Software Development for Setup Planning of Rotational Part

Sumit Dwivedi*[1] and Shahnawaz Alam[2]

ABSTRACT

An innovative approach was developed to solve the problem of setup planning, which is the most critical problem in process planning for discrete metal parts. Setup planning is the act of preparing detailed work instructions for setting up a part. The major objective of this research is to improve the performance of CAPP systems by developing a systematic approach to generate practical setup plans based on tolerance analysis. A comprehensive literature review on tolerance control in CAPP was conducted. It was found that tolerance chart analysis, a traditional tolerance control technique, is reactive in nature and can be greatly improved by solving the problem of setup planning. In order to develop a theoretically sound foundation for tolerance analysis-based setup planning, the problem of tolerance stack up in NC machining was analyzed in terms of manufacturing error analysis. Guidelines for setup planning were then developed based on the analysis. To systematically solve the setup planning problem, a graph theoretical setup planning algorithm for rotational parts was then developed for automated and integrated setup planning and fixture design. Its efficiency and effectiveness evaluated. The result is promising. The algorithms were then computerized. A setup planning program was developed under the Microsoft Windows environment using C.

***Keywords**: Setup Planning; CAPP; NC machining*

1. INTRODUCTION

1.1 Automation and Machining Planning

Machining planning is an important link between design and manufacturing. It is responsible for successful and efficient translation of design information into products. Automated design and manufacturing with intensive use of computers resulting in CAD (Computer Aided Design) and CAM (Computer Aided Manufacturing) respectively have been important developments of the last three decades. Recently, over last one decade, automation of machining planning has emerged as a bridge between CAD & CAM.

Automated machining has been reported to offer several advantages including shorter lead time, higher quality of machined part, lower part cost and more flexibility in machining planning. The machining planning as the planning activity performed between product drawing and actual machining is complex and ill defined. Machining planning, process planning, process engineering, and machine routing are some of the titles given to the same activity.

Machining planning is the systematic determination of the methods and means by which a product is to be manufactured economically. The machining planning involves several or all of the following functions:

i) **Blank Selection:** Given the finished part geometry, a raw material form must be selected.

ii) **Feature Recognition:** The final part geometry is analyzed to identify various features for machining.

1*. Sumit Dwivedi Department of Mechanical Engineering, B.N.College of Engineering and Technology, Lucknow, Uttar Pradesh, India Email-smtdwvd@gmail.com

2. Shahnawaz Alam, Ph.D., Department of Mechanical Engineering, Integral University, Lucknow, Uttar Pradesh, India, Email-sulehaq@gmail.com

iii) **Operation Selection:** For each feature in the finished part, a set of machining operation, capable of producing it economically must be selected.

iv) **Operation Sequencing:** The order of applying these operations must be determined. This sequencing is influenced by several factors such as accessibility, setup and tolerances.

v) **Machine and Cutting Tool Selection:** For each machining operation or a group of machining operations to be performed, a machine and a cutting tool must be selected.

vi) **Setup Planning:** A series of orientations of the work piece, locating faces and feature to be produced in each work piece orientation must be determined.

vii) **Picture Design / Selection:** For each setup detailed fixture configuration must be designed, and / or a suitable set of fixture from the available list should be selected.

viii) **Operation Parameter Selection:** Operation parameter such as feed rate, spindle speed and depth of cut, must be selected. These can be selected from machining data hand books.

ix) **NC Code Generation and Part Programming:** NC codes to drive NC machines must be generated

1.2 Importance of Automated Maching Planning

Presently parts, which qualify mass production is not more than 25% of the total value of metal working production in all industrialized metal working nations .One half of the remaining 75% is produced in job or lots production. The no. of these parts is increasing each year and their complexity and requirement for accuracy are also increasing. Social and technological trends such as demands for customized products, shorter products lives, higher reliability of products, along with superior process tolerances and a wider variety of materials, are some of driving forces behind the need of smaller batch sizes.

Accordingly industry is leaning toward flexible manufacturing system (FMS) to meet the demand of smaller batch sizes and the ever in increasing demand for productivity improvement.

As FMS advances, less people are seen on shop floor, but in office next to shop floor, many people are seen working in machining planning in a labor intensive way. The manual machining is flexible to disturbances such as engineering changes and emergency orders. The flexibility of FMS manual machining planning has another problem, since process planning is based on the previous experiences of planner personal preference, extent of shop knowledge interpretation of design requirements and other judgment factors, process plans are often inaccurate, inconsistent and faulty.

Automating this labor intensive machining planning offers benefits of more than just saving labor cost of manual machining planning. More importantly, it offers the possibility of shorter lead time, higher quality of machined parts, lower parts cost, scheduling flexibility and flexibility to disturbances.

