Министерство образования и науки РФ

Государственное образовательное учреждение высшего профессионального образования (ВолгГТУ)

Кафедра ЭВМиС

Лукьянов В. С., Жариков Д. Н., Гаевой С. В., Королев А. Д., Шафран Ю. В.

Моделирование Grid-систем

Волгоград, 2014 г.

Оглавление

1 Описание Grid-систем.....5
1.1 Основные понятия.....5
1.2 Классификации Grid.....7
1.3 Проблема распределения ресурсов.....12
2 Моделирование Grid-систем 14
2.1 Обзор моделей 14
2.2 GridModel 21
2.3 Реализованные законы распределения случайных величин.....27
2.4 Процесс моделирования в GridModel.....33
2.5 Цепи событий имитационного моделирования.....35
3 Стратегии распределения заданий.....42
3.1 Стратегии выбора оптимального кластера для исполнения.....42
3.2 Способы использования стратегий.....46
3.3 Анализ стратегий распределения заданий.....48
4 Модель надёжности 52
5 Результат разработки.....63
6 Лабораторная работа №1.....66
6.1 Тема работы.....66
6.2 Цель работы.....66
6.3 Рекомендации 66
6.4 Краткое описание работы.....67
6.5 Практическая часть.....68
6.6 Содержание отчёта.....71
6.7 Контрольные вопросы.....76
7 Лабораторная работа №2.....77
7.1 Тема работы.....77
7.2 Цель работы.....77
7.3 Рекомендации 77

7.4 Краткое описание работы....78
7.5 Практическая часть....79
7.6 Содержание отчёта....81
7.7 Контрольные вопросы....87
8 Лабораторная работа №3....88
8.1 Тема работы....88
8.2 Цель работы....88
8.3 Рекомендации 88
8.4 Краткое описание работы....89
8.5 Практическая часть....90
8.6 Содержание отчёта....94
8.7 Контрольные вопросы....95
9 Лабораторная работа №4*....96
9.1 Тема работы....96
9.2 Цель работы....96
9.3 Рекомендации 96
9.4 Краткое описание работы....97
9.5 Практическая часть....101
9.6 Содержание отчёта....105
9.7 Контрольные вопросы....110
10 Примеры выполнения лабораторных работ....112
10.1 Лабораторная работа №1. Нулевой вариант....112
10.2 Лабораторная работа №2. Пример топологии для индивидуального задания....129
ПРИЛОЖЕНИЕ А. Доказательство формулы генерации двух независимых чисел, распределённых по нормальному закону, без использования суммирования (ЦПТ) и спецфункций....132
ПРИЛОЖЕНИЕ Б. Написание генератора случайных чисел, равномерно распределённых по интервалу от нуля до единицы....135

ПРИЛОЖЕНИЕ В. Пример сборки результата при параллельном исполнении 136
Список использованных источников.....138

1 Описание Grid-систем

1.1 Основные понятия

Недавно появившийся, но уже успевший стать популярным английский термин Grid означает среду, в которой объединены находящиеся в разных местах глобальной телекоммуникационной сети вычислительные установки и которая предназначена для выполнения распределенных приложений, использующих ресурсы этих установок. Так же, как и Интернет, Grid создавалась как технология совместных научных исследований, а позже была адаптирована к задачам электронного бизнеса. Программные инструменты для совместного использования научных ресурсов, ставшие стимулом для разработки первых Grid-сред, предусматривают объединения вычислительных ресурсов и ресурсов хранения для анализа данных, которые требуют больших объемов расчетов и широкой функциональности, а также объединение опыта за счет совместной визуализации больших наборов научных данных. Самая важная роль концепции Grid состоит в том, чтобы предложить решения новых задач, связанных с созданием надежных, масштабируемых и защищенных распределенных систем. Эти задачи возникают в связи с явной тенденцией к декомпозиции и распределению по сети ранее монолитных, централизованных служб. Технологии и концепции Grid быстро набирают популярность, выходя за рамки научных сообществ [2].

Грид (англ. grid — решётка, сеть) — согласованная, открытая и стандартизованная компьютерная среда, которая обеспечивает гибкое, безопасное, скоординированное разделение вычислительных ресурсов и ресурсов хранения информации, которые являются частью этой среды, в рамках одной виртуальной организации [1].

Грид вычисления — это форма распределённых вычислений, в которой «супер и виртуальный компьютер» представлен в виде кластера соединённых с

помощью сети, слабосвязанных компьютеров, работающих вместе для выполнения огромного количества заданий (операций, работ). Эта технология была применена для решения научных, математических задач, требующих для решения значительных вычислительных ресурсов. Грид вычисления используются также и в коммерческой инфраструктуре для решения таких трудоёмких задач как экономическое прогнозирование, сейсмоанализ, разработка и изучение свойств новых лекарств [1].

Грид является географически распределённой инфраструктурой, объединяющей множество ресурсов разных типов (процессоры, долговременная и оперативная память, хранилища и базы данных, сети), доступ к которым пользователь может получить из любой точки, независимо от места их расположения [1].

Сегодня трудно найти сколько-нибудь развитую страну, в которой не были бы развернуты национальные Grid-проекты, имеющие целью создание инфраструктуры, обеспечивающей удаленный доступ к разнообразным вычислительным ресурсам независимо от места расположения потребителя. В близкой перспективе Grid претендует на роль вычислительного инструментария в различных сферах человеческой деятельности, аналогично тому, как подобным инструментарием стали ПК и Internet[7].

1.2 Классификации Grid

Grid – среда, в которой объединены находящиеся в разных местах глобальной телекоммуникационной сети вычислительные установки и которая предназначена для выполнения распределенных приложений, использующих ресурсы этих установок .

Идея грид-компьютинга возникла вместе с распространением персональных компьютеров, развитием интернета и технологий пакетной передачи данных на основе оптического волокна, а также технологий локальных сетей (Gigabit Ethernet). Полоса пропускания коммуникационных средств стала достаточной, чтобы при необходимости привлечь ресурсы другого компьютера. Учитывая, что множество подключенных к глобальной сети компьютеров большую часть рабочего времени простаивает и располагает ресурсами, большими, чем необходимо для решения их повседневных задач, возникает возможность применить их неиспользуемые ресурсы в другом месте [1].

Существуют несколько классификаций распределенных вычислительных систем. Одна из них дана в [15]. Из-за различных требований к функциональности существует большое разнообразие таких систем. Для упрощения рассмотрения распределенные системы условно подразделяются на несколько типов.

Во-первых, это классификация по размерам систем и способам администрирования:

Кластер - это несколько десятков компьютеров, объединенных с помощью локальной сети. Администрирование осуществляется вручную.

Распределенная система корпоративного уровня – десятки и даже сотни компьютеров, при работе которых необходимо устанавливать правила совместного использования ресурсов. Масштаб таких систем, как правило,

небольшой, и можно обходиться прямыми административными мерами для организации работы ресурсов и пользователей.

Глобальная система (грид-система) – огромное количество компьютеров, число которых может достигать нескольких миллионов, распределенных по миру и объединенных глобальной сетью. Административное программное обеспечение, встроено в промежуточное программное обеспечение.

Во-вторых, распределенные системы можно рассматривать с точки зрения их функциональности, т.е. по типу предоставляемых ресурсов и видам прикладных задач, под решение которых они оптимизированы. Таким образом, можно выделить следующие системы [15]:

- Вычислительные системы (Computational Grid) – это тип систем, в которых основным компьютерным ресурсом является мощность процессора.
- Информационные системы (Data Grid) – это тип систем, в которых основным компьютерным ресурсом является объем памяти данных. Эти системы рассматриваются как огромные хранилища данных.

Последователи классификации [15] выделяют еще и третий тип систем - так называемые Передаточные грид (Network Grid или Delivery Grid). В этой системе узлы представляют собой передатчики (router), повышающие производительность и отказоустойчивость сети передачи данных между источником и приемником.

К сожалению, на сегодняшний день невозможно обозначить четкие границы между этими типами систем. В рамках этой работы производилось моделирование вычислительных грид-систем.

Примеры систем по классификации [15] даны в табл. 1.2.1.

