Федеральное агентство связи (Россвязь)

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Сибирский государственный университет телекоммуникаций и информатики»

(СибГУТИ)

09.03.01 Информатика и вычислительная техника

Профиль: Программное обеспечение средств вычислительной техники и автоматизированных систем

**РАСЧЁТНО-ГРАФИЧЕСКОЕ ЗАДАНИЕ**

по дисциплине «Архитектура вычислительных систем»

Вариант № 24

Выполнил:

студент гр. ИП-611 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /Макаревич А.А./

подпись

Проверил:

доцент кафедры ВС

к.т.н. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /А.В. Ефимов /

ОЦЕНКА, подпись

Новосибирск, 2018

Содержание

[1. Задание 1 3](#_Toc533860925)

[1.1 Ответ на задание 1 3](#_Toc533860926)

[2. Задание 2 14](#_Toc533860927)

[2.1 Ответ на задание 2 14](#_Toc533860928)

[3. Задание 3 19](#_Toc533860929)

[3.1 Ответ на задание 3 19](#_Toc533860930)

[Список использованных источников 23](#_Toc533860931)

# 1. Задание 1

Осуществить анализ архитектуры мультипроцессорных вычислительных систем. Привести пример функциональной структуры суперВС (из списка Top500).

## 1.1 Ответ на задание 1

**Назначение и область применения архитектур мультипроцессорных вычислительных систем**

В настоящее время сфера применения многопроцессорных вычислительных систем (МВС) непрерывно расширяется, охватывая все новые области в различных отраслях науки, бизнеса и производства. Стремительное развитие кластерных систем создает условия для использования многопроцессорной вычислительной техники в реальном секторе экономики.

Если традиционно МВС применялись в основном в научной сфере для решения вычислительных задач, требующих мощных вычислительных ресурсов, то сейчас из-за бурного развития бизнеса резко возросло количество компаний, отводящих использованию компьютерных технологий и электронного документооборота главную роль. В связи с этим непрерывно растет потребность в построении централизованных вычислительных систем для критически важных приложений, связанных с обработкой транзакций, управлением базами данных и обслуживанием телекоммуникаций.

Можно выделить две основные сферы применения описываемых систем:

- обработка транзакций в режиме реального времени (OLTP, on-line transaction processing);

- создание хранилищ данных для организации систем поддержки принятия решений (Data Mining, Data Warehousing, Decision Support System).

Система для глобальных корпоративных вычислений - это, прежде всего, централизованная система, с которой работают практически все пользователи в корпорации, и, соответственно, она должна все время находиться в рабочем состоянии. Как правило, решения подобного уровня устанавливают в компаниях и корпорациях, где даже кратковременные простои сети могут привести к громадным убыткам. Поэтому для организации такой системы не подойдет обыкновенный сервер со стандартной архитектурой, вполне пригодный там, где нет жестких требований к производительности и времени простоя. Высокопроизводительные системы для глобальных корпоративных вычислений должны отличаться такими характеристиками, как повышенная производительность, масштабируемость, минимально допустимое время простоя.

Наряду с расширением области применения по мере совершенствования МВС происходит усложнение и увеличение количества задач в областях, традиционно использующих высокопроизводительную вычислительную технику. В настоящее время выделен круг фундаментальных и прикладных проблем, эффективное решение которых возможно только с использованием сверхмощных вычислительных ресурсов. Этот круг, обозначаемый понятием *«Grand challenges»*, включает следующие задачи:

* предсказания погоды, климата и глобальных изменений в атмосфере;
* науки о материалах;
* построение полупроводниковых приборов;
* сверхпроводимость;
* структурная биология;
* разработка фармацевтических препаратов;
* генетика;
* квантовая хромодинамика;
* астрономия;
* транспортные задачи;
* гидро- и газодинамика;
* управляемый термоядерный синтез;
* эффективность систем сгорания топлива;
* геоинформационные системы;
* разведка недр;
* наука о мировом океане;
* распознавание и синтез речи;
* распознавание изображений.

Многопроцессорные вычислительные системы могут существовать в различных конфигурациях. Наиболее распространенными типами МВС являются:

* системы высокой надежности;
* системы для высокопроизводительных вычислений;
* многопоточные системы.

МВС являются идеальной схемой для повышения надежности информационно-вычислительной системы. Благодаря единому представлению отдельные узлы или компоненты МВС могут незаметно для пользователя заменять неисправные элементы, обеспечивая непрерывность и безотказную работу даже таких сложных приложений, как базы данных.

