тольяттинский
государственный
университет

Н.В. Хрипунов

МНОГОПОТОЧНОЕ ПРОГРАММИРОВАНИЕ

ВЫПОЛНЕНИЕ КУРСОВОЙ РАБОТЫ

Учебно-методическое пособие

Тольятти
Издательство ТГУ
2025

Министерство науки и высшего образования
Российской Федерации
Тольяттинский государственный университет

Н.В. Хрипунов

МНОГОПОТОЧНОЕ ПРОГРАММИРОВАНИЕ. ВЫПОЛНЕНИЕ КУРСОВОЙ РАБОТЫ

Учебно-методическое пособие

Тольятти
Издательство ТГУ
2025

УДК 378.091.313(075.8)+004.42(075.8)
ББК 74.480.278я73+32.973я73
Х 933

Рецензенты:
канд. экон. наук, доцент, исполнительный директор Высшей школы интеллектуальных систем и кибертехнологий Поволжского государственного университета сервиса *О.А. Филиппова*;
д-р техн. наук, доцент, профессор кафедры «Прикладная математика и информатика» Тольяттинского государственного университета *С.В. Мкртычев*.

Х 933 Хрипунов, Н.В. Многопоточное программирование. Выполнение курсовой работы : учебно-методическое пособие / Н.В. Хрипунов. – Тольятти : Издательство ТГУ, 2025. – 51 с. – ISBN 978-5-8259-1728-3.

В учебно-методическом пособии приведены методические указания к курсовой работе по курсу «Многопоточное программирование», в том числе описание процедуры выбора и формулировки темы курсовой работы, указания к структуре и содержанию курсовой работы, требования к оформлению и критерии оценки курсовой работы.

Предназначено студентам бакалавриата по направлениям подготовки 09.03.03 «Прикладная информатика», 02.03.03 «Математическое обеспечение и администрирование информационных систем», 01.03.02 «Прикладная математика и информатика» очной, заочной форм обучения.

Может быть полезно преподавателям, инженерно-техническим работникам.

УДК 378.091.313(075.8)+004.42(075.8)
ББК 74.480.278я73+32.973я73

Рекомендовано к изданию научно-методическим советом Тольяттинского государственного университета.

ISBN 978-5-8259-1728-3

© Хрипунов Н.В., 2025
© ФГБОУ ВО «Тольяттинский государственный университет», 2025

НОРМАТИВНЫЕ ССЫЛКИ

В настоящем пособии использованы ссылки на следующие локальные акты ТГУ:

– Положение о курсовой работе (курсовом проекте);

– Методические указания по оформлению выпускных квалификационных работ по программам бакалавриата, программам специалитета, программам магистратуры.

ВЫБОР И ФОРМУЛИРОВКА ТЕМЫ КУРСОВОЙ РАБОТЫ

Общий порядок разработки, алгоритм выбора и согласования темы курсовой работы регламентируются Положением о курсовой работе (курсовом проекте).

С целью обеспечения уникальности тем курсовых работ при выборе темы работы используется трехэтапный подход: выбор стратегии выполнения работы, выбор технологии (технологий), выбор задачи (задач).

Первая стратегия – стратегия МТОЗ – состоит в сравнении эффективности различных технологий при решении какой-либо одной задачи (много технологий, одна задача).

Вторая стратегия – стратегия ОТМЗ – состоит в сравнении эффективности одной технологии при решении различных задач или при решении задачи различными методами (одна технология, много задач).

Перечень технологий формируется из технологий, изучаемых в лекционном материале и на практических занятиях, например, OpenMP, C++11 Threads и т. п.

Перечень задач содержит задачи, для решения которых могут быть использованы как однопоточная, так и многопоточная технологии, например, инверсия изображений, умножение матриц, расчет числа Пифагора и т. п.

Пример формулировки темы по стратегии МТОЗ: «Сравнение эффективности технологий OpenMP и C++11 Threads на примере задачи умножения матриц».

Пример формулировки темы по стратегии ОТМЗ: «Сравнение эффективности технологии OpenMP в решении задач численного интегрирования методом трапеций и методом Монте-Карло».

По согласованию с руководителем курсовой работы допускается обоснованный выбор темы, основанной на стратегии, технологии (технологиях) и задаче (задачах), не включенных в перечни.

ТРЕБОВАНИЯ К СТРУКТУРЕ КУРСОВОЙ РАБОТЫ

Общие требования к структуре курсовой работы регламентируются Положением о курсовой работе (курсовом проекте).

Теоретический раздел работы содержит теоретическую информацию, связанную с темой работы.

Практический раздел работы состоит из двух подразделов. В первом выполняется разработка программ, во втором – сравнительная оценка эффективности программ.

Типовая структура основной части работ для тем по стратегии МТОЗ (сравнение эффективности технологий «Технология 1» и «Технология 2» на примере задачи «Задача 1»):

1. Постановка задачи на исследование.

1.1. Практическая значимость задачи.

1.2. Обзор методов решения задачи.

1.3. Реализация последовательной программы.

2. Проектирование и разработка параллельных программ.

2.1. Обзор технологий разработки параллельного программного обеспечения.

2.2. Разработка параллельных алгоритмов.

2.3. Реализация параллельной программы «Технология 1».

2.4. Реализация параллельной программы «Технология 2».

3. Экспериментальное исследование эффективности разработанного программного обеспечения.

3.1. Методика оценки эффективности программного обеспечения.

3.2. Оценка воспроизводимости эксперимента.

3.3. Планирование и проведение экспериментальных вычислений.

3.4. Анализ результатов эксперимента.

Типовая структура основной части работ для тем по стратегии ОТМЗ (сравнение эффективности технологии «Технология 1» в решении задач «Задача 1» и «Задача 2»):

1. Постановка задачи на исследование.

1.1. Практическая значимость технологии.

1.2. Обзор методов решения задач.

1.3. Реализация последовательной программы «Задача 1».
1.4. Реализация последовательной программы «Задача 2».

2. Проектирование и разработка параллельных программ.

2.1. Разработка параллельных алгоритмов.
2.2. Реализация параллельной программы «Технология 1» для «Задача 1».
2.3. Реализация параллельной программы «Технология 1» для «Задача 2».

3. Экспериментальное исследование эффективности разработанного программного обеспечения.

3.1. Методика оценки эффективности программного обеспечения.
3.2. Оценка воспроизводимости эксперимента.
3.3. Планирование и проведение экспериментальных вычислений.
3.4. Анализ результатов эксперимента.

По согласованию с руководителем курсовой работы допускается обоснованное изменение типовой структуры основной части работы.

ТРЕБОВАНИЯ К СОДЕРЖАНИЮ КУРСОВОЙ РАБОТЫ

В курсовой работе используется академический стиль письма, которому, в частности, присущи:

– безличная форма (не «мы рассчитали», а «рассчитано»);
– отсутствие необоснованной оценочной составляющей (не «безусловно, наилучшим методом расчета является», а «методом расчета является» или «одним из методов расчета является»);
– отсутствие прямого обращения к читателю (не «вы можете видеть», а «видно»).

Заимствованные фрагменты должны быть взяты в кавычки, после которых размещается ссылка на источник. Сам источник с указанием URL должен присутствовать в списке используемой литературы и используемых источников.

В общем случае результаты расчётов округлять до трех значащих цифр. Исключение составляют случаи, когда значащие цифры после третьей имеют принципиальное значение, например, при оценке точности расчета числа Пифагора.

В конце каждого раздела приводятся выводы по разделу, в которых в краткой форме отражаются результаты расчёта, полученные в разделе.

Введение

Введение должно содержать:

– вводный текст (общее описание решаемой задачи, отражающее ее актуальность);
– формулировку предмета исследования (это должен быть процесс, например, «процесс решения...»);
– формулировку объекта исследования (это, как правило, программа, модуль, информационная система);
– цель курсовой работы («разработать параллельную программу для...»);
– задачи, решаемые в ходе выполнения курсовой работы для достижения указанной цели (как правило, задачи соответствуют разделам работы);

– краткое описание структуры работы («работа состоит из трех разделов. В первом разделе выполнено...»).

Пример вводного текста (стратегия МТОЗ): «Умножение матриц является распространенной операцией компьютерной обработки информации, встречающейся в таких вычислительных задачах, как обработка изображений, машинное обучение и т. п. Производители вычислительных машин стараются увеличивать производительность своих продуктов. Они могут увеличивать ускорение вычислений посредством ускорения центральных процессоров или посредством увеличения их количества. Из-за определенных ограничений, таких как увеличение потребляемой энергии и ограничение пропускной способности, производители компьютеров стали увеличивать производительность за счет увеличения числа чипов и увеличения числа ядер в одном чипе. В свете изложенного работа, направленная на применение многопоточных технологий к операции матричного умножения, является актуальной».

Далее приведен пример описания объекта, предмета, цели и задач работы (стратегия ОТМЗ):

«Предмет исследования – увеличение производительности численного интегрирования за счет распараллеливания вычислений.

Объект исследования – программы численного интегрирования методом трапеций и методом Монте-Карло по технологии OpenMP.

Цель курсовой работы – повышение производительности численного интегрирования методом трапеций и методом Монте-Карло по технологии OpenMP.

Для достижения цели в курсовой работе решены следующие задачи:

– проведен анализ теоретических основ численного интегрирования и технологий многопоточных вычислений;
– разработаны программы, реализующие численное интегрирование методом трапеций и методом Монте-Карло в один поток и многопоточные по технологии OpenMP;
– выполнено теоретическое и экспериментальное исследование сравнительной эффективности разработанных программ».

