



НИЖЕГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМ. Н.И. ЛОБАЧЕВСКОГО

ЦЕНТР КОМПЕТЕНЦИЙ ONEAPI В ННГУ

ИНСТИТУТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МАТЕМАТИКИ И МЕХАНИКИ



НИЖЕГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ им. Н.И. ЛОБАЧЕВСКОГО
ЦЕНТР КОМПЕТЕНЦИЙ oneAPI в ННГУ
ИНСТИТУТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МАТЕМАТИКИ И МЕХАНИКИ

***ВВЕДЕНИЕ В АНАЛИЗ ПРОИЗВОДИТЕЛЬНОСТИ
И ОПТИМИЗАЦИЮ ПРОГРАММ***

**Практическое использование Intel C++ Compiler
в среде разработки Microsoft Visual Studio.
Компиляция и сборка из командной строки**

Содержание

- ❑ Цель работы
- ❑ Возможности инструментария
 - Intel C++ Compiler
- ❑ Тестовая задача
- ❑ Использование Intel C++ Compiler

ЦЕЛЬ РАБОТЫ

Цель работы

- ❑ Научиться использовать инструментарий для повышения производительности программ

Задачи:

- ❑ Рассмотреть основные возможности Intel C++ Compiler
- ❑ На примере тестовой задачи из области вычислительной математики выполнить оптимизацию кода с использованием инструментария

INTEL C++ COMPILER

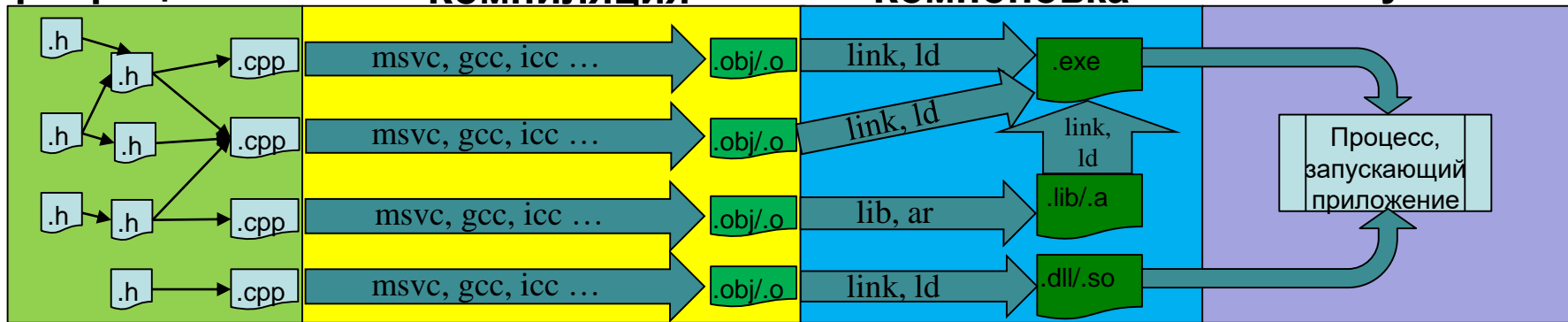
Сборка и запуск приложения

Препроцессинг

Компиляция

Компоновка

Запуск



- Удаление комментариев
- `#include`
- `#define`
- `#if`, `#ifdef` ...
- Лексический анализ
- Синтаксический анализ
- Семантический анализ
- Оптимизация
- Генерация кода
- Создание исполняемых файлов, статических и динамических библиотек

Возможности компилятора

❑ Основные возможности:

- Высокоуровневая оптимизация
- Межпроцедурная оптимизация
- Автоматическое распараллеливание кода
- Векторизация для SSE, SSE2, SSE3, SSE4, AVX, AVX2, AVX512
- Оптимизация с учетом данных профилировки

❑ Дополнительные библиотеки:

- MKL (Math Kernel Library), IPP (Integrated Performance Primitives), DAAL (Data Analytics Acceleration Library)
- TBB (Threading Building Blocks), MPI

❑ Поддерживаемые язык и стандарты (19.2 в составе OneAPI HPC Toolkit):

- Полная поддержка C++11, C++14, C++17. Частичная C++20
- Полная поддержка OpenMP 4.5. Частичная OpenMP 5.0&5.1

ТЕСТОВАЯ ЗАДАЧА

Тестовая задача и метод решения

- Вычисляется результат перемножения двух квадратных матриц:

$$C = A * B$$

где $C, A, B \in \mathbb{R}^{N \times N}$

- Параметры тестовой задачи:

- $N = 1024$
- A, B – случайные матрицы
- Измеряется время одного умножения матриц

