrom)with the new superior
sol.(Xnewchrom)

for j=1:1:20;
   New=[Xnewchrom;Nchrom];
   NewChrom4=[New];
end

disp('The new chrom is:');
NewChrom4

%END OF LOCAL SEARCH STEP
```

APPENDIX C-4: UPDATE TRIAL STEP

```
%START UPDATE TRAIL STEP

%load file from localS.m

localS;

disp('Step4: Update Trail:')

%separate the solution

for i=1:12;

Sep=[Xnewchrom(i,1:5);Xnewchrom(i,5+1:10);Xnewchrom(i,16-5:15)];

%convert the binary number into decimal number

n=bin2dec(Sep);

%convert the decimal number into parameter values

fV=(4800*(n(1,1))/31)+4800;
ff=(0.1*(n(2,1))/31)+0.1;
fa=(2*(n(3,1))/31)+2;

%change the parameters matrix position

VX(1,i)=fV;
fX(1,i)=ff;
aX(1,i)=fa;
end

%constant values for calculate the objective function

Lb = 10;
La = 90;
Ln = 200;
D = 160;
Z = 4;
W = 30;
Tr = 0.12;
Tcm = 1;

%formula to find total length

L = Lb + Ln + La;

for t = 1:12;
VX(1,t);
```

```
%formula to find machining time

for p=1:12;
MachTime = (pi*D*L*(W/aX(1,p)))/(1000*Z*VX(1,p)*fX(1,p));
Tm(p,:)=[MachTime];
end
end
Tm;

for h=1:12;
    Tm(h,:);

%formula to find the total time

for r=1:12;
TotTime = Tm(r,:) + Tcm + Tr;
Ted(r,:)=[TotTime];
end
end
Ted;

%find the objective function

for d = 1:12;
Q = (1/Ted(d,:));
FXnew(d,:)=[Q];
end
FXnew;

for i=1:12;

Sep=[Xold(i,1:5);Xold(i,5+1:10);Xold(i,16-5:15)];

%convert the binary number into decimal number

n=bin2dec(Sep);

%convert the decimal number into parameter values

fV=(4800*(n(1,1))/31)+4800;
ff=(0.1*(n(2,1))/31)+0.1;
fa=(2*(n(3,1))/31)+2;

%change the parameters matrix position

VX(1,i)=fV;
fX(1,i)=ff;
aX(1,i)=fa;
end
```

```
%constant values for calculate the objective function

Lb = 10;
La = 90;
Ln = 200;
D = 160;
Z = 4;
W = 30;
Tr = 0.12;
Tcm = 1;

%formula to find total length

L = Lb + Ln + La;

for t = 1:12;
VX(1,t);

%formula to find machining time

for p=1:12;
MachTime = (pi*D*L*(W/aX(1,p)))/(1000*Z*VX(1,p)*fX(1,p));
Tm(p,:)=[MachTime];
end
end
Tm;

for h=1:12;
    Tm(h,:);

%formula to find the total time

for r=1:12;
TotTime = Tm(r,:) + Tcm + Tr;
Ted(r,:)=[TotTime];
end
end
Ted;

%find the objective function

for d = 1:12;
Q = (1/Ted(d,:));
FXold(d,:)=[Q];
end
FXold;

disp('The fitness value for new solution,FXnew is:');
FXnew
disp('The fitness value for old solution,FXold is:');
FXold
```

```
%calculate the new age for each solution

for i=1:1:12;
    if FXnew < FXold;
        AgeNew(i,:) = Age + 1;
    else FXnew > FXold;
        AgeNew(i,:) = Age - 1;
    end
end

disp('The new age for the solution is:')
AgeNew

%calculate the new pheromone for each solution

for i=1:1:12;
    phNew(i,:)= ((FXnew(i,:) - FXold(i,:))/FXold(i,:))*ph;
end

disp('The new pheromone value,phNew is:');
phNew

%Calculate the pheromone average using the new value

PhAve=mean(phNew);

disp('The new pheromone average is:');
PhAve

%END OF UPDATE TRIAL STEP
```

APPENDIX C-5: TERMINATION STEP

```
%START TERMINATION STEP

for b=1:100;%100 GENERATION

%load file from updateT

updateT;

disp('Step5: Termination')

for j=1:20;

%separate the solution number

for i=1:1:20;
       w=NewChrom4(i,:);
       for i=j;
           G=[NewChrom4(i,1:5);NewChrom4(i,5+1:10);NewChrom4(i,16-
5:15)];
       end
end

%convert the binary number into decimal number

n=bin2dec(G);