МВС для высокопроизводительных вычислений предназначены для параллельных расчетов. Имеется много примеров научных расчетов, выполненных на основе параллельной работы нескольких недорогих процессоров, обеспечивающих одновременное проведение большого числа операций.

МВС для высокопроизводительных вычислений обычно собраны из многих компьютеров. Разработка таких систем – процесс сложный, требующий постоянного согласования таких вопросов, как инсталляция, эксплуатация и одновременное управление большим числом компьютеров, технических требований параллельного и высокопроизводительного доступа к одному и тому же системному файлу (или файлам), межпроцессорной связи между узлами и координации работы в параллельном режиме. Эти проблемы проще всего решаются при обеспечении единого образа операционной системы для всего кластера. Однако реализовать подобную схему удается далеко не всегда, и обычно она применяется лишь для небольших систем.

Многопоточные системы используются для обеспечения единого интерфейса к ряду ресурсов, которые могут со временем произвольно наращиваться (или сокращаться). Типичным примером может служить группа web-серверов.

Главной отличительной особенностью многопроцессорной вычислительной системы является ее производительность, т.е. количество операций, производимых системой за единицу времени.

Чтобы дать более полное представление о многопроцессорных вычислительных системах, помимо высокой производительности необходимо назвать и другие отличительные особенности. Прежде всего, это необычные архитектурные решения, направленные на повышение производительности (работа с векторными операциями, организация быстрого обмена сообщениями между процессорами или организация глобальной памяти в многопроцессорных системах и др.).

**Программные реализации**

**Многопроцессорная обработка с SISD**

SISD (single instruction stream / single data stream) – одиночный поток команд и одиночный поток данных. К этому классу относятся последовательные компьютерные системы, которые имеют один центральный процессор, способный обрабатывать только один поток последовательно исполняемых инструкций. В настоящее время практически все высокопроизводительные системы имеют более одного центрального процессора, однако каждый из них выполняет несвязанные потоки инструкций, что делает такие системы комплексами SISD-систем, действующих на разных пространствах данных. Для увеличения скорости обработки команд и скорости выполнения арифметических операций может применяться конвейерная обработка. В случае векторных систем векторный поток данных следует рассматривать как поток из одиночных неделимых векторов. Примерами компьютеров с архитектурой SISD могут служить большинство рабочих станций Compaq, Hewlett-Packard и Sun Microsystems.

**Многопроцессорная обработка SIMD**

В компьютере с одиночным потоком команд и множественным потоком данных один процессор обрабатывает поток команд, каждая из которых может выполнить параллельные вычисления на множестве данных.

Многопроцессорная обработка SIMD хорошо подходит для параллельной или векторной обработки, в которой большой набор данных может быть разделён на части, которые обрабатываются идентичными, но независимыми операциями. Одиночный поток команд направляет операцию модулей мультипрограммирования для выполнения однотипных манипуляций одновременно на потенциально большом количестве данных.

Для определённых типов вычислительных приложений этот тип архитектуры может дать значительное увеличение производительности с точки зрения затраченного времени. Однако, недостаток этой архитектуры состоит в том, что большая часть системы начинает простаивать при выполнении программ или системных задач, которые не могут быть разделены на модули (подзадачи), которые могут быть обработаны параллельно.

Кроме того, программы должны быть тщательно и специально написаны, чтобы иметь возможность максимально задействовать возможности архитектуры. Часто применяются специальные оптимизирующие компиляторы, спроектированные чтобы создать код специально для использования в этой среде. Некоторые [компиляторы](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D0%B8%D0%BB%D1%8F%D1%82%D0%BE%D1%80) в этой категории обеспечивают специальные конструкции или расширения, чтобы позволить программистам непосредственно определять операции, которые будут выполнены параллельно (например, операторы [DO FOR ALL](https://ru.wikipedia.org/w/index.php?title=DO_FOR_ALL&action=edit&redlink=1) в [Фортране](https://ru.wikipedia.org/wiki/%D0%A4%D0%BE%D1%80%D1%82%D1%80%D0%B0%D0%BD), используемого на [ILLIAC IV](https://ru.wikipedia.org/w/index.php?title=ILLIAC_IV&action=edit&redlink=1), который был многопроцессорным суперкомпьютером с SIMD-архитектурой).

Многопроцессорная обработка SIMD находит широкое применение в некоторых областях, таких как [компьютерное моделирование](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D0%BE%D0%B5_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5), но малополезна в универсальном настольном компьютере и бизнес-задачах.

