Министерство науки и высшего образования Российской Федерации
Нижнекамский химико-технологический институт (филиал)
федерального государственного бюджетного образовательного учреждения
высшего образования
«Казанский национальный исследовательский технологический университет»
(НХТИ ФГБОУ ВО «КНИТУ»)

ДОКУМЕНТ ПОДПИСАН
ЭЛЕКТРОННОЙ ПОДПИСЬЮ

Сертификат: 02C581C300FBAC24844729BEECFD34536D
Владелец: Ахметов Ильдар Гумерович
Действителен: с 30.03.2021 до 30.06.2022

УТВЕРЖДАЮ
Заместитель директора по УР
_____ Н.И. Никифорова
«12» 04 2021 г.

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ

по дисциплине

Б1.О.12 «Архитектура параллельных вычислительных систем»
(наименование дисциплины (модуля))

09.04.01 «Информатика и вычислительная техника»
(код и наименование направления подготовки/ специальности)

Автоматизированные системы обработки информации и управления
(наименование профиля)

магистр
квалификация

очная
форма обучения

Нижнекамск, 2021

Составитель ФОС:
зав. кафедрой ИСТ  О.В. Матухина

ФОС рассмотрен и одобрен на заседании кафедры ИСТ, реализующей подготовку основной образовательной программы,
протокол от 15.03.2021.

Зав. кафедрой ИСТ  О.В. Матухина

Эксперт:
Матухина О.В., зав. кафедрой ИСТ НХТИ ФГБОУ ВО «КНИТУ» 

Перечень компетенций и индикаторов достижения компетенций с указанием этапов формирования в процессе освоения дисциплины

Компетенция:

ОПК-3. Способен самостоятельно решать задачи управления в сфере наукоемких технологий и экономики инноваций на базе последних достижений науки и техники.

Индикаторы достижения компетенции:

ОПК- 3.1. Знает последние достижения науки и техники в своей сфере профессиональной деятельности, основы инновационного менеджмента, механизмы управления наукоемкими производствами;

ОПК-3.2. Умеет планировать инновационные процессы хозяйствующих субъектов, мезоэкономических систем и национальных хозяйств; организовывать научно-исследовательские и опытно-конструкторские работы на предприятии, опираясь на последние достижения науки и техники; координировать развитие наукоемких производств с научно-техническим прогрессом;

ОПК-3.2. Владеет методами планирования, организации, мотивации и контроля инновационной деятельности хозяйствующих субъектов, в том числе высокотехнологичных компаний; методами интенсификации развития наукоемких отраслей и национальных инновационных систем.

Компетенция:

ОПК-9. Способен осуществлять профессиональную эксплуатацию оборудования и приборов для решения задач управления.

Индикаторы достижения компетенции:

ОПК-9.1. Знает функциональное назначение и принципы работы электронно-вычислительных машин, контрольно-измерительных приборов и других технических средств, используемых в процессе реализации управленческих функций; конфигурацию аппаратных средств и программного обеспечения, необходимых для решения управленческих задач в сфере профессиональной деятельности

ОПК – 9.2. Умеет эксплуатировать специализированное оборудование и приборы в процессе реализации профессиональных управленческих задач;

ОПК – 9.3. Владеет правилами профессиональной эксплуатации специализированного оборудования и приборов, используемых в рамках решения управленческих задач.

Индикаторы достижения компетенции	Этапы формирования в процессе освоения дисциплины				Наименование оценочного средства
	Лекции	Практические занятия	Лабораторные занятия	Курсовой проект (работа)	
ОПК-3.1	Разделы дисциплины 1-4.	Не предусмотрен учебным планом	Разделы дисциплины 2-4.	Не предусмотрен учебным планом	Расчетно-графические работы, лабораторные работы, экзамен
ОПК-3.2	Разделы дисциплины 1-4.	Не предусмотрен учебным планом	Разделы дисциплины 2-4.	Не предусмотрен учебным планом	Расчетно-графические работы, лабораторные работы, экзамен
ОПК-3.3	Разделы дисциплины 1-4.	Не предусмотрен учебным планом	Разделы дисциплины 2-4.	Не предусмотрен учебным планом	Расчетно-графические работы, лабораторные работы, экзамен
ОПК-9.1	Разделы дисциплины 1-4.	Не предусмотрен учебным планом	Разделы дисциплины 2-4.	Не предусмотрен учебным планом	Расчетно-графические работы, лабораторные работы, экзамен
ОПК-9.2	Разделы дисциплины 1-4.	Не предусмотрен учебным планом	Разделы дисциплины 2-4.	Не предусмотрен учебным планом	Расчетно-графические работы, лабораторные работы, экзамен
ОПК-9.3	Разделы дисциплины 1-4.	Не предусмотрен учебным планом	Разделы дисциплины 2-4.	Не предусмотрен учебным планом	Расчетно-графические работы, лабораторные работы, экзамен

Перечень оценочных средств по дисциплине

Оценочные средства	**Кол-во**	**Min, баллов (базовый уровень)**	**Max, баллов (повышенный уровень)**
Лабораторные работы	3	18	30
Расчетно-графические работы	2	18	30
Экзамен	1	24	40

Шкала оценивания

Цифровое выражение	**Выражение в баллах:**	**Словесное выражение**	**Критерии оценки индикаторов достижения при форме контроля:**
			экзамен
5	87 - 100	Отлично	Оценка «отлично» выставляется студенту, если теоретическое содержание курса освоено полностью, без пробелов; исчерпывающе, последовательно, четко и логически стройно излагает материал; свободно справляется с задачами, вопросами и другими видами применения знаний; использует в ответе дополнительный материал все предусмотренные программой задания выполнены, качество их выполнения оценено числом баллов, близким к максимальному; анализирует полученные результаты; проявляет самостоятельность при выполнении заданий
4	74 - 86	Хорошо (	Оценка «хорошо» выставляется студенту, если теоретическое содержание курса освоено полностью, необходимые практические компетенции в основном сформированы, все предусмотренные программой обучения учебные задания выполнены, качество их выполнения достаточно высокое. Студент твердо знает материал, грамотно и по существу излагает его, не допуская существенных неточностей в ответе на вопрос.
3	60 - 73	Удовлетворительно	Оценка «удовлетворительно» выставляется студенту, если теоретическое содержание курса освоено частично, но пробелы не носят существенного характера, большинство предусмотренных программой заданий выполнено, но в них имеются ошибки, при ответе на поставленный вопрос студент допускает неточности, недостаточно правильные формулировки, наблюдаются нарушения логической последовательности в изложении программного материала.
2	Ниже 60	Неудовлетворительно	Оценка «неудовлетворительно» выставляется студенту, если он не знает значительной части программного материала, допускает существенные ошибки, неуверенно, с большими затруднениями выполняет практические работы, необходимые практические компетенции не сформированы, большинство предусмотренных программой обучения учебных заданий не выполнено, качество их выполнения оценено числом баллов, близким к минимальному