1.3 Setup Planning

Setup planning is defined as how to orient and fix a part so that all the features on the part can be machined successfully with required tolerance and surface finish. Setup planning is the critical bridge between general process planning and detailed operation planning. It is also the intimate upstream of fixture planning. In earlier works, setup planning has been carried out mostly based on tool approach direction.

The purpose of setup is to locate and fix a part in a definite manner on a machine tool so that machining can take place. The theoretically exact point, axis, or plan used to locate the part is referred to as a setup datum. When setups and setup datum of a process plan are not selected properly, a tolerance chart analysis might find that the process plan cannot guarantee parts to be made within the specified tolerance. Hence tolerance information from part decision should be taken into consideration in setup planning.

1.4 Objective and Scope of the Present Work

The motivation of this work is automation of process planning for machining, which is generally performed manually. In spite of many research efforts in the past towards the automatic generation of the plans for machining, setup planning and fixture design is one of the least studied areas and it remains a major missing link in automated machining planning.

Integration of process planning with automated setup planning and fixture design is one of the major areas, requiring immediate and apt attention of the researcher in this field. At present, almost most of the computer aided process planning (CAPP) system do not have the facility of automatic setup planning and fixture designing, as this part of CAPP is quite complicated and carries a lot of bottleneck needed to be overcome. Hence, automated setup planning and fixture designing stands as major research issues in the context of process planning.

In the present work an algorithm based on graph theory concepts has been developed for automated and integrated setup planning and fixture design. The work has been divided into two parts: setup planning and fixture design which are then integrated. In earlier works, setup planning has been carried out mostly based on tool approach direction. Tolerance relations are taken in to consideration after selection of setup plan. In this case selected set up plan(s) is (are) not competent to offer required tolerances then the effort may go waste. In addition if such selected setups plans are used then generally machine tools with very high precision are needed, which directly increase the production cost. In the present work, tolerance relation has been considered during selection of setup planning .Setup planning consists of following task:

1. Setup formation- All the machining features are grouped together according to their tool approach direction and tolerance relations.
2. Datum selection- For each setup, datum surfaces (primary, secondary, and tertiary) are selected,

3. Setup sequencing- Selected setups are sequenced on the basis of number of features and tolerance relations.

This approach gives more comprehensive setup plan. Basic research issues involve in fixture design are the selection of locating and clamping positions for a given work piece along with its setup position(s), in order to achieve accurate locating, and total restraint of the work piece, no interference between fixture, work piece and the cutting tool, and goodness of the design of the fixture.

Fixture design consists of the following task:-

1. Selection of locating surface,
2. Selection of clamping surface,
3. Selection of locating position,
4. Selection of clamping position

2. PROPOSED ALGORITHM

A. Setup Planning Algorithm For Rotational Parts

In this section, a setup planning algorithm for rotational parts is developed and implemented. Because secondary features such as chamfers and keyways do not influence setup planning; only primary features are considered. Primary features of a rotational part are cylindrical surface and vertical surface. it is assumed that (1) the total approach direction of each feature is given ,(2) each setup requires one (and only one) vertical feature and one (and only one) cylindrical feature as locating datum, and (3) a feature may or may not exist on the stock. The following notations are adopted

n = no. of features within the part

A_i = tool approach direction, defined as follows:

$$A_i = \begin{cases} 1, \text{ if feature i can be machined only from the left} \\ 2, \text{ if feature I can be machined only from the right} \end{cases}$$

0, otherwise

in which i=1, 2,…,n

E_i feature type, defined as follows:

$$E_i = \begin{cases} 1, & \text{if feature } i \text{ is a vertical feature} \\ 2, & \text{if feature } i \text{ is a cylindrical feature} \\ 0, & \text{otherwise} \end{cases}$$

in which i=1, 2,…, n

S_i stock shape, defined as follows:

$$S_i = \begin{cases} 1, & \text{if feature } i \text{ exits on the stock} \\ 0, & \text{otherwise} \end{cases}$$

In which i=1, 2, …, n

T= [t_{ij}] tolerance matrix, defined as follows:

$$t_{ij} = \begin{cases} ä_{ij}, & \text{if features } i \text{ and } j \text{ have tolerance } ä_{ij} \\ 0, & \text{otherwise} \end{cases}$$

in which i = 1, 2, …,n
j = 1, 2, …,n,

The algorithm is as follows:

