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

Факультет компьютерных наук

Департамент программной инженерии

**Рабочая программа дисциплины**

Научно-исследовательский семинар

**Современные архитектуры вычислителей**

для образовательной программы “Программная инженерия”

направления подготовки 09.03.04 “Программная инженерия”

уровень - бакалавр

Разработчик программы

Баканов В.М.,д.т.н., профессор, vbakanov@hse.ru

Одобрена на заседании департамента программной инженерии “\_25\_”\_мая\_ 2018 г.

Руководитель департамента Авдошин С.М. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Утверждена Академическим советом образовательной программы

“\_25\_”\_мая\_ 2018 г., № протокола\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Академический руководитель образовательной программы

Шилов В.В. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Москва, 2018

*Настоящая программа не может быть использована другими подразделениями университета и другими вузами без разрешения подразделения-разработчика программы.*

|  |  |
| --- | --- |
| Автор | Баканов Валерий Михайлович |
| Число кредитов | 3 |
| Контактная работа (час.) | 100 |
| Самостоятельная работа (час.) | 1,68 |
| Курс | 2 |
| Формат изучения дисциплины | Без использования онлайн курса |

**I. ЦЕЛЬ, РЕЗУЛЬТАТЫ ОСВОЕНИЯ ДИСЦИПЛИНЫ И ПЕРЕКВИЗИТЫ**

К цели научно-исследовательского семинара (НИС) "Современные архитектуры вычислителей" руководитель семинара отно́сит:

* формирование у студентов профессиональных компетенций, связанных как с общей методологией научного исследования, так и с частными аспектами основ математического моделирования и последующей компьютерной реализацией разработанных моделей вычислительных систем разной архитектуры и исполняемых на них алгоритмов
* приобретение студентами навыков работы с научными публикациями (статьями, главами книг, препринтов), в том числе и на английском языке, самостоятельного научного исследования, связанного с анализом, разработкой, программной реализацией (напр., самостоятельная разработка или использование программных сред технической и/или математической направленности) и последующим анализом функционирования вычислительных систем разной архитектуры
* формирование у студентов интереса к исследовательской работе и первоначальных навыков подготовки материала по определённым (предлагаемым) темам и написания отчёта в виде научной статьи (в соответствии с шаблоном оформления статей в формате IEEE – примеры доступны студентам через страницы научно-исследовательского семинара в LMS).

Научно-исследовательский семинар (НИС) "Современные архитектуры вычислителей" предполагает в аспекте общей методологии научного исследования знакомство участников семинара (студентов образовательной программы "Программная инженерия") с основными этапами проведения научного исследования:

* предварительный анализ литературы (состояние проблемы, достигнутые к текущему моменту времени результаты)
* чёткая постановка задачи исследования, предложение собственных вариантов решения задачи (возможных подходов к решению поставленной задачи) и их сравнительный анализ с существующими аналогами
* разработка математических моделей процесса, определение критериев и параметров оптимизации, реализация моделей с помощью выбранных программных средств, проведение вычислительных экспериментов, первичная обработка результатов
* понимание получаемых результатов, их интерпретация, оформление в виде отчёта (сообщение на семинаре, научная статья) с последующей презентацией последнего (желательно).

В процессе обсуждения тем используются слайды и сопроводительные материалы (примеры, учебные программы, системы компьютерной симуляции и др.), специально подготовленные для дискуссий в рамках НИС; ссылки на указанные материалы, отдельные публикации, полезные InterNet-источники, а также программы доступны студентам через систему управления обучением (LMS) НИУ ВШЭ, [http:/lms.hse.ru](http://lms.hse.ru). Студентам настоятельно рекомендуется также активно задействовать электронные ресурсы библиотеки НИУ ВШЭ (<http://library.hse.ru/e-resources/e-resources.htm>) для ознакомления и использования отдельных публикаций ведущих издательств при выполнении работ.

**II. СОДЕРЖАНИЕ УЧЕБНОЙ ДИСЦИПЛИНЫ**

Участникам семинара предлагаются для обсуждения и последующего исследования следующие укрупнённые темы, каждая из которых допускает достаточно широкую детализацию и определённый выбор публикаций (для самостоятельного изучения), а также примеров и задач разного уровня сложности.

Данная тематика соответствует целям формирования у студентов наиболее общих, базирующихся на математических основах, принципов организации и анализа процесса вычислений и является теоретическим базисом для читаемого на 2 курсе лекционного курса дисциплины “Архитектура вычислительных систем и курсов следующих годов обучения:

* Методы формального описания алгоритмов (графы различных типов, их сравнительные достоинства и недостатки). Графы распространённых алгоритмов. Открытая энциклопедия свойств алгоритмов AlgoWiki.
* Анализ графов алгоритмов с целью выявления скрытого параллелизма. Математические модели анализа графов. Ярусно-параллельная форма информационного графа алгоритмов и её возможности при анализе алгоритмов.
* Постановка и решение задач оптимизации процесса выполнения программ на конкретных вычислительных системах. Формулировка целей и задач оптимизации, выбор параметров оптимизации, методы решения оптимизационных задач. Исследовательские программные пакеты METIS и ParMETIS (университет Миннесоты, США), V-Ray и Parus (МГУ, Россия), SPF@home, DATA-FLOW (ВШЭ, Россия) - назначение, возможности, области применения.
* Разработка рациональных планов выполнения параллельных программ на гетерогенных вычислительных системах. Цели и задачи оптимизации. Примеры постановки и решения задач с использование пакета SPF@home.
* Математическое моделирование вычислителей пото́ковой (DATA-FLOW) архитектуры. Постановка задач оптимизации. Выбор параметров оптимизации. Примеры моделирования и оптимизации режимов выполнения задач с помощью программного симулятора потокового вычислителя.

