

**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:	Верификация программного обеспечения
по направлению:	Информатика и вычислительная техника
профиль подготовки:	Прикладная математика и информатика Физтех-школа Прикладной Математики и Информатики кафедра системного программирования
курс:	2
квалификация:	магистр

Семестры, формы промежуточной аттестации:

2 (весенний) - Дифференцированный зачет

3 (осенний) - Дифференцированный зачет

Аудиторных часов: 120 всего, в том числе:

лекции: 60 час.

семинары: 60 час.

лабораторные занятия: 0 час.

Самостоятельная работа: 150 час.

Всего часов: 270, всего зач. ед.: 6

Программу составили:

А.С. Камкин, канд. физ.-мат. наук

Н.В. Пакулин, кандидат наук

Программа обсуждена на заседании кафедры системного программирования 01.05.2024

Аннотация

Курс представляет собой введение в методы верификации программного обеспечения. Цель курса – познакомить с предметом верификации ПО, представить широкую палитру существующих методов и подходов, а также осветить преимущества и ограничения, присущие методам верификации. В рамках курса рассматриваются методы статического анализа программ, методы проверки моделей (model checking), методы динамического анализа программ и различные варианты функционального тестирования.

1. Цели и задачи

Цель дисциплины

- познакомить студентов с базовыми принципами и методами формальной верификации.
- сформировать у студентов навыки необходимые для практического использования рассмотренных методов.

Задачи дисциплины

- объяснение роли формальной верификации для построения корректных и надежных программ, формирование базовых знаний в этой области;
- обучение студентов методам формальной спецификации программ (пред- и постусловия, темпоральные утверждения);
- обучение студентов методам формализации поведения программ (формализация семантики языков программирования, использование формальных моделей);
- обучение студентов методам формальной верификации программ (дедуктивная верификация программ, проверка моделей).

2. Перечень формируемых компетенций

Освоение дисциплины направлено на формирование следующих компетенций:

Код и наименование компетенции	Индикаторы достижения компетенции
УК-3 Способен организовывать и руководить работой команды, вырабатывая командную стратегию для достижения поставленной цели	УК-3.1 Организует и координирует работу участников проекта, способствует конструктивному преодолению возникающих разногласий и конфликтов
	УК-3.2 Учитывает в своей социальной и профессиональной деятельности интересы, особенности поведения и мнения (включая критические) людей, с которыми работает/взаимодействует, в том числе посредством корректировки своих действий
	УК-3.3 Способен предвидеть результаты (последствия) как личных, так и коллективных действий
	УК-3.4 Способен планировать командную работу, распределять поручения членам команды, организовывать обсуждение разных идей и мнений
ПК-3 Владеет навыками участия в научных дискуссиях, выступления с сообщениями и докладами устного, письменного и виртуального (размещение в информационных сетях) характера, представления материалов собственных исследований	ПК-3.1 Знает основы ведения научной дискуссии и формы устного научного высказывания
	ПК-3.2 Умеет вести корректную дискуссию в области информационных технологий задавать вопросы и отвечать на поставленные вопросы по теме научной работы
	ПК-3.3 Имеет практический опыт участия в научных студенческих конференциях, очных, виртуальных, заочных обсуждениях научных проблем в области информационных технологий
ПК-1 Готов к включению в профессиональное сообщество; способен	ПК-1.1 Знает принципы построения научной работы, методы сбора и анализа полученного материала, способы аргументации; владеет навыками подготовки научных обзоров, публикаций, рефератов и библиографий по тематике проводимых исследований на русском и английском языке

проводить под научным руководством локальные исследования на основе существующих методов в конкретной области профессиональной деятельности	ПК-1.2 Умеет решать научные задачи с пониманием существующих подходов к верификации моделей программного обеспечения в связи с поставленной целью и в соответствии с выбранной методикой
	ПК-1.3 Имеет практический опыт выступлений и научной аргументации при анализе объекта научной и профессиональной деятельности

3. Перечень планируемых результатов обучения по дисциплине (модулю)

В результате освоения дисциплины обучающиеся должны

знать:

- место и роль формальной верификации в процессе построения корректных программ;
- методы формальной спецификации и верификации программ;
- современные средства формальной верификации программ;
- связь методов формальной верификации с методами смежных дисциплин: математической логики, дискретной математики, программной инженерии.

