МИНИСТЕРСТВО НАУКИ И ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение
высшего профессионального образования
«Московский физико-технический институт (государственный университет)»
МФТИ (ГУ)
Кафедра «Системное программирование»

«УТВЕРЖДАЮ»
Проректор по учебной работе
_____ О.А.Горшков
____ _____ 2012 г.

РАБОЧАЯ УЧЕБНАЯ ПРОГРАММА

по дисциплине: Конструирование оптимизирующих компиляторов
по направлению: 010900 «Прикладные математика и физика»
профиль подготовки: Системное программирование
факультеты: ФУПМ
кафедра Системного программирования
курс: 3 и 4 (бакалавриат)
семестры: весенний и осенний **диф. зачет 6 семестр**
экзамен 7 семестр
Трудоёмкость в зач. ед.: вариативная часть – **4 зач. ед**.;
в т.ч.:

лекции: вариативная часть – **66 час**,
практические (семинарские) занятия: нет,
лабораторные занятия: нет.
мастер классы, индивид. и групповые консультации: нет,
самостоятельная работа: вариативная часть – **18 час,**
курсовые работы: нет,
подготовка к экзамену: вариативная часть – **1 зач. ед.**

ВСЕГО АУДИТОРНЫХ ЧАСОВ 82

Программу составил доцент, к.ф.-м.н., Гайсарян С. С.
Программа обсуждена на заседании кафедры Системного программирования

«____» _____2012 г.

Заведующий кафедрой академик В. П. Иванников

ОБЪЁМ УЧЕБНОЙ НАГРУЗКИ И ВИДЫ ОТЧЁТНОСТИ.

Вариативная часть, в т.ч. :	__4___ зач. ед.
Лекции	__66___ часов
Практические занятия	__нет___ часов
Лабораторные работы	__нет___ часов
Индивидуальные занятия с преподавателем	__нет___ часов
Самостоятельные занятия	__18__ часов
Итоговая аттестация	**Диф.зачет 5 семестр, экзамен 7 семестр - 1 зач. ед.**
ВСЕГО	**4 зач. ед. 100 час**

1. ЦЕЛИ И ЗАДАЧИ

Цель курса – Целью курса является изучение основ построения оптимизирующих статических и динамических компиляторов современных языков программирования, учитывающих особенности архитектур современных компьютеров.

Задачами данного курса являются:

- освоение студентами базовых знаний в области оптимизирующей компиляции программ***;***
- приобретение теоретических знаний в области теории графов, теории решеток, методов сбора статистики, используемых при разработке методов анализа и трансформации программ;
- оказание консультаций и помощи студентам в проведении собственных исследований и разработок в областях, использующих компиляторные технологии;
- приобретение навыков работы на современных неоднородных распределенных компьютерных системах.

2. МЕСТО ДИСЦИПЛИНЫ В СТРУКТУРЕ ООП БАКАЛАВРИАТА

Дисциплина _ **Конструирование оптимизирующих компиляторов** ***_ включает в себя разделы, которые могут быть отнесены к вариативным части цикла _Б.3_ кода УЦ ООП.***

Дисциплина **Конструирование оптимизирующих компиляторов** ***базируется на циклах Б.2 курса 1,2,3 базовой и вариативных частях.***

КОМПЕТЕНЦИИ ОБУЧАЮЩЕГОСЯ, ФОРМИРУЕМЫЕ В РЕЗУЛЬТАТЕ ОСВОЕНИЯ ДИСЦИПЛИНЫ

Освоение дисциплины **_ Конструирование оптимизирующих компиляторов _** ***направлено на формирование следующих общекультурных и профессиональных интегральных компетенций бакалавра:***

а) общекультурные (ОК):

- способность анализировать научные проблемы и физические процессы, использовать на практике фундаментальные знания, полученные в области естественных и гуманитарных наук (ОК-1);
- способность осваивать новую проблематику, терминологию, методологию, овладевать научными знаниями, владеть навыками самостоятельного обучения (ОК-2);
- способность логически точно, аргументировано и ясно формулировать свою точку зрения, владеть навыками научной и общекультурной дискуссий (ОК-3);
- готовность к творческому взаимодействию с коллегами по работе и научным коллективом, способность и умение выстраивать межличностное взаимодействие, соблюдая уважение к товарищам и проявляя терпимость к иным точкам зрения (ОК-4).