**V.1. Программный инструментарий, применяемый при проведении научно-исследовательского семинара**

**V.1.1. Программная система SPF@home**

* Назначение и основные возможности системы SPF@home: проведение тонкого информационного анализа алгоритмов (заданных информационными графами - ИГА) на наличие скрытого параллелизма (методом представле́ния их в ярусно-параллельной форме - ЯПФ) и разработка рациональных планов выполнения этих алгоритмов на гомогенном или гетерогенном поле параллельных вычислителей. Для последнего случая предусмотрено сопоставление операторов и вычислителей по произвольному числу параметров (напр., требуемый размер памяти заданного типа, вычислительная сложность алгоритма, необходимость обработки данных с плавающей запятой, возможность векторизации вычислений и т. д.); сопоставле́ния проводятся по принципу “попада́ния в заданный диапазон”, результаты сравнения по отдельным параметрам связаны логическим отношением конъюнкции. Для бо́льшего приближения к реальности предусмотрена возможность нагру́зки узлов (*операторов*) и дуг (*линий передачи данных*) графа числовыми ме́триками, которые могут интерпретироваться Исследователем сообразно ситуации (в примитиве как время выполнения операторов и передачи данных).
* Методы реализации: выполнение заданных пользователем преобразований над формами представле́ния графов алгоритмов с использованием скриптового языка Lua; в данном случае язык Lua является обёрткой относительно набора API-вызовов (до 70 штук) системы SPF@home. Входной информацией для обработки является ИГА исследуемого алгоритма в формате списка сме́жных вершин (формат файлов .edg), выходная информация выводится в текстовом (и частично графическом) виде. Собственно стратегия преобразований разрабатывается Исследователем и реализуется на Lua.
* Техническая информация о системе SPF@home: система (программный пакет) разработан для платформы Win’32 GUI, исполняемый код представлен загрузочным файлом spf\_client.exe формата PE (*Portable Executable*) размером 2,9 Mбайт; язык программирования С++ , язык пользовательского интерфейса и руководства по применению – русский.
* Правила использования: комплект программной системы SPF@home доступен для свободной вы́грузки c WEB-сайта автора рабочей программы в виде архива <http://vbakanov.ru/spf@home/content/spf@home.rar> (дополнительная информация см. <http://vbakanov.ru/spf@home/spf@home.htm> и <http://vbakanov.ru/poems_04.htm>).

Табл. 1 иллюстрирует некоторые этапы применение программной системы SPF@home для обработки ИГА простейшего алгоритма (определение действительных корней полного квадратного уравнения а🞨x2+b🞨x+c=0) в ходе выполнении сценария минимизации ширины ЯПФ.

Таблица 1. Исходные данные и примеры преобразований ИГА в системе SPF@home для гомогенного поля параллельных вычислителей

|  |  |  |
| --- | --- | --- |
| *Исходный файл* squa\_eq\_2.edg *описания ИГА (перечень*  *комплементарных*  *вершин графа)* | *ЯПФ алгоритма в*  *каноническом виде* | *ЯПФ алгоритма после*  *применения стратегии*  *минимизации ширины ЯПФ при условии постоянства высоты (один из вариантов решения)* |
| 21  100 109  100 110  101 104  102 107  102 108  103 105  104 105  105 106  106 107  106 108  107 109  108 110  *111 100*  *111 101*  *112 102*  *112 103*  *113 104*  *114 100*  *114 103*  *115 101*  *116 102* | -=- Всего ярусов = 6 -=-  0/6: «111 «112 «113 «114 «115 «116  1/4: 100 101 102 103  2/1: 104  3/1: 105  4/1: 106  5/2: 107 108  6/2: 109» 110» | -=- Всего ярусов = 6 -=-  0/6: «111 «112 «113 «114 «115 «116  1/2: 101 103  2/1: 104  3/2: 105 102  4/2: 106 100  5/2: 107 108  6/2: 109» 110» |

Вещественные корни находятся по формуле x1,x2=(-b ± sqrt(b2-4🞨a🞨c))/(2🞨a), где sqrt – операция взятия квадратного корня. Полный алгоритм содержит 11 операторов (17 - с учетом присвоения начальных значений a,b,c,2,4,-1), нумерация операторов начинается с 100.

Список операторов каждого яруса ЯПФ предваря́ется информационной подстрокой формата “N/M:” , где N – номер яруса, M – число операторов на ярусе (исходные данные располагаются на 0-вом ярусе). Символ ‘«’ указывает на входные данные, ‘»’ – на выходные (результат работы алгоритма).

В результате минимизации ширины ЯПФ графа показано, что рассматриваемый алгоритм (программа) может быть выполнена на 2-х независимых (параллельных) вычислителях вместо 4-х (случай канонической формы ЯПФ). Результат получен методом переноса операторов между ярусами ЯПФ без нарушения информационных зависимостей в алгоритме.

В настоящее время набор готовых алгоритмов (файлы формата .edg) включает перемножение матриц, умножение матриц на вектор, решение систем линейных алгебраических уравнений (СЛАУ) прямым методом Гаусса, МНК-аппроксимация линейной и параболической зависимостями, вычисление коэффициентов парной корреляции, вычисление элементов рекурре́нтных последовательностей (чи́сла Фибоначчи, “трибоначчи”, “квадроначчи”), алгоритм сдва́ивания (отдельно для определённого размера обрабатываемых данных каждый) и др. С целью минимизации работы возможен импорт ИГА из нижеописанной программной системы DATA-FLOW в формат .edg файлов пакета SPF@home.

**V.1.1.1 Типовые темы исследований, проводи́мых с использованием программной системы SPF@home**

* Разработать и реализовать на языке Lua стратегию построения каркаса (обобщённого плана выполнения) параллельной программы при условии минимизации необходимого числа параллельных вычислителей при требовании неувеличения числа ярусов (фактически времени выполнения параллельной программы) для гомогенного поля параллельных вычислителей.
* Разработать и реализовать на языке Lua стратегию преобразования ЯПФ для выполнения программы на заданном гетерогенном поле параллельных вычислителей.
* Разработать и реализовать на языке Lua стратегию построения каркаса параллельной программы при условии минимизации числа параллельных вычислителей при требовании минимального увеличения числа ярусов для гомогенного поля параллельных вычислителей.

