OPTIMIZING CNC TURNING PROCESS USING REAL CODED GENETIC ALGORITHM AND DIFFERENTIAL EVOLUTION

Pinkey Chauhan, Kusum Deep, Millie Pant (2011)

Zhou Yidong

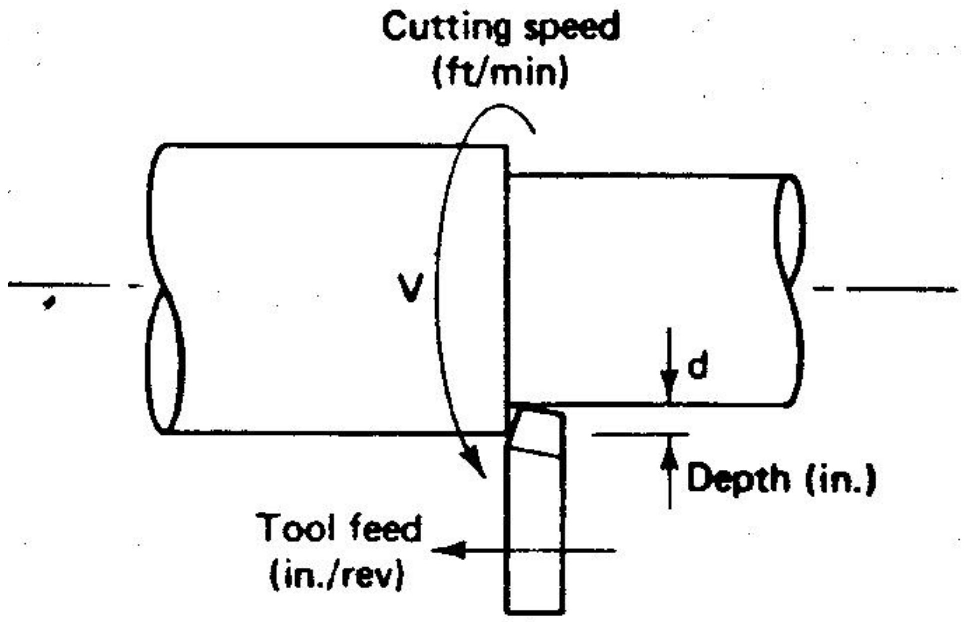
2016.02.23

Contents

Introduction

❖ Problem review

- Develop efficient and economic CNC machining
  - *Control all process by computer*
  - *Meet the highly accuracies*
  - *Introduces flexibility in processing*
- Optimizing CNC turning machine parameter
  - *Cutting speed*
  - *Feed rate*
  - *Depth of cutting*



Introduction

❖ Problem description

- Problem
  - Optimizing CNC turning process parameters
- Objective
  - Minimum production time
- Decision variable
  - Feed rate
  - Cutting speed
- Algorithms
  - An real coded genetic algorithm
  - Differential evolution
- constraints
  - Decision variables' lower and upper bound
  - Cutting force
  - Cutting power
  - Tool chip Interface temperature
  - Surface roughness

Introduction

❖ Notations

Symbol	*Significance*	*Numerical value*
D	diameter of the workpiece (*mm*)	152
L	length of the workpiece (*mm*)	203
V	cutting speed (*m/min*)	-
V_{min},	minimum allowable cutting speed	30
V_{max}	maximum allowable cutting speed	200
f	feed rate (*mm/rev*)	-
f_{min},	minimum allowable feed rate	0.254
f_{max}	maximum. allowable feed rate	0.762
R_a	surface roughness (*μm*)	-
R_{max}	max. surface roughness of rough and finished cut	50
P_{max}	max. power of the machine (*kW*)	5
F_{max}	max. cutting force (N)	900
Θ_{max}	max. temperature of tool workpiece interface (°C)	550
doc	depth of cut (*mm*)	-
doc_{min},	minimum allowable depth of cut (finish)	2.0
doc_{max}	maximum. allowable depth of cut (finish)	5.0
T	tool life (*min*)	
t_m	machining time (*min*)	
t_{cs}	tool change time (*min/edge*)	0.5
t_h	loading and unloading time (*min/pass*)	1.5
t_r	quick return time (*min/pass*)	0.13
T_u	total production time (*min*)	-
C_0	operating cost (*R_s/piece*)	3.5
C_t	tool cost per cutting edge (*R_s/edge*)	17.5
C_T	total production cost (*R_s/edge*)	-
a1, a2, a3, K	constants used in tool life equation	0.29 ; 0.35 ; 0.25; 193.3

The machining optimization model

❖ **The machining optimization model**

✓ Machining processing is shaping of metal parts by removing unwanted material. Should satisfy given quality specifications as accuracy, surface finish and surface integrity with an objective of minimum production time.