б) профессиональные (ПК):

- способность применять в своей профессиональной деятельности знания, полученные в области физических и математических дисциплин, включая дисциплины: алгоритмы и языки программирования, программирование на языке ассемблера, математическая логика, теория графов, линейная алгебра (ПК-1);
- способность понимать сущность задач, поставленных в ходе профессиональной деятельности, использовать соответствующее открытое программное обеспечение и алгоритмы для их постановки и решения (ПК-3);
- способность использовать знания в области физических и математических дисциплин для дальнейшего освоения дисциплин в соответствии с профилем подготовки (ПК-4);
- способность работать с современным программным обеспечением, приборами и установками в избранной области (ПК-5).

3. КОНКРЕТНЫЕ ЗНАНИЯ, УМЕНИЯ И НАВЫКИ, ФОРМИРУЕМЫЕ В РЕЗУЛЬТАТЕ ОСВОЕНИЯ ДИСЦИПЛИНЫ

В результате освоения дисциплины «Конструирование оптимизирующих компиляторов» обучающийся должен:

1. Знать:

- фундаментальные понятия, теории современного системного программирования;
- структуру и состав современных оптимизирующих компиляторных сред (примеры – GCC, LLVM и др.);
- цели, задачи и методы машинно-независимой, машинно-ориентированной статической и динамической оптимизации программ в процессе их компиляции;
- принципы применения компиляторных сред для решения других задач программной инженерии: выявление дефектов и аудит программ, запутывание программ и др. ;

2. Уметь:

- разрабатывать, обосновывать и реализовывать новые методы и алгоритмы машинно-независимой оптимизации программ;
- разрабатывать и реализовывать новые языки и их оптимизирующие компиляторы для новых архитектур процессоров, в том числе специализированных;
- применять компиляторные методы и компиляторные среды для решения задач обратной инженерии, защиты программного кода, обнаружения дефектов в программах и др.;

3. Владеть:

- навыками освоения большого объема информации;

- навыками самостоятельной работы в Интернете;
- культурой разработки и реализации системного программного обеспечения современных компьютеров;
- навыками грамотной разработки новых языков программирования и их программного обеспечения;

4. СТРУКТУРА И СОДЕРЖАНИЕ ДИСЦИПЛИНЫ

Структура дисциплины

Перечень разделов дисциплины и распределение времени по темам

№ темы и название	Количество часов
1. Постановка задачи оптимизации программ в компиляторах. Локальная и глобальная оптимизации	2
2. Глобальная оптимизация: анализ потока данных.	13
3. Методы ускорения анализа потока данных. Выделение областей графа потока управления	11
4. Методы ускорения анализа потока данных.	9
5. Межпроцедурный анализ указателей	5
6. Другие оптимизирующие преобразования. Применение статического анализа потоков данных в задачах инженерии программ.	3
7. Особенности архитектуры современных компьютеров и задача генерации оптимального кода.	5
8. Выдача команд	5
9. Распределение и назначение регистров.	7
10. Планирование кода.	7
11. Параллельное выполнение циклов.	9
12. Динамическая и адаптивная оптимизация в компиляторах времени выполнения	7
ВСЕГО (зач. ед.(часов))	100 час (3 зач. ед.)