Грид вычисления — это форма распределённых вычислений, где мощная ЭВМ представлена сетью слабосвязанных компьютеров, работающих вместе для выполнения огромного количества заданий. Эта технология была применена для решения научных, математических задач, требующих для

решения значительных вычислительных ресурсов. Грид вычисления используются также и в коммерческой инфраструктуре для решения таких трудоёмких задач как экономическое прогнозирование, сейсмоанализ, разработка и изучение свойств новых лекарств.

Согласно [6], мы имеем следующие факты. Наиболее характерными свойствами этой информационно-вычислительной среды являются:

- масштабы вычислительного ресурса (объем памяти, количество процессоров), которые многократно превосходят ресурсы отдельного компьютера или одного вычислительного комплекса;
- гетерогенность среды; в ее состав могут входить компьютеры различной мощности, работающие под управлением различных операционных систем и собранные на различной элементной базе;
- пространственное (географическое) распределение информационно-вычислительного ресурса;
- объединение ресурсов, которые не могут управляться централизованно (в случае, если они не принадлежат одной организации);
- использование стандартных, открытых, общедоступных протоколов и интерфейсов.
- обеспечение информационной безопасности.

К прикладным задачам, которые могут использовать GRID, в частности, относятся:

- сложное моделирование;
- совместная визуализация очень больших наборов научных данных;
- распределенная обработка в целях анализа данных;
- связывание научного инструментария с удаленными компьютерами и архивами данных.

Наиболее эффективно применение Grid`ов, по-видимому, для решения следующих задач:

- распределенные высокопроизводительные вычисления, решение очень крупных задач, требующих максимальных процессорных ресурсов, памяти и т.д.;
- «высокопоточные» вычисления, позволяющие организовать эффективное использование ресурсов для небольших задач, утилизируя временно простаивающие компьютерные ресурсы;
- проведение крупных разовых расчетов;
- вычисления с привлечением больших объемов распределенных данных, например, в метеорологии, астрономии, физике высоких энергий;
- коллективные вычисления: одновременная работа нескольких взаимодействующих задач разных пользователей.

Мировой опыт построения GRID-систем выявляет следующие проблемы:

- объединение разнородных систем;
- совместное использование данных;
- динамическое выделение ресурсов;
- переносимость приложений в гетерогенной среде;
- обеспечение информационной безопасности.

В настоящий время выделяют три основных типа GRID-систем [1]:

1) GRID на основе использования добровольно предоставляемого свободного ресурса персональных компьютеров (добровольная GRID);
2) Научная GRID — хорошо распараллеливаемые приложения программируются специальным образом (например, с использованием Globus Toolkit);

3) GRID на основе выделения вычислительных ресурсов по требованию (Enterprise GRID или коммерческая GRID) — обычные коммерческие приложения работают на виртуальном компьютере, который, в свою очередь, состоит из нескольких физических компьютеров, объединённых с помощью GRID-технологий.

Таблица 1.2.1 — Примеры классификации систем из [15]

Тип системы	Вычислительная система	Информационная система
Кластер	Кластер ИРЭ РАН, кластер ВЦ РАН	-
Распределенная система корпоративного уровня	Системы на основе IBM WAS	Lotus Notes
Глобальная система	Системы на основе Globus, gLite	WWW, eDonkey, eMule

1.3 Проблема распределения ресурсов

Эвристические оценки для распределения ресурсов, или эвристики, хорошо описаны в [24]. Ниже освещаются только основные моменты.

Последние работы в области Grid позволяют приложениям использовать вычислительные ресурсы, принадлежащие различным организациям, распределенным по различным странам и континентам. Одним из видов ресурсов Grid являются однородные многопроцессорные системы (кластеры), которые могут состоять из сотен или даже тысяч процессоров.

Двумя главными субъектами в системе Grid являются поставщики и потребители ресурсов. Оба субъекта имеют собственные стратегии. Потребители ресурсов применяют стратегии решения своих прикладных задач в зависимости от требуемого времени и наличного бюджета. Поставщики ресурсов используют стратегию получения наибольшей выгоды от вложенных средств. Владельцы ресурсов стараются максимизировать использование своих ресурсов.

Кластеры и задачи в [24] описываются следующим образом.

Процессоры одного кластера одинаковы, однако процессоры различных кластеров могут иметь различные объемы памяти и производительность. Таким образом, кластер C_i можно охарактеризовать количеством процессоров m_i, объемом памяти каждого процессора S_i и производительностью процессоров r_i по отношению к некоторому процессору, выбранному за стандарт. Стоимость K_i характеризует каждый кластер и равна стоимости единицы времени одного процессора этого кластера. Каждый кластер имеет очередь задач, которые должны быть выполнены.

Задачи T_j являются независимыми - во время их выполнения не происходит коммуникаций между задачами, что позволяет выполнять задачи в любом порядке. Мы предполагаем также, что каждая задача может быть решена

с использованием только одного кластера (нет необходимости в совместных вычислениях на нескольких кластерах). Каждая задача может быть описана количеством требуемых процессоров p_j, максимальным временем выполнения t_j (на p_j стандартных процессорах), требованиями к памяти M_j и предельным временем. Когда время выполнения задачи T_j превышает t_j, задача T_j должна быть прервана и отправлена обратно брокеру. Стоимость (или важность) задачи I_j определяется как функция от количества процессоров p_j и времени $t(T_j)$.

Приведенное ниже описание является очень похожим, за несколькими исключениями:

- требования к памяти пока не учитываются;
- отсутствие предельного времени выполнения задачи;
- отдельное поле приоритета задачи;
- прерывание выполнения задачи осуществляется только при отказе узла;
- при прерывании задачи она отправляется в очередь кластера.

Подобно [24] система распределения задач является иерархической:

- распределение задач по кластерам;
- распределение многопроцессорных задач на кластере.

Первый уровень обслуживается брокерами, а второй - кластерами (локальными планировщиками кластеров). Рассматриваются различные алгоритмы для этих уровней и оценивается эффективность системы в целом.

Говоря о стратегиях распределения заданий, будем иметь в виду исключительно первый пункт. Т. к. кластеры гомогенны (однородны) внутри, все узлы равнозначны (в плане надежности тоже). Однако это НЕ значит, что ы не будем затрагивать внутреннюю структуру кластеров при расчете оценок.

2 Моделирование Grid-систем

2.1 Обзор моделей

В связи с массовым распространением распределенных вычислительных систем стала актуальной проблема простого и в то же время эффективного использования таких систем. Поэтому необходим анализ алгоритмов работы подобных систем. Данный вопрос охватывает многие аспекты, в т. ч. анализ аппаратного и программного обеспечения вычислительных центров, вопросы стандартизации и т.д.

Для моделирования подобных распределенных систем используются различные подходы, ни один из которых не может претендовать на исчерпывающее описание, а позволяет описать какой-либо аспект функционирования или структуры системы.

В ИСП РАН в рамках работ по параллельным вычислениям и Grid-технологиям разрабатывается среда моделирования распределенных вычислительных систем. Кроме того, совместно с ВЦ РАН и МФТИ на базе существующих вычислительных ресурсов ведутся работы по созданию стенда как для работ проведения по верификации модели, так и для поддержки учебного процесса базовых кафедр МФТИ и ВМК МГУ[14].

Нами было проведено собственное исследование существующих программ, предложенных для моделирования грид-систем. Производилось сравнение следующих программных продуктов: MicroGrid[28, 34], SimGrid[32], ChicSim[33], GridSim[25, 26], OptorSim[30], Bricks[31], ИСП РАН[14], MONARCH[27]. После сопоставления имеющихся средств моделирования Grid – систем было решено создать свою [4, 11, 12, 16]. Не видим смысла проводить здесь все сравнение, приведем лишь ключевые моменты.

Укажем задаваемые параметры конфигурации для каждого из этих приложений.

В MicroGrid существуют следующие компоненты системы:

- вычислительные узлы;
- сеть;
- информационные сервисы;

Для MicroGrid задаются:

- параметры реальной системы (IP, ОС, количество процессоров [=1]);
- список узлов реальной системы (имя, частота ЦПУ, количество процессоров [=1], память [не используется]);
- коэффициент замедления виртуального времени моделирования относительно реального времени;
- список узлов вирутальной системы (имя, идентификатор, частота ЦПУ, количество ЦПУ [=1], память [не используется], управление трафиком);
- топология сети в формате SSFNET DML.

В SimGrid существуют следующие компоненты системы:

- список вычислительных ресурсов (скорость, доступность);
- список линий связи (задержка, пропускная способность);
- параметры данных (объёмы);
- топология сети;
- загрузка системы задачами;
- параметры отказов.