- **The objective function:**

$$T_u = t_m + t_{cs}(t_m / T) + t_r + t_h z$$

where cutting time per pass is

$$(t_m) = \frac{\pi DL}{1000\, Vf}$$

Taylor's tool life equation is given by

$$V\, f^{a1}\, doc^{a2}\, T^{a3} = K$$

T_u = Total production time
t_{cs} = Tool changing time (min/edge)
t_r = Quick return time(min/pass)
t_h = Loading and unloading time (min/pass)

The machining optimization model

❖ **The machining optimization model**

- **Constraints during machining:**
  - The bound of parameters depth of cut, feed rate and cutting speed:

    (i) $doc_{\min} \leq doc \leq doc_{\max}$

    (ii) $f_{\min} \leq f \leq f_{\max}$

    (iii) $V_{\min} \leq V \leq V_{\max}$

  - Constraints imposed on required machining features:

    I. Cutting force(preventing tool chatter or deflection)

    $$F = 844\ V^{-0.10133}\ f^{0.725}\ doc^{0.75} \leq F_{\max}$$

    II. Cutting power

    $$P = 0.0373\ V^{0.91}\ f^{0.78}\ doc^{0.75} \leq P_{\max}$$

    III.Working temperature

    $$74.96\ V^{0.4}\ f^{0.2}\ doc^{0.105} \leq \theta_{\max}$$

    IV.Surface roughness

    $$14.785\ V^{-1.52}\ f^{1.004}\ doc^{0.25} \leq R_{\max}$$

methodology

❖ **LXPM: A real coded GA**

- **Chromosomes representation**

  Decision variables are encoded as real numbers

  Individual:$X^1 = ({x_1}^1, {x_2}^1, {x_3}^1, \dots, {x_n}^1)$

- **Laplace crossover**

  Parents: $X^1 = ({x_1}^1, {x_2}^1, {x_3}^1, \dots, {x_n}^1)$ and $X^2 = ({x_1}^2, {x_2}^2, {x_3}^2, \dots, {x_n}^2)$

  Offspring: $Y^1 = ({y_1}^1, {y_2}^1, {y_3}^1, \dots, {y_n}^1)$ and $Y^2 = ({y_1}^2, {y_2}^2, {y_3}^2, \dots, {y_n}^2)$

  A random number $\beta_i = \begin{cases} a - b\log(u_i), r_i \le 0.5; \\ a + b\log(u_i), r_i \ge 0.5, \end{cases}$

  Where a is *a* location parameter and $b > 0$ is a scaling parameter, uniform random numbers u_i, r_i $\in [0,1]$

  Crossover operation:

$$y_i^1 = x_i^1 + \beta_i |x_i^1 - x_i^2| \qquad y_i^2 = x_i^2 + \beta_i |x_i^1 - x_i^2|$$

methodology

❖ LXPM: A real coded GA

- **Power mutation:**

  Let $\bar{x}$ be a parent solution then x is a mutated solution.

$$x = \begin{cases} \bar{x} - s(\bar{x} - x^l), & t < r; \\ \bar{x} + s(x^u - \bar{x}), & t \geq r. \end{cases}$$

  r are uniform random number ∈ $[0,1]$, s follow the power distribution $s = (s1)^p$, where s_1 is a random number between 0 and 1,p is the index of mutation. $t = \frac{\bar{x}-x^l}{x^u-\bar{x}}$ and x^l, x^u being the lower and upper bound of Decision variables.