Краткая характеристика оценочных средства

№ п/п	*Наименование оценочного средства*	*Краткая характеристика оценочного средства*	*Представление оценочного средства в фонде*
1	*2*	*3*	*4*
1.	Лабораторная работа	Это вид учебной работы, целью которой является изучение (исследование, измерение) характеристик лабораторного объекта. Цель лабораторных занятий: освоение изучаемой учебной дисциплины; приобретение навыков практического применения знаний учебной дисциплины (дисциплин) с использованием технических средств и (или) оборудования	Темы лабораторных работ, контрольные вопросы по теме лабораторной работы, вопросы к коллоквиуму
2.	Расчетно-графическаяработа	Средство проверки умений применять полученные знания по заранее определенной методике для решения задач или заданий по модулю или дисциплине в целом.	Комплект заданий для выполнения расчетно-графической работы
3.	Экзамен	Система стандартизированных заданий, позволяющая автоматизировать процедуру измерения уровня знаний и умений обучающегося.	Фонд тестовых заданий

Министерство науки и высшего образования Российской Федерации
Нижнекамский химико-технологический институт (филиал)
федерального государственного бюджетного образовательного учреждения
высшего образования
«Казанский национальный исследовательский технологический университет»

Факультет информационных технологий
Кафедра информационных систем и технологий
Направление подготовки 09.04.01 «Информатика и вычислительная техника»

Учебным планом по направлению подготовки 09.04.01 «Информатика и вычислительная техника» для обучающихся предусмотрено проведение лабораторных работ по дисциплине «Архитектура параллельных вычислительных систем».

Лабораторная работа 1
Параллельное программирование для систем с общей памятьюс использованием технологии OpenMP.

Цель работы: –изучить технологию OpenMP и основы разработки параллельных программ для многоядерных процессоров, написать и отладить на языке С параллельную программу для решения поставленной задачи на системе с общей памятью.

1.1. Основные сведения об OpenMP

Технология OpenMP [10, 11] в настоящее время является одним из наиболее популярных средств параллельного программирования для компьютеров с общей памятью, базирующейся на традиционных последовательных языках программирования и использовании специальных комментариев. За основу берётся последовательная программа, а для создания её параллельной версии пользователю предоставляется набор директив, функций и переменных окружения. Технология OpenMP нацелена на то, чтобы пользователь имел один и тот же вариант программы как для параллельного, так и для последовательного режима выполнения. Параллелизм в OpenMP реализуется с помощью многопоточности. При запуске программы создается единственный «главный» (master) поток, который затем создает набор «подчиненных» (slave) потоков, и вычисления распределяются между всеми потоками. Предполагается, что потоки выполняются параллельно на машине с несколькими процессорами (ядрами), причём количество процессоров/ядер не обязательно должно быть больше или равно количеству потоков. Другими словами, потоки параллельной программы могут обычным образом конкурировать между собой за время процессора/ядра.

Важным достоинством технологии OpenMP является возможность реализации так называемого инкрементального программирования, когда программист постепенно находит участки в программе, содержащие ресурс параллелизма, с помощью предоставляемых механизмов делает их параллельными, а затем переходит к анализу следующих участков. Таким образом, распараллеленные участки постепенно охватывают всё большую часть программы. Этот подход значительно облегчает процесс адаптации последовательных программ к параллельным компьютерам, а также отладку и оптимизацию программ.

Модель исполнения параллельной программы, подготовленной с помощью технологии OpenMP, можно сформулировать следующим образом:

- Программа содержит набор последовательных и параллельных областей (или секций или регионов).
- В начальный момент времени создается главный поток, выполняющий впоследствии все последовательные области программы.
- При входе в параллельную область главным потоком выполняется операция fork, порождающая совокупность подчиненных потоков. Каждый поток имеет свой уникальный числовой идентификатор (главному потоку соответствует 0). При распараллеливании циклов все параллельные потоки исполняют один и тот же

код, но с разными данными. В общем случае потоки могут исполнять различные фрагменты кода.

- При выходе из параллельной области всеми потоками выполняется операция join. Завершается выполнение всех потоков, кроме главного.

На рис. 1. проиллюстрировано создание и исполнение двух параллельных областей. В первой области все потоки приходят к моменту выхода из нее практически одновременно, во второй – в разные моменты времени (кАк правило, это более соответствует реальной картине).

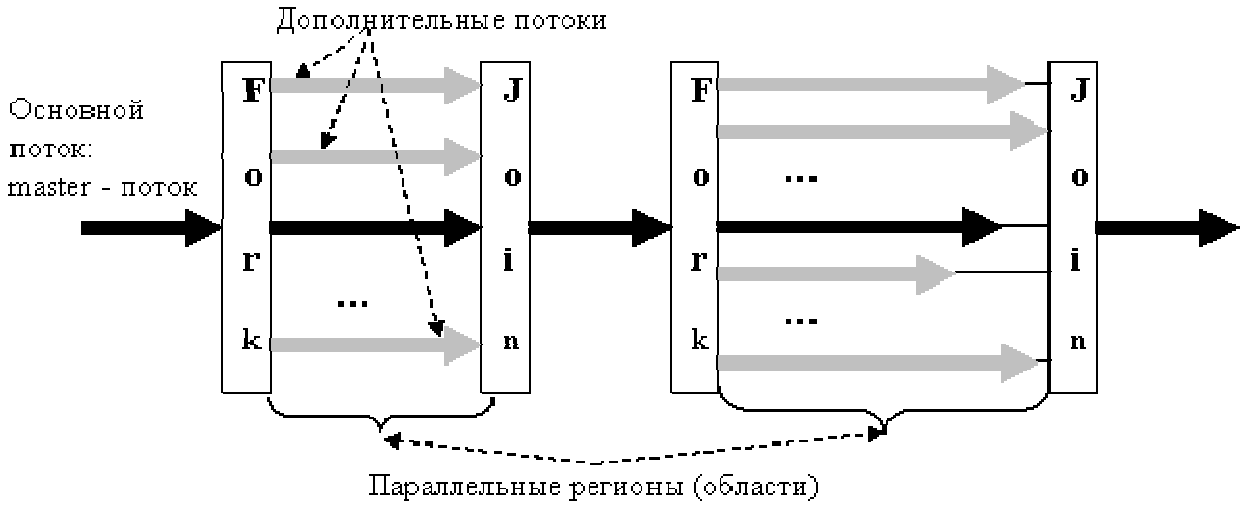

Рис. 1. Создание двух параллельных регионов

Обычно допускается возможность образования вложенных параллельных областей, когда поток, созданный как дополнительный, становится master-потоком для новой параллельной области, как показано на рис. 2.