Квалификатором (уточни́телем) темы является обычно указание на конкретный обрабатываемый алгоритм (представленный в форме информационного графа формата .edg) из вышеприведённого списка. Особо поощряется представле́ние новых алгоритмов в виде файлов указанного формата в целях расширения базы описаний алгоритмов.

Исследовательская работа оформляется по правилам оформления исследовательских работ (см. ссылку ниже) и предоставляется ведущему научно-исследовательский семинар преподавателю в электронном или печатном виде.

На рис. 1 приведены варианты графической интерпретации результатов преобразования формы информационных графов алгоритмов в системе SPF@home.

|  |  |
| --- | --- |
| a) | б) |
| в) | г) |

Рисунок 1. Графическая интерпретация результатов преобразования формы информационных графов алгоритмов в системе SPF@home:

а) ярусно-параллельная форма (канонический вид) ИГА определения действительных корней полного квадратного уравнения с указанием возможных диапазонов перемещения оператор по ярусам ЯПФ (широкие серые вертикальные линии)

б)  копия экрана части рабочего окна программной системы SPF@home с отрисовкой шири́н ЯПФ по ярусам в виде ленточной диаграммы

в) ленточные диаграммы ширин ярусов ЯПФ до и после преобразования (слева и справа от вертикальной пунктирной линии соответственно) для программы вычисления действительных корней полного квадратного уравнения при условии сохране́ния высоты ЯПФ (условие неувеличения времени выполнения и минимизации ресурсов вычислительной системы – в данном случае количества параллельных вычислителей)

г) ленточные диаграммы ширин ярусов ЯПФ до и после преобразования (масштаб по оси ординат скорректирован) для программы решения СЛАУ десятого порядка прямым методом Гаусса при условии непревышения шириной ЯПФ значения 10 (получен план выполнения программы на 10 параллельных вычислителях, при этом высота ЯПФ возросла с 63 до 141)

**V.1.1.2. Типовые работы участников научно-исследовательского семинара по разработке информационных графов алгоритмов (ИГА)**

Для ясного понимания студентами методов представления алгоритмов в гра́фовой форме даются задания по представлению заданных алгоритмов в форме информационного графа алгоритма (ИГА). ИГА является ориентированным (все дуги напра́влены), параметризованным по размерности входных данных и детерминированным (в нашем случае). Студент должен понимать, что ИГА не является единственной формой представления алгоритма и обладает определёнными недостатками.

Общепринятыми этапами формализации алгоритма (программы) в виде ИГА являются следующие:

* Декомпозиция (разбие́ние на фрагменты). Перед декомпозицией решается вопрос об определении величины зёрен (гра́нул, блоков) программы, которые будут выполняться ато́мно (недели́мо); в дальнейшем эти зёрна могут служить гранулами при параллелизации выполнения алгоритма. В общем случае принято 4 уровня параллелизации (в зависимости от величины характерного размера зерна):

1) параллелизм на уровне би́тов,

2) параллелизм на уровне машинных команд,

3) параллелизм на уровне программных потоков,

4) параллелизм уровня заданий.

Выбор конкретного варианта обусловлен в основном параметрами оборудования, на котором предполагается выполнять программу (в меньшей степени – свойствами самой программы).

* Представле́ние программы в виде блоков (фактически зёрен), связанных информационными связями (связи типа “результат - операнд”). После этого этапа говорят, что программа предста́влена информационным графом алгоритма (ИГА).

В качестве простейшего показательного примера рассмотрим процесс построения ИГА для алгоритма нахождения вещественных корней полного квадратного уравнения а🞨x2+b🞨x+c=0 (где a,b,c – заданные вещественные коэффициенты) в виде x1,x2=(-b ± sqrt(b2-4🞨a🞨c))/(2🞨a), где sqrt – функция вычисления квадратного корня. В качестве зерна выбираем арифметическое действие (фактически уровень машинной команды).

В табл. 2 приведён результат декомпозиции рассматриваемого алгоритма (решение требует 11 арифметических операций, исходными данными являются коэффициенты a, b, c уравнения и необходимые для решения константы 2, 4, -1; результат – вещественные корни x1, x2). Заметим, что практически всегда существуют варианты декомпозиции в соответствии с имеющимся в распоряжении набором команд.

Таблица 2. Результат разбиения на операции  (декомпозиция) алгоритма нахождения вещественных корней полного квадратного уравнения (нумерация операторов на́чата с 100).

|  |  |  |
| --- | --- | --- |
| *№ оператора* | *Действие* | *Примечание* |
| 100 | a2**←**2a | а2 – рабочая переменная |
| 101 | a4**←**4a | а4 – рабочая переменная |
| 102 | b\_neg**←**(-1)b | b\_neg – рабочая переменная, |
| 103 | bb**←**b**^**2 | bb – рабочая переменная |
| 104 | ac4**←**a4c | aс4 – рабочая переменная |
| 105 | p\_sqr**←**bb-a4 | p\_sqr – рабочая переменная |
| 106 | sq**←**sqrt(p\_sqr) | sq – рабочая переменная |
| 107 | w1**←**b\_neg+sq | w1 – рабочая переменная |
| 108 | w2**←**b\_neg-sq | w2 – рабочая переменная |
| 109 | x1**←**w1/a2 | x1 – первый корень уравнения |
| 110 | x2**←**w2/a2 | x2 – второй корень уравнения |

На рис. 2 приведён информационные граф, полученный согласно данных табл. 2 (форма представления графа соответствует последовательному выполнению операций).