- **Constraint handling**

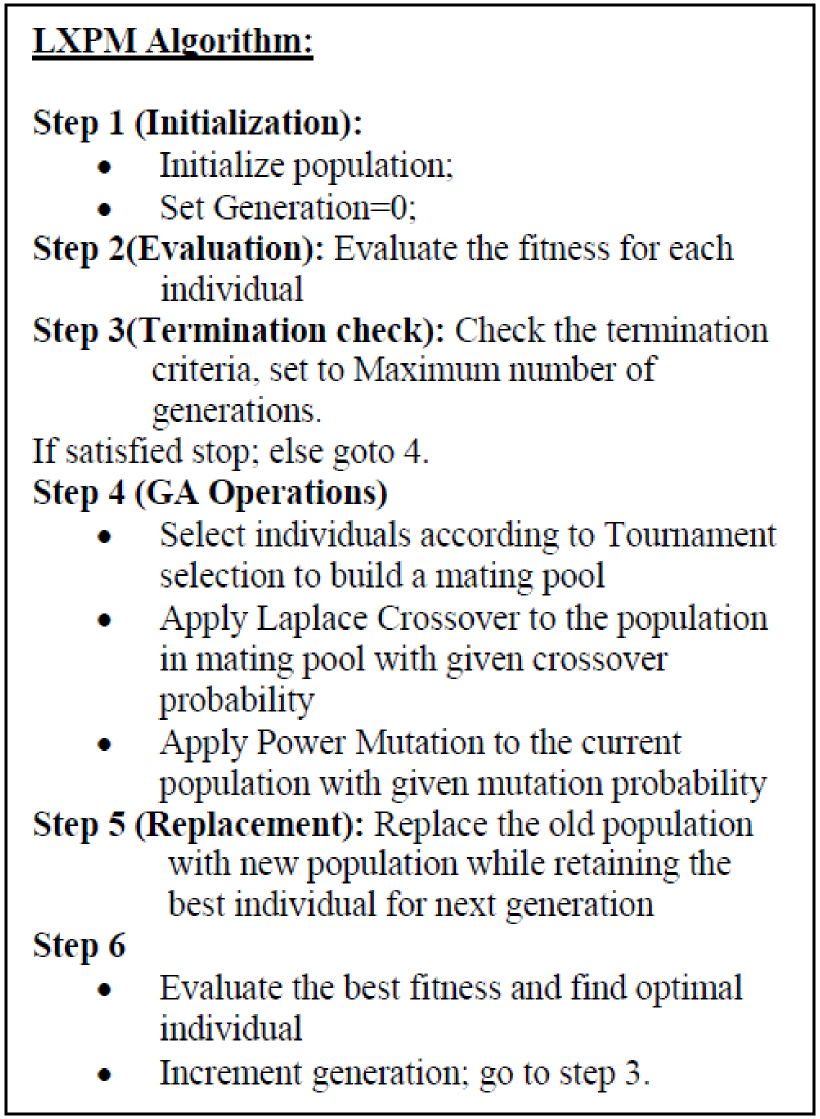
  Penalty function:

$$fitness\ (X_i) = \begin{cases} f(X_i), & if\ X_i\ is\ feasible \\ f_{worst} + \sum_{j=1}^{m} \left| \Phi_j(X_i) \right|, & otherwise \end{cases}$$

  F_{worst} is the objective value of the worst feasible solution currently available in the population. $\varphi_j(x_i)$ refer to value of the left hand side of the inequality constraint($\varphi_j(x_i) = g_j(x_i)$).

$$g_j(\vec{x}) \geqslant 0, \quad j = 1, \ldots, J,$$

methodology

❖ 

LXPM Algorithm:

Step 1 (Initialization):
- Initialize population;
- Set Generation=0;

Step 2(Evaluation): Evaluate the fitness for each individual

Step 3(Termination check): Check the termination criteria, set to Maximum number of generations.

If satisfied stop; else goto 4.

Step 4 (GA Operations)
- Select individuals according to Tournament selection to build a mating pool
- Apply Laplace Crossover to the population in mating pool with given crossover probability
- Apply Power Mutation to the current population with given mutation probability

Step 5 (Replacement): Replace the old population with new population while retaining the best individual for next generation

Step 6
- Evaluate the best fitness and find optimal individual
- Increment generation; go to step 3.

- **The other parameter setting**
  - a. Population = D*10
  - b. Generation =200
  - c. Run = 100
  - d. Crossover rate(CR) = varies from 0.86 to 0.9
  - e. Mutation rate(p_m) = varies from 0.006 to 0.06

methodology

❖ Differential evolution (DE)

- **Mutation**
- Produce a trial vector $u_i(t)$ corresponding to each individual of the current population by mutating a target vector $X_i(t)$ with a weighted differential. This trial vector is then used by crossover operator to produce offspring.