уметь:

- описывать условия корректности программ в форме пред- и постусловий;
- аналитически доказывать корректность программ;
- строить формальные модели компьютерных систем;
- описывать свойства реагирующих систем в виде формул темпоральной логики;
- применять инструментальные средства формальной верификации.

владеть:

- навыками аналитической верификации программ;
- навыками использования средств дедуктивной верификации программ;
- навыками использования средств проверки моделей.

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	Динамический анализ программ.	5	5		12
2	Модели программных систем.	5	5		13
3	Стандарты жизненного цикла ПО.	5	5		12
4	Статический анализ программ.	5	5		12
5	Тестирование с использованием моделей.	5	5		13
6	Тестирование.	5	5		13
7	Дедуктивная верификация программ.	7	8		18
8	Принципы формальной верификации.	7	8		19
9	Проверка моделей (model checking).	8	6		19
10	Связь между разными методами верификации.	8	8		19
Итого часов		60	60		150
Подготовка к экзамену		0 час.			
Общая трудоёмкость		270 час., 6 зач.ед.			

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

Семестр: 2 (Весенний)

1. Динамический анализ программ.

Методы контроля потока управления в бинарных исполнимых файлах. Обнаружение утечек памяти. Выявление ошибок синхронизации.

Методы, основанные на разрешении ограничений. DART, Avalanche.

2. Модели программных систем.

Введение в моделирование программ. История вопроса.

Исполнимые модели. Конечные автоматы, расширенные конечные автоматы. Диаграммы состояний UML. Недетерминизм. Последовательная и параллельная композиции. Проблема взрыва числа состояний.

Введение в сети Петри.

Логические модели. Тройки Хора. Аксиоматические модели. Темпоральные логики. Формулы состояний и формулы последовательностей. Логики LTL, CTL, CTL*. Интерпретация формул на моделях.

Алгебраические модели. Алгебры термов, эквивалентность термов. Переписывание.

3. Стандарты жизненного цикла ПО.

Базовые понятия о качестве программного обеспечения. Стандарты процессов жизненного цикла программного обеспечения. Место верификации в жизненном цикле.

Стандарты и модели жизненного цикла: ISO 9000, ISO/IEC 12207, CMM, DO 178, Orange Book, Common Criteria.

Представление о методах верификации ПО. Связи между инспекцией, тестированием, моделированием, статическим анализом,

Ревью кода. Организация процесса ревью, сбор результатов, оценка результатов.

4. Статический анализ программ.

Представление о статическом анализе. Статическая и динамическая семантика языка программирования. Базовый статический анализ на этапе компиляции.

Методы статического анализа. Абстрактная интерпретация. Построение и анализ графа потока управления.

Проверка на моделях. Формализация требований средствами темпоральной логики. Верификация формул на автоматной модели программы или алгоритма. Построение контрпримеров.

Доказательство корректности. Контрактные спецификации как теоремы. Доказательство теорем на основе кода программы. Доказательство интегральных свойств ПО на основе контрактов отдельных компонентов.

5. Тестирование с использованием моделей.

Виды моделей, пригодные для тестирования. Применение моделей в тестирование. Задача извлечения тестов. Задача построения оракула. Критерии покрытия, основанные на моделях.

Технология UniTESK. Контрактные спецификации, пред- и постусловия. Генерация тестовых последовательностей из частично заданных автоматов тестов.

6. Тестирование.

Задачи тестирования. Классификация тестирования по размеру целевых систем: модульное, компонентное, системное, интеграционное. Место тестирования в процессах жизненного цикла.

