Федеральное государственное автономное образовательное учреждение высшего образования
"Национальный исследовательский университет "Высшая школа экономики"

Факультет компьютерных наук
Департамент программной инженерии

Рабочая программа дисциплины
"Верификация программного обеспечения"

для образовательной программы «Системное программирование»
направления подготовки 09.04.04 «Программная инженерия»
уровень – магистр

Разработчик программы
Камкин А.С, к.ф.-м.н., kamkin@ispras.ru

Одобрена на заседании департамента программной инженерии «___»_____ 2017 г.
Руководитель департамента Авдошин С.М. _____

Утверждена Академическим советом образовательной программы
«___»_____ 2017 г., № протокола _____

Академический руководитель образовательной программы
Петренко А.К. _____

Москва, 2017

1 Область применения и нормативные ссылки

Настоящая программа учебной дисциплины устанавливает минимальные требования к знаниям и умениям студента и определяет содержание и виды учебных занятий и отчетности.

Программа предназначена для преподавателей, ведущих данную дисциплину, учебных ассистентов и студентов образовательной программы «Системное программирование» направления подготовки 09.04.04 «Программная инженерия», изучающих дисциплину «Верификация программного обеспечения».

Программа разработана в соответствии с образовательным стандартом Национального исследовательского университета «Высшая школа экономики» по направлению 09.04.04 «Программная инженерия».

2 Цели освоения дисциплины

Цель курса – изучение базовых принципов и методов верификации программного обеспечения (ПО); приобретение навыков, необходимых для практического применения методов верификации ПО.

Задачами данного курса являются:

- освоение базовых принципов верификации ПО;
- приобретение теоретических знаний в области дедуктивной верификации программ;
- приобретение теоретических знаний в области проверки моделей (model checking);
- приобретение практических навыков работы с инструментами дедуктивной верификации;
- приобретение практических навыков работы с инструментами проверки моделей.

3 Компетенции обучающегося, формируемые в результате освоения дисциплины

В результате освоения дисциплины студент должен:

1. Знать:

– базовые принципы формальной верификации ПО;
– методы формализации семантики языков программирования;
– методы дедуктивной верификации программ;
– методы проверки моделей компьютерных систем.

2. Уметь:

– описывать условия корректности программ в форме пред- и постусловий;
– аналитически доказывать частичную и полную корректность программ;
– строить формальные модели компьютерных систем;
– описывать свойства реагирующих систем в виде формул темпоральной логики.

3. Иметь навыки (приобрести опыт):

– спецификации и моделирования ПО;
– аналитической верификации программ;
– применения средств дедуктивной верификации программ;
– применения средств проверки моделей компьютерных систем.

В результате освоения дисциплины студент осваивает следующие компетенции:

Компетенция	Код по ОС ВШЭ	Уровень формирования компетенции	Дескрипторы – основные признаки освоения (показатели достижения результата)	Формы и методы обучения, способствующие формированию и развитию компетенции	Форма контроля уровня сформированности компетенции
Способен рефлексировать (оценивать и перерабатывать, анализировать и синтезировать) освоенные научные методы и способы деятельности для применения на практике.	СК-М1	РБ	Владеет	Лекционные занятия, самостоятельная работа	Контрольные работы, домашние задания, экзамен
Способен предлагать концепции, модели, создавать и апробировать новые способы и инструменты профессиональной деятельности для применения на практике.	СК-М2	РБ	Владеет	Лекционные занятия, самостоятельная работа	Контрольные работы, домашние задания, экзамен
Способен к самостоятельному освоению новых методов исследований, изменению научного и производственного профиля своей деятельности.	СК-М3	РБ	Владеет	Лекционные занятия, самостоятельная работа	Контрольные работы, домашние задания, экзамен
Способен отбирать и разрабатывать методы анализа объектов профессиональной деятельности на основе общих тенденций развития программной инженерии.	ИК-М1.1.НИД (ПИ)	СД	Применяет	Практические занятия, самостоятельная работа	Контрольные работы, домашние задания, экзамен
Способен проводить анализ, синтез, оптимизацию решений с целью обеспечения качества объектов профессиональной деятельности.	ИК-М1.2.НИД (ПИ)	СД	Применяет	Практические занятия, самостоятельная работа	Контрольные работы, домашние задания, экзамен
Способен планировать, управлять и контролировать выполнение требований.	ИК-М2.1.АД (ПИ)	СД	Применяет	Практические занятия, самостоятельная работа	Контрольные работы, домашние задания, экзамен
Способен выполнять проектную деятельность в области программной инженерии на основе системного подхода , уметь строить и использовать модели для описания и прогнозирования различных явлений, осуществлять их качественный и количественный анализ.	ИК-М3.1.ПД (ПИ)	СД	Применяет	Практические занятия, самостоятельная работа	Контрольные работы, домашние задания, экзамен

4 Место дисциплины в структуре образовательной программы

Данная дисциплина относится к базовой части цикла дисциплин магистерской программы «Системное программирование».

