

**Федеральное государственное автономное образовательное
учреждение высшего образования
«Московский физико-технический институт
(национальный исследовательский университет)»**

УТВЕРЖДЕНО

**Директор физтех-школы
прикладной математики и
информатики
А.М. Райгородский**

| | |
|----------------------------|---|
| | Рабочая программа дисциплины (модуля) |
| по дисциплине: | Методы реализации алгоритмов |
| по направлению: | Прикладная математика и информатика |
| профиль подготовки: | А1360: Передовые методы искусственного интеллекта Физтех-школа Прикладной Математики и Информатики кафедра алгоритмов и технологий программирования |
| курс: | 2 |
| квалификация: | бакалавр |

Семестр, формы промежуточной аттестации: 4 (весенний) - Дифференцированный зачет

Аудиторных часов: 60 всего, в том числе:

лекции: 30 час.

семинары: 30 час.

лабораторные занятия: 0 час.

Самостоятельная работа: 75 час.

Всего часов: 135, всего зач. ед.: 3

Программу составил: И.Д. Степанов, ассистент

Программа обсуждена на заседании кафедры алгоритмов и технологий программирования 12.02.2024

Аннотация

Данный курс посвящен изучению принципов и техник эффективной реализации алгоритмов на современных компьютерных системах. Студенты получают знания и навыки, необходимые для анализа сложности алгоритмов, оптимизации кода, использования параллельных вычислений и работы с различными структурами данных. Дисциплина включает подробное освещение теоретической стороны алгоритмов, разбор и тренировка решений практических задач, а также предполагает самостоятельное изучение студентами материала предмета через решение домашних теоретических и практических задач. Для освоения курса необходимы базовые понимания о понятии алгоритма и работе компьютера; также требуется достаточная осведомленность в простейших определениях и терминах дискретной математики.

1. Цели и задачи

Цель дисциплины

- сформировать у студентов понимание принципов и методов реализации алгоритмов на современных вычислительных системах, а также развить навыки анализа и выбора оптимальных решений для конкретных задач.

Задачи дисциплины

- научить разбираться в особенностях процессоров: конвейеры, суперскалярность, многоядерность;
- научить оценивать временную и пространственную сложность алгоритмов;
- научить выбирать оптимальный алгоритм для конкретной задачи;
- научить разрабатывать и реализовывать параллельные алгоритмы.

2. Перечень формируемых компетенций

Освоение дисциплины направлено на формирование следующих компетенций:

| Код и наименование компетенции | Индикаторы достижения компетенции |
|---|---|
| ОПК-2 Способен использовать современные информационные технологии и программные средства при решении задач профессиональной деятельности, соблюдая требования информационной безопасности | ОПК-2.2 Знает и умеет применять численные математические методы и прикладное программное обеспечение для решения научных задач в профессиональной области |
| | ОПК-2.1 Способен применять современные вычислительную технику и сервисы сети Интернет в области (сфере) профессиональной деятельности |

3. Перечень планируемых результатов обучения по дисциплине (модулю)

В результате освоения дисциплины обучающиеся должны

знать:

- принципы организации памяти (иерархия, кэширование, виртуальная память);
- устройство и функционирование процессоров (конвейеры, суперскалярность, многоядерность);
- модели параллельных вычислений (многопоточность, распределенные вычисления);
- параллельные алгоритмы и методы их реализации;
- структуры данных: массивы, списки, деревья, хеш-таблицы и т.д.

уметь:

- определять временную и пространственную сложность алгоритма;
- сравнивать алгоритмы по эффективности и выбирать оптимальный вариант;
- выбирать подходящий язык программирования для конкретной задачи.

владеть:

- навыками анализа и выбора оптимального алгоритма для решения конкретной задачи;
- навыками реализации алгоритмов на различных языках программирования;
- навыками оптимизации алгоритмов на уровне алгоритмов, кода и параллелизма.

4. Содержание дисциплины (модуля), структурированное по темам (разделам) с указанием отведенного на них количества академических часов и видов учебных занятий

4.1. Разделы дисциплины (модуля) и трудоемкости по видам учебных занятий

| № | Тема (раздел) дисциплины | Трудоемкость по видам учебных занятий, включая самостоятельную работу, час. | | | |
|-----------------------|---|---|----------|-----------------|----------------|
| | | Лекции | Семинары | Лаборат. работы | Самост. работа |
| 1 | Строковые алгоритмы | 6 | 6 | | 15 |
| 2 | Архитектура современных вычислительных систем | 6 | 6 | | 15 |
| 3 | Алгоритмические оптимизации | 6 | 6 | | 15 |
| 4 | Оптимизации на уровне кода | 6 | 6 | | 13 |
| 5 | Параллельные алгоритмы | 6 | 6 | | 17 |
| Итого часов | | 30 | 30 | | 75 |
| Подготовка к экзамену | | 0 час. | | | |
| Общая трудоёмкость | | 135 час., 3 зач.ед. | | | |

4.2. Содержание дисциплины (модуля), структурированное по темам (разделам)

Семестр: 4 (Весенний)

1. Строковые алгоритмы

Зет- и префикс-функция строки. Алгоритмы их нахождения за линейное время. Алгоритм Манакера. Структура данных бор. Алгоритм Ахо-Корасик. Поиск вхождения шаблона в строку и шаблонов в текст. Подсчёт числа вхождений.