Step 1- Setup Formation

```
Find $t_{pq} = \min [t_{ij}]$, in which $i = 1, 2, …,n; j =1,2, …,n; A_i = 0; A_j \neq 0; t_{ij} \neq 0$
If no such $t_{pq}$ can be found then
for i = 1 to n do
if $A_i$=0 then
let $l= \sum_{j=1}^{n} X_j$, in which $X_j = \begin{cases} 1, & \text{if } A_j=1 \\ 0, & \text{otherwise} \end{cases}$
let $r= \sum_{j=1}^{n} X_j$, in which $X_j = \begin{cases} 1, & \text{if } A_j=2 \\ 0, & \text{otherwise} \end{cases}$
if l< r then
let $A_i$=1
else
        go to step 2
else
        let $A_p=A_q$
        let $t_{pq}=0;t_{qp}=0$
        repeat step 1
```

Step 2-Datum Section

```
Step 2.0
let $t_{ij}$=0 if $A_i=A_j$, in which i= 1,2,….,n;  j=1,2,…,n
let $d_{ij}$=0, in which i=1,2,…n;   j=1,2,…,n
let T '=[$t_{ij}$]=T
Step 2.1
find $t_{pq}$'=min [$t_{ij}$'], in which i=1,2,..n; j=1,2,…..,n; $t_{ij}' \neq 0$
if no such $t_{pq}$' can be found then
for i=1 to 2 do
for j=1 to 2do
if $d_{ij}$=0 then
find feature w such that $A_w$=i &$E_w$=j
let $d_{ij}=1;D_{ij}=w$
go to step 3
else
        let $t_{pq}'=0$; $t_{qp}'=0$
        if $d_{ApEp}$=0 then
        let $d_{ApEp}$=1
        $D_{ApEp}$=p
        if $d_{AqEq}$=0 then
        let $d_{AqEq}$=1
        $D_{AqEq}$=q

if $\sum_{i=1}^{2} \sum_{j=1}^{2} d_{ij}=4$ then
go to step 3
else
        go to step 2.1
```

Step 3- Setup Sequencing

```
for i=1 to 2 do
for j=1 to 2 do
```

$$\text{let } R_{ij} = \sum_{k=1}^{n} x_k, \text{ In which } x_k = \begin{cases} 1, \text{if } t_{Dijk} ?\ 0 \\ 0, \text{ otherwise} \end{cases}$$

find $R_{pq}=\max[R_{ij}]$

$$\text{let } Z = \begin{cases} 1, \text{ if } p=2 \\ 0, \text{ otherwise} \end{cases}$$

Step 4 –Generate Setup Plan Based on Stock Shape

```
Let c=0
    if $S_{DZ1} \neq 1$ then
        find feature f so that $A_f=Z$, $E_f=1$, &$S_f=1$
        if such a feature cannot be fo und then
        let c= 1
    else
        let $D_{Z1}=f$
    if $S_{DZ2} \neq 1$ then
        find feature f so that $A_f=Z$, $E_f=2$, & $S_f=1$
        if such a feature cannot be fo und then
   let c=1
else
   if $c \neq 1$ then
   let $D_{Z2}=f$
Let $S_l = \varnothing$ ; $S_r = \varnothing$
Let $\mathbf{S_L}=\mathbf{S_L} \cup \{i\}$ if $A_i=1$, in which i=1,2,…..,n
Let $\mathbf{S_R}=\mathbf{S_R} \cup \{i\}$ if $A_i=2$, in which i=1,2,…..,n
   if c= 1 then
   Machine features $D_{Z1}$& $D_{Z2}$
if z=1 then
```

Features within S_R are to be machine in the first setup using features D_{11} & D_{12} as the locating datum. Features within S_L are to be machine in the seco nd setup using features D_{21} &D_{22} as the locating datum.
else
features within S_L are to be machined in the fir st setup using features D_{21} &D_{22} as the locating datum; featur es with in S_R to be machined in the second s etup using features D_{11}&D_{12} as the locating datum.

The algorithm can deal with di fferent type of rotational parts .It was implemented using C++ under the Microsoft Windows environment .The input for the software is-
(1) The no. of features
(2) Features type of each feat ures
(3) Tool approach directions o f each feature
(4) Stock shape information
(5) Tolerance information.
The output is a brief descript ion of how to setup the part.