Код программы

// см. приложенный код

```
#pragma once
#include <random>
using namespace std;

class Matrix
{
public:
    Matrix(int _size) {
        data = new double[_size * _size];
        size = _size;
        for (int i = 0; i < size * size; i++) {
            data[i] = 0.0;
        }
    }
    Matrix(const Matrix& oth) {
        data = new double[oth.size * oth.size];
        size = oth.size;
        std::copy(oth.data, oth.data + oth.size * oth.size,
data);
    }

    double& at(int i, int j) const {
        return data[i * size + j];
    }
    double at(int i, int j) const {
        return data[i * size + j];
    }
}
```

```
double* getData() {
    return data;
}

void generateRandomMatrix() {
    random_device rd;
    mt19937 gen(rd());
    normal_distribution<double> uniform_dis(0.0, 1e-4);
    for (int i = 0; i < size; i++) {
        for (int j = 0; j < size; j++) {
            data[i*size + j] = uniform_dis(gen);
        }
    }
}

Matrix& operator=(const Matrix& oth) {
    if (this != &oth) {
        if (oth.size != size) {
            delete data;
            size = 0;
            data = new double[oth.size * oth.size];
            size = oth.size;
        }
        copy(oth.data, oth.data + oth.size * oth.size, data);
    }
    return *this;
}
```

Код программы (2)

// см. приложенный код

```
~Matrix() {
    delete data;
    size = 0;
}

friend Matrix operator* (const Matrix& x, const
Matrix& y);
private:
    double* data;
    int size;
};

Matrix operator* (const Matrix& x, const Matrix& y) {
    if (x.size == y.size) {
        int size = x.size;
        Matrix res(size);
#pragma omp parallel for
        for (int i = 0; i < size; i++)
            for (int k = 0; k < size; k++)
#pragma omp simd
                for (int j = 0; j < size; j++)
                    res.at(i, j) += x.at(i, k) * y.at(k, j);
        return res;
    } else
        throw std::string("different size in *");
}
```

```
#include <chrono>
#include "matrix.h"
#include <iostream>
using namespace chrono;

int main() {
    int size = 2048;
    Matrix a(size), b(size);
    a.generateRandomMatrix();
    b.generateRandomMatrix();

    time_point<system_clock> start, end;
    start = system_clock::now();
    Matrix c = a * b;
    end = system_clock::now();

    int elapsed_seconds = duration_cast<milliseconds>(end -
                                                    start).count();
    cout << "Total time: " << elapsed_seconds / 1000.0 << " sec"
        << endl;
    return 0;
}
```

ИСПОЛЬЗОВАНИЕ INTEL C++ COMPILER

Запуск компилятора

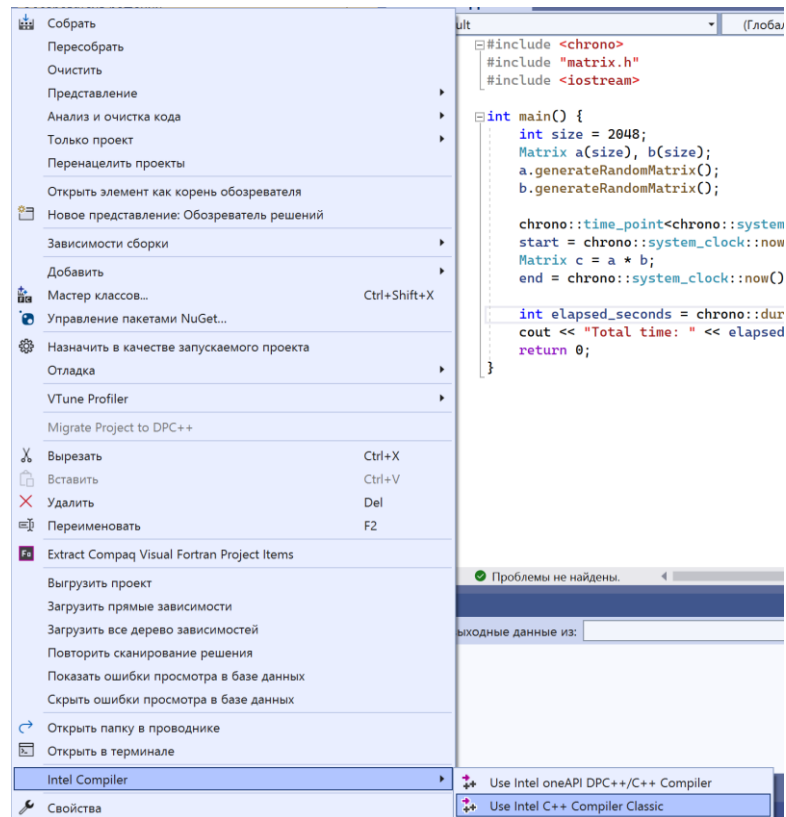
❑ Интеграция в Visual Studio:

Свойство проекта -> Intel Compiler ->
Use Intel C++ Compiler Classic

(Use Intel OneAPI DPC++/C++ Compiler)

❑ Intel C++ Compiler можно использовать в консоли:

- Переменные окружения компилятора должны быть подгружены из `compilervars.bat/compilervars.sh` (Windows/Linux) с указанием архитектуры
- Из командной строки доступны: `icl/icc` (Windows/Linux) – классическая версия `dpcpp` и `icx` – LLVM версия



Режимы оптимизации

Windows	Linux	Описание
/Od	-O0	Отключает оптимизацию
/O1	-O1	Оптимизация по размеру. Включить оптимизации, но не использовать методы, увеличивающие размер файлов
/O2	-O2	Оптимизация по скорости
/O3	-O3	Оптимизация по скорости и размеру файлов. Включить более агрессивные методы оптимизации, которые подходят не для всех программ
/Os	-Os	Используется в сочетании с предыдущими ключами. Включить оптимизации по скорости, но не применять оптимизации, которые могут увеличить размер при незначительном выигрыше в скорости
/fast	-fast	Устанавливает -xHost -O3 -ipo -no-prec-div -static -fp-model fast=2 параметры для повышения производительности: <ul style="list-style-type: none">• -ipo (включает межпроцедурную оптимизацию между файлами)• -static (предотвращает связь с динамическими библиотеками)

Работа программы в разных режимах

- ❑ Запускается тестовая задача размера 2048. OpenMP включено. В таблицах представлено наименьшее время из 3 запусков

Visual Studio 2022

Od (Debug)	O2 (Release)
33.7 сек	4.93 сек

Intel C++ Compiler 19.2

Od/O0	O1	O2	Os (+O2)	O3	fast
41 сек	5.03 сек	1.32 сек	1.33 сек	1.19 сек	0.84 сек

- ❑ Сравнивать время в Debug режимах бесполезно
- ❑ Агрессивная оптимизация не всегда полезна

Векторная оптимизация

Windows	Linux	Описание
/QxARCH	-xARCH	Скомпилировать код, оптимизированный под ARCH
/QaxARCH	-axARCH	Добавить отдельную ветку кода, оптимизированную под ARCH

□ В качестве ARCH может быть:

- Любая архитектура Intel, начиная с архитектуры Tremont. В таком случае компилятор будет учитывать микроархитектурные особенности вычислительных ядер
- Наборы инструкций: SSE3, SSE4.2 (по умолчанию в командной строке), AVX, CORE-AVX2, CORE-AVX512, COMMON-AVX512
- Host (по умолчанию при интеграции в Visual Studio) – скомпилироваться под текущую вычислительную машину

Полезные ключи компиляции

Windows	Linux	Описание
/Qansi-alias	-ansi-alias	При оптимизации считать, что имена указывают на разную память. Если это не так, скомпилируется неверный код.
/Qopenmp	-qopenmp	Включить поддержку OpenMP
/Qvec	-vec	Включить векторизацию при компиляции
/Qopt-report	-qopt-report	Выводить отчет оптимизации. Можно настроить вывод разных уровней оптимизации
/fp:keyword	-fp-model keyword	Использовать одну из моделей точности при вычислениях. По умолчанию используется агрессивный метод <i>fast=1</i> , что позволяет переставлять действия без сохранения точности
/Qprec-div	-prec-div	Включает полную точность при делении чисел с плавающей запятой по стандарту IEEE
/debug	-debug	Добавляет отладочную информацию

ДЕМОНСТРАЦИЯ

Литература

1. Intel C++ Compiler Developer Guide and Reference: <https://software.intel.com/en-us/cpp-compiler-developer-guide-and-reference>

Авторский коллектив

- ❑ Мееров Иосиф Борисович, к.т.н., доцент, зам. зав. каф. МОСТ
- ❑ Сысоев Александр Владимирович, к.т.н., доцент каф. МОСТ
- ❑ Линев Алексей Владимирович, зав. лаб. интернета вещей, каф. ПРИН
- ❑ Волокитин Валентин Дмитриевич, программист лаборатории СТиВВ, каф. МОСТ
- ❑ Козинов Евгений Александрович, к.т.н., преподаватель каф. МОСТ
- ❑ Панова Елена Анатольевна, инженер лаборатории СТиВВ, каф. МОСТ

Контакты

Нижегородский государственный университет

<http://www.unn.ru>

Центр компетенций oneAPI в ННГУ

<http://hpc-education.unn.ru/ru/центр-компетенций-oneapi-в-ннгу>

Институт информационных технологий, математики и механики

<http://www.itmm.unn.ru>