**Многопроцессорная обработка MISD**

Многопроцессорная обработка с множественным потоком команд и одиночным потоком данных предлагает главным образом преимущество избыточности, так как модули мультипрограммирования выполняют одни задачи на одних данных, уменьшая возможности неправильных результатов, если один из модулей выходит из строя. Архитектура MISD позволяет сравнивать результаты вычислений в целях обнаружения отказов. Кроме избыточности и отказоустойчивости у этого типа многопроцессорной обработки немного преимуществ. К тому же он весьма дорог. Он не увеличивает производительность.

**Многопроцессорная обработка MIMD**

Архитектура многопроцессорной обработки MIMD является подходящей для большого числа разнообразных задач, в которых реализовано полностью независимое и параллельное выполнение команд, касающихся различных наборов данных. По этой причине и потому что это просто осуществить, MIMD преобладает в многопроцессорной обработке.

Обработка разделена на несколько потоков, каждый с собственным аппаратным состоянием процессора, в рамках единственного определённого программным обеспечением процесса или в пределах множественных процессов. Поскольку система имеет несколько потоков, ожидающих выполнения (системные или пользовательские потоки), эта архитектура эффективно использует аппаратные ресурсы.

В MIMD могут возникнуть проблемы взаимной блокировки и состязания за обладание ресурсами, так как потоки, пытаясь получить доступ к ресурсам, могут столкнуться непредсказуемым способом. MIMD требует специального кодирования в операционной системе компьютера, но не требует изменений в прикладных программах, кроме случаев когда программы сами используют множественные потоки (MIMD прозрачен для однопоточных программ под управлением большинства операционных систем, если программы сами не отказываются от управления со стороны ОС). И системное и пользовательское программное обеспечение, возможно, должны использовать программные конструкции, такие как [семафоры](https://ru.wikipedia.org/wiki/%D0%A1%D0%B5%D0%BC%D0%B0%D1%84%D0%BE%D1%80_(%D0%B8%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)), чтобы препятствовать тому, чтобы один поток вмешался в другой, в случае если они содержат ссылку на одни и те же данные. Такое действие увеличивает сложность кода, снижает производительность и значительно увеличивают количество необходимого тестирования, хотя обычно не настолько чтобы свести на нет преимущества многопроцессорной обработки.

Подобные конфликты могут возникнуть на аппаратном уровне между процессорами, и должен обычно решаться аппаратными средствами, или с комбинацией программного обеспечения и оборудования.

**Основные преимущества и недостатки архитектуры вычислительных систем**

**SMP архитектура**

SMP (symmetric multiprocessing) – симметричная многопроцессорная архитектура. Главной особенностью систем с архитектурой SMP является наличие общей физической памяти, разделяемой всеми процессорами.



Рисунок 1. Схематический вид SMP архитектуры

Память служит, в частности, для передачи сообщений между процессорами, при этом все вычислительные устройства при обращении к ней имеют равные права и одну и ту же адресацию для всех ячеек памяти. Поэтому SMP-архитектура называется симметричной. Последнее обстоятельство позволяет очень эффективно обмениваться данными с другими вычислительными устройствами. SMP-система строится на основе высокоскоростной системной шины (SGI PowerPath, Sun Gigaplane, DEC TurboLaser), к слотам которой подключаются функциональные блоки типов: процессоры (ЦП), подсистема ввода/вывода (I/O) и т.п. Для подсоединения к модулям I/O используются уже более медленные шины (PCI, VME64).

Наиболее известными SMP-системами являются SMP-cерверы и рабочие станции на базе процессоров Intel (IBM, HP, Compaq, Dell, ALR, Unisys, DG, Fujitsu и др.) Вся система работает под управлением единой ОС (обычно UNIX-подобной, но для Intel-платформ поддерживается Windows NT). ОС автоматически (в процессе работы) распределяет процессы по процессорам, но иногда возможна и явная привязка.