B. Software of Setup Planning for Rotational Part

```
//Software For Setup Planning of Rotational Part//

 #include<stdio.h>
 #include<stdlib.h>
 #include<conio.h>

void main()
{
        int i,j,k,p,q,n,R_max,l,r,f,f_ flag,z,w,c,SL_size,
        SR_size,nam, tot;
        int A[20],A1[20],E[20],s[20],x [20],
        SL[20], SR[20],clam[20];
        int d[10][10],D[10][10],R[10][ 10];
        clrscr();

        float t[10][10],tt[10][10];
        float min_t,min_tt;
  int a,b,comp,cmp,ch,g;
   float e;
do
{  clrscr();
printf("\n******************** *********\n\n");
printf ("Before runnining the  software one should have following informations-\n\n");
printf ("1. No of features Pre sent in Stock.\n2. Type of feature  (Vertical/Cylindrical/Other).\n3. Tool approach
Direction for each feature  (Left/Right/Both).\n4.
Stock Information  (Present/Not present).\n5.
Tolerance
Matrix.\n6. Numbering of LEFT and RIGHT side
features should be in order separately.\n\n");
printf("******************************\n\n");
          printf("Enter NO. of features present in
          stock/job\n");
          scanf ("%d",&n);
          printf ("\nEnter feature type of each
          feature.\n (Vertical= 1,Cylindrical= 2,
          Other= 0) \n");
          for(i=1;i<=n;i++)
          {
                    scanf("%d",&E[i]);
          }
          printf("\nEnter tool approach direction for
          each feature.\n (Left= 1, Right= 2, Both Side=
          0) \n");
          for(i=1;i<=n;i++)
          {
                    scanf("%d",&A[i]);
                    A1[i]=A[i];
          }
          printf("\nEnter stock shape information for
          each feature.\n(If Feature Present=1, If Not
          Present=0) \n");
          for(i=1;i<=n;i++)
          {
                    scanf("%d",&s[i]);
          }
          for(i=1;i<=n;i++)
          {
          for(j=1;j<=n;j++)
          {
          t[i][j]=0;
          }

          }
do
{
printf("\nPress 1 TO ENTER TOLERANCE
MATRIX ELEMENTS.\n(One may SEE and EDIT
the matrix during process).");
printf("\n\nPress 2 TO DISPALY PRESENT
TOLERANCE MATRIX\n");
scanf("%d",&ch);
switch(ch)
{
    case 1:
    do
    {
    printf("\nEnter location(a=Row,b=Column) for
    tolerance value in tolerance matrix\n");
    scanf("%d%d",&a,&b);
    printf("Enter Tolerance value for above
    location\n");
    scanf("%f",&e);
     t[a][b]=e;
     t[b][a]=e;
    printf("\nDo you want to enter more
    elememts.\n IF YES=pres 5, IF NO=press any
    other number.\n");
    scanf("%d",&g);
   }while(g==5);
    break;
    case 2:
    printf("Present Tolerance Matrix is- \n");
          for(i=1;i<=n;i++)
          {
 for(j=1;j<=n;j++)
           {
            printf("%0.2f \t",t[i][j]);
           }
          printf("\n");
          }
     break;

default:
          printf("WRONG CHOICE\n");
}
printf("\nTo SEE and EDIT the tolerance matrix =
PRESS 5\nTo get final SETUP PLANNING output
= PRESS ANY NUMBER\n");
scanf("%d",&cmp);
}while(cmp==5);
```

```
printf("                  * FINAL RESULT *\n");

printf("\nFeatures present in Stock/Job\n");
            printf("%d",n);
            printf("\n\nFeature type of each feature.\n
           (Vertical= 1,Cylindrical= 2, Other= 0) \n");
            for(i=1;i<=n;i++)
            {
                        printf("%d \t",E[i]);
            }
            printf("\n\nTool approach direction for each
           feature.\n (Left= 1, Right= 2, Both Side= 0)
           \n");
            for(i=1;i<=n;i++)
            {
                        printf("%d \t",A[i]);

            }
            printf("\n\nStock Shape information for
           each feature.\n(If Feature Present=1, If Not
           Present=0) \n");
            for(i=1;i<=n;i++)
            {
                        printf("%d \t",s[i]);
           }
printf("\n\nFinal Tolerance Matrix \n");
            for(i=1;i<=n;i++)
            {
             for(j=1;j<=n;j++)
              {
               printf("%0.2f \t",t[i][j]);
              }
     printf("\n");
            }
      break;

default:
            printf("WRONG CHOICE\n");
}
printf("\nTo SEE and EDIT the tolerance matrix =
PRESS 5\nTo get final SETUP PLANNING output
= PRESS ANY NUMBER\n");
scanf("%d",&cmp);
}while(cmp==5);
printf("                  * FINAL RESULT *\n");

printf("\nFeatures present in Stock/Job\n");
            printf("%d",n);
            printf("\n\nFeature type of each feature.\n
(Vertical= 1,Cylindrical= 2, Other= 0) \n");
            for(i=1;i<=n;i++)
            {
                        printf("%d \t",E[i]);
            }
printf("\n\nTool approach direction for each feature.\n
(Left= 1, Right= 2, Both Side= 0) \n");
            for(i=1;i<=n;i++)
            {
                        printf("%d \t",A[i]);
            }
            printf("\n\nStock Shape information for each
           feature.\n(If Feature Present=1, If Not
           Present=0) \n");
            for(i=1;i<=n;i++)
            {
                        printf("%d \t",s[i]);
            }
printf("\n\nFinal Tolerance Matrix \n");
            for(i=1;i<=n;i++)
            {
             for(j=1;j<=n;j++)
              {
               printf("%0.2f \t",t[i][j]);
              }
     printf("\n");

            }

            step1 : min_t=10;
            for(i=1;i<=n;i++)
            {
              for(j=1;j<=n;j++)
               {
                        if((min_t>t[i][j])&&(t[i][j]!=0)&&
                        (A[i]==0) &&(A[j]!=0))
                        {
```