Далее приведен пример краткого описания структуры работы (стратегия МТОЗ):

«Работа состоит из трех разделов. В первом разделе выполнено описание теоретических основ и областей практического применения операции нахождения обратной матрицы, разработаны алгоритм и однопоточная программа, проведено тестирование программы.

Во втором разделе разработаны, запрограммированы и протестированы два параллельных алгоритма нахождения обратной матрицы по технологиям OpenMP, C++11 Threads.

В третьем разделе проведен теоретический анализ эффективности разработанных параллельных программ, составлен план компьютерного эксперимента по исследованию эффективности, проведен анализ результатов эксперимента».

Постановка задачи на исследование (по стратегии МТОЗ)

В подразделе 1.1 «Практическая значимость задачи» необходимо показать актуальность задачи, выполнить обзор современных научных разработок по задаче, определить области применения задачи.

Пример выполнения подраздела 1.1 (фрагмент): «Обратная матрица – это такая матрица, при умножении на которую исходная матрица даст единичную матрицу.

Операция вычисления обратной матрицы широко используется в обратных задачах – задачах, где значения параметров модели должны быть получены из наблюдаемых данных. Обратная задача часто возникает в таких областях, как геофизика, астрономия, медицинская визуализация, компьютерная томография, дистанционное зондирование Земли, спектральный анализ, теория рассеяния и задачи по неразрушающему контролю.

Сами по себе матрицы получили широкое распространение. Матрицу можно использовать везде, где данные можно записать в виде прямоугольной таблицы. Так, в программировании матрицы – это двумерный массив, в математике матрицы применяются для компактной записи и решения системы линейных и дифференциальных уравнений, в экономике математическую модель экономических объектов и процессов удобно записывать в матричном виде, в реляционных базах данных все данные представляются в виде таблиц».

В подразделе 1.2 «Обзор методов решения задачи» необходимо выполнить математическое описание задачи и методов ее решения.

Пример выполнения подраздела 1.2: «Для того чтобы вычислить определённый интеграл

$$\int_a^b f(x)dx \tag{1}$$

методом Монте-Карло, рассматривают случайную величину u, равномерно распределённую на отрезке интегрирования $[a, b]$. Тогда $f(u)$ также будет случайной величиной, причём её математическое ожидание выражается формулой:

$$Ef(u) = \int_a^b f(x)\varphi(x)dx, \tag{2}$$

где $\varphi(x)$ – плотность распределения случайной величины u, равная $\frac{1}{b-a}$.

Таким образом, искомый интеграл выражается формулой:

$$\int_a^b f(x)dx = (b-a)Ef(u). \tag{3}$$

Математическое ожидание случайной величины $f(u)$ можно оценить, смоделировав эту случайную величину и посчитав выборочное среднее. Для этого берется N точек, равномерно распределённых на $[a, b]$, для каждой точки u_i вычисляется $f(u_i)$. Затем вычисляется выборочное среднее:

$$\frac{1}{N}\sum_{i=1}^{N} f(u_i). \tag{4}$$

В итоге оценка интеграла:

$$\int_a^b f(x)dx \approx \frac{b-a}{N}\sum_{i=1}^{N} f(u_i). \tag{5}$$

Точность оценки зависит только от количества точек N {\displaystyle N}N. Этот метод имеет и геометрическую интерпретацию. Он очень похож на описанный выше метод, с той разницей, что вместо равномерного разделения области интегрирования на маленькие интервалы и суммирования площадей получившихся «столбиков» область интегрирования забрасывается случайными точками, на каждой из которых строится такой же «столбик» с определением его ширины по формуле

$$\frac{b-a}{N}.$$

Метод Монте-Карло для вычисления одномерных интегралов обычно не применяется, так как для получения высокой точности более удобны квадратурные формулы. Этот метод оказывается более эффективным при вычислении кратных интегралов, когда кубатурные формулы для достижения малой погрешности слишком громоздки и требуют большого объема вычислений.

Формулы метода Монте-Карло для вычисления кратных интегралов:

– для двойного интеграла:

$$\int_a^b \int_c^d f(x,y)dxdy = \frac{(b-a)(d-c)}{N} \sum_{i=1}^{N} f(x_i, y_i); \tag{6}$$

– для m-кратного интеграла:

$$\int_{a_1}^{b_1} \int_{a_2}^{b_2} \dots \int_{a_m}^{b_m} f(x^1, x^2, \dots x^m)\, dx^1 dx^2 \dots dx^m \approx$$
$$\approx \frac{(b_1 - a_1)(b_2 - a_2) \dots (b_m - a_m)}{N} \sum_{i=1}^{N} f\left(x_i^1, x_i^2 \dots x_i^m\right). \tag{7}$$

Для реализации последовательного алгоритма расчета интеграла по методу Монте-Карло необходимо задать кратность интеграла, область интегрирования и количество итераций вычисления».

В подразделе 1.3 «Реализация последовательной программы» должны присутствовать следующие элементы:

– текстовое описание алгоритма;

– графическое изображения алгоритма;

– текстовое описание выполняемого кода;

– фрагменты кода (полный код приводится в приложении);

– текстовое описание условий тестирования;

– результаты тестирования (снимки экрана, графики, таблицы. Снимки экрана должны быть читаемыми. Области, не содержащие информации, необходимо обрезать. Снимки экранов последовательных запусков программы на разных настройках рекомендуется объединять на одном рисунке);

– вывод по результатам тестирования (в тестирование по времени расчета для однопоточной программы следует в качестве базового для тестов многопоточных программ принять результат с временем расчета от 10 до 60 секунд).

Пример текстового описания алгоритма (фрагмент): «На начальном этапе создается матрица размером N на N и заполняется случайными числами. Далее объявляются используемые в вычислениях переменные, после чего начинается обход матрицы в цикле по переменным i от 0 до $N - 1$ и j, которые отвечают за индексы матрицы. Далее нужно получить значение переменной, на которое впоследствии будет умножаться элемент матрицы. Если элемент матрицы, на который необходимо делить, окажется равным нулю, то происходит обнуление этой переменной. Если этот элемент не равен нулю, то выполняется деление. После этого шага выполняется обход столбцов матрицы по переменной k от i до $N - 1$. Затем вычитается из элемента с индексами j, k элемент матрицы с индексами i, k, умноженный на посчитанную ранее переменную».

Пример графического изображения алгоритма показан на рисунке 1.

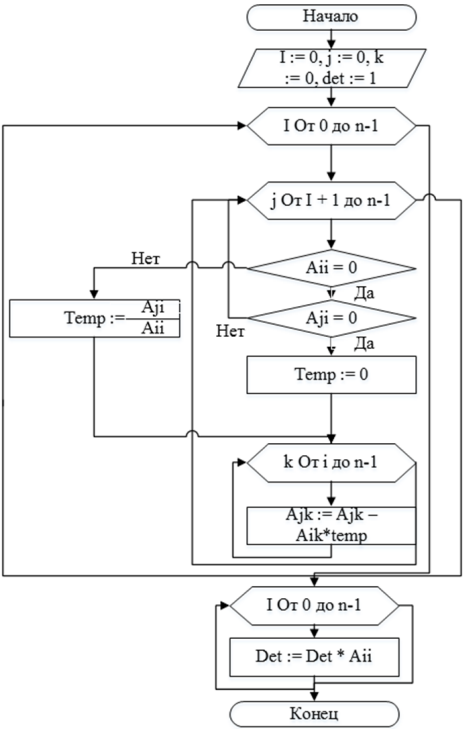

Рисунок 1 – Последовательное нахождение определителя матрицы

Пример текстового описания выполняемого кода и фрагментов кода (фрагмент): «Для реализации последовательной программы на языке C++ программа должна иметь методы для работы с матрицами:

– выделение памяти для матрицы;
– вывод матрицы на экран;
– получение копии матрицы;
– заполнение матрицы случайными числами;
– освобождение памяти, выделенной на матрицу.

Для выделения памяти для матрицы реализуется функция allocateMemory. Сначала выделяется память под массив указателей, а потом в цикле для каждого указателя выделяется свой участок памяти. В данной функции выделяется память для $N \times N$ элементов матрицы типа double (листинг 1).

Листинг 1 – функция выделения памяти для матрицы:

```
double** allocateMemory(int N) // выделить память под массив
{
    double **array = new double*[N];
    for (int i = 0; i < N; ++i)
        array[i] = new double[N];
        return array;
}
```

Для получения копии матрицы реализована функция getCopyMatrix. В данной функции выделяется память для новой матрицы с помощью функции allocateMemory, новая матрица заполняется элементами изначальной матрицы в двух вложенных циклах (листинг 2).

Листинг 2 – функция получения копии матрицы:

```
double **getCopyMatrix(double **matrix, int N) // копирует
матрицу
{
      double** returnMatrix = allocateMemory(N);
            for (int i = 0; i < N; ++i)
            for (int j = 0; j < N; ++j)
                  returnMatrix[i][j] = matrix[i][j];
      return returnMatrix;
}»
```

Пример текстового описания и результатов тестирования: «Для тестирования корректности программы умножения матрицы

на вектор использована матрица размером 4×4 и вектор из четырех элементов, заполненные случайными целыми от 0 до 50. Результаты расчета (рисунок 2) показывают, что умножение матрицы на вектор проведено правильно.