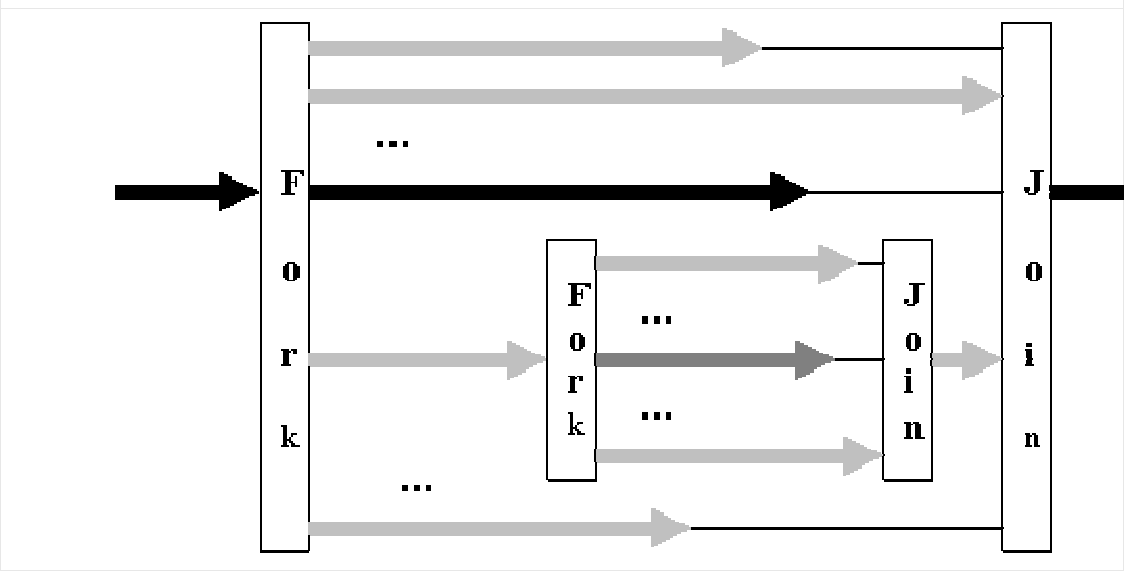

Рис. 2. Создание вложенных параллельных регионов

Под OpenMP понимается совокупность следующих компонент:

- Директивы компилятора – используются для создания потоков, распределения работы между потоками и их синхронизации. Директивы включаются программистом в исходный текст программы.

- Подпрограммы библиотеки времени выполнения – используются для установки и определения атрибутов потоков. Вызовы этих подпрограмм включаются программистом в исходный текст.

- Переменные окружения – используются для управления поведением параллельной программы. Переменные окружения задаются для среды выполнения параллельной программы соответствующими средствами операционной системы.

Использование директив компилятора и подпрограмм библиотеки времени выполнения подчиняется правилам, которые различаются для разных языков программирования. Совокупность таких правил для одного языка программирования называется привязкой к языку. Технология OpenMP создана и поддерживается для языков C, C++ и Fortran. В этой лабораторной работе предполагается использование языка C/C++ в среде разработки Microsoft Visual Studio (начиная с версии 2005). Для того, чтобы ее компилятор нужным образом реагировал на директивы OpenMP и строил параллельную программу, в командную строку его запуска должна быть включена опция /openmp (например, путем включения флажка «OpenMP support» в свойствах проекта на закладке Configuration properties\C/C++\Language).

Для того, чтобы в программе на языке C/C++ стали доступны возможности технологии OpenMP, в нее нужно включить заголовочный файл omp. h:

#include <omp.h>

Если эту программу транслировать компилятором, не поддерживающим технологию OpenMP, то она будет построена в обычном последовательном (однопоточном) варианте, поскольку все директивы, задающие распараллеливание, оформляются в виде так называемых прагм (рекомендаций), и просто игнорируются такими компиляторами.

В программах на языке C/C++ все прагмы, имена функций и переменных окружения OpenMP начинаются со строки «omp». Формат директивы:

#pragma omp директива [опция_1[, опция_2, ...]]

Каждая директива вместе со всеми ее опциями обязательно должна занимать ровно одну строку текста. Действие некоторых директив распространяется только на один оператор (или блок операторов, заключенных в фигурные скобки), непосредственно следующий за директивой в тексте программы. Для таких директив будет указан

<структурный блок кода>.
Перечень **директив OpenMP** включает в себя:

1. Директива задания параллельно выполняемой секции:

#pragma omp parallel [опция_1[, опция_2, ...]]
<структурный блок кода>

С помощью опций этой директивы можно указать:

– требуемое количество потоков n (*num_threads(n)*);
– условие, при котором параллельная область действительно создается (*if(условие)*);

– список общих переменных для всех потоков данной секции (*shared(список переменных)*);

– список переменных, которые будут локальными в каждом потоке секции (*private(список переменных)*), причем их начальные значения не будут определены;

– список переменных, которые будут локальными в каждом потоке секции (*firstprivate(список переменных)*), причем в качестве их начальных значений будут установлены значения одноименных переменных из главного потока;

– способ назначения класса памяти *default(shared|none)* всем переменным потоков, которым класс не назначен явно с помощью опции *shared* (слово *none* означает, что класс памяти всех локальных переменных должен быть задан явно); в реализациях для языка Fortran могут назначаться классы *private* и *firstprivate*;

– список переменных, объявленных директивой *threadprivate* (см. ниже), которые при входе в параллельную секцию инициализируются значениями соответствующих переменных в потоке-мастере;

– оператор сведения и список общих переменных *reduction(оператор : список переменных)*; для каждой указанной в списке переменной создаются локальные копии в каждом потоке; локальные копии инициализируются соответственно типу оператора (для аддитивных операций ноль или его аналоги, для мультипликативных операций единица или её аналоги); над всеми локальными копиями каждой переменной после завершения параллельной секции будет выполнен заданный оператор сведения, результаты будут занесены в одноименные общие переменные; в качестве оператора можно указывать: +, –, *, &, |, ^, &&, ||.