Стандарты на процессы тестирования. Планирование тестирования, разработка тестов, оценка результатов. Тестовые покрытия. Покрытия по коду, ветвлениям, пространствам входных параметров.

Методология тестирования xUnit. Введение в Junit. Разработка на основе тестов. Тестирование асинхронных систем и обратных интерфейсов. Заглушки.

Компонентное тестирование. Задачи интеграционного и системного тестирования.

Семестр: 3 (Осенний)

7. Дедуктивная верификация программ.

Основные понятия дедуктивного анализа программ. Аксиомы и правила вывода (тройки Хоара). Понятие аннотированной программы. Верификация как поиск доказательства.

Проблема индукции при выводе свойств циклов. Невыводимость свойств цикла из его структуры. Понятие инварианта цикла. Примеры и задания.

Инструменты дедуктивной верификации программ. Язык ACSL (ANSI C Specification Language). Платформы Frama-C для статического анализа C-программ. Плагин Jessie для дедуктивного анализа C-программ (платформа Why). Примеры и задания.

Метод индуктивных утверждений Флойда. Синтаксис и семантика блок-схем.

Доказательство частичной корректности блок-схем. Точки сечения. Индуктивные утверждения. Условия верификации. Примеры и задания.

Метод фундированных множеств Флойда. Доказательство полной корректности блок-схем. Оценочные функции. Условия завершимости. Примеры и задания.

Верификация последовательных программ на языках программирования. Примеры и задания.

Автоматизация дедуктивного анализа программ. Синтез инвариантов циклов. Генерация условий верификации.

Дедуктивная верификация параллельных программ. Семантика чередований. Справедливость планировщика.

8. Принципы формальной верификации.

Общая схема формальной верификации. Формальная спецификация требований. Формальная модель повеления. Соответствие поведения требованиям.

Примеры методов формальной верификации. Дедуктивная верификация. Проверка моделей. Проверка эквивалентности.

Формализация условий корректности. Пред- и постусловия (программный контракт). Частичная корректность. Полная корректность.

Формализация семантики языков программирования.

Операционная семантика.

Аксиоматическая семантика.

Метод доказательного программирования Дейкстры.

9. Проверка моделей (model checking).

Синтаксис и семантика темпоральной логики линейного времени (LTL). Основные тождества. Выражение свойств реактивных систем в логике LTL. Свойства безопасности (safety), живости (liveness), справедливости (fairness). Примеры и задания.

Инструменты проверки моделей. Язык Promela (Process/Protocol Meta Language). Инструмент проверки моделей SPIN. Примеры и задания.

Введение в метод проверки моделей для логики LTL. Моделирование реактивных систем структурами Крипке. Множество допустимых траекторий. Контрольный автомат. Проверка выполнимости формулы. Примеры и задания.

Теоретико-автоматный подход к проверке моделей для логики LTL. Автоматы Бюхи. Построение автомата Бюхи для структуры Крипке. Построение автомата Бюхи для формулы LTL. Построение синхронной композиции автоматов Бюхи. Проверка пустоты языка, допускаемого автоматом Бюхи. Примеры и задачи.

10. Связь между разными методами верификации.

Тестирование программ (методы черного и белого ящика). Тестирование на основе моделей.
Дедуктивная верификация. Проверка моделей.

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

Необходимое оборудование для лекций и практических занятий: компьютер и мультимедийное оборудование (проектор, звуковая система).

Необходимое программное обеспечение: программный пакет Frama-C, программный пакет Spin.