**Основные преимущества SMP-систем:**

**-** простота и универсальность для программирования. Архитектура SMP не накладывает ограничений на модель программирования, используемую при создании приложения: обычно используется модель параллельных ветвей, когда все процессоры работают независимо друг от друга. Однако можно реализовать и модели, использующие межпроцессорный обмен. Использование общей памяти увеличивает скорость такого обмена, пользователь также имеет доступ сразу ко всему объему памяти. Для SMP-систем существуют довольно эффективные средства автоматического распараллеливания;

**-** простота эксплуатации. Как правило, SMP-системы используют систему кондиционирования, основанную на воздушном охлаждении, что облегчает их техническое обслуживание;

**-** относительно невысокая цена.

**Недостатки:**

**-** системы с общей памятью плохо масштабируются.

Этот существенный недостаток SMP-систем не позволяет считать их по-настоящему перспективными. Причиной плохой масштабируемости является то, что в данный момент шина способна обрабатывать только одну транзакцию, вследствие чего возникают проблемы разрешения конфликтов при одновременном обращении нескольких процессоров к одним и тем же областям общей физической памяти. Вычислительные элементы начинают друг другу мешать. Когда произойдет такой конфликт, зависит от скорости связи и от количества вычислительных элементов. В настоящее время конфликты могут происходить при наличии 8-24 процессоров. Кроме того, системная шина имеет ограниченную (хоть и высокую) пропускную способность (ПС) и ограниченное число слотов. Все это очевидно препятствует увеличению производительности при увеличении числа процессоров и числа подключаемых пользователей.

В реальных системах можно задействовать не более 32 процессоров. Для построения масштабируемых систем на базе SMP используются кластерные или NUMA-архитектуры. При работе с SMP-системами используют так называемую парадигму программирования с разделяемой памятью (shared memory paradigm).

**MPP архитектура**

MPP (massive parallel processing) – массивно-параллельная архитектура. Главная особенность такой архитектуры состоит в том, что память физически разделена. В этом случае система строится из отдельных модулей, содержащих процессор, локальный банк операционной памяти (ОП), коммуникационные процессоры (рутеры) или сетевые адаптеры, иногда – жесткие диски и/или другие устройства ввода/вывода. По сути, такие модули представляют собой полнофункциональные компьютеры (см. рис.3.2). Доступ к банку ОП из данного модуля имеют только процессоры (ЦП) из этого же модуля. Модули соединяются специальными коммуникационными каналами. Пользователь может определить логический номер процессора, к которому он подключен, и организовать обмен сообщениями с другими процессорами. Используются два варианта работы операционной системы (ОС) на машинах MPP-архитектуры. В одном полноценная операционная система (ОС) работает только на управляющей машине (front-end), на каждом отдельном модуле функционирует сильно урезанный вариант ОС, обеспечивающий работу только расположенной в нем ветви параллельного приложения. Во втором варианте на каждом модуле работает полноценная UNIX-подобная ОС, устанавливаемая отдельно.

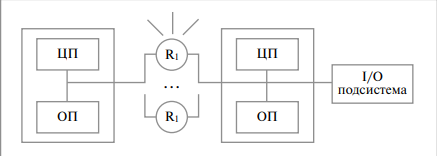


Рисунок 2. Схематический вид архитектуры с раздельной памятью

**Главным преимуществом** систем с раздельной памятью является хорошая масштабируемость: в отличие от SMP-систем, в машинах с раздельной памятью каждый процессор имеет доступ только к своей локальной памяти, в связи с чем не возникает необходимости в потактовой синхронизации процессоров. Практически все рекорды по производительности на сегодня устанавливаются на машинах именно такой архитектуры, состоящих из нескольких тысяч процессоров (ASCI Red, ASCI Blue Pacific).

**Недостатки:**

**-** отсутствие общей памяти заметно снижает скорость межпроцессорного обмена, поскольку нет общей среды для хранения данных, предназначенных для обмена между процессорами. Требуется специальная техника программирования для реализации обмена сообщениями между процессорами;

- каждый процессор может использовать только ограниченный объем локального банка памяти;

- вследствие указанных архитектурных недостатков требуются значительные усилия для того, чтобы максимально использовать системные ресурсы. Именно этим определяется высокая цена программного обеспечения для массивно-параллельных систем с раздельной памятью.

Системами с раздельной памятью являются суперкомпьютеры МВС-1000, IBM RS/6000 SP, SGI/CRAY T3E, системы ASCI, Hitachi SR8000, системы Parsytec. Машины последней серии CRAY T3E от SGI, основанные на базе процессоров Dec Alpha 21164 с пиковой производительностью 1200 Мфлопс/с (CRAY T3E-1200), способны масштабироваться до 2048 процессоров. При работе с MPP-системами используют так называемую Massive Passing Programming Paradigm – парадигму программирования с передачей данных (MPI, PVM, BSPlib).