2. Архитектура современных вычислительных систем

Иерархия памяти: регистры, кэш-память, основная память, внешняя память. Принцип локальности и кэширование. Виртуальная память и страничная организация. Конвейеры команд и суперскалярность. Многоядерные процессоры и параллелизм на уровне инструкций. Модели параллелизма: разделяемая память, распределенная память, гибридные модели.

3. Алгоритмические оптимизации

Эффективные структуры данных: массивы, списки, стеки, очереди, деревья, хеш-таблицы; выбор оптимальной структуры данных для конкретной задачи. Алгоритмические трюки. Динамическое программирование; жадные алгоритмы; разделяй и властвуй; алгоритмы с возвратом.

4. Оптимизации на уровне кода

Особенности архитектуры процессора: векторные инструкции и SIMD-расширения; конвейеры и предсказание ветвлений; кэш-память и оптимизация доступа к данным. Оптимизация компилятора: разматывание циклов, встраивание функций, удаление мертвого кода; векторизация и параллелизация кода.

5. Параллельные алгоритмы

Модели параллельного программирования: OpenMP, MPI, CUDA, OpenCL; параллелизм задач и параллелизм данных. Разработка параллельных алгоритмов: разбиение задачи на подзадачи; балансировка нагрузки и минимизация коммуникаций; синхронизация и обработка данных.

5. Описание материально-технической базы, необходимой для осуществления образовательного процесса по дисциплине (модулю)

Учебная аудитория, оснащенная компьютером и мультимедийным оборудованием (проектор, звуковая система), а так же персональными компьютерами для обучающихся.

6. Перечень рекомендуемой литературы

Основная литература

1. Введение в теорию алгоритмов и структур данных, Электронная версия печатной публикации / М. А. Бабенко, М. В. Левин. — Москва, МЦНМО, 2016

Дополнительная литература

7. Перечень ресурсов информационно-телекоммуникационной сети "Интернет", необходимых для освоения дисциплины (модуля)

Не используются

8. Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине (модулю), включая перечень необходимого программного обеспечения и информационных справочных систем (при необходимости)

Для успешного освоения дисциплины, студент использует следующие программные средства:

- компилятор языка C++;
- библиотеки для работы с данными: NumPy, Pandas (Python); Eigen, Armadillo (C++).

9. Методические указания для обучающихся по освоению дисциплины (модуля)

Успешное освоение дисциплины требует:

- посещения студентом всех видов аудиторных занятий;
- ведения конспекта в ходе лекционных занятий;
- качественной самостоятельной подготовки к практическим занятиям, активной работы на них;
- активной самостоятельной и аудиторной работы студента;
- своевременной сдачи преподавателю заданий по аудиторным видам работ.

ОЦЕНОЧНЫЕ МАТЕРИАЛЫ ПО ДИСЦИПЛИНЕ (МОДУЛЮ)

по направлению: Прикладная математика и информатика
профиль подготовки: АІ360: Передовые методы искусственного интеллекта
Физтех-школа Прикладной Математики и Информатики
кафедра алгоритмов и технологий программирования
курс: 2
квалификация: бакалавр

Семестр, формы промежуточной аттестации: 4 (весенний) - Дифференцированный зачет

Разработчик: И.Д. Степанов, ассистент

1. Компетенции, формируемые в процессе изучения дисциплины

| Код и наименование компетенции | Индикаторы достижения компетенции |
|---|---|
| ОПК-2 Способен использовать современные информационные технологии и программные средства при решении задач профессиональной деятельности, соблюдая требования информационной безопасности | ОПК-2.2 Знает и умеет применять численные математические методы и прикладное программное обеспечение для решения научных задач в профессиональной области |
| | ОПК-2.1 Способен применять современные вычислительную технику и сервисы сети Интернет в области (сфере) профессиональной деятельности |

2. Показатели оценивания компетенций

В результате изучения дисциплины «Методы реализации алгоритмов» обучающийся должен:

знать:

- принципы организации памяти (иерархия, кэширование, виртуальная память);
- устройство и функционирование процессоров (конвейеры, суперскалярность, многоядерность);
- модели параллельных вычислений (многопоточность, распределенные вычисления);
- параллельные алгоритмы и методы их реализации;
- структуры данных: массивы, списки, деревья, хеш-таблицы и т.д.