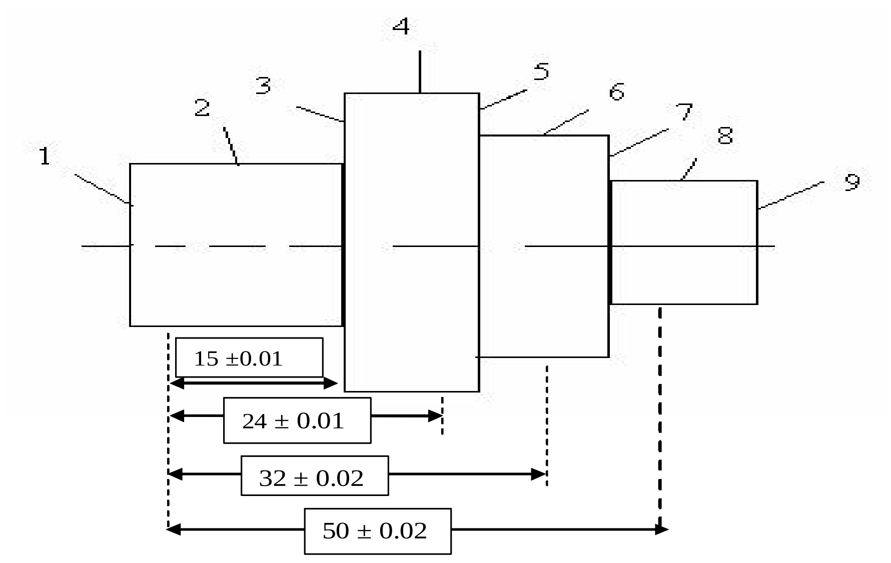
```
                    min_t=t[i][j];
                    p=i;q=j;
                  }
              }
          }
          if(min_t==10)
          {
                for(i=1;i<=n;i++)
                {
                      if(A[i]==0)
                      {
                            l=0;r=0;x[0]=0;
                            x[1]=1;x[2]=0;
                            for(j=0;j<n;j++)
                            {
                                  l=l+x[A[j]];
                            }
                            x[1]=0;x[2]=1;
                            for(j=0;j<n;j++)
                            {
                                 r=r+x[A[j]];
                            }

                            if(l<r)
                            A[i]=1;
                            else
                            A[i]=2;
                      }
                }
                goto step2;

          }
          else
          {
                A[p]=A[q];
                t[p][q]=0;
                t[q][p]=0;
                goto step1;
          }
step2:
   for(i=1;i<=n;i++)
    {
       for(j=1;j<=n;j++)
```

