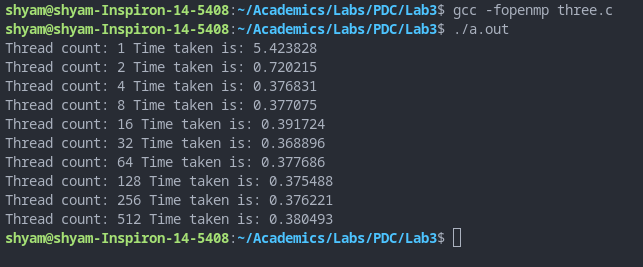
Vector



PF

1.734425

0.734425

**Vector add**

#include <stdio.h>

#include "omp.h"

#include<time.h>

#define N 600000

int main()

{

float a[N],b[N],c[N];

int i;

int m=650000;

//clock\_t t;

float start,end,exec;

printf("Name: Shyam Sundaram\nReg num: 19BCE1560\nPDC Lab:\n\n");

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

{

a[i]=(i+1)\*1.0;

b[i]=(i+1)\*2.0;

}

//t=clock();

//start=omp\_get\_wtime();

int thread[]={1,2,4,8,16,32,64,128,256,512};

float serial;

for(int t=0;t<10;++t)

{

omp\_set\_num\_threads(thread[t]);

start=omp\_get\_wtime();

#pragma omp parallel default(none), private(i,m), shared(a,b,c)

{

#pragma omp for

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

{

for(int j=0;j<1000;++j)

c[i]=a[i]+b[i];

}

}

//t=clock()-t;

//double exec=((double)t)/CLOCKS\_PER\_SEC;

end=omp\_get\_wtime();

exec=end-start;

if(t==0) serial=exec;

printf("Thread count: %d Time taken is: %f ",thread[t],exec);

float pf=(1-(exec/serial))/(1-(1/thread[t]));

printf(" PF = %f ",pf);

float s=1-pf;

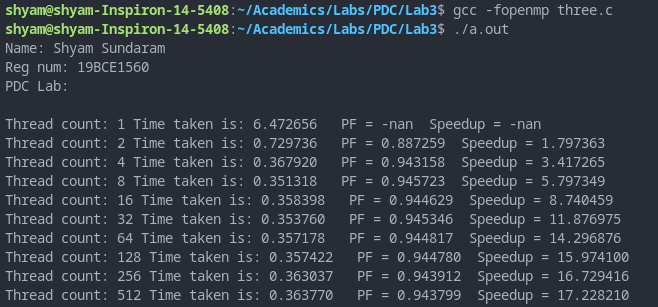
float speedup=1/(s+(pf/thread[t]));

printf(" Speedup = %f\n",speedup);

}

return 0;

}



**Matrix add**

#include <stdio.h>

#include "omp.h"

#include<time.h>

#define ROWS 2500

#define COLS 250

int main()

{

float a[ROWS][COLS],b[ROWS][COLS],c[ROWS][COLS];

printf("Name: Shyam Sundaram\nReg num: 19BCE1560\nPDC Lab:\n\n");

for(int i=0;i<ROWS;++i)

for(int j=0;j<COLS;++j)

{

a[i][j]=i\*10+j;

b[i][j]=j\*10+i;

}

int thread[]={1,2,4,8,16,32,64,128,256,512};

float serial;

for(int t=0;t<10;++t)

{

omp\_set\_num\_threads(thread[t]);

float start=omp\_get\_wtime();

#pragma omp parallel for shared(a,b,c) //reduction(+: c)

for(int i=0;i<ROWS;++i)

for(int j=0;j<COLS;++j)

{

for(int j=0;j<1000;++j)

c[i][j]=a[i][j]+b[i][j];

}

float end=omp\_get\_wtime();

float exec=end-start;

if(t==0) serial=exec;

printf("Thread count: %d Time taken is: %f",thread[t],exec);

float pf=(1-(exec/serial))/(1-(1/thread[t]));

printf(" PF = %f ",pf);

float s=1-pf;

float speedup=1/(s+(pf/thread[t]));

printf(" Speedup = %f\n",speedup);

}

return 0;

}