**Гибридная архитектура NUMA**

Главная особенность гибридной архитектуры NUMA (nonuniform memory access) – неоднородный доступ к памяти. Гибридная архитектура совмещает достоинства систем с общей памятью и относительную дешевизну систем с раздельной памятью. Суть этой архитектуры – в особой организации памяти, а именно: память физически распределена по различным частям системы, но логически она является общей, так что пользователь видит единое адресное пространство. Система построена из однородных базовых модулей (плат), состоящих из небольшого числа процессоров и блока памяти. Модули объединены с помощью высокоскоростного коммутатора. Поддерживается единое адресное пространство, аппаратно поддерживается доступ к удаленной памяти, т.е. к памяти других модулей. При этом доступ к локальной памяти осуществляется в несколько раз быстрее, чем к удаленной. По существу архитектура NUMA является MPP (массивно-параллельной ) архитектурой, где в качестве отдельных вычислительных элементов берутся SMP (cимметричная многопроцессорная архитектура) узлы. Доступ к памяти и обмен данными внутри одного SMP-узла осуществляется через локальную память узла и происходит очень быстро, а к процессорам другого SMP-узла тоже есть доступ, но более медленный и через более сложную систему адресации. Структурная схема компьютера с гибридной сетью: четыре процессора связываются между собой при помощи кроссбара в рамках одного SMP-узла. Узлы связаны сетью типа «бабочка» (Butterfly):

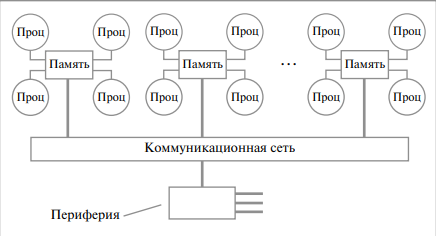


Рисунок 3. Структурная схема компьютера с гибридной связью

Впервые идею гибридной архитектуры предложил Стив Волох, он воплотил ее в системах серии Exemplar. Вариант Воллоха – система, состоящая из восьми SMP-узлов. Фирма HP купила идею и реализовала на суперкомпьютерах серии SPP. Идею подхватил Сеймур Крей (Seymour R.Cray) и добавил новый элемент – когерентный кэш, создав так называемую архитектуру cc-NUMA (Cache Coherent Non-Uniform Memory Access), которая расшифровывается как «неоднородный доступ к памяти с обеспечением когерентности кэшей».

**Выводы**

Анализ архитектуры мультипроцессорных ВС позволяет сделать нижеследующие выводы.

1. Основная тенденция в области архитектуры мультипроцессорных ВС - повышение степени полноты воплощения принципов модели коллектива вычислителей (параллелизма, программируемости структуры и конструктивной однородности).

2. Архитектурные возможности ЭП неуклонно наращиваются, их структура претерпела трансформацию от простейших конфигураций без памяти до элементарных машин - композиции из мощных микропроцессоров, оперативной памяти и внешних запоминающих устройств (и даже устройств ввода-вывода информации).

3. Мультипроцессорные ВС, начавшие свою историю как композиции из нескольких процессоров, превратились в системы c массовым параллелизмом.

4. Современные мультипроцессорные ВС - это распределенные средства обработки информации, они имеют множество процессоров и распределённyю память. Более того, в них и коммутатор (или другой ресурс), через который осуществляется взаимодействие процессоров, может быть распределенным. Программное обеспечение таких ВС также является распределенным.

5. Высокопроизводительные ВС рассматриваемого класса представляют собой суперсистемы: это множество мощных микропроцессоров, конвейеров или матричных процессоров или даже объединение мультипроцессоров.

Эволюционное развитие архитектуры мультипроцессорных ВС привело к «революционной» модернизации первоначальных канонов. Любая современная высокопроизводительнaя ВС в зависимости от функционального уровня рассмотрения может выглядеть как MISD- или SIMD-, или MIMD-система, более того, ее функциональная структура одновременно обладает свойствами конвейерных, матричных и мультипроцессорных систем. Исследователи параллельных вычислительных технологий, архитекторы и создатели промышленных ВС (независимо от их первоначальных архитектурных концепций) пришли к необходимости создания распределённых ВС c программируемой структурой.

**Пример функциональной структуры суперВС.**

Примером функциональной структурой суперВС является IBM Blue Gene/L. Конфигурация модели IBM Вlue Gene/L 2008 г. обладает следующими показателями: число процессорных ядер - 212 992, пиковая производительность - 596,38 TFLOPS (478,2 TFLOPS на UNPACK), емкость памяти 53,248 - Тбайта, пропускная способность канала «процессор-память» - 5,5 Гбайт/c, энергопотребление - около 3,25 Мегаватт, занимаемая площадь - 4060 квадратных футов (377,2 ), цена - 208 млн долл.