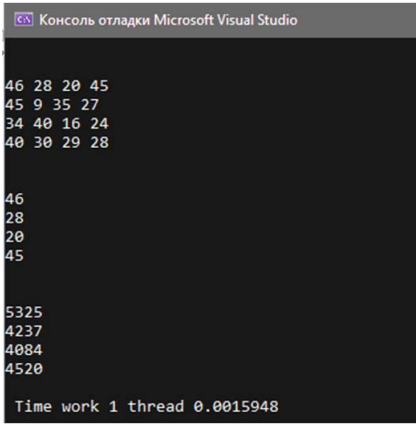

Рисунок 2 – Результаты тестирования корректности умножения матрицы на вектор

Для тестирования по времени расчёта использован системный таймер Visual Studio и отключен вывод на экран исходных массивов и результатов расчета. Результаты тестирования по времени расчета представлены в таблице 1.

Таблица 1 – Результаты тестирования по времени

Размер матрицы	Время расчёта, с
4 × 4	0,00159
400 × 400	0,0117
2000 × 2000	0,304
5000 × 5000	1,89
10000 × 10000	25,9
15000 × 15000	192

По результатам тестирования однопоточной программы умножения матрицы на вектор в качестве базового размера задачи выбран размер матрицы 10000 × 10000 со временем расчета в один поток 25,9 с».

Постановка задачи на исследование (по стратегии ОТМЗ)

В подразделе 1.1 «Практическая значимость технологии» необходимо:

– дать краткое описание применяемой вычислительной технологии;
– проанализировать особенности, достоинства и недостатки применяемой вычислительной технологии в сравнении с аналогами;
– описать область применения технологии.

В подразделе 1.2 «Обзор методов решения задач» необходимо выполнить математическое описание задач (задачи) и методов решения. Должно быть описано не менее двух методов.

В подразделе 1.3 «Реализация последовательной программы» должны присутствовать следующие элементы:

– текстовое описание алгоритма;
– графическое изображения алгоритма;
– текстовое описание выполняемого кода;
– фрагменты кода (полный код приводится в приложении);
– текстовое описание условий тестирования;
– результаты тестирования (снимки экрана, графики, таблицы. Снимки экрана должны быть читаемыми. Области, не содержащие информации, необходимо обрезать. Снимки экранов последовательных запусков программы на разных настройках рекомендуется объединять на одном рисунке);
– вывод по результатам тестирования.

Структура подраздела 1.4 идентична структуре подраздела 1.3 (примеры приведены на с. 13–16). Результаты тестирования по времени должны покрывать диапазон времени расчета от 2 до 120 с. Базовый размер задач для последующего тестирования параллельных программ следует по возможности выбирать одинаковым.

Проектирование и разработка параллельных программ (по стратегии МТОЗ)

Подраздел 2.1 «Обзор технологий разработки параллельного программного обеспечения» должен содержать краткий обзор (3–5 страниц) с указанием достоинств и недостатков не менее трех технологий, а также обоснование выбора технологии параллельного программирования для последующей реализации параллельной программы.

Подраздел 2.2 «Разработка параллельных алгоритмов» должен содержать:

– анализ последовательных задач (рассмотреть параллельность по данным, по коду);
– обоснование парадигмы параллельного программирования;
– обоснование критических секций, условных блокировок, использования библиотек, мьютексов, семафоров, мониторов, блокировок, потокобезопасных коллекций и т. п.

Пример выполнения подраздела 2.2: «Код последовательной программы состоит из нескольких циклов. Выполнение программы происходит итеративно, размер матрицы известен, поэтому для распараллеливания выбрана парадигма итеративного параллелизма. При итеративном параллелизме несколько процессов будут выполнять циклические вычисления.

Из технологии OpenMP потребуется следующий функционал:

– #pragma omp parallel for – создание параллельной области для распараллеливания цикла;
– reduction – для подсчёта значений. К переменной применяется оператор, указанный в скобках;
– private, shared – для разделения данных на частные и общие.

Для распараллеливания с помощью C++11 Parallel Technologies потребуется:

– создать поток с помощью конструктора класса thread;
– использование метода join() для синхронизации потоков;
– использование объектов класса mutex для разграничения доступа к хранилищу для потоков, что даёт устойчивость к гонке данных».

Подраздел 2.3 «Реализация параллельной программы „Технология 1“» должен содержать:

– текстовое описание параллельного алгоритма;
– графическое изображение параллельного алгоритма;
– текстовое описание параллельного выполняемого кода;
– фрагменты параллельного кода (полный код приводится в приложении);
– текстовое описание условий тестирования (на корректность и на время расчета);
– результаты тестирования (снимки экрана, графики, таблицы);
– вывод по результатам тестирования.

Пример текстового описания параллельного алгоритма: «Последовательная программа основывается на циклах с параметром. Для параллельного выполнения программы необходимо написать перед циклом #pragma omp parallel for (рисунок 3). Также необходимо использовать дополнительные директивы reduction, private, shared.

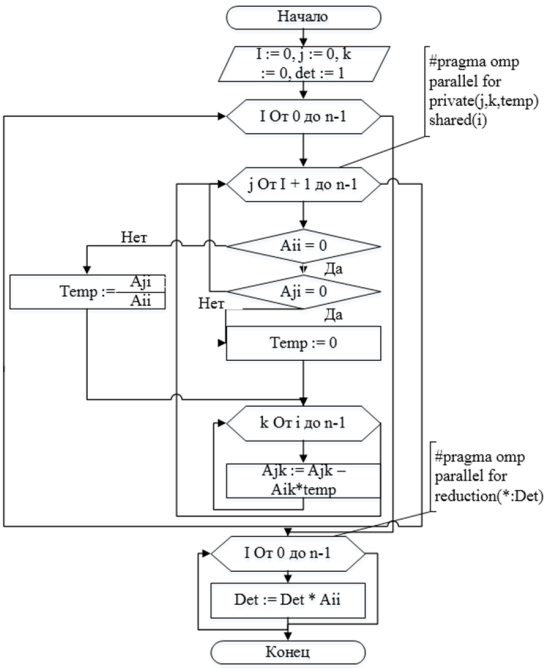

Рисунок 3 – Схема параллельного алгоритма расчета определителя на OpenMP

Директива private применяется к переменным *j*, *k*, *temp* для создания локальной копии в каждом потоке. Директива shared используется для определения переменной *i* в качестве общей для потоков. Директива reduction применяется при вычислении определителя с применением оператора умножения».

Пример описания тестирования (фрагмент): «Для тестирования параллельной программы численного интегрирования методом трапеций выбран интеграл <...> в пределах от <...> до <...>. Аналитическое решение данного интеграла равно 2,5.

Для тестирования на корректность расчёта выбран диапазон интервалов от 5 до 10 с шагом 1. Результаты расчета приведены в таблице 2.

Таблица 2 – Результаты тестирования на корректность

Число интервалов	Результат расчета	Абсолютное отклонение
5	2,04	0,46
6	2,15	0,35
7	2,24	0,26
8	2,64	0,14
9	2,61	0,11
10	2,44	0,06

Успешность теста оценивается графически (рисунок 4) по схождению расчетного значения интеграла к аналитическому значению.

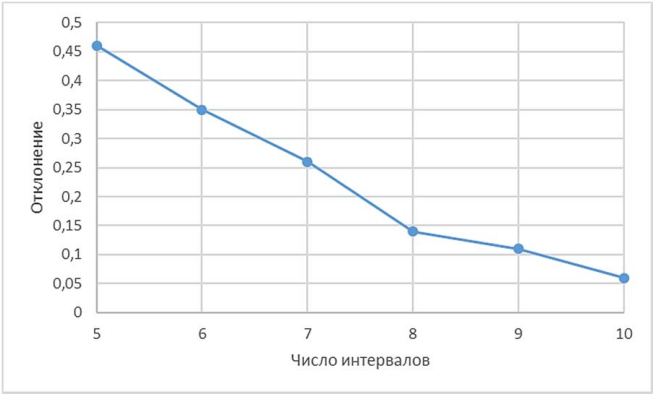

Рисунок 4 – Зависимость отклонения от числа интервалов

Для тестирования по времени расчета выбрано число интервалов 20 000. Время расчета составило 16 с. Снимок экрана с результатом показан на рисунке (номер рисунка). При тестировании однопоточной программы время расчета составило 32 с.

По результатам тестирования многопоточной программы расчета интеграла методом трапеций по технологии openMP установлено:

– многопоточная программа корректна, результат приближается к аналитическому при увеличении числа интервалов;
– многопоточная программа эффективна, расчет выполняется в 2 раза быстрее, чем в один поток».

Подраздел 2.4 «Реализация параллельной программы „Технология 2“» по структуре аналогичен подразделу 2.3.

Проектирование и разработка параллельных программ (по стратегии ОТМЗ)

Подраздел 2.1 «Разработка параллельных алгоритмов» по структуре аналогичен подразделу 2.2 (с. 18).

Подраздел 2.2 «Реализация параллельной программы „Технология 1“» для задачи «Задача 1» по структуре аналогичен подразделу 2.3 (с. 19–21).

Подраздел 2.3 «Реализация параллельной программы „Технология 1“» для задачи «Задача 2» по структуре аналогичен подразделу 2.3 (с. 19–21).

Экспериментальное исследование эффективности разработанного программного обеспечения

Подраздел 3.1 «Разработка методики оценки эффективности программного обеспечения» должен содержать:

– текстовое описание содержания работ;
– описание программно-аппаратного обеспечения (конфигурация компьютера и используемая ОС, число программных процессоров);
– варьируемые параметры, пределы и интервалы варьирования;
– измеряемые величины (время выполнения программы);
– расчётные величины (коэффициент эффективности, коэффициент ускорения).

В текстовом описании содержания работ приводятся информация по сравниваемым программам и цель экспериментального исследования.

Пример текстового описания (стратегия МТОЗ): «Для оценки эффективности представлены 3 программы. Введем обозначения для каждой из программ:

1. T0 – программа численного интегрирования методом трапеций последовательная.
2. T1 – программа численного интегрирования методом трапеций параллельная по технологии openMP.
3. T2 – программа численного интегрирования методом трапеций параллельная по технологии C++11 Threads.