ВИД ЗАНЯТИЙ

ЛЕКЦИИ:

№ п.п.	Темы	Трудоёмкость в зач. ед. (количество часов)
1	Постановка задачи оптимизации программ в компиляторах. Локальная и глобальная оптимизации	2
2	Глобальная оптимизация: анализ потока данных.	13
3	Методы ускорения анализа потока данных. Выделение областей графа потока управления	11
4	Методы ускорения анализа потока данных.	9
5	Межпроцедурный анализ указателей	5
6	Другие оптимизирующие преобразования. Применение статического анализа потоков данных в задачах инженерии программ.	3

7	Особенности архитектуры современных компьютеров и задача генерации оптимального кода.	5
8	Выдача команд	5
9	Распределение и назначение регистров.	7
10	Планирование кода.	7
11	Параллельное выполнение циклов.	9
12	Динамическая и адаптивная оптимизация в компиляторах времени выполнения	7
ВСЕГО (зач. ед.(часов))		66 час (2 зач. ед.)

ВИДЫ САМОСТОЯТЕЛЬНОЙ РАБОТЫ

№ п.п.	Темы	Трудоёмкость в зач. ед. (количество часов)
1	- Изучение теоретического курса – выполняется самостоятельно каждым студентом по итогам каждой из лекций, результаты контролируются преподавателем на лекционных занятиях, используются конспект (электронный) лекций, учебники, рекомендуемые данной программой, методические пособия.	14 час.
2	- Решение задач по заданию преподавателя – решаются задачи, выданные преподавателем по итогам лекционных занятий, и сдаются в конце семестра, используются конспект (электронный) лекций, учебники, рекомендуемые данной программой, а также сборники задач, включая электронные, учебно-методические пособия.	4 час.
3	- Подготовка к дифференцированному зачету и экзамену	1 зач. ед.
ВСЕГО (зач. ед.(часов))		18 час. + 1 зач. ед.

Содержание дисциплины

№ п/п	Название модулей	Разделы и темы лекционных занятий	Содержание	Объем	
				Аудиторная работа (зачетные единицы/ часы)	Самостоятельная работа (зачетные единицы/ часы)
1	I. **Машин-**	Постановка задачи оптимизации	Задача компиляции. Неоптимизирующий компилятор. Основные фазы компиляции.	2	1

	но-независимая оптимизация программ	программ в компиляторах. Локальная и глобальная оптимизации	Выявление ошибок в процессе компиляции и сообщения о них. Промежуточное представление программы – трехадресный код (четверки). Граф потока управления и алгоритм его построения. Локальная и глобальная оптимизации. Метод нумерации значений (локальный) как основа локальной оптимизации. Недостаточность локальной оптимизации.		
2		Глобальная оптимизация. Анализ потока данных.	Состояние программы. Путь выполнения (трасса). Прямой и обратный обход программы. Передаточная функция инструкции. Композиция передаточных функций. Передаточная функция базового блока. Определение переменной. Постановка задачи о достигающих определениях. Понятие консервативности анализа. Передаточные функции задачи о достигающих определениях (передаточные функции класса gen-kill). Замкнутость класса передаточных функций gen-kill относительно композиции. Система уравнений для задачи о достигающих определениях и ее решение методом итераций. Итеративный алгоритм для вычисления достигающих определений. Применение достигающих определений: нахождение выражений, инвариантных относительно циклов. Алгоритм обнаружения кода, инвариантного относительно цикла. Анализ живых переменных: передаточные функции, система уравнений и итерационный алгоритм. Анализ доступных выражений: передаточные функции, система уравнений и итерационный алгоритм. Анализ потока данных. Обоснование итерационных алгоритмов. Понятие полурешетки. Связь между операцией сбора и полурешеточным отношением порядка. Понятие верхнего (наибольшего) элемента полурешетки. Реализация полурешеток с помощью конечных множеств. Операции объединения и пересечения множеств как реализации операции сбора.	10	3