6. Перечень рекомендуемой литературы

Основная литература

1. Д. Грис. Наука программирования. - М.: Мир, 1984.
2. Р. Андерсон. Доказательство правильности программ. - М.: Мир, 1982.
3. Э. М. Кларк, О. Грамберг, Д. Пелед. Верификация моделей программ. Model Checking. - М.: МЦНМО, 2002 г.
4. Ю. Г. Карпов. Model Checking. Верификация параллельных и распределенных программных систем. БХВ-Петербург, 2010.
5. С.В. Силицын, Н.Ю. Налютин. Верификация программного обеспечения. - М.: ИНТУИТ-Бином, 2008. 368 с.
6. В. В. Кулямин. Технологии программирования. Компонентный подход. - М.: ИНТУИТ-Бином, 2007. 463 с.
7. Э. М. Кларк, О. Грамберг, Д. Пелед. Верификация моделей программ. Model Checking. - М.: МЦНМО, 2002 г.

Дополнительная литература

1. K.R. Apt, F.S. de Boer, E.-R. Olderog. Verification of Sequential and Concurrent Programs, Springer, 2009.
2. C. Baier, J.-P. Katoen. Principles of Model Checking. The MIT Press, 2008.
3. M. Ben-Ari. Mathematical Logic for Computer Science. Springer, 2012.
4. Котляров В.П. Основы тестирования программного обеспечения. - М.: ИНТУИТ.РУ, 2006. 360 с.
5. Мендельсон Э. Введение в математическую логику. Москва: Наука, 1984.
6. Миронов А. М., Жуков Д.Ю. Математическая модель и методы верификации программных систем, Информационные технологии и вычислительные системы, 2005, 220 стр.
7. Ю. Г. Карпов. Model Checking. Верификация параллельных и распределенных программных систем. БХВ-Петербург, 2010.
8. В.В. Липаев. Системное проектирование сложных программных средств для информационных систем. Издание второе. // Серия "Управление качеством". - М: СИНТЕГ, 2002.-268 с
9. В.В. Липаев. Качество программных средств. - М: Янус-К, 2002. 400 с.
10. В.В. Липаев. Выбор и оценивание характеристик качества программных средств. Методы и стандарты. - М.:СИНТЕГ, 2001.-228 с.

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

Необходимое программное обеспечение: программный пакет Frama-C, программный пакет Spin.

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

Необходимое программное обеспечение: программный пакет Frama-C, программный пакет Spin.

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

Успешное освоение курса «Верификация программного обеспечения» требует самостоятельной работы студента. В программе курса приведено минимально необходимое время для работы студента над темой.

Самостоятельная работа включает в себя:

- чтение и конспектирование рекомендованной литературы;
- проработку учебного материала (по конспектам лекций, учебной и научной литературе);
- решение задач, предлагаемых студентам на лекциях и практических занятиях;
- подготовку к контрольным, самостоятельным работам и тестам.

Руководство и контроль за самостоятельной работой студента осуществляется в результате анализа итогов самостоятельных работ и тестов, а также с помощью индивидуальных консультаций.

Показателем владения материалом служит умение решать прикладные задачи, возникающие при выполнении НИР. Для формирования умения применять теоретические знания на практике студенту необходимо решать как можно больше задач.

ОЦЕНОЧНЫЕ МАТЕРИАЛЫ ПО ДИСЦИПЛИНЕ (МОДУЛЮ)

по направлению:	Информатика и вычислительная техника
профиль подготовки:	Прикладная математика и информатика Физтех-школа Прикладной Математики и Информатики кафедра системного программирования
курс:	<u>2</u>
квалификация:	магистр

Семестры, формы промежуточной аттестации:

- 2 (весенний) - Дифференцированный зачет
- 3 (осенний) - Дифференцированный зачет

Разработчики:

А.С. Камкин, канд. физ.-мат. наук
Н.В. Пакулин, кандидат наук

1. Компетенции, формируемые в процессе изучения дисциплины

Код и наименование компетенции	Индикаторы достижения компетенции
УК-3 Способен организовывать и руководить работой команды, вырабатывая командную стратегию для достижения поставленной цели	УК-3.1 Организует и координирует работу участников проекта, способствует конструктивному преодолению возникающих разногласий и конфликтов
	УК-3.2 Учитывает в своей социальной и профессиональной деятельности интересы, особенности поведения и мнения (включая критические) людей, с которыми работает/взаимодействует, в том числе посредством корректировки своих действий
	УК-3.3 Способен предвидеть результаты (последствия) как личных, так и коллективных действий
	УК-3.4 Способен планировать командную работу, распределять поручения членам команды, организовывать обсуждение разных идей и мнений
ПК-3 Владеет навыками участия в научных дискуссиях, выступления с сообщениями и докладами устного, письменного и виртуального (размещение в информационных сетях) характера, представления материалов собственных исследований	ПК-3.1 Знает основы ведения научной дискуссии и формы устного научного высказывания
	ПК-3.2 Умеет вести корректную дискуссию в области информационных технологий задавать вопросы и отвечать на поставленные вопросы по теме научной работы
	ПК-3.3 Имеет практический опыт участия в научных студенческих конференциях, очных, виртуальных, заочных обсуждениях научных проблем в области информационных технологий
ПК-1 Готов к включению в профессиональное сообщество; способен проводить под научным руководством локальные исследования на основе существующих методов в конкретной области профессиональной деятельности	ПК-1.1 Знает принципы построения научной работы, методы сбора и анализа полученного материала, способы аргументации; владеет навыками подготовки научных обзоров, публикаций, рефератов и библиографий по тематике проводимых исследований на русском и английском языке
	ПК-1.2 Умеет решать научные задачи с пониманием существующих подходов к верификации моделей программного обеспечения в связи с поставленной целью и в соответствии с выбранной методикой
	ПК-1.3 Имеет практический опыт выступлений и научной аргументации при анализе объекта научной и профессиональной деятельности

2. Показатели оценивания компетенций

В результате изучения дисциплины «Верификация программного обеспечения» обучающийся должен:

знать:

- место и роль формальной верификации в процессе построения корректных программ;
- методы формальной спецификации и верификации программ;
- современные средства формальной верификации программ;
- связь методов формальной верификации с методами смежных дисциплин: математической логики, дискретной математики, программной инженерии.

уметь:

- описывать условия корректности программ в форме пред- и постусловий;
- аналитически доказывать корректность программ;
- строить формальные модели компьютерных систем;
- описывать свойства реагирующих систем в виде формул темпоральной логики;
- применять инструментальные средства формальной верификации.

владеть:

- навыками аналитической верификации программ;
- навыками использования средств дедуктивной верификации программ;
- навыками использования средств проверки моделей.

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

С целью контроля освоения обучающимися учебного материала проводится устный опрос в начале занятия по теме прошлой лекции или в конце занятия по пройденной теме.

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

Вопросы к дифференцированному зачёту в 10 семестре:

1. Формальная верификация программ. Пред- и пост-условия. Понятие частичной и полной корректности программы.
2. Операционная семантика языка программирования. Описание операционной семантики языка while.
3. Аксиоматическая семантика языка программирования. Описание аксиоматической семантики языка while.
4. Метод Дейкстры построения корректных программ. Слабейшее предусловие и его свойства.
5. Аксиоматическая система Хоара. Проблема индукции при выводе свойств циклов. Понятие инварианта цикла.
6. Метод индуктивных утверждений Флойда. Доказательство частичной корректности блок-схем.
7. Метод фундированных множеств Флойда. Доказательства полной корректности блок-схем.
8. Автоматизация дедуктивной верификации программ. Синтез инвариантов циклов. Генерация условий верификации.
9. Формальная спецификация C-программ на языке ACSL. Обзор основных возможностей языка ACSL и платформы Frama-C.
10. Параллельные программы. Семантика асинхронных чередований. Справедливость планировщика. Дедуктивная верификация алгоритма Петерсона.

