AI techniques for tool path optimization in CNC.

Problem Statement: Optimizing the tool path in CNC machining. Determination of optimal cutting parameters can enhance the machining results to reach high efficiency and minimize the machining cost. One of the factors that govern the productivity of a CNC machine is the tool path travel during cutting a work piece. Optimizing the tool path can save a lot of time and other costs (like amount of coolant) leading to overall increased productivity. Suppose we need to machine different areas of a workpiece, then we can use AI techniques to find the optimal path to go through all the areas.

Defining the problem: Initial state is describes (In:Start)

Possible actions available: ACTION(In:a)={Go:b, Go:c, Go:d…}, where b,c and d are possible states. For example, ACTION(In:Area 2)= {Go:Area 1, Go:Area 3}

Transition model, RESULT(s, a) where s is the state the action is currently in and a is the action performed by the agent. For example RESULT(In:Area 2, Go:Area 3) = (In:Area 3)

The goal test, {In: g}, where g is the goal state.

The cost function, *c(s, a, s')*, where s is the current state and a is

the action performed by the agent to reach state s'.

AI Modelling: This problem statement can be modelled as a

heuristic search problem.

Solution Approach: We need to machine different areas on the workpiece. We denote these areas as Area1, Area2, Area3 etc. We are given costs when moving from an area to another. We have to minimize the cost taken to reach the end goal after machining all the required areas.

Let us consider the areas we are required to machine using the CNC as 'nodes' and the time taken by the tool to finish machining the current area and reach the next one as 'cost'.

Suppose we are given a fixed number of nodes and the costs between each and every node. We are also given the starting node, to which we have to return after we have travelled through all the other nodes. We are required to find the shortest path, such that all nodes are reached at the least possible total cost. We can find such a path using the following code.

```
#include<stdio.h>

void search(int* C, int* A, int* path, int* fpath, int *sum, int*fsum, int flag, int n, int b, int a, int *sc){
        int i,k;
        flag++;
```

```
for(k=0;k<n;k++)

if(*(C+n*flag+k)==0){  //Checking if any node of (flag+1)-th row is not already reached.

*(C+n*flag+k)=k+1;   //Placing the new node in that Vertex.

*sum=*sum+*(A+n*b+k);   //Updating total covered path cost.

*(path+flag-1)=k;   //Adding the vertex to path.


/*Updating the Board w.r.t. the newly covered node*/

if(flag<n){

for(i=flag+1;i<n;i++)

            *(C+i*n+k)=k+1;

}


/*Recursively call search function*/

if(flag<n-1)

           search(C,A,path,fpath,sum,fsum,flag,n,k,a,sc);


/*Storing new solution to 'fpath'-array if found.*/

if(flag==n-1){

*sum=*sum+*(A+n*k+a);

if(*sum==*fsum){

            *sc=*sc+1;

      for(i=0;i<n-1;i++)

          *(fpath+(*sc)*(n-1)+i)=*(path+i);   //Updating final path direction.

}

else if(*sum<*fsum){

            *fsum=*sum; //Updating covered path cost.

         *sc=0;

            for(i=0;i<n-1;i++)
```

```
                *(fpath+i)=*(path+i);   //Updating final path direction.

            }

            *sum=*sum-*(A+n*k+a);

            }


            /*Removing the previous node and undoing all its effect*/

            for(i=flag;i<n;i++)

            *(C+n*i+k)=0;

            *sum=*sum-*(A+n*b+k);   //Substructing last added path cost.

            }

}


int main(){

            int n,i,j,a,sc=0;

            printf("Please enter the number of nodes:");

            scanf("%d",&n);

            int A[n][n],C[n][n],sum=0,fsum=9999,path[n-1],fpath[1000][n-1];

            printf("\nPlease enter all the values of Adjacency Matrix:\n");

            for(i=0;i<n;i++){

            printf("\n");

            for(j=0;j<n;j++)

            scanf(" %d",&A[i][j]);

            }


            printf("\nPlease enter the Source (destination as well) node number:");

            scanf("%d",&a);

            for(i=0;i<n;i++)

            for(j=0;j<n;j++)
```

```
        C[i][j]=0;

        for(i=0;i<n;i++)

        C[i][a-1]=a;


    search(C,A,path,fpath,&sum,&fsum,0,n,a-1,a-1,&sc);


        printf("\n\nMinimum cost = %d.",fsum);

        for(i=0;i<=sc;i++){

        printf("\n\n\tpath direction type %d: %d -->",i+1,a);

        for(j=0;j<n-1;j++)

        printf(" %d -->",fpath[i][j]+1);

        printf(" %d.",a);

        }

        printf("\n\n\n");

}
```