并行与分布式计算作业

Homework-1

姓名：陈恩婷
班级：行政 1 班
学号：19335015

一、 问题描述

在第一次课程中已经讲到，早期单节点计算系统并行的粒度分为:Bit 级并行，指令级并行和线程级并行。现代处理器如 Intel、ARM、AMD、Power 以及国产 CPU 如华为鲲鹏等，均包含了并行指令集合。1. 请调查这些处理器中的并行（向量）指令集，并选择其中一种如 AVX, SSE 等进行编程练习。此外，现代操作系统为了发挥多核的优势，支持多线程并行编程模型，请利用多线程的方式实现 N 个整数的求和，编程语言不限，可以是 Java，也可以是 C/C++。

二、 解决方案

用 C/C++进行编程，代码如下：

(1) Avx 编程

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

int main( int argc, char const* argv[] ) {
    __m256 float_vec_0 = _mm256_set_ps( 8.0, 9.0, 10.0, 11.0, 12
.0, 13.0, 14.0, 15.0 );
    float* flo1 = (float*)&float_vec_0;
    printf( "float1:\t\t%f, %f, %f, %f, %f, %f, %f, %f\n", flo1[
 0 ], flo1[ 1 ], flo1[ 2 ], flo1[ 3 ], flo1[ 4 ], flo1[ 5 ], flo
1[ 6 ], flo1[ 7 ] );

    __m256 float_vec_1 = _mm256_set_ps( 17.0, 18.0, 19.0, 20.0,
21.0, 22.0, 23.0, 24.0 );
    float* flo2 = (float*)&float_vec_1;
    printf( "float2:\t\t%f, %f, %f, %f, %f, %f, %f, %f\n", flo2[
 0 ], flo2[ 1 ], flo2[ 2 ], flo2[ 3 ], flo2[ 4 ], flo2[ 5 ], flo
2[ 6 ], flo2[ 7 ] );

    __m256 float_result = _mm256_add_ps( float_vec_0, float_vec_
1 );
```

```
    float* flo = (float*)&float_result;
    printf( "float:\t\t%f, %f, %f, %f, %f, %f, %f, %f\n", flo[ 0
 ], flo[ 1 ], flo[ 2 ], flo[ 3 ], flo[ 4 ], flo[ 5 ], flo[ 6 ],
flo[ 7 ] );

    return 0;
}
```

用 gcc -mavx avx.c -o avx 进行编译。

(2) 多线程方式实现 N 个整数求和

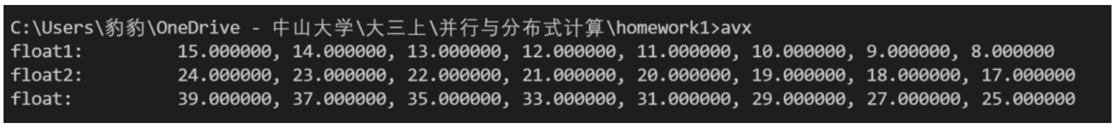
```
#include<omp.h>
#include<iostream>
using namespace std;