			Диаграммы полурешеток. Декартовы произведения полурешеток. Понятие структуры потока данных. Понятие замкнутости семейства передаточных функций. Замкнутость семейств передаточных функций для достигающих определений, живых переменных и доступных выражений. Монотонные и дистрибутивные структуры потока данных. Дистрибутивность структур потока данных для достигающих определений, живых переменных и доступных выражений. Обобщенный итеративный алгоритм и его свойства. Понятие максимальной фиксированной точки. Сходимость обобщенного итеративного алгоритма к максимальной фиксированной точке. Идеальное решение уравнений потока данных. Решение сбором по всем путям для дистрибутивных и монотонных структур потока данных. Консервативность максимальной фиксированной точки. Пример недистрибутивной, но монотонной структуры потока данных – распространение констант. Полурешетка для проблемы распространение констант. Семейство передаточных функций, его монотонность и недистрибутивность. Итерационный алгоритм распространения констант. Исключение частично избыточных выражений методом анализа ожидаемых выражений. Четырехэтапный алгоритм отложенного перемещения кода. Достоинства и недостатки подхода. Предварительный этап – ликвидация критических ребер. Первый этап – анализ ожидаемых выражений. Второй этап – анализ доступных выражений. Третий этап – анализ откладываемых выражений. Четвертый этап – анализ используемых выражений (исключение мертвого кода).		
3		Методы ускорения анализа потока данных.	Структурный анализ графа потока управления. Глубинное остовное дерево и его обход. Нумерация узлов графа потока управления. Классификация ребер графа потока управления. Алгоритм построения	8	3

		Выделение областей графа потока управления	глубинного остовного дерева и упорядочения графа потока управления в глубину. Нумерация узлов графа потока управления (в глубину). Доминаторы. Свойства отношения доминирования. Итеративный алгоритм вычисления доминаторов. Дерево доминаторов и алгоритм его построения. Классификация ребер графа потока управления. Понятие естественного цикла. Алгоритм построения естественного цикла для заданного обратного ребра. Вложенность естественных циклов. Гнезда циклов. Сильно связанные компоненты. Алгоритм построения всех максимальных сильно связанных компонентов заданного графа потока управления. Приводимые графы потока управления. Неприводимые области (собственные и несобственные) графа потока управления. Примеры неприводимых областей. Глубина графа потока управления. Понятие области. Виды областей Алгоритм построения иерархии областей для приводимых графов потока управления. Дерево управления. Алгоритм построения восходящего порядка областей графа потока управления. Другие способы структурирования графов потока управления: интервальный анализ, «структурный анализ» с помощью шаблонов и др. Анализ потоков данных на основе областей. Построение передаточных функций областей с помощью операций композиции, сбора и замыкания. Замкнутость структуры потока данных относительно операций композиции, сбора и замыкания. Трехэтапный алгоритм анализа потока данных на основе областей (на примере достигающих определений). Метод расщепления узлов для обработки неприводимых графов потока управления.		
4		Методы ускорения анализа потока данных.	**Форма статического единственного присваивания** (SSA-форма). Определение SSA-формы. Понятие ϕ-функции. Свойства ϕ-функции. Базовый алгоритм преобразования промежуточного представления	6	3

			программы в SSA-форму. Недостатки базового алгоритма. Пример применения алгоритма. Форма статического единственного присваивания (SSA-форма). Квазиоптимальная SSA-форма. Граница доминирования. Алгоритм построения границ доминирования. Построение множества глобальных имен и других вспомогательных множеств. Размещение ф-функций. Переименование переменных. Восстановление кода из SSA-формы. Проблема потери копий **Глобальная нумерация значений.** Два подхода к реализации глобальной нумерации значений: нумерация значений, основанная на хэшировании и нумерация значений, основанная на классификации значений. Нумерация значений, основанная на хэшировании. Нумерация значений в расширенном базовом блоке. Повторное использование результатов для блоков, входящих в несколько путей. Механизм контекстно-ориентированных хэш-таблиц. Алгоритм нумерации значений на основе доминаторов. Нумерация значений, основанная на классификации значений. Понятие конгруэнтности ориентированных графов. Объединение нумерации значений с построением SSA-формы.		
5		Межпро-цедурный анализ указателей	Внутрипроцедурный (глобальный). Проблемы, связанные с обработкой членов структур, элементов массивов и данных, доступных по указателям, в том числе – динамических данных. Понятие алиаса. Алиасы в языке Си. Алиасы в языке Java. Глобальный (внутрипроцедурный) анализ алиасов: первая фаза – обнаружение алиасов, вторая фаза – распространение алиасов (задача анализа потока данных). Недостаточность глобального анализа алиасов. Межпроцедурный анализ алиасов. Способы задания графа вызовов. Чувствительность к потоку и контексту вызова. Контекстно-нечувствительный межпроцедурный анализ. Контекстно-чувствительный	4	1