уметь:

- определять временную и пространственную сложность алгоритма;
- сравнивать алгоритмы по эффективности и выбирать оптимальный вариант;
- выбирать подходящий язык программирования для конкретной задачи.

владеть:

- навыками анализа и выбора оптимального алгоритма для решения конкретной задачи;
- навыками реализации алгоритмов на различных языках программирования;
- навыками оптимизации алгоритмов на уровне алгоритмов, кода и параллелизма.

3. Перечень типовых (примерных) вопросов, заданий, тем для подготовки к текущему контролю

Примеры типовых контрольных заданий:

1. Дан следующий алгоритм вычисления факториала числа:

```
def factorial(n):  
    if n == 0:  
        return 1  
    else:  
        return n * factorial(n-1)
```

Реализуйте данный алгоритм на языке программирования C++ или Python.

2. Дан одномерный массив $A(n)$. Найти $\max(a_2, a_4, \dots, a_{2k}) + \min(a_1, a_3, \dots, a_{2k+1})$.

3. Дана последовательность действительных чисел a_1, a_2, \dots, a_n . Указать те ее элементы, которые принадлежат отрезку $[c, d]$.

4. Дана последовательность целых положительных чисел a_1, a_2, \dots, a_n . Найти произведение только тех из них, которые больше заданного числа m . Если таких чисел нет, то выдать сообщение об этом.

5. Даны действительные числа a_1, a_2, \dots, a_n . Среди них есть положительные и отрицательные. Заменить нулями те числа, величина которых по модулю больше максимального числа, т.е. $|a_i| > \max\{a_1, a_2, \dots, a_n\}$.

4. Перечень типовых (примерных) вопросов и тем для проведения промежуточной аттестации обучающихся

1. Что такое алгоритм? Какие свойства должен иметь алгоритм?
2. Опишите основные методы анализа сложности алгоритмов. Что такое нотация "О-большое"?

3. Сравните временную и пространственную сложность следующих структур данных: массив, связный список, двоичное дерево поиска.
4. В чем заключается принцип локальности? Как он используется для оптимизации доступа к памяти?
5. Опишите основные этапы конвейера команд процессора. Как конвейер влияет на производительность?
6. Какие существуют модели параллельных вычислений? Приведите примеры задач, которые эффективно решаются с использованием параллелизма.
7. В чем заключается идея динамического программирования? Приведите пример задачи, которая решается с помощью этого метода.
8. Как особенности архитектуры процессора (например, векторные инструкции) могут быть использованы для оптимизации кода?
9. Опишите основные принципы разработки параллельных алгоритмов. Какие проблемы могут возникнуть при параллелизации алгоритмов?
10. Как использовать профилировщик для анализа производительности кода? Какие показатели производительности следует учитывать при оптимизации кода?

Критерии оценивания

Оценка "отлично" (10) выставляется если полностью и вовремя решены все задачи без ошибок. Продemonстрирован грамотный подход к решению задач, реализованы оптимальные алгоритмы, код оформлен в едином удобочитаемом стиле.

Оценка "отлично" (9) выставляется если полностью и вовремя решены все задачи без ошибок. Продemonстрирован грамотный подход к решению задач, реализованы оптимальные алгоритмы.

Оценка "отлично" (8) выставляется если полностью и вовремя решены все задачи без ошибок. Продemonстрирован грамотный подход к решению задач.

Оценка "хорошо" (7) выставляется если полностью решены все задачи. Допущены несущественные ошибки.

Оценка "хорошо" (6) выставляется если полностью решено большинство задач. В некоторых задачах допущены и не исправлены ошибки, либо некоторые задачи решены частично.

Оценка "хорошо" (5) выставляется если полностью решено две трети задач. В некоторых задачах допущены и не исправлены ошибки, либо некоторые задачи решены частично.

Оценка "удовлетворительно" (4) выставляется если полностью решено более половины задач. В остальных задачах допущены и не исправлены ошибки, либо некоторые задачи решены частично.

Оценка "удовлетворительно" (3) выставляется если полностью решено более половины задач.

Оценка "неудовлетворительно" (2) выставляется если решено менее половины задач.

Оценка "неудовлетворительно" (1) выставляется если не решено ни одной задачи.

5. Методические материалы, определяющие процедуры оценивания знаний, умений, навыков и (или) опыта деятельности

Во время проведения дифференцированного зачета обучающиеся могут пользоваться программой дисциплины.