Для SimGrid задаются:

- узлы;
- линии связи.

В GridSim существуют следующие компоненты системы:

- пользователи (генераторы задач);
- диспетчеры (распределяют задачи пользователей);

- ресурс: ПК, кластеры, рабочие станции (вычислительные мощности);
- информационные сервисы (хранят информацию о ресурсах);
- ввод и вывод (обмен данными): линия, маршрутизатор, пакет, диспетчер пакетов, качество обслуживания, генератор фонового трафика.

Для GridSim задаются:

- параметры задач (типы, время исполнения, количество копий данных, политики диспетчеризации, объём входных и выходных данных, количество операций ввода/вывода, частоты появления, крайние сроки исполнения);
- параметры ресурсов (количество процессоров, стоимость вычислений, производительность, политика диспетчеризации, загруженность, объём памяти, часовой пояс);
- параметры сети (алгоритмы передачи данных) ;
- есть возможность моделирования логических линий.

В OptorSim существуют следующие компоненты системы:

- узлы (вычислительный ресурс, хранилище данных, модуль дублирования данных);
- диспетчер задач;
- сеть.

Для OptorSim задаются:

- топология GRID-системы;
- параметры ресурсов;
- параметры задач и соответствующих им файлов;
- параметры и алгоритмы доступа к данным, передачи данных, оптимизации дублирования данных, загрузки сети посторонним трафиком.

В ChicSim имеются следующие компоненты системы:

- пользователь (генератор задач, привязанный к узлу);
- узел (глобальный диспетчер определяет узел для исполнения, локальный диспетчер задач определяет процессор для исполнения, диспетчер данных);
- файлы.

Для ChicSim задаются:

- количество пользователей, узлов, заданий;
- параметры узлов (количество процессоров, производительность процессора, объём памяти);
- пропускная способность линий связи (одинаковая для всех линий).

В Bricks имеются следующие компоненты системы:

- вычислительная среда (клиенты, сеть, сервер);
- диспетчер (модуль диспетчеризации, модуль дублирования данных, модуль мониторинга сети, модуль мониторинга сервера, база данных ресурсов).

Для Bricks задаются:

- параметры сетевых каналов связи (пропускная способность, загруженность, динамика параметров);
- параметры серверов (производительность, загруженность, динамика параметров);
- топология сети;
- параметры задач (объём входных и выходных данных, количество операций).

В ИСП РАН имеются следующие компоненты системы::

- потоки задач;
- диспетчер;

- кластеры;
- сетевые соединения;
- хранилища данных.

Для ИСП РАН задаются:

- параметры потока задач (свойства задачи, промежутки между генерацией задач, количество задач, промежуток до генерации первой задачи);
- топология сети;
- количество и алгоритмы диспетчеров;
- наличие и количество хранилищ данных;
- параметры кластеров (количество узлов, количество ядер).

В MONARC имеются следующие компоненты системы:

- региональные центры (множество ЦПУ);
- хранилища данных на дисках и на магнитных лентах;
- сетевые соединения (локальные и глобальные сети);
- пользователи (генераторы задач);
- задача;
- анализатор и диспетчер задач;
- информационные сервисы.

Для MONARC задаются:

- топология сети;
- параметры линий связи (задержка, пропускная способность);
- параметры генерации новых данных;
- параметры копирования данных в региональные центры (часть от общего объёма);

- параметры региональных центров (количество ЦПУ, производительность, объём памяти и др.).

На основе выше изложенного были сформулированы аналогичные требования к нашей системе.

В GridModel имеются следующие компоненты системы:

- кластер (вычислительные мощности, генератор задач, узел сети);
- линия связи (связывает два кластера);
- диспетчер (распределяет задачи между кластерами).

Эти программные продукты обладают рядом особенностей и дополнительных свойств, которые тоже надо упомянуть.

Особенность MicroGrid – это допустимость GRID-системы из нескольких сотен узлов. Дополнительная возможность MicroGrid – автоматическая генерация топологии и ресурсов. Особенность SimGrid –возможность создать сеть до 100 000 моделируемых узлов.

Дополнительные возможности SimGrid:

- готовые модели GRID-систем с заданными параметрами;
- автоматическая генерация топологии и ресурсов.

Особенности GridSim:

- можно создать сеть до 1 000 узлов;
- нет ограничений на количество и вид параллелизма задач.

Дополнительные возможности GridSim:

- учёт расположения узлов в различных часовых поясах;
- есть возможность резервирования ресурсов.
- поддерживаются статическая и динамическая диспетчеризация

Особенность ИСП РАН в том, что модель реализована в виде конечного автомата. Дополнительная возможность ИСП РАН – возможность добавления

собственных потоков задач, алгоритмов диспетчеризации в виде пользовательских модулей.

Особенность MONARC в том, что выполнение моделируемых задач может прерываться, приостанавливаться и возобновляться, имитируя переключение ЦПУ. Особенность MONARC в том, что политики распределения задач учитывают надёжность элементов GRID-системы и могут комбинироваться с учетом степени влияния каждого.

2.2 GridModel

В нашем случая сделана попытка проанализировать алгоритмы распределения заданий по вычислительным центрам Grid-системы. Были рассмотрены уже предложенные алгоритмы и выработаны несколько своих собственных. Так же было принято учесть влияние показателей надежности элементов системы на ее производительность. В результате исследований был создан программный продукт «Имитационная модель Grid-систем» (GridModel) [4, 11, 12, 16].

Актуальность проблемы такова:

- Технологии и концепции Grid быстро набирают популярность, выходя за рамки научных сообществ, играют важную роль для научных и инженерных задач, для организации распределенных вычислений.
- Они делают возможным решение новых задач, связанных с созданием надежных, масштабируемых и защищенных распределенных систем.
- Развитые страны разворачивают национальные Grid-проекты, имеющие целью создание инфраструктуры. Grid может стать вычислительным аналогом электрической, железнодорожной или почтовой сети, аналогично тому, как подобным стал Internet.

Целью нашей работы является создание имитационной модели Grid-системы, удовлетворяющей нижеизложенным требованиям. Эти требования являются подцелями:

- реализовать возможность задавать параметры системы при помощи GUI (Graphic User Interface, графический интерфейс пользователя);
- реализовать заданный набор стратегий распределения заданий;
- учесть надежность элементов системы.

Для этого необходимо решить следующие задачи:

- рассмотреть существующие имитационные модели Grid-систем, выявить используемые принципы моделирования и стратегии распределения задач;
- реализовать GUI;
- рассмотреть имеющиеся модели надежности систем и выработать модель для данной задачи.

Проблема моделирования:

- Стремительное усложнение инфраструктуры распределенных информационно-аналитических систем, которое наблюдается в настоящее время, привело к необходимости развития подходов, методов и средств моделирования этих систем.
- Для моделирования подобных распределенных систем используются различные подходы, ни один из которых не может претендовать на исчерпывающее описание, а позволяет описать только какой-либо аспект функционирования или структуры системы.

При моделировании необходимо учесть:

- возможность создания сети при помощи GUI;
- распределение заданий по сети;
- способ выбора оптимального узла сети для исполнения;
- надежность элементов системы.

В GridModel существуют следующие компоненты системы:

- топология сети;
- кластера с набором параетров (название, количество узлов, количество ядер, частота ЦПУ, количество одновременно восстанавливаемых узлов, математические распределения для интервалов починки узлов, промежутков времени между отказами узлов, появлениями задач, для необходимого задачам количества узлов, объёма памяти под входные и выходные данные, для сложности задачи);

- линии связи с параметрами (пропускная способность, протяжённость, математические распределения для промежутков времени между отказами, для интервалов восстановления);
- стратегии распределения задач;
- диспетчер задач.

Принципиальные отличия нашей программы от аналогов:

- возможность задавать параметры системы по средствам GUI, а не текстовых файлов;
- реализация ниже указанного набора стратегий;
- учет надежности элементов системе;
- возможность адаптации нашей модели для нашей задачи.

Дополнительные возможности GridModel:

- комбинирование нескольких встроенных политик распределения для многокритериального выбора в рамках одной новой стратегии;
- визуализация загрузки моделируемой GRID-системы во времени;
- возможность задания потока задач для каждого кластера в файле специального формата.