			анализ на основе аннотаций. Контекстно-чувствительный анализ на основе классификации и клонирования. Контекстно-чувствительный анализ ссылок		
6		Другие оптимизирующие преобразования. Применение статического анализа потоков данных в задачах инженерии программ.	Оптимизация циклов: классификация и обработка индуктивных переменных, развертка циклов, исключение ненужных (избыточных) проверок условий окончания цикла. Оптимизация потока управления. Оптимизация возвратов из рекурсивных процедур. Открытое вставление процедур. Порядок применения оптимизирующих преобразований. Режимы компиляции. Распознавание программ: восстановление документации разработчика по исходному коду программы. Запутывание (обфускация) программ на языках высокого уровня. Нахождение критических ошибок и уязвимостей.	4	1
7	II Машинно-ориентированная и динамическая оптимизация.	Особенности архитектуры современных компьютеров и задача генерации оптимального кода.	**Особенности архитектуры современных компьютеров, учитываемые при генерации объектного кода (обзор).** RISC и CISC. Конвейер потока команд: блокировки конвейера, система сброса конвейера, конфликты по данным. Векторные регистры и векторные команды. Вырезки из массивов. Конвейер данных. Диспетчер и выдача команд. Блок предсказания переходов. Конвейерное выполнение. Out-of-Order-процессоры. VLIW и EPIC. Кэш, локальность, упреждающая выборка (prefetching). Проблемы генерации оптимизированного кода. **Промежуточное представление низкого уровня** (последовательность трехадресных инструкций). Операции низкого уровня. Модель целевой машины (целевой язык). Набор команд. Псевдорегистры. Режимы адресации: прямая, косвенная, индексированная адресации. Стоимость команд и стоимость	4	1

			программы. Задачи генератора кода: распределение памяти, выбор команд, распределение регистров, выбор оптимального порядка команд (планирование кода).		
8		Выдача команд	Распределение памяти: статическое и динамическое. Статическое выделение памяти. Дескрипторы регистров и адресов. Генерация кода для базового блока. Генерация кода для вызовов процедур и возвратов из них (соглашения о связях). Выбор команд – построение отображения программы в промежуточном представлении на последовательность команд целевой машины. Выбор команд путем переписывания дерева. Действия, шаблоны, схема трансляции дерева. Поиск соответствий с помощью синтаксического анализа. Числа Ершова. Алгоритм генерации кода для размеченного дерева выражения. Вычисление выражений при недостаточном количестве регистров. Замощение дерева. Генерация кода с использованием динамического программирования. Покадровая оптимизация: устранение лишних инструкций; оптимизация потока управления; алгебраические упрощения; использование машинных идиом. Исключение недостижимого кода.	6	1
9		Распределе-ние и назначение регистров.	Распределение и назначение регистров в пределах базового блока (Функция getReg ()). Глобальное распределение регистров. Интервалы жизни значений переменных. Построение интервалов жизни. Оценка стоимости сброса. Конфликтные ситуации и граф конфликтов. Построение графа конфликтов. Раскраска графа конфликтов сверху вниз и снизу вверх. Распределение и назначение регистров. Алгоритм раскраски	4	1