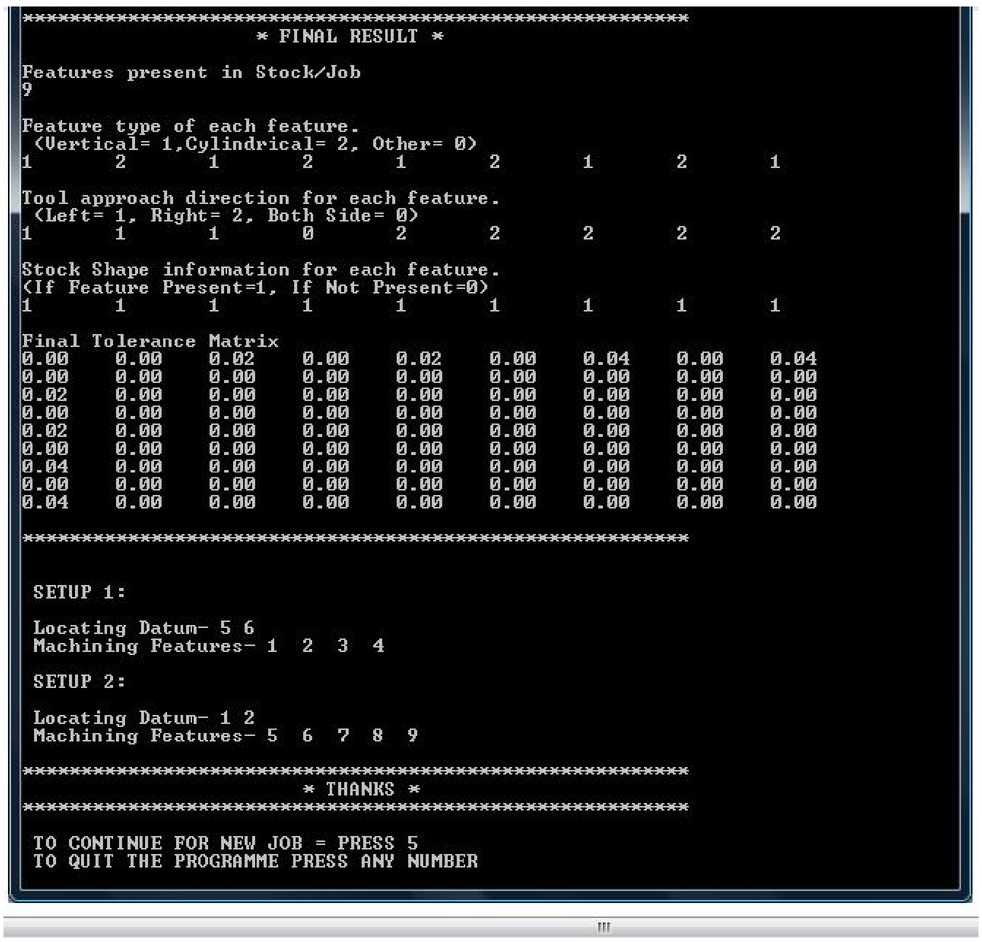
```
    {
          if(A[i]==A[j])
           {
                t[i][j]=0;
           }
          }
    }

d[1][1]=0;d[1][2]=0;d[2][1]=0;d[2][2]=0;
D[1][1]=0;D[1][2]=0;D[2][1]=0;D
[2][2]=0;
for(i=1;i<=n;i++)
{
 for(j=1;j<=n;j++)
  {
   tt[i][j]=t[i][j];
  }
}
step21:min_tt=10;
for(i=1;i<=n;i++)
{
  for(j=1;j<=n;j++)
   {
    if((min_tt>tt[i][j])&&tt[i][j]!=0)
      {
           min_tt=tt[i][j];
           p=i;q=j;
      }
    }
}


if(min_tt==10)
{
  w=0;l=0;
  for(i=1;i<=2;i++)
   {

    for(j=1;j<=2;j++)
     {
           if(d[i][j]==0)
           {
             for(k=1;k<=n;k++)
              {
                    if((A[k]==i)&&(E[k]==j))
                     {
                         w=k;
                         if((i==1)&&(j==2))
              {
                int w1,w2;
                w1=(D[i][j-1])+1;
                w2=(D[i][j-1])-1;
                if((w1==w)||(w2==w))
                 {
                   d[i][j]=1;
                              D[i][j]=w;
                   break;
                 }
               }
              else
              {
               int w1,w2;
                w1=(D[i][j-1])+1;
                w2=(D[i][j-1])-1;
                if((w1==w)||(w2==w))
                 {
                   d[i][j]=1;
                              D[i][j]=w;
                   break;
                 }

               }

                     }

                   }
                 }

             }
           }

          goto step3;
 }
 else
 {
        tt[p][q]=0;tt[q][p]=0;
        if(d[A[p]][E[p]]==0)
        {
         d[A[p]][E[p]]=1;
         D[A[p]][E[p]]=p;

        }
        if(d[A[q]][E[q]]==0)
        {
         d[A[q]][E[q]]=1;
         D[A[q]][E[q]]=q;
        }

        nam=d[1][1]+d[1][2]+d[2][1]+d[2][2];
        if(nam==4)
        goto step3;
 else
     goto step21;

   }

     step3:
    for(i=1;i<=2;i++)
          {
              for(j=1;j<=2;j++)
              {
                 if(D[i][j]!=0)
                  {
                      R[i][j]=0;
                      for(k=1;k<=n;k++)
                      {
                       if(t[D[i][j]][k]!=0)
                        x[k]=1;
                       else
                       x[k]=0;
                       R[i][j]+=x[k];
                      }
                  }
                 else
                  {
                      R[i][j]=0;
                  }
              }
          }
 R_max=R[1][1];
 for(i=1;i<=2;i++)
  {
        for(j=1;j<=2;j++)
        {
                if(R[i][j]>=R_max)
                {
                        R_max=R[i][j];
                        p=i;q=j;
                }
        }
  }
 if(p==2) z=1;
 else z=2;

 step4:c=0;
 if(s[D[z][1]]!=1)
  {
        f_flag=0;
        for(k=1;k<=n;k++)
        {
        if((A1[k]==z)&&(E[k]==1)
         &&(s[k]==1))
        {
                f=k;f_flag=f_flag+1;
          }
        }
        if(f_flag==0)
        c=1;
        else
        D[z][1]=f;