  Select a target vector X_{i_1} from population, random select two individuals X_{i_2}, X_{i_3} ,such that $i \neq i_1 \neq i_2 \neq i_3$. The trial vector is calculated as:

$$u_i(t) = X_{i_1}(t) + \beta(X_{i_2}(t) - X_{i_3}(t))$$

  Where $\beta \in (0, i+)$ is the scale factor.

smaller value of β leads to smaller step sizes that increases the computational time of algorithm, on the other hand the larger value of β provides faster convergence but may result in premature convergence.

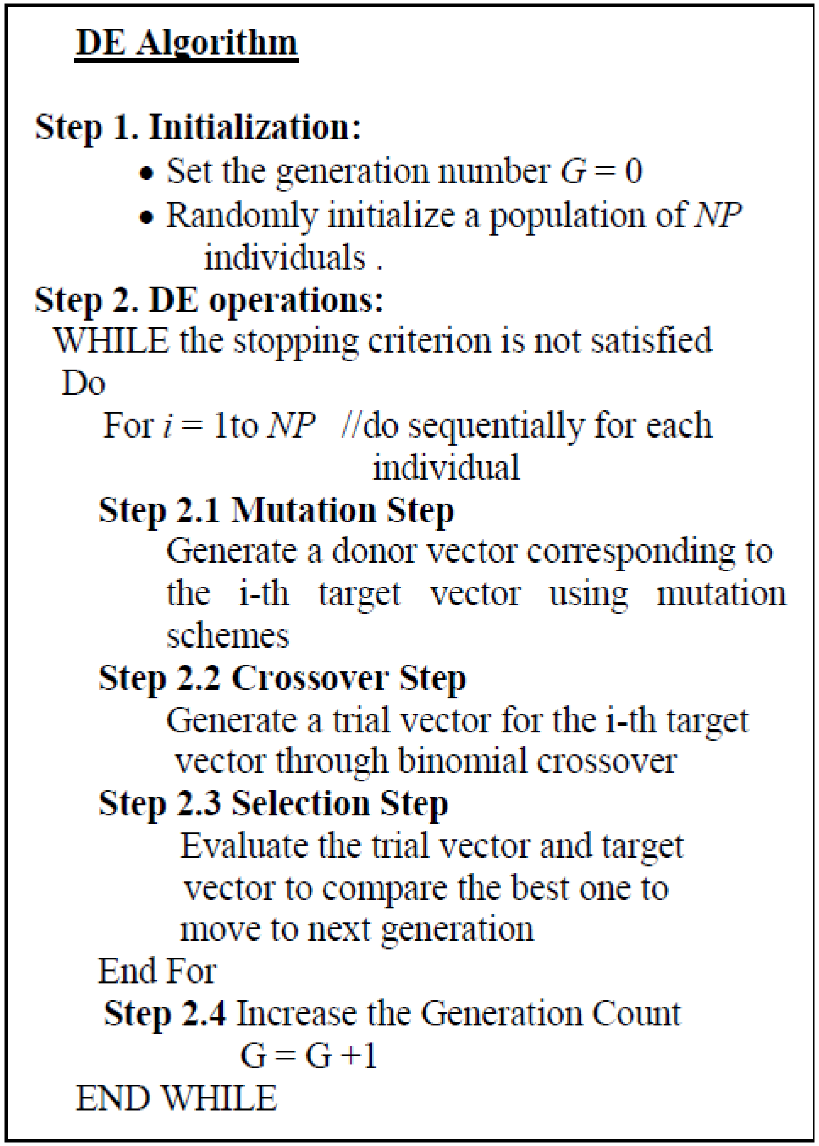
- **Crossover**
- Combines the trial vector $u_i(t)$ and the parent vector $X_i(t)$, to produce offspring as :

$$X'_{ji}(t) = \begin{cases} u_{ji}(t) & if\ randb(j) \leq CR\ \ or\ \ j = rnbr(i) \\ X_{ji}(t) & if\ randb(j) > CR\ \ or\ \ j \neq rnbr(i) \end{cases}$$

- Where $randb(j) \in [0,1]$ is the jth evaluation of random number generator.$rnbr(i)$ is a randomly chosen index $\in [1,2,\dots,d]$, which ensures that offspring, has at least one component from trial vector $u_i(t)$.
- **Selection**
- Selection operator decides which individual should be forwarded to next generation.