Модель должна выполнять следующие действия:

1 Реальная система представляет собой вычислительную сеть, состоящую из вершин (кластеров) и линий связи.
2 На кластерах имеются свои узлы, на которых выполняются задания. Кластеры предполагаются гомогенными внутри.
3 На каждом кластере есть стационарный пуассоновский поток заданий. Пуассоновский он является, т.к. в нашем приближении обладает двумя свойствами:

   а) Приход одного задания не влияет на все остальные (отсутствие последействия).

b) При длине интервала, стремящейся к нулю, вероятности прихода двух и более заданий являются бесконечно малыми более высоких порядков, чем вероятность прихода одного задания.

Стационарный пуассоновский поток имеет еще одно свойство:

c) Стационарность (вероятность попадания на интервал некоторого числа точек не зависит от положения отрезка на временной оси).

4 Кластер имеет характеристики, по которым можно оценить время выполнения на нем определенного задания (у каждого узла кластера есть некоторое количество ядер, производительность одного ядра и т.д.)

5 Задание обладают такими характеристиками (характеристики обладают некоторыми случайными величинами, распределенными по определенным законам): приоритет (распределен по закону близкому к нормальному, т.к. чаще встречаются задания среднего приоритета), количество необходимых для выполнения узлов кластера, затраты на выполнение, размер задания и размер рассчитанной информации.

6 Кластер так же имеет очередь заданий на выполнение. Эта очередь является приоритетной (в соответствии с приоритетами задач). Приоритет заданий является относительным (появление более приоритетного задания не отменяет выполнение менее приоритетного).

7 На каждом кластере вводится так же пуассоновский поток отказов узлов. В случае отказа узла в процессе выполнения задания решение прекращается, результаты аннулируются, задание возвращается в очередь или начинает сразу же выполняться на других узлах кластера (если есть свободные).

8 Кластеры способны обмениваться заданиями с целью повышения производительности. Кластер, рассчитавший задание другого кластера, должен отправить ответ с результатами расчетов.

9 Каждая линия также имеет свой поток отазов и некоторое время восстановления, которое в частном случае можно считать константой

(это выражается в подключении резервной линии) или нормально-распределенной величиной.

10 Необходимо обосновать метод оценки оптимального кластера для выполнения задания по трем вариантам критериев:

a) Время исполнения (непосредственно само время исполнение + время пересылки задания и ответа) должно быть минимальным.

b) Показатели надежности должны быть максимальными, т.е. вероятность отказа аппаратуры во время исполнения задания или линии связи во время пересылки должна быть минимальной.

c) Сочетание этих двух крайних критериев.

На основе проведённого анализа предметной области и существующих программных средств для моделирования GRID-систем можно сформулировать следующие требования к проектируемой системе (Речь идет не о модели, а об описании программной реализации модели):

1 Система должна представлять собой самостоятельное приложение, не требующее для выполнения своих функций стороннего программного обеспечения, за исключением среды выполнения.

2 Система должна обладать графическим интерфейсом (GUI), позволяющим задавать параметры моделируемой GRID-системы в наглядной форме.

3 Система должна обеспечивать сохранение и загрузку файлов с параметрами моделируемой GRID-системы, а также сохранение файлов с результатами моделирования (Результаты моделирования получаются после набора достаточного объема данных для получения достоверных результатов с заданной точностью и заданным уровнем доверия).

4 Система должна обеспечивать моделирование работы GRID-системы с неизменными параметрами при разных стратегиях распределения задач и их комбинациях.

5 Система должна обеспечивать сбор и представление статистики в наглядной форме (в виде таблиц, диаграмм и графиков).

6 Система должна обеспечивать возможность просмотра смоделированной работы GRID-системы в наглядной форме.

7 Приложение должно выполняться в среде Microsoft .NET Framework и состоять из двух файлов исполняемого кода. Оконное приложение GridModel (GridModel.exe), которое непосредственно запускает пользователь, реализует графический интерфейс системы, работу с файлами и т. д. Оно использует динамически подключаемую библиотеку GridLibrary (GridLibrary.dll), в которой находятся классы, реализующие алгоритмы моделирования.

Система допускает выполнение трёх видов моделирования:

- моделирование работы одной и той же GRID-системы при разных стратегиях;
- оценка показателей надежности кластеров (работа каждого кластера моделируется отдельно);
- оценка показателей надежности линий связи (работа каждой линии связи моделируется отдельно).

Для разработки системы были выбраны следующие инструменты:

- среда Microsoft .NET Framework 3.5;
- Microsoft Visual Studio 2008 Pro.

2.3 Реализованные законы распределения случайных величин

В программе реализованы следующие законы распределения случайной величины. Будем считать, что R – это случайное число равномерно распределённое по $[0;1)$. Используемый ГСЧ устроен так, что число 0 может быть выдано, а 1 — нет (см. Приложение Б). Будем считать, что событие вероятности P происходит, если $R<P$. Таким образом, если $P=1$, то событие происходит всегда, а если $P=0$, то — никогда.

Из набора N взаимоисключающих событий $\{A_0, A_1, \dots, A_{N-1}\}$ происходит событие k, если $\sum_{i=0}^{k-1} P_i \le R < \sum_{i=0}^{k} P_i$ [1], где P_i — вероятность i-го события. Число R генерируется единожды и сравнивает поочерёдно с суммами. Иными словами, надо найти минимальное k, при котором $R<\sum_{i=0}^{k} P_i$. События могут не образовывать полную группу. В этом случае ни одно событие может не произойти.

В частном случае задача может быть сильно облегчена в плане реализации, если существует алгебраическое или подобное ему выражение k через R. В этом случае, нет необходимости производить поочерёдные сравнения.

Формула остается справедливой при бесконечном дискретном наборе событий, но практическая реализация становится затруднительной, если нет вышеописанной легко рассчитываемой зависимости k от R. Надо просто убрать маловероятные события и оставить конечное их число для последовательного рассмотрения.

Случайную величину обозначим t, $f(t)$ — плотность распределения величины, $F(t)$ — функцию распределения.

---

1 Логично предполагаем, что $\sum_{i=0}^{-1} P_i = 0$.

Выражения, где случайная величина стоит слева от знака равенства, есть способы её генерации.

Введем обозначения для законов распределения:

1) H(m, s) - нормальный с матожиданием (m) и СКО (s);

$$t \in (-\infty\,; +\infty) \wedge t \in \mathbb{R} \qquad (2.3.1)^2$$

$$f(t) = \frac{1}{s\sqrt{2\pi}} \cdot \exp\left(-\frac{(t-m)^2}{2s^2}\right) \qquad (2.3.2)$$

$$F(t) = \frac{1}{2} + \Phi\left(\frac{t-m}{s}\right) \qquad (2.3.3)^3$$

$$t = m + s \cdot \Phi^{-1}\left(R - \frac{1}{2}\right),\ R \in (0\,;1) \qquad (2.3.4)^4$$

$$t = m + s \cdot \sqrt{\frac{12}{N}} \cdot \left(\sum_{i=0}^{N-1} R_i - \frac{N}{2}\right),\ N \geq 6 \qquad (2.3.5)^5$$

$$t = m \pm s \cdot \sqrt{2 \cdot P^{-1}\left(\frac{1}{2}, R\right)} \qquad (2.3.6)^6$$

---

2 Время до события может принимать отрицательные значения? Нет. Но можно сделать так, что бы вероятность принятия отрицательного значения была минимальной. Однако это не отменяет необходимость осуществления проверки в программе на отрицательные значения. Будем считать, что при получении отрицательного значения требуется сгенерировать значение заново. Если отрицательное значение будет слишком «вероятным», то это процесс может занять длительное время. Если p – вероятность получения отрицательного значения, а T – время генерации значения и проверки, то среднее время на генерацию неотрицательного значения $\frac{T}{1-p}$.

3 Будем считать, что Φ — нечётная функция: Φ(0)=0, Φ(∞)= 0.5, Φ(-∞)= -0.5. В некоторых источниках полагают: Φ(0) = 0.5, Φ(∞) = 1, Φ(-∞) = 0.

4 В принципе расчёт по этой формуле возможен, но есть более простой и менее затратный путь в виде следующей формулы. Условие $R \in (0\,;1)$ можно обеспечить на уровне генерации случайного числа, а можно просто пропускать нулевые значения, как мы пропускаем отрицательные (см. Приложение Б).

