TSN3151: Laboratory 05

Part 1: Routing

Using the e-cube routing (also known as a deadlock free routing algorithm), determine the route taken in a 6-dimensional Hypercube network from Node 8 to Node 53. Your final answer should be stated in Node (decimal) number, show your steps clearly.

Part 2: Compare speedup using `MPI_Wtime()`

In your last week's laboratory sheet, you are asked to code for the parallel computation of π. Similarly to doing the parallel code to compute π, write the serial code for numerical integration for π calculation. Determine the serial time and compare it with MPI parallelized program by using `MPI_Wtime()` function.

Depending on your setup, you may not get a better performance as you are running it on a single machine.

Part 3: Calculating something else than Pi

We want to compute the area below the curve

$$f(x) = \sin\left(\frac{x}{2}\right) + 1$$

between π/2 and 2π. That is to say,

$$\int_{x=\pi/2}^{2\pi} \sin\left(\frac{x}{2}\right) + 1$$

This means that, instead of starting from 0 and going to 1, we need to start from π/2 and go till 2π. So our "for" loop is a little bit more complicated, as can be seen from the serial code below.

```
#include<stdio.h>
#include<math.h>

float f(float x){
        return (sin (x/2) +1);
}

int main(){

        double a,b,w,sum=0,middle;
        int i,n;

        printf("Enter the number of intervals : ");
        scanf("%d",&n);

        a=M_PI/2;
        b=2*M_PI;

        //step length (width of each slice)
        w=(b-a)/n;
        middle = w/2;

        for(i=0; i<n; i++)
        {       sum += f((a+i*w)+middle);
        }
        sum = sum * w;

        printf("Answer : %f \n",sum);

        return 0;
}
```

Convert the above to an MPI program. After you have it working correctly, you can add the `MPI_Wtime()` function to see how much faster it runs.