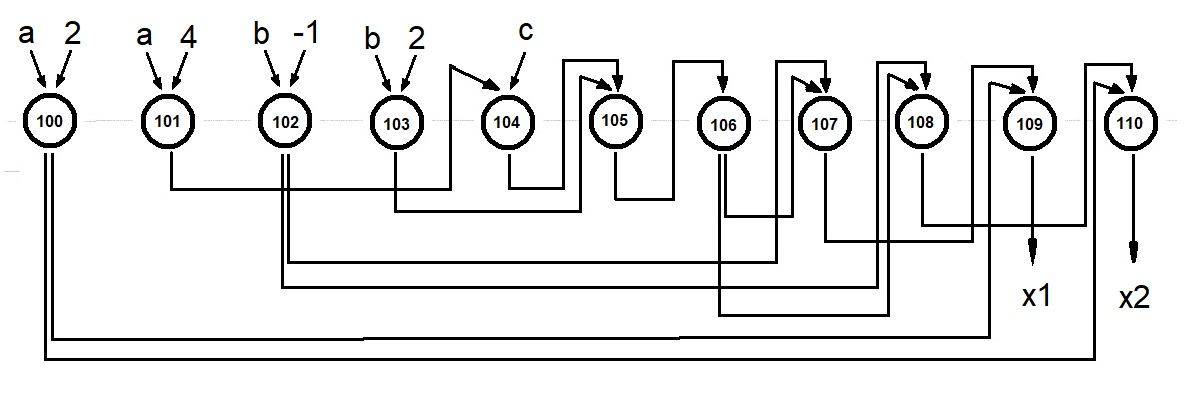


Рисунок 2. Полученный после декомпозиции и учёта информационных связей в программе информационный граф алгоритма (ИГА) нахождения вещественных корней полного квадратного уравнения.

Последним этапом является представле́ние ИГА в максимально компактной форме. Для этого используется метод представления графа в виде списка дуг, задаваемых комплементарными (соответствующими одной дуге) вершинами. Например, соединяющая вершины 101 и 104 дуга может быть представлена в виде пары номеров вершин с одним или большим числом пробелов-разделителей (напр., строкой вида “101 104”, где порядок важен, ибо указывает на направление передачи информации).

Реализация этого этапа состоит в перечислении дуг графа, содержание полученного файла (формат .edg) для данного алгоритма приведено в левом столбце табл. 1. Первой строкой файла является общее число дуг (применяется для удобства программного считывания файла), первые 12 пар вершин соответствуют информационным связам между операторами, оставшиеся – присвоению начальных значений (111↔a, 112↔b, 113↔c, 114↔2, 115↔4, 116↔(-1) соответственно).

Примеры разработанных участниками научно-исследовательского семинара ИГА-файлов (.edg формат) приведены в табл. 3.

Таблица 3. Примеры содержимого разработанных студентами файлов информационных графов алгоритмов.

|  |  |  |  |
| --- | --- | --- | --- |
| *mnk\_5.edg* | *doubling\_32.edg* | *m\_matr\_vec\_5.edg* | *slau\_10.edg* |
| 66  100 138  100 141  100 146  101 111  101 119  101 128  102 111  102 120  102 129  103 112  103 122  103 131  104 113  104 124  104 133  105 114  105 126  105 135  106 115  106 119  107 115  107 120  108 116  108 122  109 117  109 124  110 118  110 126  111 112  112 113  113 114  114 137  114 140  114 144  115 116  116 117  117 118  118 137  118 145  119 121  120 121  121 123  122 123  123 125  124 125  125 127  126 127  127 138  128 130  129 130  130 132  131 132  132 134  133 134  134 136  135 136  136 141  137 139  138 139  139 143  140 142  141 142  142 143  143 144  144 145  145 146 | 62  100 132  101 132  102 133  103 133  104 134  105 134  106 135  107 135  108 136  109 136  110 137  111 137  112 138  113 138  114 139  115 139  116 140  117 140  118 141  119 141  120 142  121 142  122 143  123 143  124 144  125 144  126 145  127 145  128 146  129 146  130 147  131 147  132 148  133 148  134 149  135 149  136 150  137 150  138 151  139 151  140 152  141 152  142 153  143 153  144 154  145 154  146 155  147 155  148 156  149 156  150 157  151 157  152 158  153 158  154 159  155 159  156 160  157 160  158 161  159 161  160 162  161 162 | 90  100 130  101 131  102 133  103 135  104 137  105 139  106 140  107 142  108 144  109 146  110 148  111 149  112 151  113 153  114 155  115 157  116 158  117 160  118 162  119 164  120 166  121 167  122 169  123 171  124 173  125 130  125 139  125 148  125 157  125 166  126 131  126 140  126 149  126 158  126 167  127 133  127 142  127 151  127 160  127 169  128 135  128 144  128 153  128 162  128 171  129 137  129 146  129 155  129 164  129 173  130 132  131 132  132 134  133 134  134 136  135 136  136 138  137 138  139 141  140 141  141 143  142 143  143 145  144 145  145 147  146 147  148 150  149 150  150 152  151 152  152 154  153 154  154 156  155 156  157 159  158 159  159 161  160 161  161 163  162 163  163 165  164 165  166 168  167 168  168 170  169 170  170 172  171 172  172 174  173 174 | 1790  100 210  100 211  100 233  100 234  100 256  100 257  100 279  100 280  100 302  100 303  100 325  100 326  100 348  100 349  100 371  100 372  100 394  100 395  100 1104  101 213  101 236  101 259  101 282  101 305  101 328  101 351  101 374  101 397  101 1094  102 215  102 238  102 261  102 284  102 307  102 330  102 353  102 376  102 399  102 1093  103 217  103 240  103 263  103 286  103 309  103 332  103 355  103 378  103 401  103 1092  104 219  104 242  104 265  104 288  104 311  104 334  104 357  104 380  104 403  104 1091  105 221  105 244  105 267  105 290  105 313  105 336  105 359  105 382  105 405  105 1090  106 223  106 246  106 269  106 292  106 315  106 338  **. . .**  **. . .**  **. . .**  1095 1096  1096 1097  1097 1098  1098 1099  1099 1100  1100 1101  1101 1102  1102 1103  1103 1104 |

В табл. 3 файл mnk\_5.edg представляет собой алгоритм вычисления коэффициентов а и b выражения y=ax+b, определённые методом наименьших квадратов по 5 парам точек x|y. Файл doubling\_32.edg соответствует алгоритму вычисления суммы 32 чисел методом сдва́ивания. Файл m\_matr\_vec\_5.edg представляет собой ИГА умножения квадратной матрицы 5-го порядка на вектор. Файл slau\_10a.edg – решение системы линейных алгебраических уравнений 10-го порядка методом Гаусса.

