

# Курсовая работа

## “Многоязыковое параллельное программирование”

### по дисциплине

## “Высокопроизводительные вычисления”

Составитель: Фёдоров С. А.

### Получаемые и закрепляемые компетенции

- Разработка оптимизированного машинного кода;
- разработка векторизованного кода;
- разработка кода с выравниванием адресов;
- разработка собственного кода целевой платформы с использованием неявных средств многопоточности;
- разработка динамически подключаемых библиотек;
- кросскомпиляция кода с инструментальной платформы на целевую;
- многоязыковое программирование;
- подготовка научно-технической документации.

### Цель и задачи работы

Цель работы -- программное обеспечение, разработанное по технологиям многоязыкового и параллельного программирования и эффективно задействующее современную целевую архитектуру. Задачи работы:

1. **Разработать техническое задание на программное обеспечение.**
  - a. Инструментальная платформа -- GNU/Linux.
  - b. Целевая платформа -- любая (например, GNU/Linux, Windows Nt, Mac OS).
  - c. Программное обеспечение разрабатывается с использованием как минимум двух технологий разработки, взаимодействующих между собой.
  - d. За основу для ТЗ можно взять:
    - i. работу, которую студент считает полезной выполнить для себя (рекомендуется);
    - ii. лабораторную работу данной или любой другой дисциплины,
    - iii. курсовую работу или курсовой проект любой другой дисциплины,
    - iv. научно-исследовательскую работу;
    - v. работу, которую студент выполняет в другом месте.



- е. Обработка данных в динамически подключаемом собственном коде архитектуры.
  - i. В нём проводится *лишь требовательная к ресурсам* обработка данных (не обязательно вся) с использованием *неявных* средств многопоточности.
  - ii. Выполняется на одном из языков программирования: **Fortran 2008/2015** (ISO/IEC 1539-1:2010), **C11** (ISO/IEC 9899:2011), **C++14** (ISO/IEC 14882:2014), **Ada 2012** (ISO/IEC 8652:2012).
  - iii. *Неявная* многопоточность задействуется встроенными средствами языка или с использованием OpenMP 4.5.
  - iv. Код компонуется в виде динамически подключаемой библиотеки.
- ф. Головная программа.
  - i. В ней проводится ввод/вывод данных.
  - ii. Организуется интерфейс пользователя (например, графический).
  - iii. Для осуществления требовательной к ресурсам обработки данных проводится *динамический* вызов собственного кода, который её проводит.
  - iv. Выполняется *по любой* выбранной технологии разработки программного обеспечения. Код может быть исполняемым или интерпретируемым, собственным или управляемым. Язык должен быть отличен от выбранного для реализации динамического кода по обработке данных.

## 2. Разработать динамически подключаемый код по обработке данных.

- а. Разработать **однопоточный неекторизованный собственный код**. Код должен реализовывать необходимые алгоритмы обработки данных (без использования векторизации и неявных средств многопоточности).
- б. Разработать тестовую программу для тестирования и отладки кода по обработке данных, выполненной по той же технологии. Компоновка с кодом по обработке данных может проводиться статически или динамически.
- с. Провести тестирование и отладку однопоточного кода.
  - i. Подготовить опорные данные, требующие на обработку *от 1 до 300 мин.* времени работы на инструментальной платформе.
  - ii. Оценить время работы кода  $T_{б/о}$ , осуществляющего обработку данных без использования оптимизации.
  - iii. Оценить эффективность кода  $P_{б/о}$ , осуществляющего обработку данных без использования оптимизации. Под



эффективностью принять показатель  $P = \text{Perf} / \text{Comp}$ , где Perf -- время работы кода на опорных данных (сек), Comp -- сложность кода (цикломатическая сложность, число строк кода без комментариев или другой показатель сложности, доступный для оценки у выбранной технологии разработки).