 }
if(D[z][2]!=0)
{
  if(s[D[z][2]]!=1)
   {
        f_flag=0;
        for(k=1;k<=n;k++)
        {
          if((A1[k]==z)&&(E[k]==2) &&
         (s[k]==1))
          {
                f=k;f_flag=1;
          }
        }
        if(f_flag==0) c=1;
        else
        {
        if(c!=1)
        D[z][2]=f;
        }
   }
  }
  else
  {
        for(k=1;k<=n;k++)
        {
        if((A1[k]==0)&&(E[k]==2))
                {

                D[z][2]=k;
                k=n;
                }
        }

  }

SL_size=1;
SR_size=1;
for(i=1;i<=n;i++)
{
 if(A[i]==1)
  {
   SL[SL_size++]=i;
 }
 else
 SR[SR_size++]=i;
}
SL_size—;
SR_size—;
if(c==1)
{
        printf("\nMachine features %d and
        %d",D[z] [1],D[z][2]);
}
if(z==1)
```

```
        {
printf("\n*****************************\n");
         printf("\n SETUP 1:  ");
         printf("\n Locating Datum- %d %d
\n",D[1][1],D[1][2]);

    if(D[1][2]==0)
     {
     printf("\n * NOT POSSIBLE TO LOCATE
THE STOCK/JOB *\n We have to add extra features
(cylindrical/vertical) or \n special Jig & Fixture to get
the desired tolerance limit.\n\n");

     }
    printf(" Machining Features-");

         for(i=1;i<=SR_size;i++)
          {
                printf(" %d ",SR[i]);
          }

         printf("\n\n SETUP 2:   ");
         printf("\n Locating Datum- %d %d
        \n",D[2][1],D[2][2]);
    if(D[2][2]==0)
     {
        printf("\n * NOT POSSIBLE TO LOCATE
THE STOCK/JOB *\n We have to add extra features
(cylindrical/vertical) or \n special Jig & Fixture to get
the desired tolerance limit.\n\n");


     }
    printf(" Machining Features-");

         for(i=1;i<=SL_size;i++)
         printf(" %d ",SL[i]);

        }
        else
        {
printf("\n*****************************\n");
        printf("\n\n SETUP 1:  \n");
           printf("\n Locating Datum- %d %d \n",D
           [2][1], D[2][2]);
           if(D[2][2]==0)
         {
           printf("\n * NOT POSSIBLE TO LOCATE
THE STOCK/JOB *\n We have to add extra features
(cylindrical/vertical) or \n special Jig & Fixture to get
the desired tolerance limit.\n\n");

         }
      printf(" Machining Features-");

            for(i=1;i<=SL_size;i++)
             {
                   printf(" %d ",SL[i]);
             }

            printf("\n\n SETUP 2: \n ");
            printf("\n Locating Datum- %d %d \n",D
           [1][1],D[1][2]);
        if(D[1][2]==0)
         {
           printf("\n * NOT POSSIBLE TO LOCATE
THE STOCK/JOB *\n We have to add extra features
(cylindrical/vertical) or \n special Jig & Fixture to get
the desired tolerance limit.\n\n");
         }
      printf(" Machining Features-");

            for(i=1;i<=SR_size;i++)
            printf(" %d ",SR[i]);

           }
printf("\n\n******************************");
printf("\n * THANKS *");
printf("\n******************************");
printf("\n\n TO CONTINUE FOR NEW JOB =
PRESS 5 \n TO QUIT THE PROGRAMME
PRESS ANY NUMBER\n");
scanf("%d",&comp);
}while(comp==5);
 getch();
}
```