При анализе файлов ИГА необходимо иметь пред собою таблицу соответствия номеров вершин графа реальным операторам алгоритма (аналогичную табл. 2). Много более эффективным методом получения файлов ИГА является разработка и отладка соответствующей программы для симулятора DATA-FLOW вычислителя с последующей конвертацией в файлы формата .edg (см. подраздел 6.1.2).

Обработка представле́ний ИГА осуществляется программной системой SPF@home, рис. 3. Исходной информацией для этой программы являются файлы ИГА (формат .edg), для эквивалентных (не изменяющих информационных связей) преобразований используются сценарии, разрабатываемые с использованием скриптового языка Lua (в состав системы SPF@home включён текстовый редактор исходного кода Lua и встроенный интерпретатор этого языка).

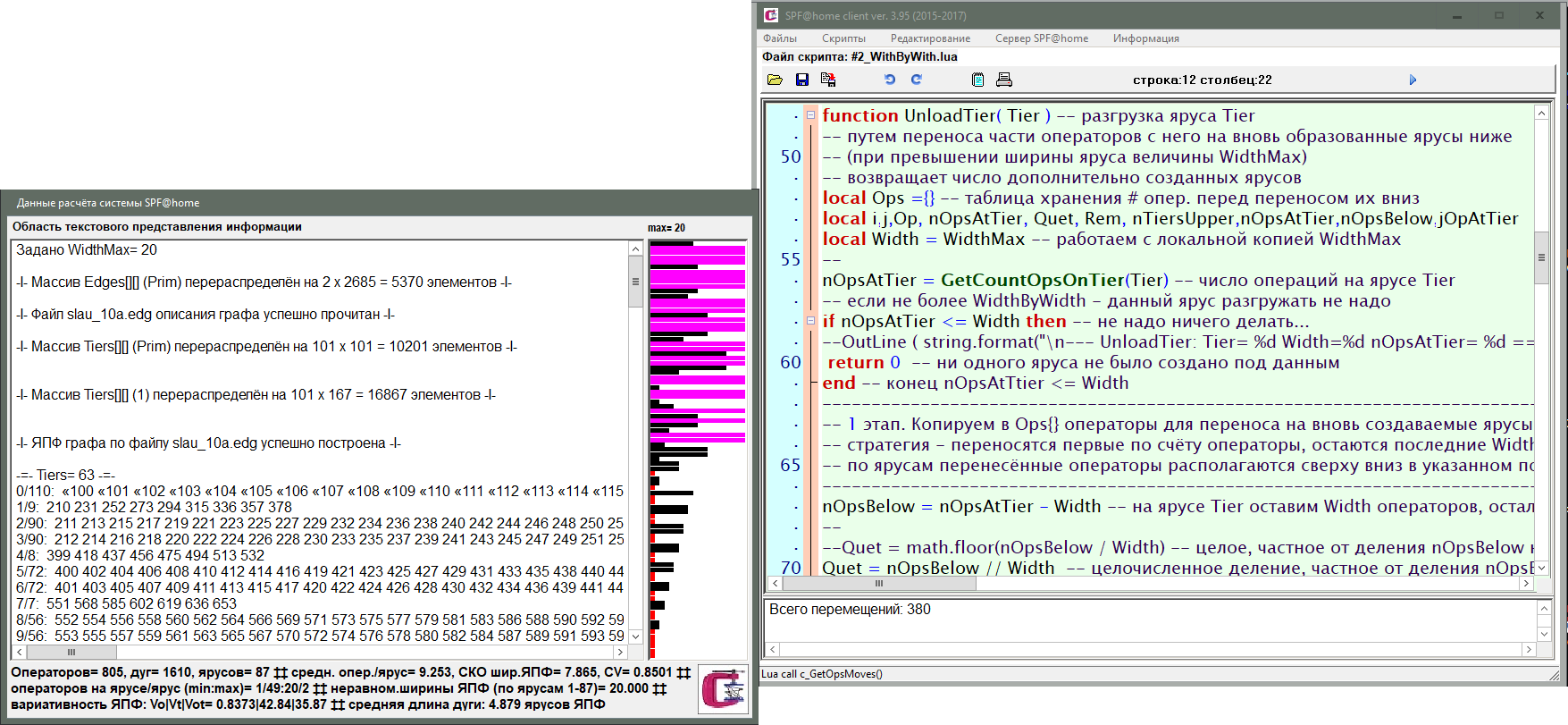


Рисунок 3. Копия экрана при работе программной системы SPF@home.

Программная система SPF@home имеет набор (до 70 штук) API-вызовов, каждый из которых может быть вызван посредством функции-обёртки на языке Lua. Система API-вызовов включает три типа функций (их описание и руководство по использованию включено в состав архива системы):

* информационные (служат для получения информации о текущих параметрах ИГА)
* акцио́нные (осуществляют эквивалентные преобразования ИГА)
* вспомогательные (работа с файловой системой, интерфейс с пользователем, вывод рассчитанных данных в соответствующих форматах).

При работе с системой SPF@home пользователь (Исследователь) разрабатывает сценарий исследования (определяет его цели и методы достижения) и далее осуществляет программирование и отладку сценария на языке Lua, реализующего поставленную цель заданными методами.

Ниже приведены примеры функций, разработанных на языке Lua (согласно синтаксису Lua двойной дефис означает начало комментария).