2. Директива определения цикла, итерации которого нужно распределить между параллельно выполняемыми потоками:

#pragma omp for [опция_1[, опция_2, ...]]
<структурный блок кода>

С использованием опций директивы for можно указать:

– список переменных, которые будут локальными в каждом потоке секции (*private(список переменных)*), причем их начальные значения не будут определены;

– список переменных, которые будут локальными в каждом потоке секции (*firstprivate(список переменных)*), причем в качестве их начальных значений будут установлены значения одноименных переменных из главного потока;

– список переменных главного потока (*lastprivate(список переменных)*), которым будут присвоены значения, полученные при выполнении последней итерации цикла;

– оператор сведения и список общих переменных *reduction(оператор : список переменных)*; для каждой указанной в списке переменной создаются локальные копии в каждом потоке; локальные копии инициализируются соответственно типу оператора (для аддитивных операций ноль или его аналоги, для мультипликативных операций единица или её аналоги); над всеми локальными копиями каждой переменной после завершения параллельной секции будет выполнен заданный оператор сведения, результаты будут занесены в одноименные общие переменные; в качестве оператора можно указывать: +, –, *, &, |, ^, &&, ||;

– способ распределения итераций цикла между потоками параллельной секции (*schedule(type[, chunk])*); параметр *chunk* этой опции определяет количество итераций на один поток (по умолчанию 1), параметр *type* указывает тип распределения и может иметь значения:

§ *static* – статический, т. е. при компиляции,

§ *dynamic* – динамический, т. е. при выполнении,

§ *guided* - динамический с уменьшением количества итераций на поток от начального, определяемого автоматически, до значения *chunk*,

§ *auto* – способ распределения выбирается компилятором или исполняющей системой,

§ *runtime* – способ распределения задается специальной переменной окружения операционной системы;

– возможность (опция *ordered*) появления в теле цикла директивы *ordered*, которая требует исполнения охваченного ею блока операторов в точности в той же последовательности, какая реализуется в последовательной версии данного цикла;

– отмену (*nowait*) неявной барьерной синхронизации потоков, достигших конца выполнения своей части итераций цикла (при отсутствии этой опции участок программы после цикла будет выполняться только тогда, когда все потоки выполнят все свои итерации;

– глубину n вложенных друг в друга циклов (*collapse(n)*), пространство итераций которых подлежит распределению между параллельными потоками (доступно только в реализации 2.5 технологии OpenMP);

Директивы *parallel* и *for* можно объединять в одну директиву, в которой можно указывать опции как директивы *parallel*, так и директивы *for*:

#pragma omp parallel for [опция_1[, опция_2, ...]]

3. Директива, указывающая на необходимость исполнения охватываемого ею участка кода (части тела цикла) в той последовательности, в которой он выполняется в чисто последовательном режиме

#pragma omp ordered
<структурный блок кода>

4. Директива задания области нециклического параллелизма (участки структурного блока кода, которые могут выполняться параллельно, выделяются с помощью описываемой далее директивы *section*):

#pragma omp sections [опция_1[, опция_2, ...]]
<структурный блок кода>

В качестве опций этой директивы можно указывать *private, firstprivate, lastprivate, reduction* и *nowait*, имеющие в точности такой же синтаксис и тот же смысл, что и у директивы *for.*

5. Директива определения участка нециклического кода для одного потока:

#pragma omp section

С помощью этой директивы внутри участка кода, охваченного директивой *sections*, выделяются отдельные фрагменты для исполнения параллельными потоками. Перед самым первым фрагментом участка нециклического кода директиву *section* можно не указывать.

6. Директива объявления списка локальных переменных потоков

#pragma omp threadprivate(список переменных)

Эта директива позволяет сделать локальные копии для статических переменных языка C/C++ (и COMMON-блоков языка Фортран), которые по умолчанию являются общими.

7. Директива создания отдельной независимой задачи (начиная с версии 2.5 OpenMP):

#pragma omp task [опция_1[, опция_2, ...]]
<структурный блок кода>

Текущий поток создает в качестве задачи ассоциированный с директивой блок операторов. Эта задача может выполняться немедленно после создания или быть отложенной на неопределённое время и выполняться по частям. Размер таких частей, а также порядок выполнения частей разных отложенных задач определяется реализацией OpenMP.

Лабораторная работа 2

Распараллеливание вычислений при решении задач матричных вычислений.

Цель работы: –изучить методы и алгоритмы разработки параллельных программ для решения задач матричных вычислений.

В лабораторной работе необхолимо разработать параллельную программу для решения задач:

1. Решение системы линейных алгебраических уравнений.

2. Сортировка массива данных различными методами.

3. Перемножение матриц.

4. Вычисление обратной матрицы.

Рекомендации к задаче № 1 – решение системы линейных алгебраических уравнений (СЛАУ) большой размерности. Матрица коэффициентов СЛАУ выдается преподавателем или создается самостоятельно с использованием программы GenSLAU, формирующей либо текстовый, либо бинарный файл (по выбору студента). И в том и в другом случае первым элементом файла является размерность матрицы N, следующие N групп элементов содержат N+1 число с плавающей точкой, из которых первые N чисел – это коэффициенты соответствующей строки матрицы, а последнее – свободный член этой строки. В бинарном формате каждым элементом является 4-х байтное вещественное число, разделителей нет. В текстовом формате разделителем между элементами является символ пробела (с кодом 0x20), а в качестве разделителя между группами элементов используется последовательность символов с кодами 0x0D, 0x0A (перевод строки, возврат каретки).На каждую работу студенту выдается рекомендуемый индивидуальный вариант разбиения матрицы коэффициентов СЛАУ между ветвями параллельной программы.

Лабораторная работа 3

Распараллеливание вычислений при численном решении аналитических задач.

Цель работы: –изучить методы и алгоритмы разработки параллельных программ для решения задач численного дифференцирования, интегрирования, моделирования.

В лабораторной работе нужно разработать параллельную программу для решения задач:

1. Численное дифференцирование.
2. Численное интегрирование.
3. Численное моделирование.

Методы и алгоритмы для решения выбирается студентом самостоятельно.

Критерии оценки

Вид контроля	Минимальное количество баллов	Максимальное количество баллов
Лабораторная работа 1	6	10
Лабораторная работа 2	6	10
Лабораторная работа 3	6	10
Итого	**18**	**30**

Министерство науки и высшего образования Российской Федерации
Нижнекамский химико-технологический институт (филиал)
федерального государственного бюджетного образовательного учреждения
высшего образования
«Казанский национальный исследовательский технологический университет»

Факультет информационных технологий
Кафедра информационных систем и технологий
Направление подготовки 09.04.01 «Информатика и вычислительная техника»

Комплект заданий для выполнения расчетно-графическихработ
по дисциплине «Архитектура параллельных вычислительных систем»

Расчетно-графическая работа 1.