5 В этой и следующей формулах рекомендуется использовать $R \in (0\,;1)$, но если его нет под руками — ничего страшного.

6 Формула требует исчисления обратной нормированной неполной нижней гамма-функции. Знак +/ - надо выбирать с равной вероятностью. Как вариант можно умножать число на $\left(2 \cdot floor\left(R' + \frac{1}{2}\right) - 1\right), R' \in [0;1)$. А можно вести расчет на основе целого случайного равномерно распределённого числа $r \in \mathbb{Z} \wedge r \in [0\,; r_{max} - 1]$. Тогда $t = m + s \cdot \sqrt{2 \cdot P^{-1}\left(\frac{1}{2}, \frac{r}{r_{max}}\right)} \cdot (2 \cdot (r \bmod 2) - 1)$. В принципе можно заменить «+» на «-». Это ничего не меняется по существу.

$$\begin{cases} t_0 = m + s\cdot\sqrt{-2\ln(1-R_0)}\cdot\cos(2\pi R_1) & (2.3.7)^7 \\ t_1 = m + s\cdot\sqrt{-2\ln(1-R_0)}\cdot\sin(2\pi R_1) & (2.3.8) \end{cases}$$

2) P(m, w) – равномерный на полуинтервале [m - w; m + w) ($w \geq 0$);

$$t \in [\text{m-w};\text{m+w}) \wedge t \in \mathbb{R} \quad (2.3.9)^8$$

$$f(t) = \frac{1}{2w} \quad (2.3.10)$$

$$F(t) = \frac{t - m + w}{2w} \quad (2.3.11)$$

$$t = (m - w) + 2wR = m + w(2R - 1) \quad (2.3.12)$$

3) П(m) – показательный с матожиданием (m);

$$\lambda = \frac{1}{m} \quad (2.3.13)$$

$$t \in [0;\infty) \wedge t \in \mathbb{R} \quad (2.3.14)$$

$$f(t) = e^{-\lambda t} \quad (2.3.15)$$

$$F(t) = 1 - e^{-\lambda t} \quad (2.3.16)$$

$$t = -\frac{1}{\lambda}\ln(1-R) = -m\ln(1-R) \quad (2.3.17)^9$$

4) К(m) – константное значение (m);

$$t \in \{m\} \wedge t \in \mathbb{R} \quad (2.3.18)$$

$$f(t) = \delta(t - m) \quad (2.3.19)$$

$$F(t) = \eta(t - m) \quad (2.3.20)^{10}$$

$$t = m \quad (2.3.21)$$

5) В(m, a) – Вейбула с математическим ожиданием m;

$$\lambda = \left(\frac{\Gamma\left(1 + \frac{1}{a}\right)}{m}\right)^a \quad (2.3.22)$$

$$t \in [0;\infty) \wedge t \in \mathbb{R} \quad (2.3.23)$$

---

7 Эти два значения независимы. Т. е. мы берем две независимые равномерно распределённые величины и получаем две независимые нормированные нормально распределённые величины. Вывод формулы дан в приложении А.

8 В связи с особенностями ГСЧ (m-w) достигается, а (m+w) — нет. Но если взять $R \in (0;1)$, границы будут не достигаться симметрично.

9 (1-R) взято вместо R затем, чтобы избежать логарифмирования нуля. Поэтому число 0 достигается.

10 Подразумевает, что функция Хевисайда η непрерывна слева, т. е. η(0)=0. δ — функция Дирака.

$$f(t)=a\lambda t^{a-1}\exp(-\lambda t^{a}) \tag{2.3.24}$$

$$F(t)=1-\exp(-\lambda t^{a}) \tag{2.3.25}$$

$$t=\sqrt[a]{-\frac{1}{\lambda}\ln(1-R)}=m\frac{\sqrt[a]{-\ln(1-R)}}{\Gamma\left(1+\frac{1}{a}\right)} \tag{2.3.26}$$

6) Л(m) – линейный с математическим ожиданием m;

$$\lambda=\frac{2}{3\cdot m} \tag{2.3.27}$$

$$t\in\left[0;\frac{2}{\lambda}\right)\wedge t\in\mathbb{R} \tag{2.3.28}$$

$$f(t)=\lambda\left(1-\frac{\lambda t}{2}\right) \tag{2.3.29}$$

$$F(t)=\lambda t-\left(\frac{\lambda t}{2}\right)^{2} \tag{2.3.30}$$

$$t=\frac{2}{\lambda}\left(1-\sqrt{1-R}\right)=3m\left(1-\sqrt{1-R}\right) \tag{2.3.31}$$[11]

$$t=\frac{2}{\lambda}\left|R_0-R_1\right|=3m\left|R_0-R_1\right| \tag{2.3.32}$$

7) Э(m, K*) – поток Эрланга с матожиданием m и порядка K[12];

$$\lambda=\frac{K+1}{m} \tag{2.3.33}$$

$$t\in[0;\infty)\wedge t\in\mathbb{R} \tag{2.3.34}$$

$$f(t)=\lambda\frac{(\lambda t)^{K}}{K!}\exp(-\lambda t) \tag{2.3.35}$$

$$F(t)=1-\exp(-\lambda t)\cdot\sum_{k=0}^{K}\frac{(\lambda t)^{k}}{k!} \tag{2.3.36}$$

$$t=-\frac{1}{\lambda}\sum_{i=0}^{K}\ln(1-R_i)=-\frac{m}{K+1}\sum_{i=0}^{K}\ln(1-R_i) \tag{2.3.37}$$[13]

---

11 (1-R) взято вместо R затем, чтобы число 0 достигалось.

12 Здесь K — это число пробрасываемых точек потока Пуассона. Поэтому показательное распределение имеет порядок ноль. Можно принять за порядок число экспоненциальных значений, которые нужно просуммировать, чтобы получить распределение Эрланга, т. е. (K + 1). В этом случае работает утверждение: порядок суммы гамма распределений с одинаковым λ равен сумме порядков с тем же λ. Но мы будем придерживаться первого варианта, считая экспоненциальное распределение распределением Эрланга нулевого порядка. Хотя во втором случае распределение Эрланга явно является частным случаем гамма-распределения.

13 С этой точки зрения, величину, распределённую по закону Эрланга, можно рассматривать как среднее арифметическое случайных величин, распределённых по экспоненциальному закону. Поэтому закон начинает становиться ближе к нормальному, значения становятся более «собранными» в районе математического ожидания.

8) T(m, w) – треугольный с математическим ожиданием m и полушириной w ($w \geq 0$). Число распределено по интервалу (m - w; m + w) с плотностью $\frac{w-|t-m|}{w^2}$. Также будем подразумевать, что при $w=0$, число всегда равно m;

$$t \in (\text{m-w};\text{m+w}) \wedge t \in \mathbb{R} \quad (2.3.38)$$

$$f(t)=\frac{w-|t-m|}{w^2} \quad (2.3.39)$$

$$F(t)=\frac{1}{2}+\frac{t-m}{w}\left(1-\frac{|t-m|}{2w}\right) \quad (2.3.40)$$

$$t=m+w(R_0-R_1) \quad (2.3.41)$$

9) Б() — бесконечность. Величина всегда равна $+\infty$. Это значит, сто событие никогда не наступает. Это можно использовать, чтобы отключить поток задач или поток отказов и т. д. Чтобы запретить исполнение заданий на кластере, надо установить ему нулевое число узлов.

$$t=+\infty \wedge t \in \overline{\mathbb{R}} \quad (2.3.42)$$

Целочисленные аналоги. Целочисленный закон выдает целочисленное значение, но его параметры могут быть вещественными. Если от параметра требуется быть целым числом, то он обозначается звёздочкой.

Вот аналоги:

10) H*(m, s) - нормальный с матожиданием (m) и СКО (s);

$$k \in (-\infty;+\infty) \wedge k \in \mathbb{Z} \quad (2.3.43)$$

$$P[k]=\frac{1}{s\sqrt{2\pi}}\cdot\exp\left(-\frac{(k-m)^2}{2s^2}\right) \quad (2.3.44)^{14}$$

$$t=\langle unknown \rangle \quad (2.3.45)^{15}$$

11) P*(m, w) – равномерный на отрезке [round(m - w); round(m + w)] ($w \geq 0$);

$$k \in [round(m-w); round(m+w)] \wedge k \in \mathbb{Z} \quad (2.3.46)$$

14 Расчёт ведётся в соответствии с локальной теоремой Лапласа.