а) Визуализация ЯПФ для данного ИГА

local function Visual() -- визуализация состояния ЯПФ

AddLineToTextFrame("=====================================")

PutTiersToTextFrame() – вывод в текстовый фрейм

PutParamsTiers() – вывести в текстовом виде параметры ЯПФ

ClearDiagrTiers() – очистить область ленточной диаграммы ширин ЯПФ по ярусам

DrawDiagrTiers() – вывести ленточную диаграмму распределения ширин ярусов ЯПФ

end -- конец функции Visual()

б) Определение яруса ЯПФ, на который рационально переместить оператор (стратегия поиска – найти ярус минимальной ширины по всей ЯПФ)

function SearchToOp\_v01(fromTier, toTier)

-- ищет (среди fromTier-toTier) и возвращает ярус, на который рационально переместить оператор

local minH=1000000000 – большое число для поиска min

local sTier=0 -- возврат, если не нашлось ни одного уровня, на который можно переносить оператор

local H,iTier

for iTier=fromTier+1,toTier do -- по ярусам ЯПФ

H=GetCountOpsOnTier(iTier) -- число операторов на ярусе iTier

if H>=bordH then -- больше среднего = неинтересно!

goto cont\_iTier -- идём на новый цикл по iTier

end

-- теперь остался вариант H < bordH (что нам и нужно!) -- ищем ярус с минимальной шириной

if H<minH then

minH=H

sTier=iTier -- это ярус c min операторов на нём

end -- конец поиска min

::cont\_iTier:: -- заканчиваем цикл

end -- конец цикла по iTier

return sTier -- возвращаем найденное...

end -- конец функции SearchToOp\_v01()

в) Поиск на ярусе ЯПФ с номером iTier оператора с максимальной вариабельностью (диапазоном возможного перемещения среди ярусов ЯПФ)

function DefOpByMaxVar(iTier)

-- находит (ПЕРВЫЙ ПО СЧЁТУ) оператор с max вариабельностью на ярусе iTier

local jOp, iOp, fromTier,toTier,varTier, iOpSave,fromTierSave,toTierSave

local maxVarTier = -1000000000 -- для поиска максимально вариативного оператора на ярусе iTier

for jOp=GetCountOpsOnTier(iTier),1,-1 do -- jOp - номер оператора на ярусе iTier (справа налево)

iOp=GetOpByNumbOnTier(jOp,iTier) -- iOp - номер оператора

fromTier=GetMinTierMaybeOp(iOp) -- верхний ярус вариативности расположения iOp

toTier=GetMaxTierMaybeOp(iOp) -- нижний ярус вариативности расположения iOp

varTier=toTier-fromTier -- вариативностью оператора iOp

if varTier>maxVarTier then -- ищем max( varTier )

iOpSave=iOp -- запомнили глобальный номер оператора с max (на ярусе iTier) вариативностью

fromTierSave=fromTier

toTierSave=toTier

maxVarTier=varTier

end -- конец if varTier>maxVarTier

end -- конец цикла for jOp…

return iOpSave,fromTierSave,toTierSave -- вернули оператор с max вариабельностью и его данные

end -- конец функции DefOpByMaxVar()

**V.1.2. Программная система DATA-FLOW**

* Назначение и основные возможности системы DATA-FLOW: программное моделирование (симуляция) выполнения программ в режиме пото́кового (data-flow) вычислителя. Исходная программа записывается в виде последовательности команд уровня Ассемблера (мнемоники команд трёхсимвольные, команды в основном трёхадресные, нотация команд в стиле AT&T – “результат праве́е операндов”), все вычисления проводятся с вещественными числами двойной точности (тип “double float”). Последовательность выполнения команд определяется условием их “готовности к выполнению” (ГКВ), которое, в свою очередь, определяется условием присваивания конкретных значений всем операндам данного оператора. Имеется возможность задавать время выполнения каждого оператора в относительных единицах - “та́ктах” (“ти́ках”) вычислителя. В системе DATA-FLOW предусмотрена возможность просмотра последовательности исполняемых операторов “вглубь”, что расширяет область исследований процесса потокового выполнения программ. Встроенная система отладки позволяет в интерактивном режиме отслеживать информационные связи в программе в обоих направлениях. Использование псевдомассивов достигается применением макросов и встроенного препроцессора. Естественно, при единичном числе вычислителей реализовывается последовательный вариант выполнения программ.
* Техническая информация о системе DATA-FLOW: система (программный пакет) разработан для платформы Win’32 GUI, исполняемый код представлен загрузочным файлом data\_flow.exe формата PE (*Portable Executable*) размером 1 Mбайт; язык программирования С++ , язык пользовательского интерфейса и руководства по применению - русский.
* Правила использования: комплект программной системы DATA-FLOW доступен для свободной выгрузки c WEB-сайта автора рабочей программы в виде архива <http://vbakanov.ru/dataflow/content/dataflow.rar> (дополнительная информация см. <http://vbakanov.ru/dataflow/dataflow.htm> и <http://vbakanov.ru/poems_04.htm#dataflow>).

Табл. 4 иллюстрирует некоторые данные выполнения программы в симуляторе DATA-FLOW вычислителя (приведены исходный текст программы и дамп памяти).