Реализовать на языке высоко уровня:

1. Гибридную архитектуруNUMA.
2. PVP- архитектуру.

Расчетно-графическая работа 2.

Организовать и выполнить расчет показателей

1. Ассоциативного процессора.
2. Конвейерного процессора.
3. Матричного процессора.

Критерии оценки

Вид контроля	Минимальное количество баллов	Максимальное количество баллов
Расчетно-графическая работа 1	9	15
Расчетно-графическая работа 2	9	15
Итого	**18**	**30**

Министерство науки и высшего образования Российской Федерации
Нижнекамский химико-технологический институт (филиал)
федерального государственного бюджетного образовательного учреждения
высшего образования
«Казанский национальный исследовательский технологический университет»

Факультет информационных технологий
Кафедра информационных систем и технологий
Направление подготовки 09.04.01 «Информатика и вычислительная техника»

Комплект экзаменационных тестовых заданий по дисциплине «Архитектура параллельных вычислительных систем»

Вариант 1.

1. Чем характеризуется многопроцессность (concurrency) в контексте параллельных вычислений?
обеспечение минимального времени выполнения одной программы
первичность пропускной способности
не требуется обеспечение максимальной изоляции процессов друг от друга
обеспечение как можно более равномерного распределения ресурсов между процессами

2. В каких ситуациях может быть реализован истинный параллелизм вычислений?
вычисления производятся на ЭВМ с одноядерным процессором в многозадачной ОС
вычисления производятся на ЭВМ с одноядерным процессором в однозадачной ОС
вычисления производятся на многопроцессорном устройстве
для вычислений применяется процессор, поддерживающий физическую векторизацию

3. Какой из режимов вычислений поддерживает классический последовательный компьютер фон Неймана?
обработка нескольких инструкций и одиночного элемента данных в каждый момент времени
обработка одиночной инструкции и нескольких потоков данных в каждый момент времени
обработка одиночной инструкции и одиночного элемента данных в каждый момент времени
обработка нескольких инструкций и нескольких потоков данных в каждый момент времени

4. Симметричный мультипроцессор характеризуется
однородным доступом к памяти всех процессорных устройств
доступом каждого процессорного устройства к отдельной части физически распределенной памяти
неоднородным доступом к памяти всех процессорных устройств
однородным доступом части процессорных устройств к части общей памяти

5. Какие факторы препятствуют получению результата с ожидаемой точностью при распараллеливании арифметических расчетов?
машинная операция сложения чисел не обладает свойством коммутативности (порядок сложения двух чисел важен)
распараллеленные алгоритмы реализуются на одноядерном процессоре
машинная операция умножения чисел не обладает свойством ассоциативности (важен порядок перемножения трёх чисел)
машинная операция умножения чисел обладает свойством коммутативности (порядок перемножения двух чисел не важен)

6. Какие из предложенных стратегий распараллеливания алгоритма нахождения среднего арифметического последовательности из 1000 чисел корректны?
последовательность разбивается на 4 части, элементы в каждой части суммируются и делятся на 1000 на отдельном процессорном устройстве, полученные значения складываются на одном процессорном устройстве и делятся на 4
последовательность разбивается на 4 равные части, элементы в каждой части суммируются, полученные значения складываются и делятся на 1000 на одном процессорном устройстве
последовательность разбивается на 4 равные части, элементы в каждой части суммируются и делятся на 1000 на отдельном процессорном устройстве, полученные значения складываются на одном процессорном устройстве

последовательность разбивается на 4 равные части, находится среднее арифметическое каждой части на отдельном процессорном устройстве, полученные значения складываются на одном процессорном устройстве и делятся на 1000

7. Выберите условия реализуемости расписания параллельного алгоритма.
на каждой вычислительной единице все операции выполняются одна за другой
количество операций, выполняющихся на каждом вычислительном устройстве, постоянно
вычислительные устройства, выполняющие разные операции, не могут обмениваться информацией между собой
каждая операция выполняется не более чем на одном вычислительном устройстве

8. Что называется длиной критического пути в графе, представляющем некоторый параллельный алгоритм?
длина максимального пути в графе
максимальная длина пути в графе, состоящего из однотипных операций
средняя высота графа алгоритма

диаметр графа алгоритма, определяющий минимально теоретически-возможное время выполнение алгоритма

9. Чем характеризуется ускорение параллельного алгоритма?
минимальное время выполнения последовательного алгоритма
отношение количества процессоров к количеству потоков исполнения
минимальное время выполнения параллельного алгоритма
размер входных данных

10. Что такое эффективность параллельного алгоритма?
отношение ускорения алгоритма к количеству процессоров
произведение минимального времени выполнения параллельного алгоритма и количества процессоров
отношение размера входных данных к размеру выходных данных
суммарное время, затрачиваемое всеми процессорами на выполнение алгоритма

11. Выберите верные утверждения.
ускорение программы с помощью параллельных вычислений зависит только от количества вычислительных узлов
суммарное время выполнения параллельной задачи не меньше времени выполнения самого длинного последовательного фрагмента
в реальных задачах добавление новых процессоров может увеличивать время расчета
в реальных задачах ускорения программы с помощью параллельных вычислений нельзя добиться добавлением вычислительных узлов

12. Какие части программы являются последовательными?
чтение входных данных с жесткого диска
запись выходных данных на несколько жестких дисков
синхронизация в параллельной программе
критическая секция в параллельной программе

13.Найдите согласно закону Густавсона ускорение масштабирования некоторой параллельной программы, если известно, что время последовательной части программы равно 1000 мс, время части программы, которая может быть распаралелена, равно 100 мс, количество процессоров равно 10. Ответ округлите до десятых.

14. Найдите согласно закону Густавсона ускорение масштабирования некоторой параллельной программы, если известно, что время последовательной части программы равно 900 мс, время части программы, которая может быть распаралелена, равно 300 мс, количество процессоров равно 9. Ответ округлите до десятых.

14. Найдите согласно закону Густавсона ускорение масштабирования некоторой параллельной программы, если известно, что время последовательной части программы равно 800 мс, время части программы, которая может быть распаралелена, равно 100 мс, количество процессоров равно 10. Ответ округлите до десятых.