15 Формула нам неизвестна, значение находится по попаданию точки на часть интервала [0;1).

$$P[k]=\frac{1}{round(m+w)-round(m-w)+1} \quad (2.3.47)^{16}$$

$$k=round(m-w)+floor\left(R\cdot(round(m+w)-round(m-w)+1)\right) \quad (2.3.48)$$

12) П*(a, b*) – целочисленный показательный;

$$k\in[b;\infty)\wedge k\in\mathbb{Z} \quad (2.3.49)$$

$$P[k]=C\cdot e^{-\frac{k-b}{a}}, \quad (2.3.50)$$

где $a>0$, а C определяется исходя из условия нормировки:

$$C=(1-e^{-\frac{1}{a}}) \quad (2.3.51)$$

тогда

$$P[k]=(1-e^{-\frac{1}{a}})\cdot e^{-\frac{k-b}{a}} \quad (2.3.52)$$

$$k=\left[-a\ln(1-R)\right]+b \quad (2.3.53)^{17}$$

Можно доказать, что

$$m=\frac{1}{e^{1/a}-1}+b \quad (2.3.54)$$

Так же можно доказать, что при больших a выполняется

$$m\approx a+b \quad (2.3.55)$$

При $b=0$ и $a\geq 10{,}4$ погрешность меньше 5%, а при $b=0$ и $a\geq 50{,}4$ — меньше 1%.

16 В знаменателе — число вариантов значения. Все варианты равновероятны.

17 Обратите внимание, что ни знак минус, ни константу нельзя выносить ни за знак целой части, ни за знак дробной.

2.4 Процесс моделирования в GridModel

Моделирование будем проводить по схеме особых событий (будем использовать цепи событий). Осуществляется некоторое не очень большое число испытаний. На основе анализа СКО полученных данных определяется необходимое количество испытаний модели. Впоследствии, в процессе моделирования, было обнаружено, что:

1 Процесс является стационарным, т.е. со временем его характеристики принимают некоторые установившиеся значения и перестают меняться. Реальная система должна удовлетворять таким параметрам, потому что именно в этом и заключается принцип стабильной работы. Такие параметры, как среднее время выполнения задание, средняя длина очереди, среднее число задействованных узлов и т.д., должны принимать некоторые постоянные конечные значения со временем.

2 Процесс является эргодичном. Это означает, что система всегда должна переходить в одно и то же стационарное состояние вне зависимости от начальных условий. В реальной системе это означает, система всегда переходит в одно установившееся состояние. Для моделируемой системы этот показатель легко проверяется: после серии реализаций статистические данные не должны меняться.

На практике это выражается в том, что с увелечением времени моделирования дисперсия анализируемых нами величин начинает сокращаться. Нам неизвестен закон распределения каждой случайной величины, однако нам известны неравенства Чебышева, одно из которых

$$p_{откл} \le \frac{D}{\varepsilon^2}, \qquad (2.4.1)$$

где $p_{откл}$ - вероятность отклонения от матожидания не менее чем на ε, D – дисперсия искомой величины.

При малых дисперсиях, т.е. при достаточно большом времени моделирования значение многих величин (средняя длина очереди, среднее время выполнения заданий и др.) перестают сильно отличаться от их матожиданий.

Если усреднить результаты значительного числа прогонок, результат становится распредленным по нормальноу закону (центральная предельная теорема). СКО обратно пропорциональна корню квадратному из числа прогонок. Т. е., чтобы сократить доверительный интервал в три раза надо в девять увеличить количество испытаний. Значит, надо в девять раз увеличить время моделирования! Если же точность слишком высока, то, сократив ее два раза, мы скрати время рассчетов в четыре раза. Поэтому не следует требовать завышенную точность.

Стоит отметить, что для оценки точности можно использовать коэффициенты Стьюдента. Стандартным доверительны интервалом будем считать интервал в 3σ, где $\sigma=\sqrt{D}$.

Программа производит все расчеты и оценки погрешностей самостоятельно. Необходимо просто понимание назначения параметров.

Схема моделирования распределения заданий приведена на рис. 2.4.1.

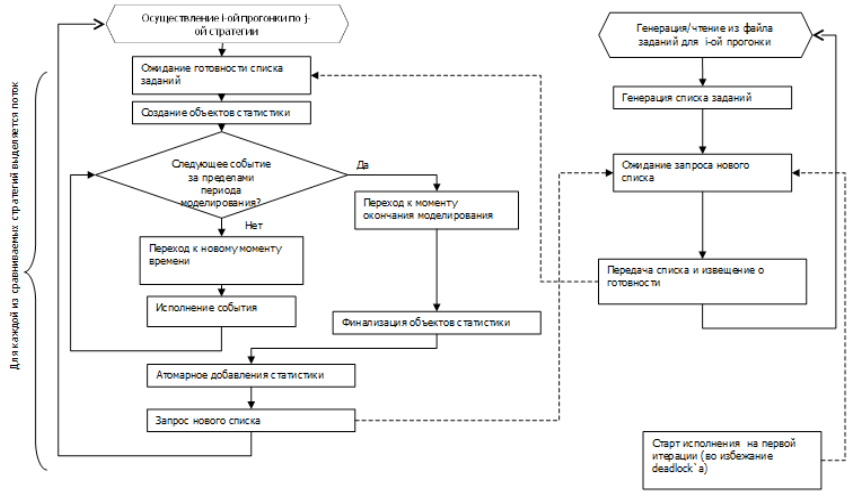


Рисунок 2.4.1 - Схема моделирования распределения заданий

2.5 Цепи событий имитационного моделирования

Имитационная модель GRID-системы представляет собой агрегат, состоящий из следующих элементов:

1) кластеры;
2) приоритетные очереди кластеров;
3) каналы связи;
4) очереди на передачу через канал связи;
5) задания;
6) сообщения (задания на исполнение или ответы, результаты расчетов заданий).

События есть особые состояние агрегата. Именно в них может происходить изменение состояние скачком. Это ключевые моменты функционирования агрегата. Они управляют логикой работы модели.

2.5.1 Схема моделирования

Моделирование можно проводить по схеме одиночной длинной реализации. Как нам известно, из курса моделирования, такое имитационное моделирование возможно при выполнении двух условий:

1) Процесс является стационарным, т.е. со временем его характеристики принимают некоторые установившиеся значения и перестают меняться. Реальная система должна удовлетворять таким параметрам, потому что именно в этом и заключается принцип стабильной работы. Такие параметры, как среднее время выполнения задание, средняя длина очереди, среднее число задействованных узлов и т.д., должны принимать некоторые постоянные конечные значения со временем.
2) Эргодичность. Это означает, что система всегда должна переходить в одно и то же стационарное состояние вне зависимости от

начальных условий. В реальной системе это означает, система всегда переходит в одно установившееся состояние. Для моделируемой системы этот показатель легко проверяется: после серии реализаций статистические данные не должны меняться.

Для проверки правильности результатов можно провести некоторое небольшое число испытаний и оценить результаты по центральной предельной теореме.

2.5.2 Кластеры

Краткая характеристика

Каждый кластер генерирует задания, время между которыми может быть описано некоторым законом распределения.

По умолчания принимается, что задания приходят на кластер под действием стационарного пуассоновского потока событий. Таким образом, время между заданиями оказывается распределенным по показательному закону. Это доказывается в курсе математики.

У каждого кластера есть некоторое количество узлов. Каждое задание тоже требует для исполнения некоторое количество узлов. Когда генерируется новое задание, кластер сначала определяет кластер, на котором это задание следует выполнять. Этот выбор определяется стратегией.

Если задание лучше выполнять на другом кластере, то рассчитывается путь до этой машины (кластера, до которых не добраться, не учитываются при выборе оптимального) и событие отправляется в соответствующий канал связи.

Если задание лучше выполнять здесь, то оно либо ставится на выполнение (если есть свободные узлы), либо отправляется в очередь.

Так же кластер может получать задания из канала связи. Получив задание, кластер проверяет, кому оно предназначается. Если ему, то задание либо ставится на выполнение (если есть свободные узлы), либо отправляется в очередь.