Таблица 4. Исходная программа и результат её выполнения в системе DATA-FLOW

|  |  |
| --- | --- |
| *Исходный файл* squa\_eq\_2.set *программы определения*  *действительных корней полного квадратного уравнения*  *для исполнения в системе DATA-FLOW* | *Дамп памяти после выполнения*  *программы* squa\_eq\_2.set |
| / точное решение: x1 = -0.45862 ; x2 = -6.5414  /  MUL A, TWO, A2 / A2 ← 2 \* A [#100]  MUL A, FOUR, A4 / A4 ← 4 \* A [#101]  MUL B, NEG\_ONE, B\_NEG / B\_NEG ← NEG\_ONE \* B [#102]  POW B, TWO, BB / BB ← B^2 [#103]  MUL A4, C, AC4 / AC4 ← A4 \* C [#104]  SUB BB, AC4, P\_SQR / P\_SQR ← BB - AC4 [#105]  SQR P\_SQR, SQ / SQ ← sqrt( P\_SQR ) [#106]  ADD B\_NEG, SQ, W1 / W1 ← B\_NEG + SQ [#107]  SUB B\_NEG, SQ, W2 / W2 ← B\_NEG – SQ [#108]  DIV W1, A2, ROOT\_1 / ROOT\_1 ← W1 / A2 [#109]  DIV W2, A2, ROOT\_2 / ROOT\_2 ← W2 / A2 [#110]  SET 1.0, A / A ← 1 / коэффициенты уравнения [#111]  SET 7.0, B / B ← 7 [#112]  SET 3.0, C / C ← 3 [#113]  SET 2, TWO / TWO ← 2 / вспомогательные величины [#114]  SET 4, FOUR / FOUR ← 4 [#115]  SET -1, NEG\_ONE / NEG\_ONE ← (-1) [#116] | A 1.00000e+00  B 7.00000e+00  C 3.00000e+00  TWO 2.00000e+00  FOUR 4.00000e+00  NEG\_ONE -1.00000e+00  A2 2.00000e+00  BB 4.90000e+01  A4 4.00000e+00  B\_NEG -7.00000e+00  AC4 1.20000e+01  P\_SQR 3.70000e+01  SQ 6.08276e+00  W1 -9.17237e-01  W2 -1.30828e+01  ROOT\_1 -4.58619e-01  ROOT\_2 -6.54138e+00 |

Каждый оператор программы в системе DATA-FLOW записывается в отдельной строке, символом начала комментария служит ‘**/**’ (в данном случае в квадратных скобках в поле комментария приведены номера операторов, соответствующие таковым в табл. 1,2).

Ход выполнения программы в системе DATA-FLOW тщательно протоколируется для дальнейшего подробного исследования. В конце выполнения программы Исследователю выдаётся в графическом виде функция интенсивности вычислений (зависимость количества параллельно выполняемых операторов от времени выполнения программы); более подробный анализ этой функции возможен путём импорта протокольных данных расчёта в MS Excel).

Программы (файлы формата .set) для системы DATA-FLOW конвертируются в файлы формата .edg, являющиеся исходными для анализа посредством SPF@home (например, приведённая в табл. 4 программа squa\_eq\_2.set после конвертации имеет имя squa\_eq\_2.edg и приведена в табл. 1).

**V.1.2.1. Типовые темы исследований, проводимых с использованием программной системы DATA-FLOW**

* Исследовать зависимость числа параллельных вычислителей, необходимых для выполнения заданной программы, от размера обрабатываемых данных. Сравнить полученные значения с теоретически предсказанными.
* Исследовать зависимость времени параллельного выполнения заданной программы от числа параллельных вычислителей. Сравнить полученные значения с теоретически предсказанными.
* Исследовать влияние на интенсивность вычислений на DATA-FLOW-вычислительной системе возможностей предпросмотра операторов “вглубь” информационного графа. Выявить и объяснить эффекты интенсификации и депрессации интенсивности.
* Сравнить план выполнения программы, полученный симуляцией DATA-FLOW-вычислительной системе, с результатами анализа системой SPF@home. Проанализировать и объяснить различия, предложить варианты модернизации принципов действия потокового вычислителя.

Квалификатором (уточни́телем) темы является обычно указание на конкретную программу (представленную в форме файла формата .set) из вышеприведённого списка. Особо поощряется разработка новых алгоритмов в виде файлов указанного формата в целях расширения базы описаний алгоритмов (программ).

Исследовательская работа оформляется по правилам оформления исследовательских работ (см. ссылку ниже) и предоставляется ведущему научно-исследовательский семинар преподавателю в электронном или печатном виде.

На рис. 4 приведены фрагменты результатов графической обработки экспериментальных данных студентами при выполнении исследований

|  |  |
| --- | --- |
| a) | б) |
| в) | г) |
| д) | е) |

Рисунок 4. Графическая интерпретация результатов исследований режимов обработки данных программным симулятором DATA-FLOW вычислителя:

а) график интенсивности вычислений для программы вычисления суммы 32 чисел методом сдваивания при выполнении на 16 параллельны вычислителях

б) то же при выполнении на 12 параллельных вычислителях

в) интенсивность вычислений *IP*(t) в функции времени *t* выполнения программы умножения матриц 10-го порядка прямым методом Гаусса при числе параллельных вычислителей *P*=1000, 700, 400

г) зависимость коэффициента ускорения *k*accel вычислений (относительно последовательного выполнения) от числа *P* параллельных вычислителей при умножении матриц порядка *N*=5,7,10 прямым методом Гаусса

д) просмотр “вглубь” операторов, исполняемых DATA-FLOW-машиной (для оператора *i*, где *n*j - количество операндов для оператора *j*)

е) результат моделирования управления интенсивностью вычислений: 1 - равноприоритетное выполнение списка ГКВ-операторов, 2 - режим интенсификации, 3 - депрессивный режим (где *P* - число параллельных вычислителей, *t* - относительное время вычислений)

**V.1.3. Порядок прохождения занятий научно-исследовательского семинара**

* На первых по расписанию занятиях преподаватель информирует о общем направлении научно-исследовательского семинара, распределяет индивидуальные задания для исследований и информирует о способах получения материалов поддержки исследований, озвучивает примерные темы докладов на семинаре.
* Занятия предполагают интерактивный режим взаимодействия с преподавателем:
* Преподаватель уточняет индивидуальные задания, рассматривает вопросы теории и их практическое применение при проведении исследований, отвечает на конкретные вопросы участников семинара. Практика работы с программными средствами и их функционал демонстрируются посредством проектора в интерактивном режиме с участниками семинара. При необходимости участники семинара сообщают (методом кратких сообщений с использованием доски или проектора слайдов) о результатах своих исследований и возникающих проблемах, проблемы обсуждаются коллективно. Для быстрой связи (в промежутках между семинарами) используются электронные средства (в основном электронная почта).
* Вторая часть семинара посвящается сообщениям студентов на определённую ведущим семинар или самопредложенную тему. Сообщение (доклад) должны содержать элемент научного исследования, доклад обсуждается участниками семинара, оценку выставляет преподаватель.
* Последние перед экзаменом занятия посвящаются анализу преподавателем выполненных исследовательских работ, при этом озвучиваются положительные и отрицательные примеры выполнения; предполагается участие в обсуждении участников семинара.