			графа конфликтов снизу вверх. Структура распределителя регистров. Примеры глобального распределения регистров		
10		Планирование кода.	Анализ зависимостей по данным. Зависимости по управлению. Граф зависимостей. Планирование списков базовых блоков. Опережающее (спекулятивное) выполнение. Использование предикатные команд. Глобальное планирование кода. Эквивалентность по управлению. Перемещение кода вверх по пути управления. Перемещение кода вниз по пути управления. Глобальное планирование кода. Алгоритм глобального планирования кода на основе областей. Развертка циклов. Уплотнение окрестностей. Агрессивные алгоритмы перемещения кода. Конвейеризация циклов. Модель процессора. Последовательное и параллельное выполнение итераций цикла. Частичная развертка цикла. Циклы с зависимыми итерациями. Ограничения программной конвейеризации. Ограничения, связанные с зависимостями по данным Алгоритм программной конвейеризации. Планирование ациклических графов зависимости данных. Планирование графов с циклическими зависимостями.	10	1
11		Параллельное выполнение циклов.	Симметричные мультипроцессоры с общей памятью (SMP). Многоядерные процессоры. Пример параллельно выполняемого цикла. Закон Амдаля. Понятие локальности данных: пространственная и временная локальность. Формальная постановка задачи распараллеливания циклов. Демонстрация некоторых приемов распараллеливания на примере параллельной программы умножения матриц.	4	1

			Распараллеливание циклов. Пространство итераций. Построение пространств итераций для гнезд циклов. Управление порядком выполнения циклов гнезда. Алгоритм исключения Фурье-Моцкина. Алгоритм вычисления границ циклов для заданного порядка выполнения. Повторное использование данных. Собственные и групповые повторные использования. Анализ зависимостей по данным. Обнаружение параллельности, не требующей синхронизации. Ограничения разбиений пространства и их решение.		
12	III. Динами-ческая оптими-зация	Динамическая и адаптивная оптимизация в компи-ляторах времени выполнения	Структура JIT-компилятора. Примеры JIT-компиляторов. Уровни оптимизации. Профилирование в JIT-компиляторах. Хранение данных о профилях. Выборочная оптимизация. Динамическая и адаптивная оптимизация в JIT-компиляторах. Профилирование с помощью «семплов». Архитектура JIT-компилятора Jikes RVM: подсистема измерений, подсистема перекомпиляции, подсистема управления процессом динамической компиляции (контроллер). Фазы JIT-компиляции. Выбор уровня перекомпиляции. Адаптивная оптимизация. Использование результатов предыдущего выполнения.	4	1

5. ОБРАЗОВАТЕЛЬНЫЕ ТЕХНОЛОГИИ

№ п/п	Вид занятия	Форма проведения занятий	Цель
1	Лекция	Изложение теоретического материала	Получение теоретических знаний по дисциплине
2	Лекция	Иложение теоретического материала с помощью презентаций	Повышение степени понимания материала
3	Лекция	Решение задач по заданию	Осознание связей между теорией

		преподавателя – решаются задачи, выданные преподавателем по итогам лекционных занятий, и сдаются в конце семестра, используются конспект (электронный) лекций, учебники, рекомендуемые данной программой, а также учебно-методические пособия	и практикой, а также взаимозависимостей разных дисциплин
4	Самостоятельная работа студента	Подготовка к к экзамену и зачету с оценкой	Повышение степени понимания материала

6. ОЦЕНОЧНЫЕ СРЕДСТВА ДЛЯ ТЕКУЩЕГО КОНТРОЛЯ УСПЕВАЕМОСТИ, ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ ПО ИТОГАМ ОСВОЕНИЯ ДИСЦИПЛИНЫ И УЧЕБНО-МЕТОДИЧЕСКОЕ ОБЕСПЕЧЕНИЕ САМОСТОЯТЕЛЬНОЙ РАБОТЫ СТУДЕНТОВ