Целью экспериментального исследования является оценка эффективности использования технологий параллельных вычислений openMP и C++11 Threads в задаче численного интегрирования методом трапеций».

Пример текстового описания (стратегия ОТМЗ): «Для оценки эффективности представлены 4 программы. Введем обозначения для каждой из программ:

1. T0 – программа численного интегрирования методом трапеций последовательная.
2. M0 – программа численного интегрирования методом Монте-Карло последовательная.
3. T1 – программа численного интегрирования методом трапеций параллельная по технологии openMP.
4. M1 – программа численного интегрирования методом Монте-Карло параллельная по технологии openMP.

Целью экспериментального исследования является оценка эффективности использования параллельных вычислений в задачах численного интегрирования методом трапеций и методом Монте-Карло».

В ходе описания программно-аппаратного обеспечения (конфигурация компьютера и используемая ОС, число процессоров) можно использовать как встроенные средства операционной системы, так и внешние программы, такие как, например, CPU-Z. Информация о системе, на базе которой выполняется эксперимент,

может быть представлена как в текстовом виде, так и в форме снимков экрана соответствующих приложений. По центральному процессору должна быть приведена следующая информация: модель, число физических ядер, число логических потоков (программных процессоров).

Варьируемые параметры, пределы и интервалы варьирования задаются относительно базового размера задачи (задач), относительно которого было рассчитано время выполнения при тестировании программ в разделах 1 и 2 курсовой работы.

Варьируемые параметры:

– для стратегии МТОЗ выбирается два варьируемых параметра, например число интервалов и число вычислительных потоков (двухфакторный компьютерный эксперимент);
– для стратегии ОТМЗ выбирается один варьируемый параметр (однофакторный компьютерный эксперимент).

Пределы варьирования задаются относительно базового размера задачи (задач), относительно которого было рассчитано время выполнения при тестировании программ в разделах 1 и 2 курсовой работы. При выборе пределов варьирования рекомендуется путём предварительного прогона программ на предельных настройках убедиться, что время расчета составляет не менее 1 секунды и не превышает 2 минут.

Интервалы варьирования:

– для стратегии МТОЗ (двухфакторный эксперимент) задаются как разность между базовым и предельным значениями параметра;
– для стратегии ОТМЗ (однофакторный эксперимент) вводятся дополнительные экспериментальные точки, в них варьируемые параметры принимают среднее значение между предельным и базовым. Интервалы варьирования задаются как полуразность между базовым и предельным значениями параметра.

Пример описания варьируемых параметров, пределов и интервалов в стратегии МТОЗ: «В качестве варьируемых параметров выбраны число интервалов n и количество потоков p (двухфакторный эксперимент). Тестирование многопоточных программ, проведенное в разделе 2 при базовых значениях $n_Б = 10\,000$ и $p_Б = 4$, показало, что время расчета по технологии openMP составляет 6,81 секунд,

по технологии C++11 Threads – 6,99 секунд. Однопоточная программа, как показано при тестировании в разделе 1, при $n_б = 10\,000$ выполняется за 20,49 секунд.

При выборе пределов варьирования начальное значение пределов:

– верхний предел по числу интервалов $n+ = 30\,000$;
– нижний предел по числу интервалов $n– = 5000$;
– верхний предел по числу потоков $p+ = 6$;
– нижний предел по числу потоков $p– = 2$.

Наибольшее время расчета ожидается при сочетании $n+$ и $p–$. По результатам предварительных прогонов программ на соответствующих значениях получено время расчета для технологии openMP 102,46 секунды, для C++11 Threads – 104,28 секунд.

Наименьшее время расчета ожидается при сочетании $n–$ и $p+$. По результатам предварительных прогонов программ на соответствующих значениях получено время расчета для технологии openMP – 1,83 секунды, для C++11 Threads – 2,01 секунды.

Время расчета в целом соответствует минимальному требуемому времени, и начальные значения пределов варьирования по $n–$ и $p+$ приняты как окончательные.

Окончательные пределы варьирования:

– верхний предел по числу интервалов $n+ = 30\,000$;
– нижний предел по числу интервалов $n– = 5000$;
– верхний предел по числу потоков $p+ = 6$;
– нижний предел по числу потоков $p–= 2$.

Значения интервалов и пределов варьирования параметров приведены в таблице 3.

Таблица 3 – Варьируемые параметры

Характеристика варьирования	Число потоков, p	Число интервалов, n
Базовый уровень	4	10 000
Верхний предел варьирования	6	30 000
Нижний предел варьирования	2	5000
Интервал варьирования, верхний/нижний	2/2	20 000/5000

Два параметра варьируются на трех уровнях».

Пример описания варьируемых параметров, пределов и интервалов в стратегии ОТМЗ: «В качестве варьируемого параметра выбрано число интервалов n (однофакторный эксперимент). Тестирование многопоточных программ, проведенное в разделе 2 при базовых значениях $n_Б = 10\,000$ по технологии openMP показало, что время расчета для метода трапеций составляет 6,81 секунд, для метода Монте-Карло – 5,16 секунд. Однопоточные программы, как показано при тестировании в разделе 1, при $n_Б = 10\,000$ выполняются за 20,49 секунд для метода трапеций и за 14,95 секунд для метода Монте-Карло.

При выборе пределов варьирования начальное значение пределов:

– верхний предел по числу интервалов $n+ = 30\,000$;

– нижний предел по числу интервалов $n– = 5000$.

Наибольшее время расчета ожидается при $n+$. По результатам предварительных прогонов программ по технологии openMP на соответствующих значениях получено время расчета 56,75 секунды для метода трапеций и 38,04 секунд для метода Монте-Карло.

Наименьшее время расчета ожидается при $n–$. По результатам предварительных прогонов программ на соответствующих значениях получено время расчета для метода трапеций 1,77 секунды, для метода Монте-Карло – 1,34 секунды.

Время расчета в целом соответствует минимальному требуемому времени, и начальные значения предела варьирования по $n–$ приняты как окончательные.

Окончательные пределы варьирования:

– верхний предел по числу интервалов $n+ = 30\,000$;

– нижний предел по числу интервалов $n– = 5000$.

Значения интервалов и пределов варьирования параметров приведены в таблице 4.

Промежуточные значения варьируемых параметров приняты как среднее значение между предельным и базовым значениями».

Таблица 4 – Варьируемые параметры

Характеристика варьирования	Число интервалов, n
Базовый уровень	10 000
Верхний предел варьирования	30 000
Нижний предел варьирования	5000
Интервал варьирования, верхний/нижний	10 000/2500

При описании измеряемых величин (в данной курсовой работе это время выполнения программы) необходимо описать программные конструкции по определению времени выполнения или используемые средства измерения времени, такие как профайлеры.

Расчетные величины – это:

– время, затраченное на реализацию последовательного алгоритма T_1;
– время, затраченное на реализацию параллельного алгоритма T_p;
– коэффициент ускорения параллельных вычислений $R_p = T_1 / T_p$;
– коэффициент эффективности распараллеливания $E_p = R_p / p$ для различного числа программных процессоров p.

Подраздел 3.2 «Оценка воспроизводимости эксперимента» должен содержать:

– результаты повтора экспериментов в центре плана;
– анализ воспроизводимости эксперимента.

Целью оценки воспроизводимости является получение информации о разбросе результатов при повторе экспериментов на одних исходных данных. Чаще всего значительный разброс при повторах имеет место при исследовании программ сортировки при рандомном задании исходных массивов.

Результаты повтора экспериментов в центре плана должны содержать результаты замеров времени выполнения программы на исходных данных, соответствующих базовым. Должно быть выполнено не менее 5 повторов.

На основании результатов повторов в центре плана выполняется анализ воспроизводимости. В случае, если размах повторов в центре плана не превышает 1 % от среднего значения, отклонения на повторах следует полагать незначительными и строить план

эксперимента без повторов. В случае, если размах составляет от 1 до 5 %, отклонения на повторах следует полагать существенными и строить план эксперимента с тремя повторами. В случае, если размах составляет более 5 %, отклонения на повторах следует полагать значительными и строить план эксперимента с пятью повторами.

Примером таблицы для анализа воспроизводимости экспериментов в стратегии МТОЗ является таблица 5, для стратегии ОТМЗ – таблица 6.

Таблица 5 – Пример результатов повторных расчетов численного интегрирования методом трапеций при числе интервалов 10 000

Наименование	Варианты исполнения программы		
	Один поток	Многопоточное, технология openMP	Многопоточное, технология C++ Threads
Время повтора 1, с	20,15	6,77	6,95
Время повтора 2, с	20,44	7,01	7,07
Время повтора 3, с	20,67	6,92	6,91
Время повтора 4, с	20,45	6,85	7,09
Время повтора 5, с	20,67	6,92	6,95
Среднее время, с	20,51	6,85	6,96
Размах, с	0,52	0,24	0,18
Отношение размаха к среднему, %	2,55	3,45	2,64

Таблица 6 – Пример результатов времени повторных расчетов по технологии openMP при числе интервалов 10 000

Наименование	Варианты исполнения программы			
	Один поток методом трапеций	Один поток методом Монте-Карло	Многопоточное методом трапеций	Многопоточное методом Монте-Карло
Время повтора 1, с	20,01	15,17	6,85	5,20
Время повтора 2, с	20,09	14,92	6,93	5,19
Время повтора 3, с	20,52	15,08	6,91	5,22
Время повтора 4, с	20,43	15,24	6,71	5,23

Окончание таблицы 6

Наименование	Варианты исполнения программы			
	Один поток методом трапеций	Один поток методом Монте-Карло	Многопоточное методом трапеций	Многопоточное методом Монте-Карло
Время повтора 5, с	20,95	15,03	6,80	5,22
Среднее время, с	20,80	15,47	6,77	4,97
Размах	0,95	0,32	0,22	0,04
Отношение размаха к среднему, %	4,55	2,06	3,27	0,85

Подраздел 3.3 «Планирование и автоматизация экспериментальных вычислений» должен содержать:

– описание плана эксперимента;

– план и результаты эксперимента.