**III. ОЦЕНИВАНИЕ**

**III.1. Формы контроля знаний студентов**

**текущий контроль:**

* обсуждение результатов подготовленных студентами сообщений по заданной преподавателем (или согласованной с ним) научно-исследовательской тематике
* демонстрация и разбор программных систем, разработанных в целях пояснения функционирования вычислителей разных архитектур
* текущие результаты заданной исследовательской работы

**итоговый контроль:** устный экзамен в конце 3 модуля, 3 час. Экзамен в основном предполагает обсуждение и защиту проведённой студентами исследовательской работы.

**III.2. Критерии оценки знаний и навыков**

**В текущем контроле** учитывается посещаемость научного семинара и активность участников (задаваемые вопросы, обсуждение тех или иных аспектов рассматриваемых тем, публикаций, предлагаемые варианты решения поставленных задач, обсуждение выступлений других участников семинара, демонстрация подготовленных программ и т.п.; за каждое сообщение выставляется оценка). При непосещении семинара автоматически ставится оценка "0" (ноль).

Все оценки выставляются по 10-ти балльной шкале. По результатам текущего контроля формируется нако́пленная оценка Онакопл. как среднеарифметическое оценок всех подготовленных сообщений на семинаре (за каждое выступление свыше 3-х добавляется 1 балл к текущей оценке).

**Результирующая оценка** Орезульт.вычисляется по формуле:

Орезульт. = 0,5 × Онакопл. + 0,5 × Оэкз. ,

где Оэкз. - оценка на экзамене; способ округления - арифметический. Компонента Оэкз. является оценкой исследовательской работы студента.

Перевод результирующей оценки в оценку по пятибалльной шкале осуществляется в соответствии со следующей таблицей:

Таблица соответствия оценок по десятибалльной и пятибалльной системам:

|  |  |
| --- | --- |
| *По десятибалльной шкале* | *По пятибалльной шкале* |
| 1 – неудовлетворительно  2 – очень плохо  3 – плохо | неудовлетворительно – 2 |
| 4 – удовлетворительно  5 – весьма удовлетворительно | удовлетворительно – 3 |
| 6 – хорошо  7 – очень хорошо | хорошо – 4 |
| 8 – почти отлично  9 – отлично  10 – блестяще | отлично – 5 |

**V. РЕСУРСЫ**

**V.1. Основная литература**

1. Э.Таненбаум, Т.Остин. Архитектура компьютера (издание 6). — СПб.: Питер, 2014. — 811 с.

**V.2. Дополнительная литература:**

1. Макконелл, Дж. Основы современных алгоритмов: учеб. пособие / Дж.Макконелл; пер. с англ. С.К.Ландо; доп. М.В.Ульянова. - 2-е изд., доп. — М.: Техносфера, 2006. — 366 с.
2. Барский А.Б. Параллельные информационные технологии: учеб. пособие / А.Б.Барский. — М.: Интернет-Университет Информационных Технологий, 2007. — 503 с.
3. Кормен Т. Алгоритмы: построение и анализ / Т.Кормен, Ч.Лейзерсон, Р.Ривест, К.Штайн; пер. с англ. под ред. И.В.Красикова. - 2-е изд. — М.; СПб.; Киев: Вильямс, 2011. — 1290 с.
4. Стивенс Р. Алгоритмы: теория и практическое применение / Р.Стивенс; пер. с англ. В.Кириленко, Р.В.Волошко. — М.: Изд-во "Э", 2016. — 544 с.

# V.3. Программное обеспечение

|  |  |  |
| --- | --- | --- |
| *№ п/п* | *Наименование* | *Условия доступа* |
| 1 | Microsoft Windows 7 Professional RUS  Microsoft Windows 10  Microsoft Windows 8.1 Professional RUS | *Из внутренней сети университета (договор)* |
| 2 | Microsoft Office Professional Plus 2010 | *Из внутренней сети университета (договор)* |
| 3 | Data\_Flow | URL: [http://vbakanov.ru/dataflow/content/ dataflow.rar](http://vbakanov.ru/dataflow/content/%20dataflow.rar) (доступ свободный) |
| 4 | SPF@home | URL: [http://vbakanov.ru/spf@home/content/ spf@home.rar](http://vbakanov.ru/spf@home/content/%20spf@home.rar) (доступ свободный) |

**V.4. Профессиональные базы данных, информационные справочные системы, интернет-ресурсы (электронные образовательные ресурсы)**

|  |  |  |
| --- | --- | --- |
| *№ п/п* | *Наименование* | *Условия доступа* |
|  | ***Интернет-ресурсы (электронные образовательные ресурсы)*** | |
| 1 | AlgoWiki. Открытая энциклопедия свойств алгоритмов. | URL: <https://algowiki-project.org/ru/> (доступ свободный) |
| 2 | Parallel.ru. Сервер Лаборатории Параллельных информационных технологий НИИВЦ МГУ. | URL: <https://parallel.ru/> (доступ свободный) |
| 3 | Оформление учебной и научной работы методическое пособие. | URL: [http://library.mstu.edu.ru/help/files/ ucheb\_work.pdf](http://library.mstu.edu.ru/help/files/%20ucheb_work.pdf) (доступ свободный) |

**V.5. Материально-техническое обеспечение дисциплины**

Учебные аудитории для лекционных занятий по дисциплине обеспечивают использование и демонстрацию тематических иллюстраций, соответствующих программе дисциплины в составе:

ПЭВМ с доступом в Интернет (операционная система, офисные программы, антивирусные программы);

мультимедийный проектор с дистанционным управлением.