Перечень контрольных вопросов для дифференцированного зачета в конце 6 семестра

1. Промежуточное представление программы. Граф потока управления и алгоритм его построения. Локальная оптимизация. Представление базового блока в виде ориентированного ациклического графа. Локальный метод нумерации значений. Виды локальной оптимизации.
2. Понятие потока данных. Состояние программы. Передаточная функция инструкции. Передаточная функция базового блока. Пути выполнения.
3. Определение и использование переменной в программе. Понятие дости-гающего определения. Передаточные функции достигающих определений. Итеративный алгоритм вычисления достигающих определений.
4. Применение достигающих определений. Обнаружение кода, инвариант-ного относительно цикла и вынесение его за пределы цикла.
5. Понятие живой (активной) переменной. Итеративный алгоритм анализа живых переменных. Понятие доступного выражения. Итеративный алго-ритм вычисления доступных выражений.
6. Полурешетки. Основные свойства полурешеток. Примеры полурешеток. Наибольшая нижняя граница и ее связь с операцией сбора. Диаграммы полурешеток.
7. Структура потока данных. Замкнутость семейства передаточных функций для достигающих определений, живых переменных и доступных выраже-ний. Монотонные и дистрибутивные структуры. Дистрибутивность структур достигающих определений, живых переменных и доступных выражений.
8. Обобщенный итеративный алгоритм. Сходимость итеративного алгоритма к решению уравнений потоков данных. Максимальная фиксированная точка. Сравнение максимальной фиксированной точки с идеальным решением уравнений потока данных и решением сбором по всем путям.
9. Распространение констант как задача потока данных. Передаточные функции структуры распространения констант, ее монотонность и недистрибутивность. Итеративный алгоритм распространения констант.
10. Обход графа потока управления. Глубинное остовное дерево. Алгоритм упорядочения графа потока в глубину. Классификация ребер графа потока управления. Доминаторы. Свойства отношения доминирования. Алгоритм вычисления доминаторов. Дерево доминаторов.
11. Обратные ребра и естественные циклы. Построение естественного цикла обратного ребра.
12. Структурный анализ графа потока управления. Понятие области. Выделение областей в графе потока. Виды областей. Построение иерархии областей для приводимых графов потока. Дерево управления. Алгоритм построения восходящего порядка областей графа потока.

13. Алгоритм анализа на основе областей: построение иерархии областей (снизу вверх) и обработка иерархии областей (сверху вниз). Пересчет передаточных функций. Пример применения алгоритма анализа достигающих на основе областей. Обработка неприводимых графов потоков.
14. Форма статического единственного присваивания (*SSA*-форма). Функции объединения значений (ϕ-функции). Определение ϕ-функции. Свойства ϕ-функций. Базовый алгоритм построения *SSA*-формы.
15. Алгоритм построения квазиоптимальной SSA-формы. Алгоритм построения границы доминирования. Алгоритм переименования переменных.
16. Алгоритм восстановления программы по ее квазиоптимальной *SSA*-форме.
17. Глобальная нумерация значений (постановка задачи). Два подхода к реализации глобальной нумерации значений. Первый подход: использование хэш-функций.
18. Глобальная нумерация значений: конгруэнтные подграфы.
19. Анализ алиасов: определение алиасов, виды алиасов. Глобальный (внутрипроцедурный) анализ алиасов.
20. Недостаточность глобального анализа алиасов. Межпроцедурный анализ алиасов. Контекстно-нечувствительный межпроцедурный анализ и его недостатки. Построение графа вызовов.
21. Межпроцедурный анализ алиасов. Чувствительность к контексту. Строки вызовов. k-ограниченный контекстно-чувствительный анализ.
22. Контекстно-чувствительный анализ на основе клонирования и на основе аннотаций.
23. Симметричные мультипроцессоры с общей памятью (SMP). Многоядерные процессоры. Закон Амдаля. Понятие локальности данных: пространственная и временная локальность. Формальная постановка задачи распараллеливания циклов.
24. Распараллеливание циклов. Пространство итераций. Построение пространств итераций для гнезд циклов. Управление порядком выполнения циклов гнезда. Алгоритм исключения Фурье-Моцкина. Алгоритм вычисления границ циклов для заданного порядка выполнения.
25. Повторное использование данных. Собственные и групповые повторные использования. Анализ зависимостей по данным. Обнаружение параллельности, не требующей синхронизации. Ограничения разбиений пространства и их решение.