Функциональная структура суперВС рассчитана на использование хост-компьютерной системы - множества хост-компьютеров. Эти компьютеры реализуют файловую систему и выполняют следующие функции: анализ функционирования, контроль, диагностику и восстановление суперВС, a также компиляцию и сервисное обслуживание. Выбор хост-компьютеров для IBM Blue Gene определяется областью применения (требуемыми производительностью и полосой пропускания, в частности).

Важной особенностью архитектуры IBM Вlue Gene является возможность одновременной работы множества пользователей. Это достигается путем выделения каждому пользователю требуемых ресурсов подсистемы из необходимого количества вычислительных узлов. Следовательно, в данной суперВС реализована возможность разбиения «пространства» вычислительных узлов на подпространства и доступа к ним пользователей. Выделение подсистем в суперВС выполняется хост-компьютерной системой.

При разработке функциональной структуры суперВС и основных ее элементов значительное внимание уделялось обеспечению надежности (RAS Reliability, Availability, Serviceability - надежности, готовности, обслуживаемости). При этом свою роль сыграли принципы простоты, однородности и модульности, a также введение избыточности, средств контроля, диагностики и восстановления.

Фактически разработчики суперВС ЭВМ Blue Gene реализовали платформу ВС c программируемой структурой, они достаточно полно воплотили и архитектурные принципы, и принципы технической реализации модели коллектива вычислителей.

# 

# 2. Задание 2

Выполнить численный расчет и построить графики для функции u(t) восстановимости и S(i,t) готовности ЭВМ для следующих количественных характеристик:

- среднего времени безотказной работы ,

- интенсивности восстановления .

## 2.1 Ответ на задание 2

**Функция восстановимости u(t)**

Функция восстановимости (или вероятность восстановления работоспособного состояния) ЭВМ - основной показатель, характеризующий надежностные способности и ЭВМ, и восстанавливающего устройства одновременно. Или, говоря иначе, эта количественная характеристика дает информацию о том, как приспособлена машина к восстановлению своей производительности (после отказа) с помощью ВУ. Пусть функция восстановимости ЭВМ будет:

(2.1.1)

Где — вероятность того, что для всякого , принадлежащего промежутку времени , производительность ЭВМ остаётся равной нулю (или вероятность того, что отказавшая ЭВМ при работе восстанавливающего устройства не будет восстановлена за время t).

Для функции справедливо:

Следовательно, u(t) является интегральной функцией распределения времени восстановления отказавшей ЭВМ.

,  (2.1.2)