Далее приведен пример подраздела 3.3 для стратегии МТОЗ (фрагмент).

Экспериментальное исследование эффективности параллельных программ вычисления интеграла методом трапеций состоит в последовательном расчете значений интеграла... при варьировании числа интервалов и количества потоков в соответствии с уровнями и интервалами варьирования (таблица...) по три повтора каждого эксперимента.

План двухфакторного эксперимента и результаты замеров времени выполнения программ представлены в таблицах 7–9.

Таблица 7 – План эксперимента и время выполнения программы в один поток C++

Номер опыта	Число интервалов, *n*	Уровень варьирования	Повтор 1, с	Повтор 2, с	Повтор 3, с	Среднее, с
1	5000	–	5,04	5,27	4,95	5,09
2	10 000	0	20,51	20,79	20,63	20,64
3	30 000	+	184,62	184,72	184,56	184,63

Таблица 8 – План эксперимента и время выполнения программы по технологии openMP

Номер опыта	Число интервалов, *n*	Уровень варьирования	Число потоков, *p*	Уровень варьирования	Повтор 1, с	Повтор 2, с	Повтор 3, с	Среднее, с
1	5000	–	2	–	3,14	3,18	3,22	3,18
2	10 000	0	2	–	12,83	12,89	12,88	12,86
3	30 000	+	2	–	102,72	102,50	102,40	102,54
4	5000	–	4	0	1,77	1,73	1,79	1,77
5	10 000	0	4	0	6,71	6,87	6,81	6,80
6	30 000	+	4	0	56,60	56,71	56,59	56,63
7	5000	–	6	+	1,81	1,79	1,83	1,81
8	10 000	0	6	+	7,04	7,14	7,03	7,07
9	30 000	+	6	+	63,42	63,28	63,87	63,52

Таблица 9 – План эксперимента и время выполнения программы по технологии C++11 Threads

Номер опыта	Число интервалов, *n*	Уровень варьирования	Число потоков, *p*	Уровень варьирования	Повтор 1, с	Повтор 2, с	Повтор 3, с	Среднее, с
1	5000	–	2	–	3,20	3,17	3,17	3,18
2	10000	0	2	–	13,18	13,28	13,34	13,27
3	30000	+	2	–	104,12	104,16	104,66	104,31
4	5000	–	4	0	1,98	1,94	1,94	1,95
5	10000	0	4	0	7,07	7,03	7,09	7,06
6	30000	+	4	0	62,89	63,13	62,88	62,97
7	5000	–	6	+	1,96	2,01	2,05	2,01
8	10000	0	6	+	7,27	7,40	7,43	7,37
9	30000	+	6	+	66,81	66,96	66,98	66,92

Далее приведен пример подраздела 3.3 для стратегии ОТМЗ (фрагмент).

Экспериментальное исследование эффективности технологии openMP при вычислении интеграла методами трапеций и Монте-Карло состоит в последовательном расчете значений интеграла... при варьировании числа интервалов в соответствии с уровнями и интервалами варьирования (таблица...) по три повтора каждого эксперимента.

План однофакторного эксперимента и результаты расчётов представлены в таблицах 10–13.

Таблица 10 – План эксперимента и время выполнения однопоточной программы вычисления интеграла методом трапеций

Номер опыта	Число интервалов, *n*	Уровень варьирования	Повтор 1, с	Повтор 2, с	Повтор 3, с	Среднее, с
1	5000	–	4,81	4,93	5,22	4,99
2	7500	–0	11,77	11,91	11,85	11,85
3	10 000	0	20,14	20,00	20,65	20,26
4	20 000	0+	81,80	81,69	81,70	81,73
5	30 000	+	184,77	184,51	184,54	184,61

Таблица 11 – План эксперимента и время выполнения однопоточной программы вычисления интеграла методом Монте-Карло

Номер опыта	Число интервалов, *n*	Уровень варьирования	Повтор 1, с	Повтор 2, с	Повтор 3, с	Среднее, с
1	5000	–	3,98	3,96	4,30	4,08
2	7500	–0	8,44	8,56	8,57	8,52
3	10 000	0	14,89	14,61	14,85	14,79
4	20 000	0+	53,55	54,20	53,73	53,82
5	30 000	+	113,72	113,74	113,66	113,71

Таблица 12 – План эксперимента и время выполнения многопоточной программы вычисления интеграла методом трапеций

Номер опыта	Число интервалов, *n*	Уровень варьирования	Повтор 1, с	Повтор 2, с	Повтор 3, с	Среднее, с
1	5000	–	1,78	1,76	1,81	1,78
2	7500	–0	3,87	3,88	3,87	3,87
3	10 000	0	6,78	6,82	6,80	6,80
4	20 000	0+	26,41	26,43	26,47	26,44
5	30 000	+	56,74	56,72	56,79	56,75

Таблица 13 – План эксперимента и время выполнения многопоточной программы вычисления интеграла методом Монте-Карло

Номер опыта	Число интервалов, *n*	Уровень варьирования	Повтор 1, с	Повтор 2, с	Повтор 3, с	Среднее, с
1	5000	–	1,30	1,37	1,34	1,33
2	7500	–0	2,82	2,87	2,83	2,84
3	10 000	0	5,13	5,14	5,16	5,15
4	20 000	0+	18,03	18,01	18,07	18,04
5	30 000	+	38,05	38,04	38,08	38,05

Подраздел 3.4 «Анализ результатов эксперимента» должен содержать:

– расчет показателей эффективности по результатам измерений;

– графическое представление и анализ результатов.

Необходимо провести анализ полученных результатов. Прокомментировать выявленное поведение контролируемых факторов при изменении варьируемых параметров.

Расчет показателей эффективности по результатам измерений выполняется от средних значений, полученных в ходе экспериментов по формулам, представленным на с. 26 настоящего пособия.

Результаты расчета представляются в табличной форме.

Примером таблиц расчетных величин для стратегии МТОЗ являются таблицы 14, 15.

Таблица 14 – Коэффициенты эффективности работы программы по технологии openMP

Число интервалов, *n*	Число процессоров, *p*	Время расчета, с		Коэффициент ускорения	Коэффициент эффективности
5000	1	5,09	–	–	–
10 000	1	20,64	–	–	–
30 000	1	184,63	–	–	–
5000	2	–	3,18	1,60	0,80
10 000	2	–	12,86	1,60	0,80
30 000	2	–	102,54	1,80	0,90
5000	4	–	1,77	2,88	0,72
10 000	4	–	6,8	3,04	0,76
30 000	4	–	56,63	3,26	0,82
5000	6	–	1,81	2,81	0,47
10 000	6	–	7,07	2,92	0,49
30 000	6	–	63,52	2,91	0,48
Среднее				2,54	0,69

Таблица 15 – Коэффициенты эффективности работы программы по технологии C++ Threads

Число интервалов, *n*	Число процессоров, *p*	Время расчета, с		Коэффициент ускорения	Коэффициент эффективности
5000	1	5,09	–	–	–
10 000	1	20,64	–	–	–
30 000	1	184,63	–	–	–
5000	2	–	3,18	1,60	0,80
10 000	2	–	13,27	1,56	0,78
30 000	2	–	104,31	1,77	0,89
5000	4	–	1,95	2,61	0,65

Окончание таблицы 15

Число интервалов, n	Число процессоров, p	Время расчета, с		Коэффициент ускорения	Коэффициент эффективности
10 000	4	–	7,06	2,92	0,73
30 000	4	–	62,97	2,93	0,73
5000	6	–	2,01	2,53	0,42
10 000	6	–	7,37	2,80	0,47
30 000	6	–	66,92	2,76	0,46
Среднее				2,39	0,66

Примером таблиц расчетных величин для стратегии ОТМЗ являются таблицы 16, 17.

Таблица 16 – Коэффициенты эффективности работы программы численного интегрирования методом трапеций

Число интервалов, n	Число процессоров, p	Время расчета, с		Коэффициент ускорения	Коэффициент эффективности
5000	1	4,99	–	–	–
7500	1	11,85	–	–	–
10 000	1	20,26	–	–	–
20 000	1	81,73	–	–	–
30 000	1	184,61	–	–	–
5000	4	–	1,78	2,80	0,70
7500	4	–	3,87	3,06	0,77
10 000	4	–	6,8	2,98	0,74
20 000	4	–	26,44	3,09	0,77
30 000	4	–	56,75	3,25	0,81
Среднее				3,04	0,76

Таблица 17 – Коэффициенты эффективности работы программы численного интегрирования методом Монте-Карло

Число интервалов, *n*	Число процессоров, *p*	Время расчета, *c*		Коэффициент ускорения	Коэффициент эффективности
5000	1	4,08	–	–	–
7500	1	8,52	–	–	–
10 000	1	14,79	–	–	–
20 000	1	53,82	–	–	–
30 000	1	113,71	–	–	–
5000	4	–	1,33	3,07	0,77
7500	4	–	2,84	3,00	0,75
10 000	4	–	5,15	2,87	0,72
20 000	4	–	18,04	2,98	0,75
30 000	4	–	38,05	2,99	0,75
Среднее				2,98	0,75

Графическое представление и анализ результатов должны выполняться в соответствии с принятой стратегией:

– для МТОЗ основной задачей графического анализа является сравнение технологий;
– для ОТМЗ основной задачей графического анализа является сравнение задач.