3. EXPERIMENT AND RESULT

3.1 Input and output for rotational part





```
*********************************************************
                    * FINAL RESULT *

Features present in Stock/Job
9

Feature type of each feature.
 (Vertical= 1,Cylindrical= 2, Other= 0)
1       2       1       2       1       2       1       2       1

Tool approach direction for each feature.
 (Left= 1, Right= 2, Both Side= 0)
1       1       1       0       2       2       2       2       2

Stock Shape information for each feature.
(If Feature Present=1, If Not Present=0)
1       1       1       1       1       1       1       1       1

Final Tolerance Matrix
0.00    0.00    0.02    0.00    0.02    0.00    0.04    0.00    0.04
0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00
0.02    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00
0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00
0.02    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00
0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00
0.04    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00
0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00
0.04    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00

*********************************************************


 SETUP 1:

 Locating Datum- 5 6
 Machining Features- 1  2  3  4

 SETUP 2:

 Locating Datum- 1 2
 Machining Features- 5  6  7  8  9

*********************************************************
                        * THANKS *
*********************************************************

 TO CONTINUE FOR NEW JOB = PRESS 5
 TO QUIT THE PROGRAMME PRESS ANY NUMBER
```

Fig.1. Input and output for rotational part

4. CONCLUSION

In the present work and effort has been done towards the automation of the machining planning. The primary purpose of the work is to automate the setup planning and fixture design process to achieve automatic machining planning.

Setup planning is the critical bridge between general process planning and detailed operation planning. It is also intimate upstream of fixture planning. Setup planning significantly affects the overall cost & quality of part machining. In the present work graph theoretic approach has been used for automated setup planning for rotational parts. The advantages of the graph theoretic approach is that it is a mathematical approach & can be easily evaluated, modified & computerized. After selection of setups, a heuristic algorithm has been developed for selection of locating & clamping surfaces & locating & clamping positions for a given work piece. Thus the present work can be summarized as follows:

1. Setups have been formed on the basis of both tool approach direction & tolerance relation between features. This approach gives more comprehensive setup plan.
2. Datum features have been selected on the basis of tolerance requirement between features so that the specified tolerance requirement can be ensure while machining. A vertical & a cylindrical surface have been selected as a datum feature for machining rotational parts, for each setup. This approach ensures proper location of the part along the axis (x, y & z).
3. Setup has been sequenced on the basis number of tolerance relation between features of the different setups, as well as number of features in a particular setup has been considered.
4. After selection of sequence of setup, position of locators & clamps has been discussed based on heuristic algorithm .A clamp for each locator has taken. The features, which are to be machined has not been selected for clamping.

REFERENCE

[1] Halevi, G. and Weill, R.D. 1995. Principles of Process Planning: A logical approach. Chapman & Hall, England.

[2] Mei, J., Zhang, H.C. and Oldham, W.J.B. A neural network approach for datum selection. Computers in Industry, 1995, 27, 53-64.

[3] Huang, S.H. and Zhang, H. Tolerance analysis in setup planning for rotational parts. Journal of Manufacturing Systems, 1996, 15, 340-350.

[4] *Kurian K. Thomas, Gray W. Fischer,* “Integrating CAD/CAM soft ware for process planning application”, Journal of Material Processing Technology, 61, 1996, PP.87-92.

[5] *S.K.Ong* and *A.Y.C.Nee*, “Automating setup planning in machining operation”, Journal of material processing technology 63, 1997, PP.151-156.

[6] *Samul H.Huang*, “Mathematical formulation for Automated Set up Planning”, Intelligent CAM Systems Laboratory, University of Toledo. 11)

[7] *H.C.Wu* and *T.C.Chag*, “Automated setup selection in feature based process planning” International Journal of Production research, 1998, Vol.36, NO.3, PP.695-712.

[8] Wang HP, Wysk, RA, A knowledge-based approach for automated process planning, Int. J.Prod. Res., 26(6), 999-1014.

[9] Shunmugam, M.S., Mahesh, P., Bhaskara Reddy S.V., “A method of preliminary planning for rotational components with C-axis features using genetic algorithm”, Comput Ind, V 48, p.p. 199–217, 2000

[10] Y.Zhang, W.HU , Y. Rong, David W.Yen, “Graph based setup planning and tolerance decomposition for computer aided fixture design”, International Journal of Production research, 2001, Vol. 39, No. 14, PP. 3109-3126.

[11] Narsingh Deo, “Graph Theory: With applications to engineering and computer science”, PHI Learning Private Limited.

[12] ISO 14649 -10:2003 Industrial automation systems and integration - Physical device control- Data model for computerized numerical controllers - Part 10: General process data, ISOTC184/SC 1, May 23, 2003.

[13] Gologlu, Cevdet. Machine capability and fixturing constraints-imposed automatic machining set-ups generation. Journal of Materials Processing Technology, 2004, 148, 83–92.

[14] Deb, S. 2005. Intelligent Computer-Aided Process Planning for Rotationally Symmetrical Parts Using Neural Network and Expert System. Ph.D. Thesis, University of Montreal, Canada.

[15]) Lan H, Liu R, Zhang C. A multi-agent-based intelligent STEP-NC controller for CNC machine tools. International Journal of Production Research 2008;46(14): 3887–907.