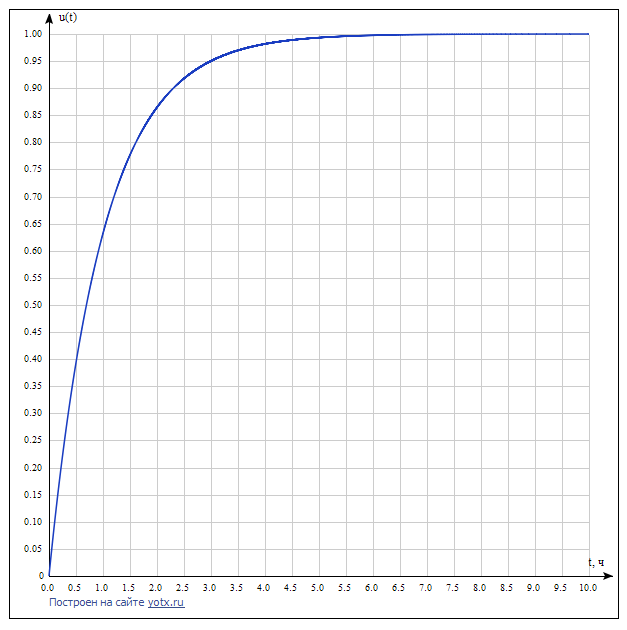
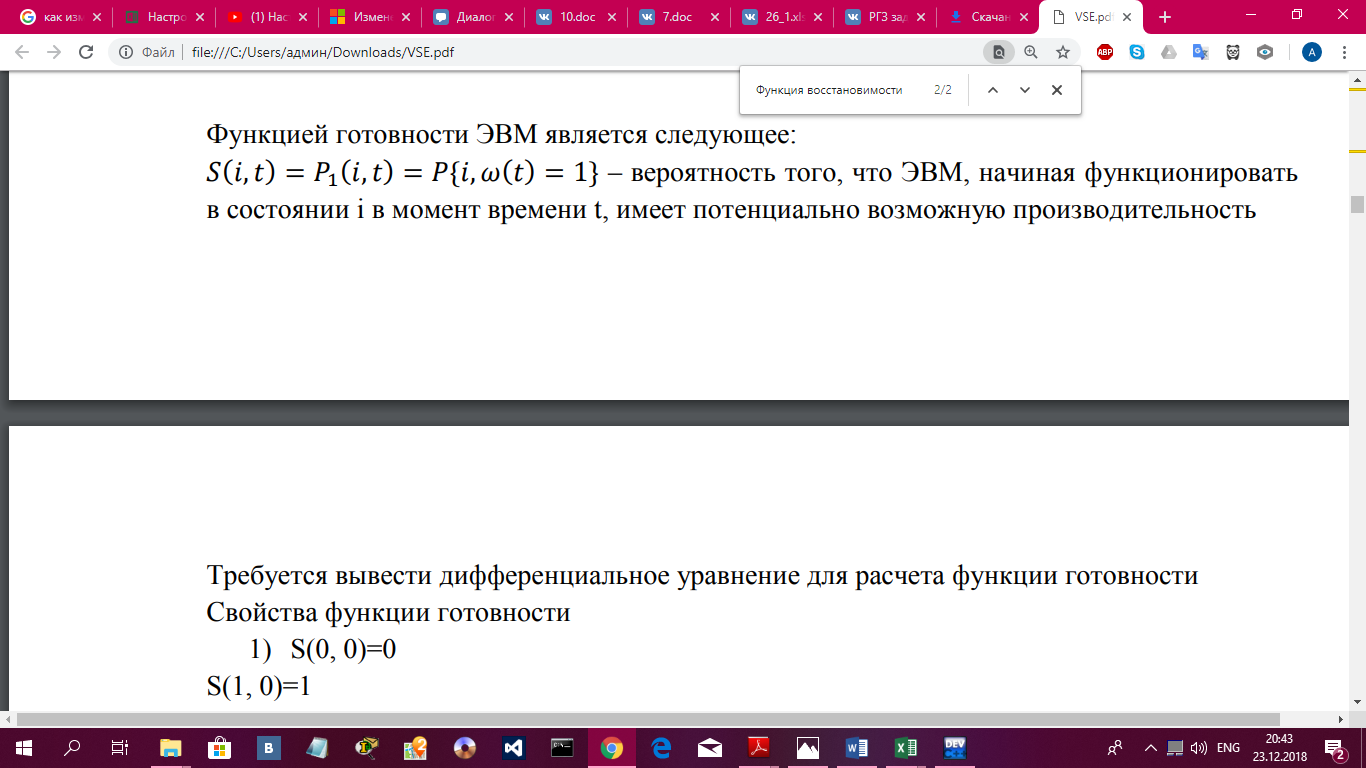


Рисунок 4. График функции восстановимости

 Поведение функции определено на всём интервале допустимых значений, так как функция восстановимости отражает вероятность восстановления работоспособного состояния, и следственно имеет диапазон значений от 0 до 1.

Функцией готовности ЭВМ является следующее:  – вероятность того, что ЭВМ, начиная функционировать в состоянии i в момент времени t, имеет потенциально возможную производительность. Требуется вывести дифференциальное уравнение для расчета функции готовности.

Свойства функции готовности

1) S(0, 0)=0;

2) S(1, 0)=1;

Теперь рассчитаем значения функции готовности. Расчет будем производить по следующим формулам:

 (2.1.3)

, (2.1.4)

для начальных состояний ЭВМ , причем  соответствует состоянию отказа, а  – работоспособному состоянию машины, где .

**Рассчитаем значения функции и построим график:**

(2.1.5)