Изучение данной дисциплины базируется на знаниях, полученных студентами при освоении учебных дисциплин:

- «Дискретная математика»,
- «Программирование»,
- «Информатика, математическая логика и теория алгоритмов»,
- «Построение и анализ алгоритмов».

5 Тематический план учебной дисциплины

№	Название раздела	Всего часов	Аудиторные часы			Самостоятельная работа
			Лекции	Семинары	Практические занятия	
1	Общие принципы формальной верификации	12	2		2	8
2	Формализация семантики языков программирования	14	2		2	10
3	Методы дедуктивной верификации программ	32	4		8	20
4	Инструментальные средства дедуктивной верификации программ	24	2		6	16
5	Параллельные программы и реагирующие системы	12	2		2	8
6	Темпоральная логика линейного времени	14	2		2	10
7	Теоретико-автоматный метод проверки моделей	32	4		8	20
8	Символический метод проверки моделей	14	2		2	10
9	Инструментальные средства проверки моделей	24	2		6	16
10	Использование формальных методов в тестировании	12	2		2	8
		190	24		40	126

6 Формы контроля знаний студентов

Тип контроля	Форма контроля	1 год				Параметры
		1	2	3	4	
Текущий	Контрольная работа		*			Письменная работа 90 минут Письменная работа 90 минут
	Домашнее задание	*	*			Верификация программы (Frama-C) Верификация протокола (Spin)
Итоговый	Экзамен		*			Экзамен, включающий устную и письменную части

7 Критерии оценки знаний, навыков

По всем видам работ выставляются 10-ти балльные оценки.

В рамках курса студентам предлагается выполнить одну контрольную работу (оценка — $O_{кр}$) и два домашних задания (оценки — $O_{дом1}$ и $O_{дом2}$). Текущая оценка ($O_{тек}$) рассчитывается как среднее арифметическое оценок $O_{кр}$, $O_{дом1}$ и $O_{дом2}$:

$$O_{тек} = 1/3 *\cdot(O_{кр} + O_{дом1} + O_{дом2}).$$

Оценка за самостоятельную работу ($O_{сам}$) отражает правильность и полноту выполнения домашних работ по темам практических занятий.

Накопленная оценка рассчитывается следующим образом:

$$O_{накопл} = 0{,}6 * O_{тек} + 0{,}4 * O_{сам}.$$

В диплом выставляет результирующая оценка по учебной дисциплине, которая формируется по следующей формуле:

$$O_{результ} = 0{,}6 * O_{накопл} + 0{,}4 * O_{экзамен},$$

где $O_{экзамен}$ — оценка за устный экзамен. Округление при вычислении результирующей оценки осуществляется в пользу студента.

8 Содержание дисциплины

№ п/п	Разделы и темы лекционных занятий	Содержание
1	Общие принципы формальной верификации	Общая схема формальной верификации. Примеры методов формальной верификации: дедуктивная верификация, проверка моделей, проверка эквивалентности. Программные контракты (пред- и постусловия). Частичная и полная корректность программ.
2	Формализация семантики языков программирования	Язык программирования while. Операционная семантика языка while. Аксиоматическая семантика языка while. Слабейшее предусловие и сильнейшее постусловие.
3	Методы дедуктивной верификации программ	Метод индуктивных утверждений. Метод фундированных множеств. Автоматизация дедуктивного анализа программ: синтез инвариантов циклов, доказательство условий верификации.
4	Инструментальные средства дедуктивной верификации программ	Язык аннотации С-программ ACSL (ANSI C Specification Language). Платформа статического анализа С-программ Frama-C. Плагин дедуктивной верификации С-программ Jessie. Платформа дедуктивной верификации Why. SMT-решатели.
5	Параллельные программы и реагирующие системы	Параллельные программы над общими переменными. Синхронный и асинхронный параллелизм. Семантика чередований. Реагирующие системы. Справедливость планировщика.
6	Темпоральная логика линейного времени	Синтаксис и семантика темпоральной логики линейного времени (LTL). Основные тождества. Выражение свойств реактивных систем в логике LTL. Свойства безопасности и живости.
7	Теоретико-автоматный метод проверки моделей	Моделирование реактивных систем структурами Крипке. Автоматы Бюхи. Построение автомата Бюхи для формулы LTL. Построение синхронной композиции автоматов Бюхи. Проверка пустоты языка, допускаемого автоматом Бюхи.
8	Символический метод проверки моделей	Символическое представление множеств и отношений. Символические алгоритмы. Двоичные решающие диаграммы (BDD) и операции над ними. Ограниченная проверка моделей.
9	Инструментальные средства проверки моделей	Язык моделирования компьютерных систем и протоколов Promela (Process/Protocol Meta Language). Инструмент проверки моделей Spin. Инструмент символической проверки моделей SMV.
10	Использование формальных мето-	Тестирование программ (методы черного и белого ящика). Тестирование на основе моделей. Использование техник символического

	дов в тестировании	исполнения для генерации тестов.

9 Образовательные технологии

Практические занятия по дисциплине проводятся в интерактивной форме и включают разбор реальных задач, возникающих в верификации ПО.