%convert the decimal number into parameter values

fV=(4800*(n(1,1))/31)+4800;
ff=(0.1*(n(2,1))/31)+0.1;
fa=(2*(n(3,1))/31)+2;

%change the parameters matrix position

VX(1,i)=fV;
fX(1,i)=ff;
aX(1,i)=fa;

end

%constant values for calculate the objective function

Lb = 10;
La = 90;
Ln = 200;
D = 160;
Z = 4;
W = 30;
Tr = 0.12;
Tcm = 1;
```

```
%formula to find total length

L = Lb + Ln + La;

for t = 1:1:20;
VX(1,t);

%formula to find machining time

for p=1:1:20;
MachTime = (pi*D*L*(W/aX(1,p)))/(1000*Z*VX(1,p)*fX(1,p));
Tm(p,:)=[MachTime];
end
end
Tm;

for h=1:1:20;
    Tm(h,:);

%formula to find the total time

for r=1:1:20;
TotTime = Tm(r,:) + Tcm + Tr;
Ted(r,:)=[TotTime];
end
end
Ted;

%find the objective function

for d = 1:1:20;
Q = (1/Ted(d,:));
ProRate(d,:)=[Q];
end
ProRate

%sort the production rate for easily to choose the best value

Q=sort(ProRate);

disp('The best PRODUCTION RATE(unit/minute) is:')
ProductionRate=Q(20,1)

%find the solution for each best production rate

for i=1:20;
    for j=1:20;
        if Q(20,1)== ProRate(j,:);
            Newc(1,:)=[NewChrom4(j,:)];
            break
        end
    end
end
bestchrom=Newc
```

```
%collect all the production rate in one matrix

s(b,:)=[ProductionRate];

%collect all the best solution in one matrix

ha(b,1:15)=[bestchrom];
end

%display the best production rate for 100 generation

disp('The best production rate for 100 generation:');
s

%display the best solution for 100 generation

disp('The best solution for 100 generation:');
ha

%sort (s) for easily to choose the best production rate in 100
generation

g=sort(s);

%plot the graph(best solution versus generation)

x=[1:100];
y=[s];
plot(x,y),xlabel('Generation'),ylabel('Best solution'),title('Best
solution vs Generation')

%arrange the fitness values to find the

for i=1:100;
    if i==1;
        BO(i,1)=s(i,:);
    elseif s(i,:)> BO(i-1,:);
        BO(i,1)=s(i,:);
    else s(i,:)<=BO(i-1,:);
        BO(i,:)=BO(i-1,:);
    end
end
BO;

%plot the graph(best overall versus generation)

x=[1:100];
y=[BO];
plot(x,y),xlabel('Generation'),ylabel('Best Overall'),title('Best
Overall vs Generation')
```

```
for w=1:100;

%separate the digit of the best solution

Sep=[ha(w,1:5);ha(w,5+1:10);ha(w,16-5:15)];

%convert the binary number into decimal number

x=bin2dec(Sep);

%convert the decimal number into parameter values

Cutting_speed=(4800*(x(1,1))/31)+4800;
Feed_rate=(0.1*(x(2,1))/31)+0.1;
Depth_of_cut=(2*(x(3,1))/31)+2;

PO(w,:)=[Cutting_speed];
HO(w,:)=[Feed_rate];
VO(w,:)=[Depth_of_cut];
end

disp('The best Cutting Speed for 100 generation:');
PO
disp('The best Feed Rate for 100 generation:');
HO
disp('The best Depth of Cut for 100 generation:');
VO

%take the best production rate value among 100 generation

disp('The best PRODUCTION RATE(unit/minute)in 100 generation is:');
g(100,1)

%choose the best solution in 100 generation

for i=1:100;
    for j=1:20;
        if g(100,1)== s(i,:);
            BestSolution(1,:)=[ha(i,1:15)];
            break
        end
    end
end
BestSolution

%separate the digit of the best solution

Sep=[BestSolution(1,1:5);BestSolution(1,5+1:10);BestSolution(1,16-
5:15)];

%convert the binary number into decimal number

x=bin2dec(Sep);
```

```
%convert the decimal number into parameter values

Cutting_speed=(4800*(x(1,1))/31)+4800;
Feed_rate=(0.1*(x(2,1))/31)+0.1;
Depth_of_cut=(2*(x(3,1))/31)+2;

disp('The best cutting speed is:');
Cutting_speed
disp('The best feed rate is:');
Feed_rate
disp('The best depth of cut is:');
Depth_of_cut

%END OF TERMINATION STEP
```