- iv. Использовать средства оптимизации, подходящие для разработанного кода (стандартные методы оптимизации, системные методы оптимизации).
- v. Оценить время работы кода  $T_{c/o}$ , осуществляющего обработку данных с использованием оптимизации.
- vi. Оценить эффективность кода  $P_{c/o}$ , осуществляющего обработку данных с использованием оптимизации.
- d. **Разработать *однопоточный векторизованный собственный код***. Код должен реализовывать необходимые алгоритмы обработки данных, сделав изменения в исходном коде, добавив поддержку векторизации.
- e. **Провести тестирование и отладку однопоточного векторизованного кода.**
  - i. Оценить время работы кода  $T_{\text{вект, без/выр'}}$ , осуществляющего обработку данных с использованием векторизации.
  - ii. Оценить эффективность кода  $P_{\text{вект, без/выр'}}$ , осуществляющего обработку данных с использованием векторизации.
- f. **Разработать *однопоточный векторизованный собственный код с использованием выравнивания адресов***. Код должен реализовывать необходимые алгоритмы обработки данных.
- g. **Провести тестирование и отладку однопоточного векторизованного кода с использованием выравнивания адресов.**
  - i. Оценить время работы кода  $T_{\text{вект, с/выр'}}$ , осуществляющего обработку данных с использованием векторизации и выравнивания адресов.
  - ii. Оценить эффективность кода  $P_{\text{вект, с/выр'}}$ , осуществляющего обработку данных с использованием векторизации и выравнивания адресов.
- h. **Разработать *многопоточный код с использованием векторизации, выравнивания адресов и неявных средств многопоточности***. Код должен реализовывать необходимые алгоритмы обработки данных (сделать изменения в исходном коде или в опциях компилятора).
- i. **Провести тестирование и отладку многопоточного векторизованного кода.**



- i. Оценить время работы многопоточного кода  $T_{\text{парал}}$ , осуществляющего обработку данных.
    - ii. Оценить эффективность многопоточного кода  $P_{\text{парал}}$ , осуществляющего обработку данных.
  - j. **Скомпоновать разработанных многопоточный код в виде динамически подключаемой библиотеки.** Если целевая и инструментальная платформы различаются, то провести кросскомпиляцию.
- 3. Разработать головную программу.**
- a. **Реализовать головную программу по выбранной технологии,** предоставляющую как минимум: интерфейс пользователя и ввод/вывод данных. Может содержать пред- или пост-обработку данных.
- 4. Организовать эффективную обработку данных.**
- a. **Разработать привязки в динамически подключаемой библиотеке.**
  - b. **Организовать вызов разработанной динамически подключаемой библиотеки для осуществления *требуемой к ресурсам* обработки данных.**
  - c. **Провести тестирование и отладку разработанного ПО.**

## Отчётность по работе

Необходимо подготовить отчёт по курсовой работе. Его согласовывают с преподавателем в электронной форме, а затем распечатывают.

1. Отчёт оформляется *строго по требованиям* [1] (формат А4).
2. Отчёт должен состоять из основных структурных элементов, включаемых в отчёты для курсовых работ, согласно требованиям [2].
3. Во введении приводятся: актуальность, цель и задача работы, краткое содержание работы.
4. Целью работы является не процесс (сборка, установка и т. п.), а основной конечный результат работы. Цель можно взять из задания на курсовую работу.
5. Задачи, необходимые для достижения цели, формулируются, исходя из основных проводимых действий («подготовка системы ...», «разработка...» и т. д.). Задачи и подзадачи можно взять из задания (они там **выделены жирным**).
6. Каждая из задач формирует одну из 4-ёх глав. В главах допустимы параграфы и пункты. В главах подробно описывается решение каждой из задач, в параграфах -- каждой из подзадач. Например, какие изменения в коде были произведены при добавлении векторизации, многопоточных средств и т. д. При этом снимки экрана *с кодом не приводятся*. -- вместо





этого приводить *части* листинга кода, выполненные моноширинным шрифтом. Полный исходный код ПО приводится в приложениях к отчёту.

7. В выводах приводятся:

- достигнутая или недостигнутая цель;
- решённые и нерешённые задачи;
- возникшие принципиальные трудности и способы их преодоления;
- выводы по скорости работы однопоточного кода без и с оптимизацией, векторизованного кода, кода с векторизацией и выравниванием адресов, многопоточного кода с векторизацией (также привести таблицу с полученными показателями скорости и эффективности кода);
- выводы по работе, которые вы вынесли лично для себя.

8. В выводах абзацы *не маркируются и не нумеруются*.

## Литература

1. Правила оформления студенческих выпускных работ и отчетов: метод. указания / сост.: В.В. Шаляпин. - СПб.: Изд-во Политехн. ун-та, 2013. – 44с.
2. Правила оформления студенческих текстовых документов: дипломных (курсовых) проектов (работ), отчётов и рефератов [Электронный ресурс] : методические рекомендации / В.И. Маслов, Л.Н. Шуткевич ; Санкт-Петербургский государственный политехнический университет, Институт металлургии, машиностроения и транспорта, Кафедра конструкторско-технологических инноваций .— Электрон. текстовые дан. (1 файл : 519 Кб) .— Санкт-Петербург, 2013 .— Загл. с титул. экрана .— Свободный доступ из сети Интернет (чтение, печать) .— Текстовый документ. — Adobe Acrobat Reader 7.0 .— <URL:<http://elib.spbstu.ru/dl/2/2976.pdf>>.