$$X_i(t+1) = \begin{cases} X'_i(t) & if\ f(X'_i(t)) \leq f(X_i(t)) \\ X_i(t) & otherwise \end{cases}$$

methodology

❖ 

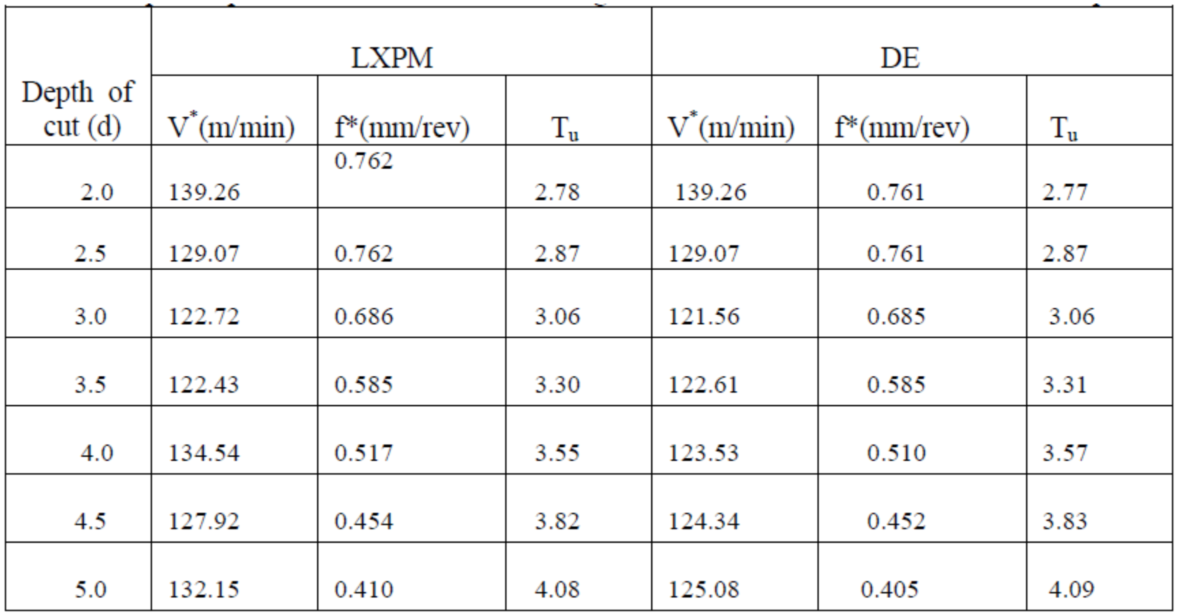
```
DE Algorithm

Step 1. Initialization:
        • Set the generation number $G = 0$
        • Randomly initialize a population of $NP$
          individuals .
Step 2. DE operations:
  WHILE the stopping criterion is not satisfied
  Do
     For $i = 1$to $NP$   //do sequentially for each
                            individual
     Step 2.1 Mutation Step
          Generate a donor vector corresponding to
          the i-th target vector using mutation
          schemes
     Step 2.2 Crossover Step
          Generate a trial vector for the i-th target
          vector through binomial crossover
     Step 2.3 Selection Step
          Evaluate the trial vector and target
          vector to compare the best one to
          move to next generation
     End For
     Step 2.4 Increase the Generation Count
               G = G +1
  END WHILE
```

- **The other parameter setting**
  a. Population = D*10
  b. Generation =200
  c. Run = 100
  d. Scaling factor(β) = 0.5
  e. Crossover rate(CR) = 0.08

Computational Results And Comparisons

❖ Computational experiment

- Experiment environment
  - VC++ on Celeron PC
  - 1.4 GHz, 1.256 GB RAM
- best parameter value obtained using LXPM and DE for different values of depth of cut