При оформлении графиков обязательно должны указываться названия осей и шкалы. Выбор предельных значений по осям должен обеспечивать хорошую визуализацию данных. Например, для представления на графике ряда 0,97, 0,96, 0,95, 0,94 пределы по вертикальной оси следует задать от 0,9 до 1. Пределы от 0 до 1 сделают плохо видимым наклон кривой.

Не следует приводить на одном графике слишком много кривых. Чаще всего в ходе анализа выполняются парные сравнения. Следует там, где это возможно, использовать усредненные данные.

Стратегия анализа – от общего к частному:

– сравнение технологий или задач по среднему значению коэффициента эффективности;

– оценка влияния варьируемых параметров на усредненную эффективность сравниваемых технологий или задач;
– оценка влияния варьируемых параметров на усредненное ускорение сравниваемых технологий или задач (выполняется для некоторых выбранных случаев);
– оценка влияния варьируемых параметров на время расчета (выполняется для некоторых выбранных случаев).

Далее приведен пример графического представления и анализа результатов в стратегии МТОЗ.

В ходе анализа необходимо рассмотреть и сравнить многопоточные технологии openMP и C++11Threads по результатам двухфакторного эксперимента.

Средние значения коэффициентов эффективности, указанные в таблицах 14 и 15, свидетельствуют о том, что по результатам эксперимента исследуемые технологии имеют приблизительно равную эффективность. Технология openMP несколько более эффективна.

Влияние варьируемых факторов на коэффициент эффективности показано на рисунках 5 и 6. На графиках приведены усредненные значения коэффициентов эффективности из таблиц 14 и 15.

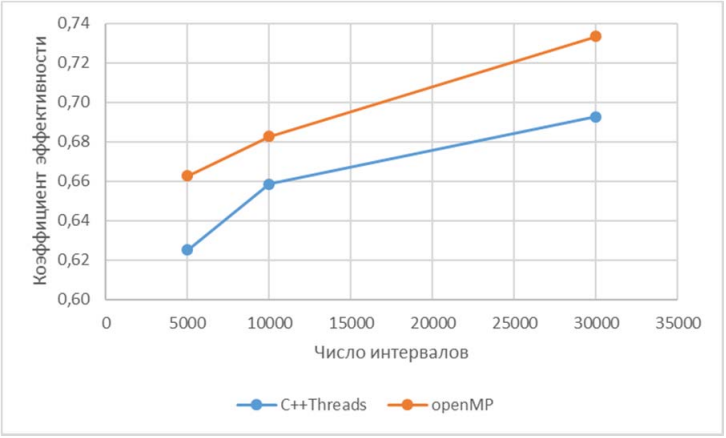

Рисунок 5 – Влияние числа интервалов на коэффициент эффективности

С ростом числа интервалов имеет место увеличение эффективности распараллеливания для обеих рассматриваемых технологий.

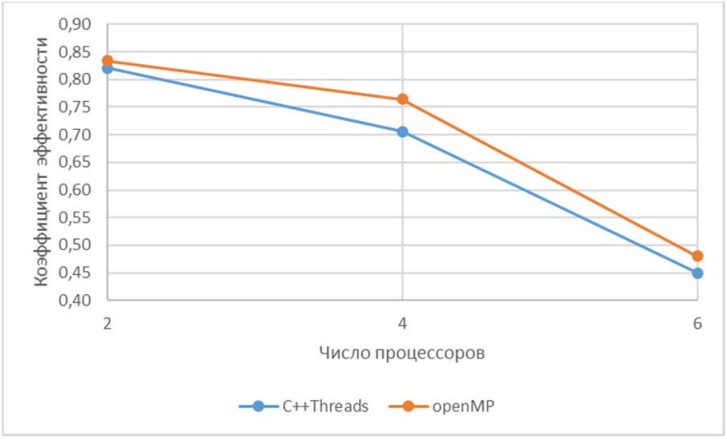

Рисунок 6 – Влияние числа процессоров на коэффициент эффективности

Это объясняется тем, что при увеличении размера задачи снижается доля накладных расходов в общем времени расчета.

С ростом числа процессоров имеет место снижение эффективности. Причем при переходе от четырех к шести процессорам имеет место значительное снижение эффективности: от 0,75 до 0,45.

Зависимость коэффициента ускорения от числа процессоров показана на рисунке 7. На графике приведены усредненные значения коэффициентов ускорения из таблиц 14 и 15.

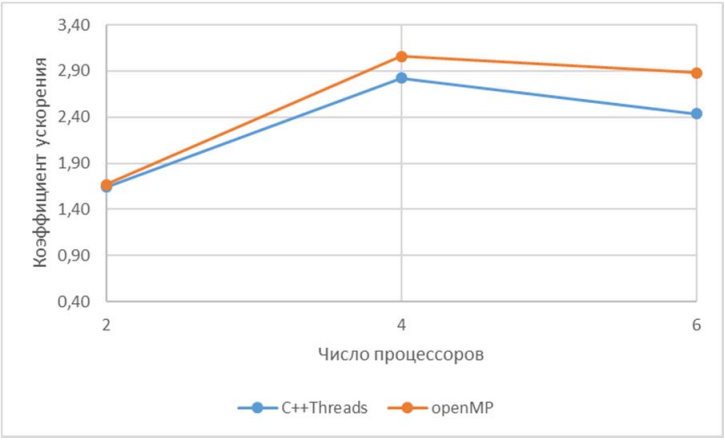

Рисунок 7 – Влияние числа процессоров на коэффициент ускорения

Зависимость времени расчета от числа процессоров показана на рисунке 8. На графике приведены усредненные значения времени расчета из таблиц 13 и 14.

Данные рисунков 7 и 8 показывают, что при переходе от четырех к шести процессорам коэффициент ускорения снижается, а время расчета увеличивается.

Это объясняется тем, что эксперименты проведены на ноутбуке с процессором, обладающим двумя физическими ядрами и допускающим по два вычислительных потока на ядро. Таким образом, 4 программных процессора являются предельным числом для распараллеливания вычислений на компьютере, использованном при проведении экспериментов.

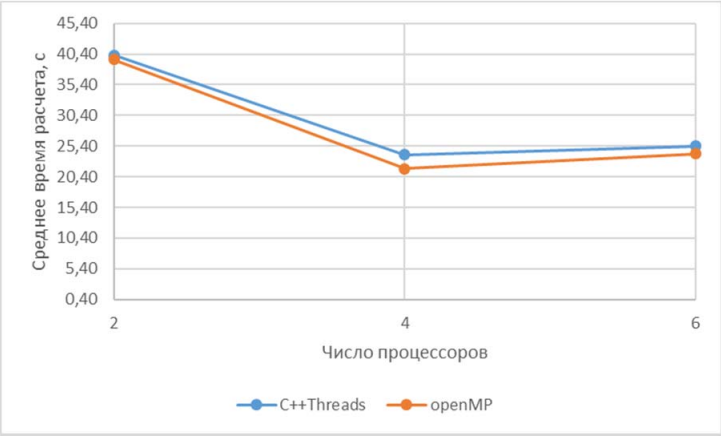

Рисунок 8 – Влияние числа процессоров на время расчета

Далее приведен пример графического представления и анализа результатов для стратегии ОТМЗ.

В ходе анализа необходимо рассмотреть и сравнить эффективность многопоточного решения задач численного интегрирования методом трапеций и методом Монте-Карло по результатам однофакторного эксперимента.

В целом средние коэффициенты эффективности, приведенные в таблицах 16 и 17, свидетельствуют о том, что исследуемые задачи численного интегрирования распараллеливаются в равной степени эффективно.

Влияние варьируемого фактора – числа интервалов – на коэффициент эффективности показано на рисунке 9.

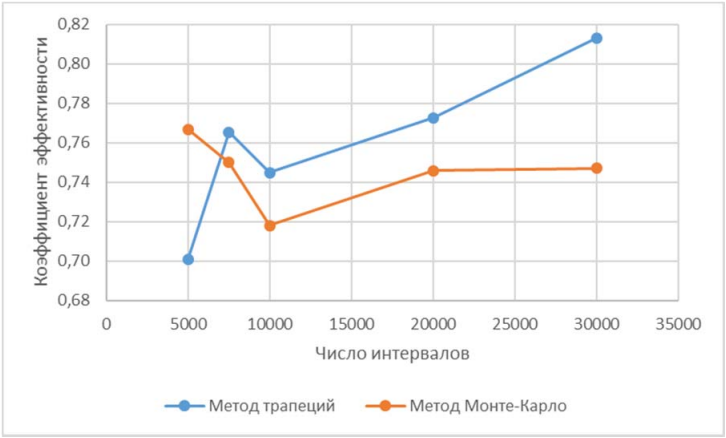

Рисунок 9 – Влияние числа интервалов на коэффициент эффективности

Влияние числа интервалов на коэффициент ускорения показано на рисунке 10.