Перечень контрольных вопросов для экзамена в конце 7 семестра

1. Генерация кода для базового блока
2. Распределение и назначение регистров
3. Глобальное распределение и назначение регистров. Построение интервалов жизни
4. Глобальное распределение и назначение регистров. Конфликтные ситуации и граф конфликтов. Построение графа конфликтов
5. Глобальное распределение и назначение регистров. Раскраска графа конфликтов сверху вниз. Раскраска графа конфликтов снизу вверх.
6. Глобальное распределение и назначение регистров. Структура распределителя регистров. Алгоритм в целом (на примере)
7. Выбор команд путем переписывания дерева.
8. Генерация оптимального кода для выражений. Числа Ершова.
9. Рекурсивный алгоритм генерации кода для размеченного дерева выражения
10. Генерация кода с использованием динамического программирования
11. Покадровая оптимизация
12. Планирование кода. Анализ зависимостей.
13. Планирование списков базовых блоков
14. Спекулятивное планирование. Опережающее выполнение.
15. Глобальное планирование кода. Алгоритм глобального планирования на основе областей
16. Глобальное планирование кода. Агрессивные алгоритмы перемещения кода
17. Программная конвейеризация циклов.

18. JIT-компиляторы. Профилирование. Сэмплирование. Выбор «горячих» участков
19. JIT-компиляторы. Архитектура JIT-компилятора
20. Параллельное выполнение циклов. Закон Амдаля. Понятие локальности данных
21. Формальная постановка задачи распараллеливания циклов.
22. Распараллеливание циклов. Блокирование: Разбиение на блоки
23. Изменение порядка циклов.
24. Обнаружение параллельности, не требующей синхронизации (на примере)

7. МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ

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

Необходимое программное обеспечение:

Обеспечение самостоятельной работы: Текстовые файлы, доступные на сайтах http://www.thefreecountry.com/programming/compilerconstruction.shtml, http://llvm.org/, http://gcc.gnu.org/, http://suif.stanford.edu/suif/suif2/,

8. НАИМЕНОВАНИЕ ВОЗМОЖНЫХ ТЕМ КУРСОВЫХ РАБОТ – УЧЕБНЫМ ПЛАНОМ НЕ ПРЕДУСМОТРЕНЫ

9. ТЕМАТИКА И ФОРМЫ ИНДИВИДУАЛЬНОЙ РАБОТЫ – УЧЕБНЫМ ПЛАНОМ НЕ ПРЕДУСМОТРЕНЫ

10. ТЕМАТИКА ИТОГОВЫХ РАБОТ – УЧЕБНЫМ ПЛАНОМ НЕ ПРЕДУСМОТРЕНЫ

11. УЧЕБНО-МЕТОДИЧЕСКОЕ И ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ

Основная литература.

1. А.В. Ахо, М.С. Лам, Р. Сети, Дж.Д. Ульман. Компиляторы: принципы, технологии и инструментарий. Издание второе. / М.: ООО «И.Д. Вильямс», 2008, ISBN: 978-5-8459-1349-4
2. K.D. Cooper and L. Torczon. Engineering a Compiler. / Morgan Kaufman Publishers, 2004, ISBN: 1-55860-698-X

Дополнительная литература.

1. Y. N. Srikant, P. Shankar. The Compiler Design Handbook. 2nd edition. CRC press – 2008 ISBN: 978-1-4200-4382-2
2. The **LLVM** Compiler Infrastructure. // http://llvm.org/
3. A GNU Manual. // http://gcc.gnu.org/
4. The SUIF 2 Compiler System. // http://suif.stanford.edu/suif/suif2/
5. Free Compiler Construction Tools. // http://www.thefreecountry.com/programming/compilerconstruction.shtml

Пособия и методические указания.

1. Слайды лекций (Интернет)

Пособие по лекциям разрабатывается.

Программу составил

С. С. Гайсарян, доцент, к.ф.–м.н.

«_____»_____2012 г.