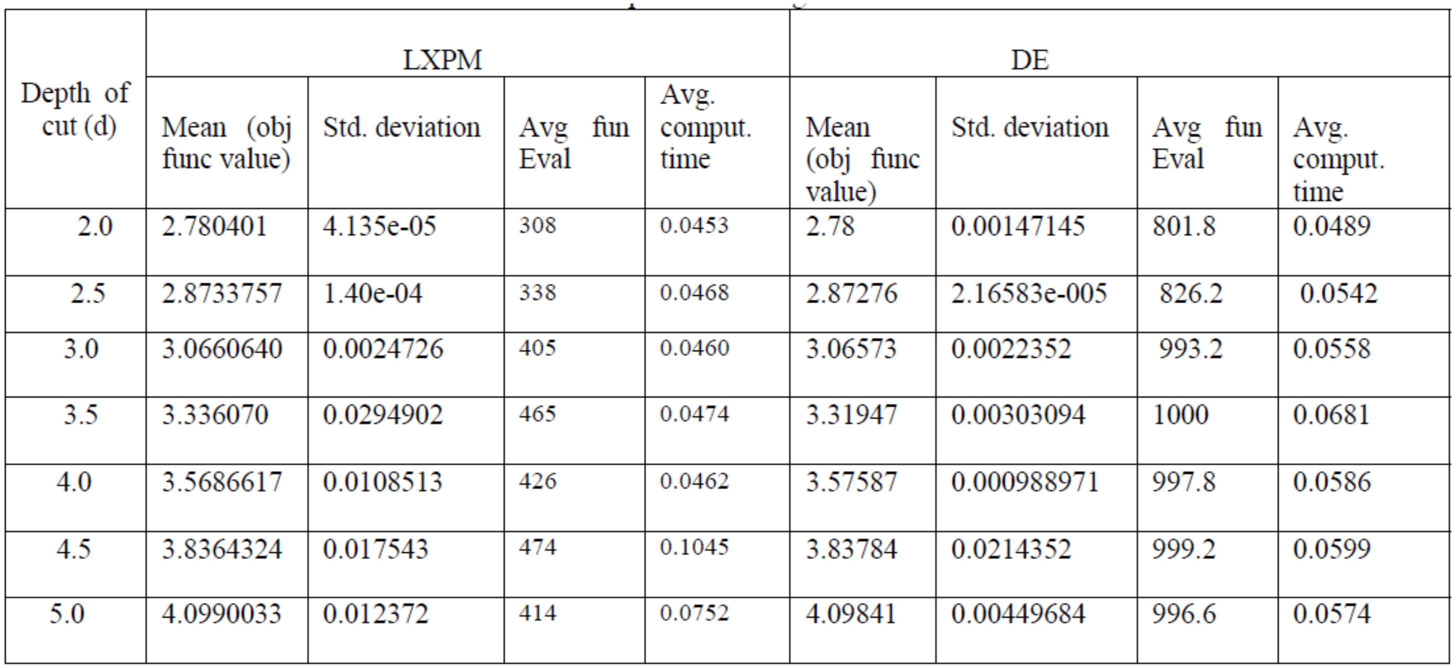


Depth of cut (d)	LXPM			DE		
	V^*(m/min)	f*(mm/rev)	T_u	V^*(m/min)	f*(mm/rev)	T_u
2.0	139.26	0.762	2.78	139.26	0.761	2.77
2.5	129.07	0.762	2.87	129.07	0.761	2.87
3.0	122.72	0.686	3.06	121.56	0.685	3.06
3.5	122.43	0.585	3.30	122.61	0.585	3.31
4.0	134.54	0.517	3.55	123.53	0.510	3.57
4.5	127.92	0.454	3.82	124.34	0.452	3.83
5.0	132.15	0.410	4.08	125.08	0.405	4.09