Вопросы к дифференцированному зачёту в 11 семестре:

1. Логика LTL. Основные тождества. Выражение свойств реагирующих систем в логике LTL. Свойства безопасности, живучести, справедливости.
2. Структуры Крипке. Представление программ с конечным числом состояний структурами Крипке. Представление параллельных программ структурами Крипке.
3. Траектории реагирующих систем. Автоматы Бюхи -регулярные языки. Построение синхронной композиции автоматов Бюхи.
4. Контрольный автомат реагирующей системы. Автомат Бюхи как контрольный автомат. Проверка пустоты языка, допускаемого автоматом Бюхи.
5. Автомат Бюхи для формулы LTL. Алгоритм построения автомата Бюхи для формулы LTL.
6. Теоретико-автоматный подход к проверке модели для логики LTL. Общая схема проверки выполнимости формулы LTL на модели системы.
7. Описание моделей компьютерных систем на языке Promela. Обзор основных возможностей языка Promela и инструмента Spin.
8. Метод проверки моделей для логики LTL.
9. Моделирование реактивных систем структурами Крипке.
10. Множество допустимых траекторий.

Критерии оценивания

Оценка отлично 10 баллов - выставляется студенту, показавшему всесторонние, систематизированные, глубокие знания учебной программы дисциплины, проявляющему интерес к данной предметной области, продемонстрировавшему умение уверенно и творчески применять их на практике при решении конкретных задач, свободное и правильное обоснование принятых решений.

Оценка отлично 9 баллов - выставляется студенту, показавшему всесторонние, систематизированные, глубокие знания учебной программы дисциплины и умение уверенно применять их на практике при решении конкретных задач, свободное и правильное обоснование принятых решений.

Оценка отлично 8 баллов - выставляется студенту, показавшему всесторонние, систематизированные, глубокие знания учебной программы дисциплины и умение уверенно применять их на практике при решении конкретных задач, правильное обоснование принятых решений, с некоторыми недочетами.

Оценка хорошо 7 баллов - выставляется студенту, если он твердо знает материал, грамотно и по существу излагает его, умеет применять полученные знания на практике, но недостаточно грамотно обосновывает полученные результаты.

Оценка хорошо 6 баллов - выставляется студенту, если он твердо знает материал, грамотно и по существу излагает его, умеет применять полученные знания на практике, но допускает в ответе или в решении задач некоторые неточности.

Оценка хорошо 5 баллов - выставляется студенту, если он в основном знает материал, грамотно и по существу излагает его, умеет применять полученные знания на практике, но допускает в ответе или в решении задач достаточно большое количество неточностей.

Оценка удовлетворительно 4 балла - выставляется студенту, показавшему фрагментарный, разрозненный характер знаний, недостаточно правильные формулировки базовых понятий, нарушения логической последовательности в изложении программного материала, но при этом он освоил основные разделы учебной программы, необходимые для дальнейшего обучения, и может применять полученные знания по образцу в стандартной ситуации.

Оценка удовлетворительно 3 балла - выставляется студенту, показавшему фрагментарный, разрозненный характер знаний, допускающему ошибки в формулировках базовых понятий, нарушения логической последовательности в изложении программного материала, слабо владеет основными разделами учебной программы, необходимыми для дальнейшего обучения и с трудом применяет полученные знания даже в стандартной ситуации.

Оценка неудовлетворительно 2 балла - выставляется студенту, который не знает большей части основного содержания учебной программы дисциплины, допускает грубые ошибки в формулировках основных принципов и не умеет использовать полученные знания при решении типовых задач.

Оценка неудовлетворительно 1 балл - выставляется студенту, который не знает основного содержания учебной программы дисциплины, допускает грубейшие ошибки в формулировках базовых понятий дисциплины и вообще не имеет навыков решения типовых практических задач.

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

Во время проведения дифференцированного зачета обучающиеся могут пользоваться программой дисциплины, а также справочной литературой, вычислительной техникой, конспектами лекций.

Дифференцированный зачет может проводиться по итогам текущей успеваемости и сдачи заданий, или путем организации специального опроса, проводимого в устной форме.