并行编程实验指导-AVX512

北京交通大学计算机与信息技术学院
2024年 6月

一、实验目的

本实验通过对SIMD中AVX512指令的使用，贯彻《高性能计算导论》课程从理论到实践的教学原则和教学目标，加深广大学生对并行程序的工作原理和实现方法的理解，使学生了解SIMD的运行机制，掌握基本的SIMD编程能力。

二、实验环境

操作系统：

- 在提供的国家超级计算无锡中心远程服务器上完成AVX512实验。
- **AVX512程序的编译与运行**
    - 以C源程序avx512_mm.c为例：
    - 编译时加上AVX512标识：
        - gcc -o avx512_mm avx512_mm.c -mavx512f
    - 提交执行：
        - sbatch run.sh ./avx512_mm
- 附：任务提交脚本run.sh，本次实验的分区为q_all

```
#! /bin/bash
#SBATCH -p q_all
#SBATCH -n 1
#SBATCH -o job.out

$1
```

三、实验内容及要求

本实验由两个实验项目组成，学生应按照实验内容及实验要求，认真完成各项实验，并完成实验报告（包括程序设计文档与清单、实验结果、实验中出现的问题、观察到的现象的解释和说明以及实验体会）。

实验一、AVX512

执行Non-AVX与AVX512的计算代码，**详细分析**实验代码与实验结果（运行时间、对比非AVX与AVX512的向量结果是否一致），并提交注释后的源代码。

非AVX的计算代码：

```
// Non-AVX
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

#define VECTOR_SIZE 16000000
```

```
float* allocate_memory(size_t size) {
    float *ptr = (float*)malloc(size * sizeof(float));
    if (ptr == NULL) {
        perror("Failed to allocate memory");
        exit(EXIT_FAILURE);
    }
    return ptr;
}

void initialize_vector(float* vec, size_t size) {
    for (size_t i = 0; i < size; ++i) {
        vec[i] = (float)i + 1.0f;
    }
}

void vector_mul(float* a, float* b, float* c, size_t size) {
    for (size_t i = 0; i < size; ++i) {
        c[i] = a[i] * b[i];
    }
}

int main() {
    float* vec_a = allocate_memory(VECTOR_SIZE);
    float* vec_b = allocate_memory(VECTOR_SIZE);
    float* vec_c = allocate_memory(VECTOR_SIZE);

    initialize_vector(vec_a, VECTOR_SIZE);
    initialize_vector(vec_b, VECTOR_SIZE);

    clock_t start = clock();
    vector_mul(vec_a, vec_b, vec_c, VECTOR_SIZE);
    clock_t end = clock();

    double time_taken = (double)(end - start) / CLOCKS_PER_SEC;
    printf("Non-AVX Time: %f seconds\n", time_taken);

    free(vec_a);
    free(vec_b);
    free(vec_c);

    return 0;
}
```

AVX512的计算代码：

```
// AVX512
#include <immintrin.h>
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

#define VECTOR_SIZE 16000000
```

```
float* allocate_aligned_memory(size_t size, size_t alignment) {
    float *ptr = NULL;
    if (posix_memalign((void**)&ptr, alignment, size * sizeof(float)) != 0) {
        perror("Failed to allocate aligned memory");
        exit(EXIT_FAILURE);
    }
    return ptr;
}

void initialize_vector(float* vec, size_t size) {
    for (size_t i = 0; i < size; ++i) {
        vec[i] = (float)i + 1.0f;
    }
}

void vector_mul_avx512(float* a, float* b, float* c, size_t size) {
    for (size_t i = 0; i < size; i += 16) {
        __m512 av = _mm512_load_ps(&a[i]);
        __m512 bv = _mm512_load_ps(&b[i]);
        __m512 cv = _mm512_mul_ps(av, bv);
        _mm512_store_ps(&c[i], cv);
    }
}

int main() {
    float* vec_a = allocate_aligned_memory(VECTOR_SIZE, 64);
    float* vec_b = allocate_aligned_memory(VECTOR_SIZE, 64);
    float* vec_c = allocate_aligned_memory(VECTOR_SIZE, 64);

    initialize_vector(vec_a, VECTOR_SIZE);
    initialize_vector(vec_b, VECTOR_SIZE);

    clock_t start = clock();
    vector_mul_avx512(vec_a, vec_b, vec_c, VECTOR_SIZE);
    clock_t end = clock();

    double time_taken = (double)(end - start) / CLOCKS_PER_SEC;
    printf("AVX-512 Time: %f seconds\n", time_taken);

    free(vec_a);
    free(vec_b);
    free(vec_c);

    return 0;
}
```

实验二、AVX512 Matrix Multiplication

补全并执行如下代码，**详细分析**实验代码与实验结果（对比矩阵乘结果是否一致、运行时间、加速比等），并提交注释后的源代码。

```
#include <immintrin.h>
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <math.h>

#define N 512

void initialize_matrix(float* matrix, size_t size) {
    for (size_t i = 0; i < size * size; ++i) {
        matrix[i] = (float)(rand() % 100) / 10.0f;
    }
}

void matrix_multiply_avx512(float* a, float* b, float* c, size_t size) {
    // TODO
}

void matrix_multiply_non_avx(float* a, float* b, float* c, size_t size) {
    // TODO
}

int compare_matrices(float* m1, float* m2, size_t size) {
    // TODO
}

double measure_time(void (*func)(float*, float*, float*, size_t), float* a, float*
b, float* c, size_t size) {
    clock_t start = clock();
    func(a, b, c, size);
    clock_t end = clock();
    return (double)(end - start) / CLOCKS_PER_SEC;
}

int main() {
    srand(time(NULL));

    float* matrix_a = (float*)aligned_alloc(64, N * N * sizeof(float));
    float* matrix_b = (float*)aligned_alloc(64, N * N * sizeof(float));
    float* matrix_c_avx = (float*)aligned_alloc(64, N * N * sizeof(float));
    float* matrix_c_non_avx = (float*)aligned_alloc(64, N * N * sizeof(float));

    initialize_matrix(matrix_a, N);
    initialize_matrix(matrix_b, N);

    // AVX-512 Matrix Multiplication
```

```
    double time_avx = measure_time(matrix_multiply_avx512, matrix_a, matrix_b,
matrix_c_avx, N);
    printf("AVX-512 Time: %f seconds\n", time_avx);

    // Non-AVX Matrix Multiplication
    double time_non_avx = measure_time(matrix_multiply_non_avx, matrix_a,
matrix_b, matrix_c_non_avx, N);
    printf("Non-AVX Time: %f seconds\n", time_non_avx);

    if (compare_matrices(matrix_c_avx, matrix_c_non_avx, N)) {
        printf("Results match.\n");
    } else {
        printf("Results do not match!\n");
    }

    free(matrix_a);
    free(matrix_b);
    free(matrix_c_avx);
    free(matrix_c_non_avx);

    return 0;
}
```

四、实验提交

1. 书面提交的设计文档封面格式见附件一。
2. 电子文档形式提交的程序源码等最后形成一个文件夹（名称为“学号姓名”）。

附件一 书面文档封面格式

并行编程实验设计文档

实验项目名称：_____

姓名　　：_____

班级　　：_____

学号　　：_____

自我评价：

_____

_____

_____

成绩：_____

北京交通大学计算机与信息技术学院