10 Оценочные средства для текущего контроля и аттестации студента

10.1 Тематика заданий текущего контроля

Промежуточный контроль проводится с помощью письменной контрольной работы и двух домашних заданий.

Примерный вариант письменной контрольной работы

1. Найдите инвариант следующего цикла:

   $\{a \geq 0\}$
   x := a;
   n := 1;
   y := 0;
   while $x \neq 0$ **do**
   y := y + n;
   n := n + 2;
   x := x – 1
   end
   $\{y = a^2\}$

2. Докажите частичную корректность следующей программы:

   $\{a \geq 0\}$
   $x := 0; y := 1;$
   while $y \leq a$ **do**
   $x := x + 1;$
   $y := y + 2*x + 1$
   end
   $\{0 \leq x^2 \leq a < (x + 1)^2\}$

3. Представьте в виде формулы LTL следующее высказывание: «между каждой парой состояний, удовлетворяющих свойству p, обязательно встретится состояние, удовлетворяющее свойству q».

4. Постройте автомат Бюхи для следующей формулы LTL: $\mathbf{GF}(\neg p \wedge \mathbf{X}q)$

Домашнее задание 1 состоит в дедуктивной верификации небольшой C-программы с помощью инструмента Frama-C (Jessie).

Домашнее задание 2 состоит в разработке Promela-модели протокола передачи данных и ее проверки с помощью инструмента Spin.

В конце учебного периода проводится экзамен, включающий устную и письменную части. Письменная часть экзамена аналогична двум контрольным работам.

Вопросы к устной части экзамена

1. Формальная верификация программ. Общая схема и основные подходы. Частичная и полная корректность программ.
2. Формальная семантика языков программирования. Операционная семантика языка программирования. Структурная операционная семантика языка while.
3. Формальная семантика языков программирования. Аксиоматическая семантика языка программирования. Аксиоматическая семантика языка while.
4. Аксиоматическая система Хоара. Инвариант цикла. Слабейшее предусловие программы и его свойства. Сильнейшее постусловие.
5. Метод индуктивных утверждений Флойда. Точки сечения. Индуктивные утверждения. Условия верификации. Схема доказательства частичной корректности блок-схемы.
6. Метод фундированных множеств Флойда. Точки сечения. Индуктивные утверждения. Условия завершимости. Схема доказательства завершимости блок-схемы.
7. Автоматизация дедуктивной верификации программ. Синтез инвариантов циклов. Эвристические методы. Методы на основе абстрактной интерпретации.
8. Автоматизация дедуктивной верификации программ. Доказательство условий верификации. Понятие разрешающей процедуры. SAT- и SMT-решатели.
9. Параллельные программы. Семантика асинхронных чередований. Справедливость планировщика. Построение асинхронной композиции систем переходов.
10. Темпоральная логика линейного времени (LTL). Синтаксис и семантика LTL. Выражение свойств реагирующих систем в логике LTL. Свойства безопасности и живучести.
11. Структуры Крипке. Вычисления и траектории структуры Крипке. Выбор уровня абстракции при моделировании компьютерных систем. Ложные сообщения и пропуски ошибок.
12. Описание ω-языков автоматами Бюхи. Проверка языка на пустоту. Синхронная композиция автоматов Бюхи. Построение автомата Бюхи по расширенному автомату Бюхи.
13. Теоретико-автоматный подход к проверке модели для LTL. Общая схема метода. Построение автомата Бюхи для LTL-формулы.
14. Использование формальных методов в тестировании. Тестирование на основе моделей. Использование техник символического исполнения для генерации тестов.

11 Учебно-методическое и информационное обеспечение дисциплины

11.1 Базовый учебник

А.С. Камкин. Введение в формальные методы верификации программ. ИСП РАН, 2017.

11.2 Основная литература

1. K.R. Apt, F.S. de Boer, E.-R. Olderog. Verification of Sequential and Concurrent Programs, Springer, 2009.
2. В.А. Непомнящий, О.М. Рякин. Прикладные методы верификации программ. М.: Радио и связь, 1988.

3. C. Baier, J.-P. Katoen. Principles of Model Checking. The MIT Press, 2008.
4. Ю.Г. Карпов. Model Checking. Верификация параллельных и распределенных программных систем. БХВ-Петербург, 2010.
5. P. Baudin, P. Cuoq, J.-C. Filliâtre, C. Marché, B. Monate, Y. Moy, V. Prevosto. ACSL: ANSI/ISO C Specification Language. Version 1.4. 2010.
6. G.J. Holzmann. The SPIN Model Checker. Primer and Reference Manual. Addison-Wesley, 2003.

11.3 Дополнительная литература

1. Э.М. Кларк, О. Грамберг, Д. Пелед. Верификация моделей программ. Model Checking. М.: МЦНМО, 2002.
1. Э. Дейкстра. Дисциплина программирования. М.: Мир, 1978.
2. Д. Грис. Наука программирования. М.: Мир, 1984.
3. Р. Андерсон. Доказательство правильности программ. М.: Мир, 1982.
4. M. Ben-Ari. Mathematical Logic for Computer Science. Springer, 2012.