15. Каковы принципы организации распределенной памяти с единым адресным пространством в мультипроцессорной системе?
все адресное пространство делится на области, соответствующие отдельным процессорам. Однако если один процессор обратился по адресу памяти другого процессора, то обмен производится с помощью специальной процедуры ОС
единое адресное пространство означает одинаковую для всех процессоров адресацию распределенной памяти

единое адресное пространство делится на области; каждая область отображает адресное пространство соответствующего процессора. Если один процессор использует адрес памяти другого процессора, то автоматически инициируется обмен, который нет необходимости предусматривать программно

16. Проследите использование базовых регистров в иерархической (стековой) структуре программы при заданном порядке вложенности процедур. Сколько базовых регистров используется при счете? Каков максимальный лексикографический уровень?

произойдет прерывание из-за недопустимости подобной структуры

произойдет прерывание по зацикливанию

базовый регистр B1 «смотрит» на область активации процедуры A. После повторного запуска рекурсивной процедуры A на ее новую область активации будет «смотреть» базовый регистр B2 и т.д. Последний, п-й, запуск процедуры A должен довершить ее выполнение. То есть, в ней запустится процедура C, на область активации которой будет «смотреть» тот же базовый регистр Bn. Далее продолжится выполнение процедуры A предыдущего запуска, на область активации которой «смотрит» базовый регистр Bn-1. В ней запустится процедура C, на область активации которой будет «смотреть» тот же базовый регистр и т.д

17. Для данного арифметического выражения составьте программу в безадресной системе команд и для автоматического распараллеливания переведите ее в трехадресную систему команд. Длина списка свободных регистров равна 6. A=(a+b)ЧсЧ(d+e). Какова длина программы в трехадресных командах? Приведите текст седьмой команды

8 команд,+ r6 r1 r2

9 команд,Сч e r1

10 команд,Сч d r6

19. Сколько и в каких комбинациях фигурируют потоки команд и потоки данных при классификации архитектур ВС?

используются все 4 возможные комбинации: SISD, характеризующая микропроцессоры INTEL; SIMD, характеризующая матричные ВС; MISD, характеризующая векторные и векторно-конвейерные системы; MIMD, характеризующая мультимикропроцессорные системы

используются все 4 возможные комбинации: ОКОД, характеризующая традиционные «скалярные» процессоры; ОКМД, характеризующая векторные, матричные и другие процессоры с многими исполнительными устройствами; МКОД, характеризующая векторно-конвейерный способ выполнения операций или конвейерный способ выполнения программ; МКМД, характеризующая многопроцессорные ВС

используются 4 возможные комбинации: ОКОД (SISD), характеризующая «скалярные» процессоры; ОКМД (SIMD), характеризующая векторные, матричные и векторно-конвейерные ВС; МКОД (MISD), характеризующая потоковую (data flow) архитектуру; МКМД (MIMD), характеризующая многопроцессорные ВС на общей оперативной памяти

20. Почему схема data flow относится к «не-фон-Неймановским» архитектурам?

потому что Фон Нейман ее не разрабатывал

потому что одно «узкое место» — счетчик команд заменено другим «узким местом» — коммуникационной сетью ВС, где проблему достижения высокой производительности обмена удается эффективно решить

из-за отсутствия счетчика команд, характеризующего ранние «классические» модели ЭВМ

21. Произведите обоснование предпочтительной формы представления алгоритма для оптимизации программы ВС, управляемой в каждом такте.Каким рекомендациям необходимо следовать при обработке массива?

не следует пользоваться алгоритмами обработки элементов массива, если они не распараллеливаются предусмотрев одновременную обработку более одного элемента массива, следует организовать конвейер этой обработки

необходимо предусмотреть одновременную обработку более одного элемента массива

22. Два процессора коммутации одновременно начинают выполнять программы в виртуальных адресах решающего поля. Составьте план программы их совместного выполнения по тактам, представив, как адресный генератор предлагает им физические адреса буферных регистров

1Чabv12+v1v3v23Чv2ev31+dfv12:v1Lv23Чv2kv3

1Чab(1,1)2+(1,1)(3,1)(4,1)3Ч(4,1)e(3,1)1+df(2,1)2:(2,1)L(1,2)3Ч(1,2)k(2,2)

1Чab(1,1)2+(1,1)(3,1)(4,1)3Ч(4,1)e(1,2)1+df(2,1)2:(2,1)L(1,2)3Ч(1,2)k(2,2)

1Чab(1,1)2+(1,1)(3,1)(3,1)3Ч(4,1)e(3,1)1+df(2,1)2:(4,1)L(1,2)3Ч(1,2)k(2,2)

23. С помощью пятиадресной команды if-then-else составьте программу коммутации для счета значения выражения:
X = bЧif (d+ c) >a then if e >b then A else B else 0
1+dc(1,1)2УСЛeb(2,1)
AB
3УСЛ(1,1)a(3,1)
(2,1)0
4Чb(3,1)(4,1)
1УСЛeb(1,1)
AB
2+dc(2,1)3УСЛ(2,1)a(3,1)
(2,1)0
4Чb(3,1)(4,1)
1+dc(1,1)2УСЛ(1,1)a(3,1)
(2,1)0
3УСЛeb(2,1)
AB
4Чb(3,1)(4,1)

24. Что понимается под параллельными вычислениями?

25. Схемы многопроцессорных систем с однородным и неоднородным доступом.

26. Рассмотрим задачу перемножения матриц. Пусть размер перемножаемой матрицы 200x200. На вычислительной системе все операции сложения и умножения выполняются одинаковое время $\tau = 2$ нсек. Латентности сети $\alpha = 500$ нсек. Пропускная способность сети $\beta = 50$ Мбайт/сек. Элементы матрицы имеют тип double и занимают w = 8 байт. Если при распараллеливании использовать алгоритм Фокса, чему будет равна теоретическая эффективность при использовании 4 процессоров:
0,55
0,77
0,98

27. Рассмотрим задачу поиска решения системы линейных уравнений. Пусть размер матрицы системы линейных уравнений 100x100. На вычислительной системе все операции сложения и умножения выполняются одинаковое время $\tau = 2$ нсек. Латентности сети $\alpha = 5$ нсек. Пропускная способность сети $\beta = 500$ Мбайт/сек. Элементы матрицы системы линейных уравнений имеют тип double и занимают w = 8 байт. Если при распараллеливании алгоритма сопряженных градиентов использовалось 4 процессора, то какое в этом случае достигается теоретическое ускорение:
3,1
1,5
0,3

28. Для разбиения графа на k частей в методе бинарного деления для решения задачи оптимального разделения графов необходимо выполнить:
k-1 деление графа пополам
k/2 делений графа пополам