В противном случае, задание пересылается дальше. Здесь возможная следующая ситуация: путь до оптимального кластера мог измениться, кластер может стать недоступным из-за обрыва связи и т.д. В этом случае анализ машины для исполнения надо провести заново. А затем либо оставить здесь, либо передать в соответствующий канал связи.

Если отказ линии связи происходит в момент передачи задания, задания возвращаются кластеру.

Когда кластер выполняет задание чужого кластера, он должен выслать ответ. Ответ тоже отправляется самым оптимальным путем. В процессе пересылки ответа, так же возможны вышеописанные проблемы. Если в процессе пересылки ответа путь до кластера назначения теряется, ответ считается потерянным.

В процессе работы кластера возможны отказы узлов. Если на отказавшем узле выполнялось задание, оно ставится в конец очереди.

Таким образом, для кластера существуют следующие события, особые состояния агрегата, которым соответствуют цепи событий.

События

1. Генерация нового задания.
    a. Определение кластера, где это задание лучше выполнять[18].
        i. Здесь.
            1. Постановка задания на выполнение или в очередь.
        ii. На другой машине.
            1. Отправка в соответствующий канал связи.
    b. Генерация времени прихода следующей задачи.

2. Получение задания из канала связи.
    a. Определение, надо ли передавать задание дальше.
        i. Задание пришло на выполнение здесь.
            1. Постановка задания на выполнение или в очередь.
        ii. Задание надо переслать дальше, и этот путь доступен.

18 Здесь и далее, если не оговорено иное: Если установлена стратегия "передавать через работающие каналы", то неработающие пути в топологии сети не участвуют.

1. Отправить задание в соответствующий канал связи.

iii. Задание надо пересылать, но путь поврежден.

1. Стратегия предавать через работающие каналы.

a. Переход к п. 1.а.

2. Стратегия ждать восстановления канала.

a. Постановка на ожидание передачи через канал связи.

3. Срыв отправки задания через канал связи (из-за поломки в момент передачи).

a. Стратегия предавать через работающие каналы.

i. Переход к п. 1.а.

b. Стратегия ждать восстановления канала.

i. Постановка на ожидание передачи через канал связи.

4. Окончание обслуживания задания

a. Определение и постановка на обслуживание заданий из приоритетной очереди, *если таковое возможно.*

b. Если задание создано другой машиной. Стратегия предавать через работающие каналы. Обратный путь существует.

i. Поставить ответ в очередь на отправку.

c. Если задание создано другой машиной. Стратегия предавать через работающие каналы. Обратный более не доступен.

i. Пометить ответ как потерянный.

d. Если задание создано другой машиной. Стратегия ждать восстановления канала.

i. Поставить ответ в очередь на отправку.

5. Отказ узла кластера

a. *Если на узле выполнялось задание*, оно переносится в конец очереди.

6. Починка узла кластера.

a. Постановка задания из очереди на выполнение, *если есть подходящие задания.*

7. Получение ответа из канал связи.
    a. *Ответ принадлежит этому кластеру.*
        i. Принятие ответа.
    b. *Ответ принадлежит другому кластеру. Вычисленный ранее путь до сих пор существует.*
        i. Отправить ответы следующему звену в пути.
    c. *Ответ принадлежит другому кластеру. Вычисленный ранее путь не существует. Стратегия ждать восстановления канала.*
        i. Поставить задание на ожидание передачи через канал.
    d. *Ответ принадлежит другому кластеру. Вычисленный ранее путь не существует. Стратегия предавать через работающие каналы. Альтернативный путь существует.*
        i. Пересылать через альтернативный путь.
    e. *Ответ принадлежит другому кластеру. Вычисленный ранее Путь не существует. Стратегия предавать через работающие каналы. Альтернативный путь не существует.*
        i. Пометить ответ как потерянный.

2.5.3 Каналы связи

Стратегии использования каналов

1. Предавать через работающие каналы. Если канал ломается, ищется альтернативный путь. Ответ может теряться.
2. Ждать починки канала. Если канал ломается, сообщения будет ждать починки канала, не изменяя своего маршрута. Ответ не может теряться.

В GridModel используется исключительно вторая стратегия.

События

Каналы связи устроены проще. У них есть только следующие события:

1. Получение задания/ответа от кластера на пересылку.
    a. *Канал занят.*
        i. Поставить в очередь.
    b. *Канал свободен.*
        i. Начать передавать.
    c. *Канал не исправен. Стратегия предавать через работающие каналы.*

i. Отвергнуть задание.

d. Канал не исправен. Стратегия ждать восстановления канала.

i. Поставить в очередь.

2. Окончание пересылки.
   a. Передать задание/ответ кластеру.

3. Отказ линии связи.
   a. Отменить передачу.
   b. Если стратегия предавать через работающие каналы.
      i. Вернуть кластеру.
   c. *Если стратегия ждать восстановления канала*[19].
      i. Вернуть в очередь на отправку.

4. Восстановление канала связи.
   a. *Если очередь не пуста (при стратегии ждать восстановления канала),* поставить сообщение на передачу.

2.5.4 Задания

Задания приходят на кластер стационарным потоком Пуассона.

1. Вероятность появление задание на временном интервале не зависит от положения этого интервала на временной оси, а зависит только от длины временного интервала.
2. Вероятность прихода некоторого количества заданий на временной интервал не зависит от того, сколько заданий пришло на другие интервалы.
3. Вероятность прихода двух и более заданий при длине интервала, стремящейся к нулю, является бесконечно малой более высокого порядка, чем вероятность прихода одного события.

Эта модель является самой простой, поэтому она принята по умолчанию.

У задания есть следующие параметры, перечисленные в табл. 2.4.1

19 Здесь видна некоторая избыточность модели: Если стратегия ждать починки канала, то при срыве передачи канал связи **не передает** задание кластеру, но у кластера такая возможность предусмотрена. Это сделано с целью модификации модели в дальнейшем.

Таблица 2.4.1 — Параметры задания

Параметр	Закон распределения
Сложность (трлн. операций с плавающей запятой, TFlop[20])	Нормальный, когда сложность заданий группируется вокруг некоторого значения, или равномерный.
Размер задания	Нормальный, когда размеры заданий обычно группируется вокруг некоторого значения, или равномерный
Размер ответа	Нормальный, когда размеры ответы обычно группируется вокруг некоторого значения, или равномерный
Приоритет	Нормальный (группируются вокруг некоторого значения), равномерный.
Требуемое число узлов	Нормальный (группируются вокруг некоторого значения), равномерный.

2.5.5 Сообщения

Имеют приоритет, соответствующий приоритету задания. Несут в себе либо задание, либо ответ. Размер сообщения определяется либо размером этого задания, либо размером ответа.

Так же у сообщения внутри есть путь, по которому задание следует в сети.

20 Так будем называть эту единицу измерения для краткости.

3 Стратегии распределения заданий

3.1 Стратегии выбора оптимального кластера для исполнения

Во всех нижеописанных стратегиях узлам присваиваются стоимости выполнения. Чем меньше стоимость, тем выгоднее выполнять задание на узле. Оценка производится только для тех кластеров, которые доступны по сети. Если два кластера получают одинаковую оценку, то выбирается случайный из них [35, 43, 44].

Понятно, что оценка должна отвечать хотя бы двум критериям:

- адекватно оценивать загруженность кластера;
- быть легко вычисляемой, т.е. высчитываемой по относительно простой формуле.

1) Случайный кластер. Каждый кластер получает себе в качестве оценки случайное число, равномерно распределенное по интервалу (0;1]. Эта система удобна, когда все задания генерируются на одном кластере или небольшой группе кластеров, а все кластеры одинаковые по производительности.

2) Наибыстрейшего выполнения:

$$t_{вып} = t_{зад} + t_{пер} + t_{пер.рез} + \sum_i \frac{t_{задi} \cdot w_i}{W} + \sum_j \frac{t_{задj} \cdot w_j}{W}, \qquad (3.1.1)$$

где $t_{зад}$ - предполагаемое время выполнения задания на кластере, $t_{пер}$ - предполагаемое время пересылки задания по сети до узла, $t_{пер.рез}$ - предполагаемое время обратной пересылки результата вычислений, $t_{задi}$ - время, оставшееся до окончания исполнения i-го задания на кластере в текущий момент времени задания, $t_{задj}$ - предполагаемое время исполнения j-го задания из очереди, w_i и w_j - количество узлов, на которых будут исполняться задания, W - количество узлов кластера. Времена оцениваются на основании

производительности кластера, сложности задания, объемов передаваемой информации и пропускной способности каналов.