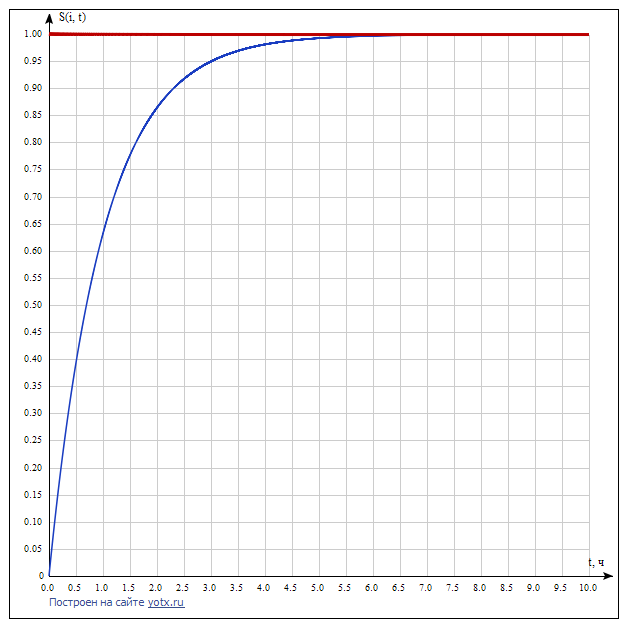
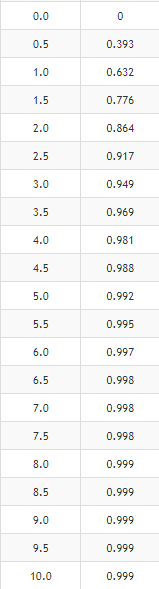


Рисунок 5. График функции готовности

Поведение функции готовности определено на всём интервале допустимых значений при .

# 

# 3. Задание 3

Построить блок-схему p-алгоритма умножения матриц:

Q[1:E;1:H] = P[1:E;1:F] \* R[1:F;1:H],

обеспечивающего распределение в элементарных машинах ВС элементов

результирующей матрицы по горизонтальным полосам.

Отыскать максимум коэффициента накладных расходов при реализации p-алгоритма на вычислительной системе, имеющей следующие параметры:

– разрядность ;

– полосу пропускания канала между машинами ;

– время выполнения операции сложения ;

– время выполнения операции умножения .

## 3.1 Ответ на задание 3

Необходимо разработать блок-схему р-алгоритма для вычисления произведения двух матриц:

Q[1:E; 1:H] = P[1:E;1:F] \* R[1:F;1:H],

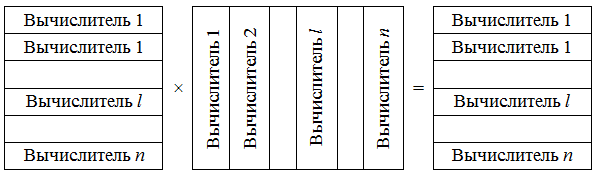
применив распределения в машинах BC матрицы R по горизонтальным полосам.

Элементы матрицы Q вычисляются по следующей формуле:

Количество столбцов F в матрице P равно количеству строк F матрице R.

P-алгоритм должен обеспечить распределение элементов матрицы Q[1: E; 1: H] по горизонтальным полосам в элементарных машинах ВС.

Распределение данных по вычислителям ВС



P \* R = Q

Для построения p-алгоритма прежде всего требуется осуществить распределение исходного массива данных. Осуществим следующие распределения:

Матрицу P разобьем на n равных горизонтальных полос, а матрицу R на n равных вертикальных полос.

α – номер передающего вычислителя, {1, 2, …, α–1, α+1, …, n} – номера принимающих вычислителей.

?

**Да**

**Нет**

**Передача**

**Передача**

**Вычисление**

**Передача**

**Передача**

**Приём**

**Да**

**Нет**

=

**КОНЕЦ**

**Да**

**Нет**

**НАЧАЛО**

**НАЧАЛО**

Рисунок 6. Блок-схема алгоритма

Коэффициент накладных расходов рассчитывается по формуле:

(3.1.1)

где t - время, расходуемое на обмен информацией;

T - время, расходуемое на вычисление логических операций.

В соответствии с алгоритмом при пересылки строки матрицы P, состоящей из E – элементов, в каждом вычислители производится E\*]H/n[ - операций умножения и (E-1)\*]H/n[ - операций сложения.

Так как используется матрицы с большой размерностью, т.е. H>>n,

следовательно, можно считать, что после пересылки одного элемента из любого вычислителя во все остальные вычислители производится ρ=]H/n[ - операций сложения и умножения.

Получаем:

(3.1.2)

В параллельных алгоритмах показатель ρ не может быть меньше единицы, в противном случае нет необходимости распараллеливания.

Итак, максимум коэффициента накладных расходов достигается при .

– время пересылки

– время умножения

- время сложения

(3.1.4)

Ответ: максимум коэффициента накладных расходов равен

# Список использованных источников

1. Хорошевский В.Г. Архитектура вычислительных систем. – М.: МГТУ им. Н.Э. Баумана, 2008. – 520 с.