29. log2k делений графа пополам
Топология полный граф сети кластерной вычислительной системы может иметь ограничения на:
одновременность выполнения коммуникационных операций
максимальный размер сообщений, отсылаемых по сети
количество процессоров в сети

30. Функция блокирующего ожидания завершения одного обмена в MPI называется:
MPI_Wait
MPI_Waitall
MPI_Waitone

31. Три схемы распараллеливания алгоритма быстрой сортировки различаются:
стратегией выбора ведущего элемента
способом рассылки ведущего элемента остальным процессорам

стратегией выбора ведущего процесса

32. Для определения угла поворота в рекурсивном инерционном методе деления пополам при решении задачи оптимального разделения графов, используется:
главная инерционная ось сети
кривые Пеано, построенные на основании сети
центр масс элементов сети

33. Под коллективными операциями в MPI понимаются:
операции передачи данными, в которых принимают участие все процессы используемого коммуникатора
операции над группами процессов
операции над коммуникаторами

34. В результате выполнения одной итерации параллельного алгоритма быстрой сортировки исходное множество процессоров разделяется на:
два подмножества процессоров по p/2 процессоров в каждом и, таким образом, исходный N-мерный гиперкуб также оказывается разделенным на два гиперкуба размерности N-1
N подмножеств процессоров и, таким образом, исходный N-мерный гиперкуб также оказывается разделенным на N гиперкубов меньшей размерности
два подмножества процессоров по p/2 процессоров в каждом и, таким образом, исходный N-мерный гиперкуб также оказывается разделенным на два гиперкуба размерности N/2

35. Комбинаторные методы решения задачи оптимального разделения графов обычно обеспечивают:
более сбалансированное разбиение и меньшее информационное взаимодействие полученных подсетей
менее сбалансированное разбиение и большее информационное взаимодействие полученных подсетей
более быстрое решение поставленной задачи

36. Операция широковещательной рассылки данных это:
операция рассылки значений ведущим процессом всем остальным процессам, все процессы получают рассылаемые данные целиком
операция рассылки значений ведущим процессом всем остальным процессам, все процессы получают часть исходных данных
операция рассылки различающихся значений ведущим процессом всем остальным процессам

37. Для поддержки упорядоченности в ходе выполнения алгоритма обобщенной быстрой сортировки процессоры должны выполнять:
операцию слияния частей блоков, получаемых после обмена данных между процессорами
операцию разделения частей блоков, получаемых после слияния
операцию выбора среднего элемента после слияния частей блоков

38. Протяженность производного типа в MPI это:
размер памяти в байтах, который нужно отводить для одного элемента рассматриваемого типа
число байтов, которые занимает один элемент данных рассматриваемого типа
смещение первого байта значений рассматриваемого типа

39. H-векторный и H-индексный способы создания данных отличаются от векторного и индексного способов тем, что:
интервалы между блоками задаются в байтах, а не в элементах исходного типа данных
разрешают использовать последовательность элементов исходного типа, между которыми могут быть одинаковые промежутки памяти
разрешают использовать разные промежутки памяти между блоками

40. Топология типа тор в MPI является частным видом топологии типа:
декартовой топологии
графа произвольного вида
полный граф

Вариант 2.

1. Что представляет собой task parallelism?
комбинация исходных вычислений в более крупную комплексную задачу
декомпозиция рекурсивной процедурой исходной задачи на несколько более мелких
декомпозиция линейной процедурой исходной задачи на несколько более мелких слияние однотипных задач в более крупную с помощью рекурсивной процедуры

2. Принципы построения вычислительной сети для достижения параллелизма?

3. Что понимается под многозадачным режимом выполнения независимых частей программы?

4. Для каких параллельных алгоритмов обработка данных при распределѐнных вычислениях более эффективна?

5. Что понимается под суперкомпьютером?

6. Как называется группа компьютеров, объединѐнных в локальную сеть и способных работать в качестве единого вычислительного ресурса?

7. Что такое "Beowulf"?

8. Типы систем по классификации Флинна.

9. Каждый процессор параллельной системы может использовать только свою локальную память, в то время как для доступа к данным, располагаемым на других процессорах, необходимо явно выполнить операции передачи сообщений. О какой системе идѐт речь?

10. Что понимается под топологией сети передачи данных?

11. Частным случаем какой топологии является топология "гиперкуб" ?

12. В каком случае два процессора имеют прямое соединение между собой в топологии "гиперкуб"?

13. Что означает понятие «стоимость»?

14. Что означает понятие «диаметр»?

15. Что означает понятие «связность»?

16. Что представляет собой декомпозиция задачи с помощью парадигмы "разделяй и властвуй"?
рекурсивное разбиение задачи на более мелкие того же типа, вплоть до элементарных
разбиение задачи с помощью линейной процедуры
слияние однотипных задач в более крупную с помощью рекурсивной процедуры
бесконечное рекурсивное разбиение задачи на более мелкие того же типа

17. Какой тип вычислительных задач называется embarrassingly parallel?
задачи, к которым не применима парадигма "разделяй и властвуй"
задачи с большим количеством внутренних связей
любые задачи, вычисление которых может быть реализовано с помощью параллельного алгоритма
задачи с большим количеством вычислительных подзадач, не имеющих зависимостей между собой

18. В каких случаях для вычислений применяется конвейерная обработка?
данные поступают непрерывным односторонним регулярным потоком
данные поступают в виде нестабильного двустороннего потока
над каждым элементом из набора данных необходимо произвести обработку в несколько стадий
данные принимаются однажды и требуют индивидуального подхода к обработке и анализу

19. В каких случаях для вычислений применяется координация на основе событий?
данные поступают в виде регулярного одностороннего потока
применяемая структура данных характеризуется непредсказуемостью взаимодействия между своими составными частями
над каждым элементом из набора данных необходимо произвести обработку в несколько стадий
применяется двусторонний поток данных

20. Выберите методы решения с локальными взаимодействиями между подзадачами.
метод Гаусса-Зейделя
метод Red-Black
редукция методом "разделяй и властвуй"
метод хаотической релаксации

21. Что означает тот факт, что соотношение между временами вычислений и синхронизации приближается к единице в некоторой вычислительной системе?
вычисления в этой системе недостаточно эффективны
требуется увеличение подзадач
требуется дополнительное разделение подзадач на более мелкие
вычисления производятся в оптимальной форме и не требуют дополнительных преобразований

22. В каких случаях для вычислений применяется динамическое планирование с балансировкой нагрузки?
число подзадач намного больше числа процессоров
количество процессоров в вычислительной системе меняется во времени
подзадачи сильно различаются по размерам
требуется равномерная загрузка процессоров, но вычислительная система содержит разнородные процессоры