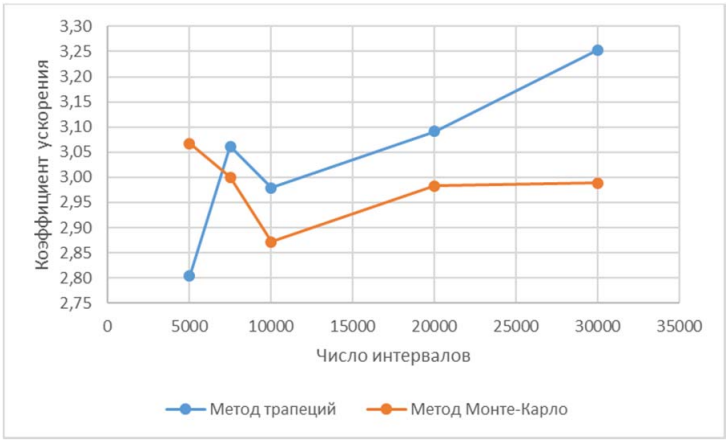

Рисунок 10 – Влияние числа интервалов на коэффициент ускорения

Поскольку расчеты выполнялись на неизменном количестве программных процессоров $p = 4$, графики на рисунках 9 и 10 отличаются только масштабом по вертикальной оси.

На интервале от 5 до 10 тысяч имеет место разнонаправленность кривых. При повышении количества интервалов имеет место возрастание эффективности и ускорения по обоим исследованным методам численного интегрирования. Положение кривых и динамика их изменения с ростом *n* показывают, что параллельная программа численного интегрирования методом трапеций более эффективна.

Влияние числа интервалов на среднее время расчета показано на рисунке 11.

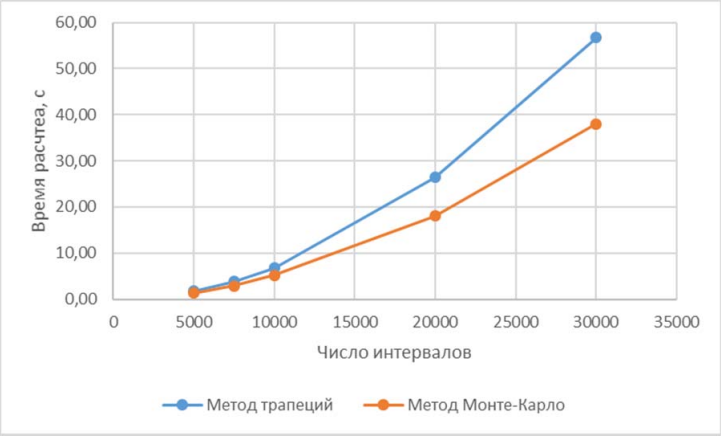

Рисунок 11 – Влияние числа интервалов на время расчета

График, приведенный на рисунке 11, показывает, что с ростом размера задачи метод Монте-Карло обеспечивает меньшее время расчета по сравнению с методом трапеций. Данные по расчету в один поток (рисунок 12) подтверждают, что метод Монте-Карло по сравнению с методом трапеций обеспечивает снижение времени расчета с ростом размера задачи.

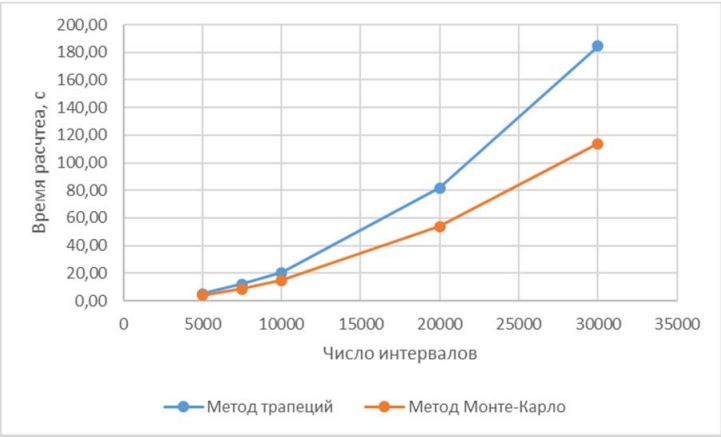

Рисунок 12 – Влияние числа интервалов на время расчета в один поток

Заключение

Заключение должно содержать:

– общее описание работы;

– выводы по работе;

– заключение о достижении цели работы.

Список источников

На все приведенные в списке источники должны быть приведены ссылки в тексте работы.

Приложения

В приложениях к курсовой работе приводится полный код разработанных программ.

ТРЕБОВАНИЯ К ОФОРМЛЕНИЮ КУРСОВОЙ РАБОТЫ

В соответствии с п. 4.2 Положения о курсовой работе (курсовом проекте), одной из задач выполнения курсовой работы является «формирование культуры написания выпускной квалификационной работы». В связи с этим при оформлении текста, рисунков, таблиц, заголовков разделов и подразделов курсовой работы необходимо соблюдать правила оформления, изложенные в Методических указаниях по оформлению выпускных квалификационных работ.

Нумерация рисунков, таблиц и листингов – сквозная в пределах работы.

Листинги оформляются моноширинным шрифтом (например, Courier New) размером 12 через одинарный интервал. Заголовок пишется над листингом и содержит номер и название листинга.

Пример оформления листинга в тексте курсовой работы: «Код функции (листинг 1), приводящий матрицу к треугольному виду, содержит внешний цикл от 0 до $n - 1$, где n – количество элементов матрицы.

Листинг 1 – Код функции приведения матрицы к треугольному виду:

```
int ComputeGaussianElimination(double **a,int n)
{
    int kolvoPerestanovok = 0;
    double pivot,gmax,pmax,temp;
    int pindmax,gindmax,i,j,k;

    for (k = 0 ; k < n-1 ; k++)
    {
        gmax = 0.0;
// поиск в столбце максимального по модулю элемента и на-
хождение его индекса строки
           for (i = k ; i < n ; i++)
           {
                temp = abs(a[i][k]);
                if (temp > pmax)
                {
                   gmax = temp;
                   gindmax = i;
                }
    }
```

```
       if (gmax == 0) return false; // Если матрица сингу-
лярная, установите флаг & quit.
       if (gindmax != k) // Поменять местами строки при не-
обходимости
      {
          kolvoPerestanovok++;
          for (j = k; j < n; j++)
          {
              temp = a[gindmax][j];
              a[gindmax][j] = a[k][j];
              a[k][j] = temp;
          }
      }
      pivot = -1.0/a[k][k]; // Вычислить точку поворота
      for (i = k+1 ; i < n; i++)
    {
             temp = pivot*a[i][k];
             for (j = k ; j < n ; j++)
             {a[i][j] = a[i][j] + temp*a[k][j];
             }
         }
    }
    return kolvoPerestanovok;
}
```

В основном тексте работы нужно приводить относительно небольшие фрагменты листинга программ в объеме, необходимом для пояснения текста. Полные листинги необходимо вынести в приложения.

КРИТЕРИИ ОЦЕНКИ КУРСОВОЙ РАБОТЫ

К проверке принимаются работы, полностью соответствующие требованиям по оформлению. При отклонении от требований по оформлению работа возвращается студенту для приведения в соответствие с требованиями по оформлению.

Первым критерием оценки курсовой работы является реализация в работе многопоточности. Если снижение времени расчета при переходе от однопоточной к параллельным программам не продемонстрировано, то цель работы не достигнута и работа оценивается на «неудовлетворительно».

Обязательным требованием является связность результатов: данные, представленные на снимках экрана, в тексте работы, в таблицах и на графиках, должны совпадать.

Предварительная оценка работ, соответствующих первому критерию, выполняется по полноте и правильности выполнения составных элементов работы.

Для оценки полноты и правильности работы может быть использован чек-лист, содержащий элементы курсовой работы (табл. 18, 19).

Таблица 18 – Перечень элементов курсовой работы для стратегии МТОЗ

Номер по порядку	Элемент курсовой работы	Оцениваемый фрагмент
1	Введение	Вводный текст
2	Введение	Предмет, объект
3	Введение	Цель, задачи
4	Введение	Описание структуры работы
5	1.1. Практическая значимость задачи	Место задачи в определенной области знаний, актуальность задачи
6	1.1. Практическая значимость задачи	Обзор современного уровня знаний по задаче

Продолжение таблицы 18

Номер по порядку	Элемент курсовой работы	Оцениваемый фрагмент
7	1.1. Практическая значимость задачи	Области применения задачи
8	1.2. Обзор методов решения задачи	Математическое описание задачи
9	1.2. Обзор методов решения задачи	Описание методов решения задачи
10	1.3. Реализация последовательной программы	Текстовое описание алгоритма и кода
11	1.3. Реализация последовательной программы	Графическое изображение алгоритма
12	1.3. Реализация последовательной программы	Фрагменты кода
13	1.3. Реализация последовательной программы	Текстовое описание условий тестирования
14	1.3. Реализация последовательной программы	Результаты тестирования
15	Выводы по разделу 1	
16	2.1. Обзор технологий разработки параллельного программного обеспечения	Обзор с указанием достоинств и недостатков не менее трех технологий
17	2.2. Разработка параллельных алгоритмов	Анализ последовательных задач
18	2.2. Разработка параллельных алгоритмов	Обоснование парадигмы параллельного программирования
19	2.2. Разработка параллельных алгоритмов	Обоснование критических секций, условных блокировок и т. п.
20	2.3. Реализация параллельной программы «Технология 1»	Текстовое описание параллельного алгоритма и кода
21	2.3. Реализация параллельной программы «Технология 1»	Графическое изображение параллельного алгоритма
22	2.3. Реализация параллельной программы «Технология 1»	Фрагменты параллельного кода