Computational Results And Comparisons

❖ Computational experiment

- An Analysis for LXPM and DE for 100 runs

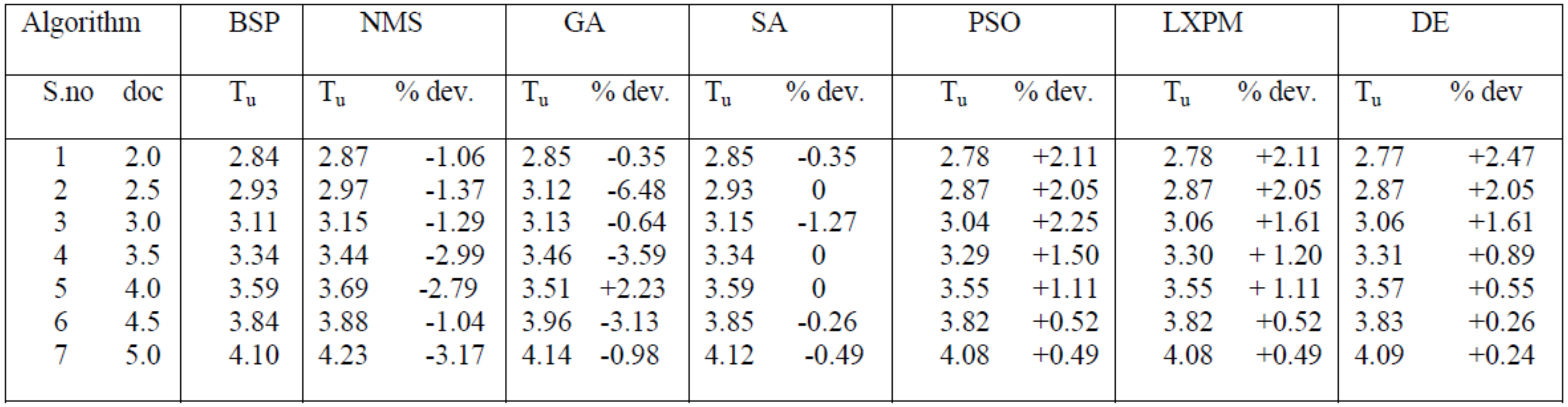


Depth of cut (d)	LXPM				DE			
	Mean (obj func value)	Std. deviation	Avg fun Eval	Avg. comput. time	Mean (obj func value)	Std. deviation	Avg fun Eval	Avg. comput. time
2.0	2.780401	4.135e-05	308	0.0453	2.78	0.00147145	801.8	0.0489
2.5	2.8733757	1.40e-04	338	0.0468	2.87276	2.16583e-005	826.2	0.0542
3.0	3.0660640	0.0024726	405	0.0460	3.06573	0.0022352	993.2	0.0558
3.5	3.336070	0.0294902	465	0.0474	3.31947	0.00303094	1000	0.0681
4.0	3.5686617	0.0108513	426	0.0462	3.57587	0.000988971	997.8	0.0586
4.5	3.8364324	0.017543	474	0.1045	3.83784	0.0214352	999.2	0.0599
5.0	4.0990033	0.012372	414	0.0752	4.09841	0.00449684	996.6	0.0574

Computational Results And Comparisons

❖ Computational experiment

- An Analysis for LXPM and DE for 100 runs



Algorithm		BSP	NMS		GA		SA		PSO		LXPM		DE	
S.no	doc	T_u	T_u	% dev.	T_u	% dev.	T_u	% dev.	T_u	% dev.	T_u	% dev.	T_u	% dev
1	2.0	2.84	2.87	-1.06	2.85	-0.35	2.85	-0.35	2.78	+2.11	2.78	+2.11	2.77	+2.47
2	2.5	2.93	2.97	-1.37	3.12	-6.48	2.93	0	2.87	+2.05	2.87	+2.05	2.87	+2.05
3	3.0	3.11	3.15	-1.29	3.13	-0.64	3.15	-1.27	3.04	+2.25	3.06	+1.61	3.06	+1.61
4	3.5	3.34	3.44	-2.99	3.46	-3.59	3.34	0	3.29	+1.50	3.30	+ 1.20	3.31	+0.89
5	4.0	3.59	3.69	-2.79	3.51	+2.23	3.59	0	3.55	+1.11	3.55	+ 1.11	3.57	+0.55
6	4.5	3.84	3.88	-1.04	3.96	-3.13	3.85	-0.26	3.82	+0.52	3.82	+0.52	3.83	+0.26
7	5.0	4.10	4.23	-3.17	4.14	-0.98	4.12	-0.49	4.08	+0.49	4.08	+0.49	4.09	+0.24

- **Analysis :**

  LXPM, PSO and DE perform better than other algorithms on the considered model.

  for different values of depth of cut, LXPM and DE give significant improvement over Binary GA and other methods such as NMS, BSP, GA and SA.

*BSP(Boundary Search Procedure)
NMS(Nelder-Mead Simplex Method)
Binary GA,
SA (Simulated Annealing)
PSO (Particle Swarm Optimization).

Conclusion

- **Optimizing machining parameters for turning process**
  - Objective function is minimizing processing time
  - Take over the constrained machining environment ensure machining performance and product quality
- **Suggested**
  - A real coded genetic algorithm
  - Differential evolution algorithms
- **Adv & Disadv**

THANK YOU