23. В каких случаях для вычислений применяется статическое планирование с балансировкой нагрузки?
число подзадач намного больше числа процессоров
количество процессоров в вычислительной системе меняется во времени
между подзадачами возникают неструктурированные взаимодействия
требуется равномерная загрузка процессоров, но вычислительная система содержит разнородные процессоры

24. Произведите обоснование предпочтительной формы представления алгоритма для оптимизации программы ВС, управляемой в каждом такте. Представьте предпочтительный ряд рабочих критериев, по которым производится включение «готовых» команд в формируемое «длинное» командное слово:
включение команды в состав «длинного» командного слова определяется предпочтительным рядом критериев: максимум времени выполнения, минимум максимального времени начала выполнения, максимум объема последующих работ
полное отсутствие подобных критериев (назначение в порядке следования при анализе потока команд) сокращает время трансляции, не приводя на практике к значительному снижению качества программы
в большинстве случаев достаточно пользоваться критерием назначения по максимальному времени выполнения операций
включение команды в состав «длинного» командного слова определяется предпочтительным рядом критериев: минимум максимального времени начала выполнения, максимум времени выполнения, максимум объема последующих работ

25. Задан трехмерный массив A[0:10; 0:10; 0:10]. Адрес начала равен 10 (в десятичной системе счисления). Найдите адрес элемента a[4, 3, 4].
444
492
531

26. Увеличение вершин:
отношение количества вершин в логической и физической топологиях
максимальное количество вершин физической топологии, на которые отображаемая вершина логической топологии
максимальное количество вершин логической топологии, которые отображаются на одну вершину физической топологии

27. В статической схеме передачи данных:
моменты и участники информационного взаимодействия фиксируются на этапах проектирования и разработки параллельных программ
структура операции передачи данных определяется в ходе выполняемых вычислений
взаимодействия могут быть, как определенны на этапе проектирования, так и определяться в ходе выполнения вычислений

28. Прием сообщений при помощи функции MPI_Recv может быть осуществлен:
от любого адресата и с любым тегом при указании специальных значений в качестве параметров вызова функции
от любого адресата, однако, тег сообщения должен быть указан однозначно
от однозначно определяемого адресата с заданным тегом

29. Рассмотрим задачу перемножения матрицы на вектор. Пусть размер перемножаемой матрицы 100x100. На вычислительной системе все операции сложения и умножения выполняются одинаковое время $\tau = 2$ нсек.

Латентности сети $\alpha = 40$ нсек. Пропускная способность сети 60 Мбайт/сек. Элементы матрицы имеют тип double и занимают w = 8 байт. Если при распараллеливании использовать разделение матрицы на строки, чему будет равна теоретическая стоимость при использовании 2 процессоров:
200000000
01313412
221313412

30. Рассмотрим задачу перемножения матриц. Пусть размер перемножаемой матрицы 100x100. На вычислительной системе все операции сложения и умножения выполняются одинаковое время $\tau = 2$ нсек. Латентности сети $\alpha = 500$ нсек. Пропускная способность сети $\beta = 50$ Мбайт/сек. Элементы матрицы имеют тип double и в системе занимают w = 8 байт. Если при распараллеливании использовать алгоритм Кеннона, чему будет равно теоретическое ускорение при использовании 4 процессоров:
2,5
1,5
1,17

31. Рассмотрим задачу поиска решения системы линейных уравнений. Размер матрицы системы линейных уравнений 10x10. На вычислительной системе все операции сложения и умножения выполняются одинаковое время $\tau = 2$ нсек. Латентности сети $\alpha = 50$ нсек. Пропускная способность сети $\beta = 60$ Мбайт/сек. Элементы матрицы системы линейных уравнений имеют тип double и в системе занимают w = 8 байт. Если при распараллеливании алгоритма Гауса использовалось 4 процессора, то какая в этом случае достигается теоретическая стоимость параллельного алгоритма:
1387870
2102751
1453075

32. В худшем случае трудоемкость быстрой сортировки оценивается выражением:
$T \sim n^2$
$T \sim nlog_2 n$
$T \sim n^{1.25}$

33. Какая из приведенных в лекции топологий (при одинаковом количестве процессоров) обладает наименьшей стоимостью:
топология полное двоичное дерево
топология двумерный решетка-тор
топология полный граф

34. Задача разделения вычислительной сети, на которую разбивается область обрабатываемых данных, между процессорами может быть сведена:
к проблеме оптимального разделения графа
к задаче поиска всех кратчайших путей
к задаче нахождения минимального охватывающего дерева

35. Какие достоинства и недостатки имеет асинхронный механизм передачи сообщений?
асинхронные механизмы передачи, как правило, могут передать большие структуры данных быстрее
неблокирующие операции могут позволить совместить процессы передачи данных и вычислений
асинхронные механизмы передачи обычно приводят к повышению сложности программирования

36. Помимо выполнения экспериментов в режиме имитации, в системе ПараЛаб предусмотрена возможность проведения реальных экспериментов в режиме удаленного доступа к вычислительному кластеру. Какие возможны операции после выполнения реальных параллельных вычислений:
сравнить результаты и оценить точность используемых в системе теоретических моделей времени выполнения параллельных алгоритмов
осуществить настройку параметров удаленного кластера
настроить параметры моделей, используемых в системе ПараЛаб

37. Пусть в решаемой задаче последовательная часть составляет четыре единицы времени, а часть, допускающая линейное распараллеливание, шесть единицы времени. Если использовать закон Амдаля, какая достигается эффективность, если используются три вычислительных элемента:
1/5
7/6

38. Соседние вершины в кольцевой топологии отображаются кодом Грея:
на соседние процессоры в гиперкубе
процессоры, находящиеся на расстоянии не более чем 2 единицы в гиперкубе
на соседние процессоры в торе

39. Этап распределения подзадач между процессорами является избыточным, если:
количество подзадач совпадает с числом имеющихся процессоров, а топология сети передачи данных представляет собой полный граф
количество подзадач больше числа имеющихся процессоров, а топология сети передачи данных представляет собой полный граф
количество подзадач совпадает с числом имеющихся процессоров, а топология сети передачи данных представляет собой гиперкуб

40. Режим передачи по готовности может быть использован только если:
операция приема сообщения уже инициирована
при достаточном малом размере сообщения, менее размера системного буфера
операция приема сообщения гарантированно будет запущена позднее момента начала передачи сообщения

Критерии оценки

Вид контроля	Минимальное количество баллов	Максимальное количество баллов
Экзамен	24	40