1 Type of process

1.1 Serial Program

A single thread program normally executed on a processing core of a CPU

1.2 Concurrency program

Many tasks or threads running at the same time independently

1.3 Parallel Program

Many tasks or threads running at the same time on many cores and there is cooperation between them

1.4 Distributed Program

Many tasks or threads running at the same time on many cores distributed in different places and there is cooperation between them

2 OpenMP (Open Multi Processing)

Cross-platform parallel programming API for C/C++ and Fortran

Example

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

int main(void){
 #pragma omp parallel
 {
       int ID = omp_get_thread_num();
       printf("Hello(%d)", ID);
       printf("World(%d)\n", ID);
 }
 return 0;
}
```

2.1 Compile

To allow complier to parallelize the code use -fopenmp

```
gcc -g -Wall -fopenmp -o hello_omp hello_omp.c
```

2.2 Summation

An example of unroll-for loop to combine results from different cores to the master core.

Here we use different type of predefined code:

1. `#pragma omp parallel` to set the following line becomes parallel part
2. `#pragma omp shared(sum)` to set to be a shared variable
3. `#pragma omp barrier` to set barrier all threads must be finished before the master thread can proceed the next instructions

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

int main(void){
        int sum[8]={};
        #pragma omp parallel shared(sum)
        {
                int ID = omp_get_thread_num();
                //int x[1000];
                int i;
                //initialization
                for(i=ID*1;i<(ID+1)*1;i++){
                        sum[ID]+=i;
                }
                printf("L0-->core:%d, sum:%d \n",ID,sum[ID]);
       #pragma omp barrier
       if(!(ID%2)){
             sum[ID]+=sum[ID+1];
             printf("L1-->core:%d, sum:%d\n",ID,sum[ID]);
       }
       #pragma omp barrier
       if(!(ID%4)){
             sum[ID]+=sum[ID+2];
             printf("L2-->core:%d, sum:%d\n",ID,sum[ID]);
       }
       #pragma omp barrier
       if(!(ID%8)){
             sum[ID]+=sum[ID+4];
             printf("L3-->core:%d, sum:%d\n",ID,sum[ID]);
       }
 }
        return 0;
}
```

3 Github

In this course, all source codes will be hosted in github as the following URL. Please, make sure you are able to reach the matterials.

```
To clone
    git clone https://github.com/wasit7/cs426

To reset all local files and pull
```

```
    git fetch --all
    git reset --hard origin/master
    git pull

To add all and commit
    git add -A && git commit

To push
    git push
```