Birla Institute of Technology & Science, Pilani
Pilani Campus

II SEMESTER 2024-2025
TAKE HOME EXERCISE-2

Course No.: CS F422 **Course Title:** Parallel Computing
Deadline: As per Canvas **Maximum Marks:** 20M

Consider the given code for Jacobi method for slving linear equations.:

(a) Identify which loops in this code are free from loop-carried dependence. Mention with justificcation in the code file itself as a comment againt each loop.

(b) Add OpenMP directives in the file so that loops can be run in parallel.

(c) Identify the best sheduling methods for each loop and mention in the code file itself as a comment againts each loop. Attach screenshots of your code printing time taken for each schedule method.

All output will be printed by executing the file "modified_jacobi.c".

Files Expected: A tar file <idno>_the2.tar containing modified_jacobi.c, makefile, screenshots.

```
#include<stdio.h>
#include<stdlib.h>
#include<math.h>
int main()
{
    float a[10][10],b[10],x[10],xn[10],sum,e;
    int i,j,n,flag=0,key;
    printf("\nThis program illustrates Gauss-Jacobi method to solve system of AX=B\n");
    printf("\nEnter the dimensions of coefficient matrix n: ");
    scanf("%d",&n);
    printf("\nEnter the elements of matrix A:\n");
    for(i=0;i<n;i++)
    {
        for(j=0;j<n;j++)
        {
            scanf("%f",&a[i][j]);
        }
    }
    printf("\nEnter the elements of matrix B:\n");
    for(i=0;i<n;i++)
    scanf("%f",&b[i]);
    printf("\nThe system of linear equations:\n");
    for(i=0;i<n;i++)
    {
        printf("\n(%.2f)x1+(%.2f)x2+(%.2f)x3=(%.2f)\n",a[i][0],a[i][1],a[i][2],b[i]);
    }
    for(i=0;i<n;i++)
    {
        sum=0;
    for(j=0;j<n;j++)
    {
        sum+=fabs(a[i][j]);
    }
    sum-=fabs(a[i][i]);
    if(fabs(a[i][i]<sum))
    {
        flag=1;
        break;
    }
    }
    if(flag==1)
    {
```

```
        printf("\nThe system of linear equations are not diagonally dominant\n");
        return main();
    }
    else
    {
        printf("\nThe system of linear equations are diagonally dominant\n");
        printf("\nEnter the initial approximations: ");
        for(i=0;i<n;i++)
        {
            //x[i]=0;
        printf("\nx%d=",(i+1));
        scanf("%f",&x[i]);
        }
        printf("\nEnter the error tolerance level:\n");
        scanf("%f",&e);
    }
    printf("x[1]\t\tx[2]\t\tx[3]");
    printf("\n");
    key=0;
    while(key<n-1)
    {
        key=0;
    for(i=0;i<n;i++)
    {
        sum=b[i];
    for(j=0;j<n;j++)
    if(j!=i)
    sum-=a[i][j]*x[j];
    xn[i]=sum/a[i][i];
    if(fabs(x[i]-xn[i])<e)
    {
        key=key+1;
    }
    }
    if(key==n)
    {
        break;
    }
    printf("%f\t %f\t %f\t",xn[0],xn[1],xn[2]);
    for(i=0;i<n;i++)
    {
        x[i]=xn[i];
    }
    }
    printf("\nAn approximate solution to the given system of equations is\n");
    for(i=0;i<n;i++)
    {
        printf("\nx[%d]=%f\n",(i+1),x[i]);
    }
    return 0;
}
```

Taken from: Gauss-Jacobi method (C) - myCompiler

--&--