Продолжение таблицы 18

Номер по порядку	Элемент курсовой работы	Оцениваемый фрагмент
23	2.3. Реализация параллельной программы «Технология 1»	Текстовое описание условий тестирования (на корректность и на время расчета)
24	2.3. Реализация параллельной программы «Технология 1»	Результаты тестирования
25	2.3. Реализация параллельной программы «Технология 1»	Вывод по результатам тестирования
26	2.4. Реализация параллельной программы «Технология 2»	Текстовое описание параллельного алгоритма и кода
27	2.4. Реализация параллельной программы «Технология 2»	Графическое изображение параллельного алгоритма
28	2.4. Реализация параллельной программы «Технология 2»	Фрагменты параллельного кода
29	2.4. Реализация параллельной программы «Технология 2»	Текстовое описание условий тестирования (на корректность и на время расчета)
30	2.4. Реализация параллельной программы «Технология 2»	Результаты тестирования
31	2.4. Реализация параллельной программы «Технология 2»	Вывод по результатам тестирования
32	Выводы по разделу 2	
33	3.1. Методика оценки эффективности программного обеспечения	Текстовое описание содержания работ
34	3.1. Методика оценки эффективности программного обеспечения	Описание программно-аппаратного обеспечения
35	3.1. Методика оценки эффективности программного обеспечения	Варьируемые параметры, пределы и интервалы варьирования
36	3.1. Методика оценки эффективности программного обеспечения	Измеряемые величины с указанием способов измерения

Окончание таблицы 18

Номер по порядку	Элемент курсовой работы	Оцениваемый фрагмент
37	3.1. Методика оценки эффективности программного обеспечения	Расчётные величины с указанием расчетных формул
38	3.2. Оценка воспроизводимости эксперимента	Результаты повтора экспериментов в центре плана
39	3.2. Оценка воспроизводимости эксперимента	Анализ воспроизводимости эксперимента
40	3.3. Планирование и проведение экспериментальных вычислений	Описание плана эксперимента
41	3.3. Планирование и проведение экспериментальных вычислений	План и результаты эксперимента
42	3.4. Анализ результатов эксперимента	Расчет показателей эффективности по результатам измерений
43	3.4. Анализ результатов эксперимента	Графическое представление и анализ результатов
44	Выводы по разделу 3	
45	Заключение	Общее описание работы
46	Заключение	Выводы по работе, заключение о достижении цели работы
47	Список литературы	Достаточное количество источников
48	Список литературы	Наличие ссылок на источники по тексту
49	Приложение	Наличие полного кода разработанных программ
50	Приложение	Правильность оформления кода

Таблица 19 – Перечень элементов курсовой работы для стратегии ОТМЗ

Номер по порядку	Элемент курсовой работы	Оцениваемый фрагмент
1	Введение	Вводный текст
2	Введение	Предмет, объект
3	Введение	Цель, задачи
4	Введение	Описание структуры работы
5	1.1. Практическая значимость задачи	Краткое описание применяемой вычислительной технологии
6	1.1. Практическая значимость задачи	Анализ особенностей, достоинств и недостатков применяемой вычислительной технологии в сравнении с аналогами
7	1.1. Практическая значимость задачи	Области применения технологии
8	1.2. Обзор методов решения задачи	Математическое описание задачи
9	1.2. Обзор методов решения задачи	Описание методов решения задачи
10	1.3. Реализация последовательной программы «Задача 1»	Текстовое описание алгоритма и кода
11	1.3. Реализация последовательной программы «Задача 1»	Графическое изображение алгоритма
12	1.3. Реализация последовательной программы «Задача 1»	Фрагменты кода
13	1.3. Реализация последовательной программы «Задача 1»	Текстовое описание условий тестирования
14	1.3. Реализация последовательной программы «Задача 1»	Результаты тестирования
15	1.4. Реализация последовательной программы «Задача 2»	Текстовое описание алгоритма и кода
16	1.4. Реализация последовательной программы «Задача 2»	Графическое изображение алгоритма

Продолжение таблицы 19

Номер по порядку	Элемент курсовой работы	Оцениваемый фрагмент
17	1.4. Реализация последовательной программы «Задача 2»	Фрагменты кода
18	1.4. Реализация последовательной программы «Задача 2»	Текстовое описание условий тестирования
19	1.4. Реализация последовательной программы «Задача 2»	Результаты тестирования
20	Выводы по разделу 1	
21	2.1. Разработка параллельных алгоритмов	Анализ последовательных задач
22	2.1. Разработка параллельных алгоритмов	Обоснование парадигмы параллельного программирования
23	2.1. Разработка параллельных алгоритмов	Обоснование критических секций, условных блокировок и т. п.
24	2.2. Реализация параллельной программы «Задача 1»	Текстовое описание параллельного алгоритма и кода
25	2.2. Реализация параллельной программы «Задача 1»	Графическое изображение параллельного алгоритма
26	2.2. Реализация параллельной программы «Задача 1»	Фрагменты параллельного кода
27	2.2. Реализация параллельной программы «Задача 1»	Текстовое описание условий тестирования (на корректность и на время расчета)
28	2.2. Реализация параллельной программы «Задача 1»	Результаты тестирования
29	2.2. Реализация параллельной программы «Задача 1»	Вывод по результатам тестирования
30	2.3. Реализация параллельной программы «Задача 2»	Текстовое описание параллельного алгоритма и кода
31	2.3. Реализация параллельной программы «Задача 2»	Графическое изображение параллельного алгоритма
32	2.3. Реализация параллельной программы «Задача 2»	Фрагменты параллельного кода

Продолжение таблицы 19

Номер по порядку	Элемент курсовой работы	Оцениваемый фрагмент
33	2.3. Реализация параллельной программы «Задача 2»	Текстовое описание условий тестирования (на корректность и на время расчета)
34	2.3. Реализация параллельной программы «Задача 2»	Результаты тестирования
35	2.3. Реализация параллельной программы «Задача 2»	Вывод по результатам тестирования
36	Выводы по разделу 2	
37	3.1. Методика оценки эффективности программного обеспечения	Текстовое описание содержания работ
38	3.1. Методика оценки эффективности программного обеспечения	Описание программно-аппаратного обеспечения
39	3.1. Методика оценки эффективности программного обеспечения	Варьируемые параметры, пределы и интервалы варьирования
40	3.1. Методика оценки эффективности программного обеспечения	Измеряемые величины с указанием способов измерения
41	3.1. Методика оценки эффективности программного обеспечения	Расчетные величины с указанием расчетных формул
42	3.2. Оценка воспроизводимости эксперимента	Результаты повтора экспериментов в центре плана и анализ воспроизводимости эксперимента
43	3.3. Планирование и проведение экспериментальных вычислений	Описание плана эксперимента
44	3.3. Планирование и проведение экспериментальных вычислений	План и результаты эксперимента
45	3.4. Анализ результатов эксперимента	Расчет показателей эффективности по результатам измерений

Окончание таблицы 19

Номер по порядку	Элемент курсовой работы	Оцениваемый фрагмент
46	3.4. Анализ результатов эксперимента	Графическое представление и анализ результатов
47	Выводы по разделу 3	
48	Заключение	Общее описание работы, выводы по работе, заключение о достижении цели работы
49	Список литературы	Достаточное количество источников, наличие ссылок на источники по тексту
50	Приложение	Наличие и правильность оформления полного кода разработанных программ

К защите допускаются курсовые работы, имеющие положительную предварительную оценку. Окончательная оценка выставляется по результатам защиты курсовой работы.

Организация защиты курсовых работ регламентирована Положением о курсовой работе (курсовом проекте).

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Формы, утвержденные Положением о курсовой работе (курсовом проекте) // ТГУ. Тольяттинский государственный университет : [сайт]. – URL www.tltsu.ru/upravlenie/educational-methodical-management/forms/forms-approved-by-the-regulations-on-course-work-course-project/?ysclid=l8h2l1wif9492728431 (дата обращения: 01.09.2022).
2. Положение о курсовой работе (курсовом проекте) : утверждено решением учёного совета от 19 декабря 2019 года № 262 / Тольяттинский государственный университет. – Тольятти, 2019. – 15 с. – URL old.tltsu.ru/upravlenie/educational-methodical-management/regulatory-documents-of-educational-process/?ysclid=l8h3ue7t1x546842592 (дата обращения: 01.09.2022).
3. Методические указания по оформлению выпускных квалификационных работ по программам бакалавриата, программам специалитета, программам магистратуры : утверждены приказом от 30 января 2020 года № 145 : с изменениями, утвержденными приказом от 17 июня 2021 года № 1180 / Тольяттинский государственный университет. – Тольятти, 2021. – 39 с. – URL: old.tltsu.ru/upravlenie/educational-methodical-management/regulatory-documents-of-educational-process/?ysclid=l8h3ue7t1x546842592 (дата обращения: 01.09.2022).

Содержание

Нормативные ссылки 3

Выбор и формулировка темы курсовой работы 4

Требования к структуре курсовой работы 5

Требования к содержанию курсовой работы 7

Требования к оформлению курсовой работы 39

Критерии оценки курсовой работы 41

Библиографический список 49

Учебное издание

Хрипунов Николай Владимирович

МНОГОПОТОЧНОЕ ПРОГРАММИРОВАНИЕ. ВЫПОЛНЕНИЕ КУРСОВОЙ РАБОТЫ

Учебно-методическое пособие

Редактор *О.П. Корабельникова*
Технический редактор *Н.П. Крюкова*
Компьютерная верстка: *Л.В. Сызганцева*
Дизайн обложки: *И.И. Шишкина*

В оформлении обложки использованы изображения от Freepik и DC Studio на сайте ru.freepik.com

Подписано в печать 04.04.2025. Формат 60×84/16.
Печать оперативная. Усл. п. л. 2,96.
Тираж 100 экз. Заказ № 1-91-22.

Издательство Тольяттинского государственного университета
445020, г. Тольятти, ул. Белорусская, 14,
тел. 8 (8482) 44-91-47, www.tltsu.ru