int main(){
    int sum = 0;
    #pragma omp parallel for num_threads(10) reduction(+: sum
)
    for ( int i = 1; i <= 100; i++ ){
        sum += i;
    }
    cout << "sum = " << sum << endl;
}
```

用 g++ -g -Wall -fopenmp add_n_integers.cpp -o add 进行编译。

三、 实验结果

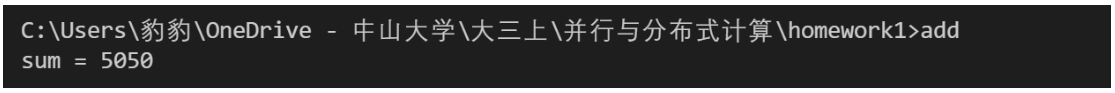
编译好后执行程序，结果分别如下：

(1) Avx 编程



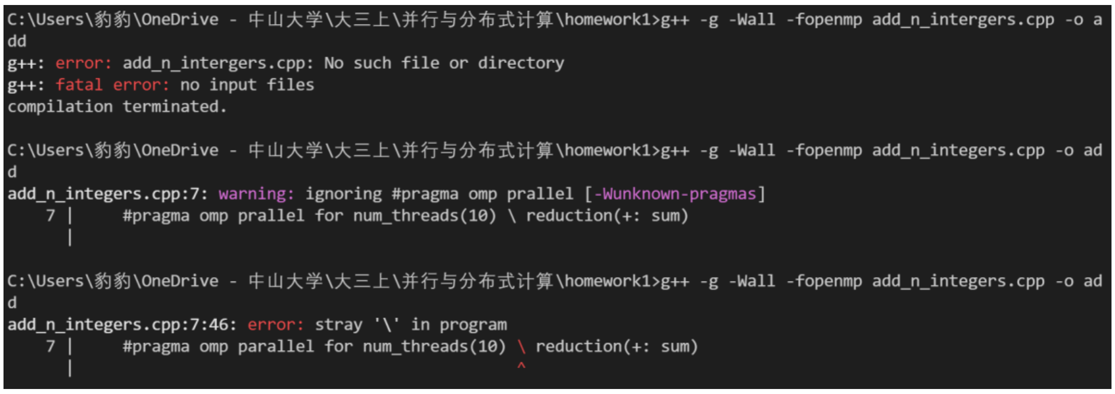
```
C:\Users\豹豹\OneDrive - 中山大学\大三上\并行与分布式计算\homework1>avx
float1:         15.000000, 14.000000, 13.000000, 12.000000, 11.000000, 10.000000, 9.000000, 8.000000
float2:         24.000000, 23.000000, 22.000000, 21.000000, 20.000000, 19.000000, 18.000000, 17.000000
float:          39.000000, 37.000000, 35.000000, 33.000000, 31.000000, 29.000000, 27.000000, 25.000000
```

(2) 多线程方式实现 N 个整数求和



```
C:\Users\豹豹\OneDrive - 中山大学\大三上\并行与分布式计算\homework1>add
sum = 5050
```

四、 遇到的问题及解决方法



```
C:\Users\豹豹\OneDrive - 中山大学\大三上\并行与分布式计算\homework1>g++ -g -Wall -fopenmp add_n_intergers.cpp -o a
dd
g++: error: add_n_intergers.cpp: No such file or directory
g++: fatal error: no input files
compilation terminated.

C:\Users\豹豹\OneDrive - 中山大学\大三上\并行与分布式计算\homework1>g++ -g -Wall -fopenmp add_n_integers.cpp -o ad
d
add_n_integers.cpp:7: warning: ignoring #pragma omp prallel [-Wunknown-pragmas]
    7 |     #pragma omp prallel for num_threads(10) \ reduction(+: sum)
      |

C:\Users\豹豹\OneDrive - 中山大学\大三上\并行与分布式计算\homework1>g++ -g -Wall -fopenmp add_n_integers.cpp -o ad
d
add_n_integers.cpp:7:46: error: stray '\' in program
    7 |     #pragma omp parallel for num_threads(10) \ reduction(+: sum)
      |                                              ^
```

如图, 在编译时分别出现了以上问题, 分别时由于文件位置放错, #pragma 一行 parallel 拼写错误, 以及代码中多写了一个"\"。修改好后问题解决了。

Write an essay describing a research problem in your major that would benefit from the use of parallel computing. Provide a rough outline of how parallelism would be used. Would you use task- or data-parallelism?

An example of using parallel computing in a research problem would be developing better online translators. We can carry this out by doing analysis on the natural language texts and their corresponding translations that exist today, so that we can discover more patterns in languages and how to translate them. This can hugely benefit from parallel computing, as this involves dealing with huge amounts of data. By using data parallelism, we will be able to process these text data much more efficiently.