3) Создатель. Кластер создатель получает оценку в 1.0, а все остальные кластера оценку в 100.0. Данная стратегия реализует выполнение заданий на том же кластере, на котором они появились.

4) Стратегия минимального риска. Среднее время работы на кластере одного узла, деленное на количество требуемых заданию узлов, представляет собой (при допущении пуассоновского потока отказов и восстановлений) среднее время работоспособного состояния этой группы узлов. Поэтому в качестве оценки было взято

$$\frac{t_{зад} \cdot w}{t_{раб}}, \tag{3.1.2}$$

где $t_{зад}$ - предполагаемое время выполнения задания на кластере, $t_{раб}$ - среднее время безотказной работы одного узла, w - количество узлов, на которых будет исполняться задание. Эта формула есть отношение предполагаемого времени выполнения задания к среднему времени работы узлов, на которых это задание будет выполняться.

5) Соотношения количества свободных узлов кластера и количества узлов, требуемых задачей. С корректируем оценку из [14]

$$\frac{w}{N+1}, \tag{3.1.3}$$

где w - количество узлов, на которых будет исполняться задание, N - количество свободных узлов кластера. Используется при быстром выполнении заданий и в отсутствие очереди.

6) Минимальной сложности на узел кластера. Оценка [14]

$$\frac{c_{зад} + \sum_j c_{задj}}{W}, \tag{3.1.4}$$

где $c_{зад}$ - сложность задания, для которого выбирается исполнитель, $c_{задj}$ - сложность j - го задания в очереди кластера, W - количество узлов кластера.

7) Оптимальной сложности. Оценка (наша модификация 6))

$$\frac{c_{зад} + \sum_j c_{задj}}{W \cdot L \cdot P}, \tag{3.1.5}$$

где $c_{зад}$ - сложность задания, для которого выбирается исполнитель, $c_{задj}$ - сложность j - го задания в очереди кластера, W - количество узлов кластера, L - количество ядер на узле, P - производительность узла.

8) Оптимального использования узлов. Оценка [14]

$$\frac{w_{зад} + \sum_j w_{задj}}{W}, \tag{3.1.6}$$

где $w_{зад}$ - количество узлов, необходимое заданию, для которого определяется кластер, $w_{задj}$ - количество узлов, необходимое j-му заданию в очереди кластера, W - количество узлов кластера.

Сразу видно, что результаты оценок имеют разную размерность, например:

- время выполнения;
- сложность на узел кластера;
- сложность на производительность;
- безразмерные величины и т. д.

Видно, что все оценки положительны, причем, чем меньше оценка, тем приоритетнее кластер.

Это основные стратегии. Они были заложены в модель с самого начала. Впоследствии были добавлены еще несколько стратегий, в соответствии с исследованиями нашей кафедры.

9) Самого мощного сегмента

$$P^{-1} \tag{3.1.7}$$

где P - производительность узла.

10) Балансировки текущей нагрузки между сегментами

$$\sum_j c_{задj} + C \qquad (3.1.8)$$

где $c_{задj}$ - сложность j - го задания в очереди кластера, C - некоторая константа, единая для всех кластеров.

11) Среднего времени выполнения подзадачи

$$\frac{1}{n}\sum_j \tau_{задj} + C \qquad (3.1.9)$$

где $\tau_{задj}$ - время выполнения выполненного задания, n - количество выполненных заданий, C - некоторая константа, единая для всех кластеров.

В начальный момент времени $n=0$ - получается неопределенность $\frac{0}{0}$, поэтому будем полагать оценку равной C. Таким образом, оценка в начальный момент времени одинакова для всех кластеров и принимает наименьшее возможное значение. Поэтому задания будут распределяться так:

- сначала случайно между всеми, пока хотя бы одно задание не выполнится;
- когда часть кластеров уже выполнила хотя бы одно задание, а часть — нет, то задания распределяются между кластерами, не выполнившими ни одного задания;
- когда все кластеры выполнили, хотя бы одно задание, задания распределяются в соответствии с оценками.

12) Количества ждущих подзадач. Оценка равняется количеству задач в очереди кластера плюс константа C.

Константа всюду добавляется, чтобы значение было все время положительным. Это потребуется нам в дальнейшем.

3.2 Способы использования стратегий

Можно оценить один и тот же кластер сразу по нескольким критериям, но для этого надо определить способ объединения оценок по различным критериям. Из курса теории принятия решений нам известно, что существуют следующие способы этого сделать:

$$W = \sum_{i=1}^{n} \alpha_i W_i \,, \tag{3.2.1}$$

$$\mathrm{W} = \prod_{i=1}^{n} W_i^{\alpha_i} \,, \tag{3.2.2}$$

$$\mathrm{W} = \min_{i=1}^{n} \{W_i^{n\alpha_i}\} \tag{3.2.3}$$

$$W = max \underset{i=1}{\overset{n}{} } \{W \overset{na_i}{i}\} \tag{3.2.4}$$

$$W = (1-\mu) \cdot max \underset{i=1}{\overset{n}{} } \{W \overset{na_i}{i}\} + \mu \cdot min \underset{i=1}{\overset{n}{} } \{W \overset{na_i}{i}\} \tag{3.2.5}$$

$$\sum_{i=1}^{n} \alpha_i = 1 \tag{3.2.6}$$

$$0 \le \mu \le 1 \tag{3.2.7}$$

где W_i -оценка по i-му критерию, α_i - вес i-го критерия, n-количество критериев, μ - коэффициент риска (чем он больше, тем меньше риск), W-объединенный критерий.

Первый критерий – это критерий аддитивной свертки. Он представляет собой усреднение оценок по различным стратегиям наподобие среднего арифметического. Только оценки по различным стратегиям имеют, в общем случае, различные веса.

Второй критерий – критерий произведений, он же — критерий мультипликативной свертки. Итоговая оценка – это среднее геометрическое оценок по различным стратегиям с различными весами критериев.

Стоит отметить, что второй критерий применим только для W_i>0. значение W_k=0 приводит к тому, что объединенный критерий произведений

принимает тоже нулевое значение и влияние значений всех остальных W_i не учитывается.

Третий критерий – критерий максиминной свертки. Должны выполняться условия $\forall W_i \in [0;1]$ и условие: чем больше W, тем приоритетнее обладающий им узел. Т.е. использование в качестве W стоимости исполнения не представляется возможным. В этой стратегии мы рассматриваем самую плохую ситуацию: узел получает наихудшую оценку, а затем из всех узлов выбирается узел с максимальной оценкой.

Четвертый критерий – критерий «азартного игрока». На W_i налагаются те же ограничения, что и в предыдущем случае. В этой стратегии мы рассматриваем самую хорошую ситуацию: узел получает наилудшую оценку, а затем из всех узлов выбирается узел с максимальной оценкой.

Пятый критерий — объединение третьего и четвертого. При граничных значениях μ он переходит в один из них.

Список способ оценки по нескольким критериям можно продолжать и далее.В общем случае неясно, какой именно из объединенных критериев необходимо использовать. Поэтому выбор должны сделать мы сами. Остановимся на мультипликативном объединении, а если оно не будет нас устраивать, воспользуемся аддитивным, минимаксным или каким-нибудь другим.

Таким образом, задавая веса используемых оценок кластеров по различным стратегиям, мы сможем подобрать оптимальную стратегию распределения заданий по кластерам.

3.3 Анализ стратегий распределения заданий

Существует несколько эвристик (эмпирических методов) для определения оптимального узла для исполнения задания. Модель создана с целью определения наиболее удачной стратегии распределения для заданной топологии сети.

Наиболее часто встречающейся топологией является топология типа «звезда». Необходимо провести анализ распределения заданий на нескольких топологиях.

Изначально были использованы следующие стратегии выбора кластера для выполнения задания:

- стратегия 1 (C1) - Создатель;
- стратегия 2 (C2) - Случайный хост;
- стратегия 3 (C3) - Наискорейшего исполнения;
- стратегия 4 (C4) - Минимального риска;
- стратегия 5 (C5) - Соотношения свободных узлов кластера;
- стратегия 6 (C6) - Минимальной сложности на узел хоста;
- стратегия 7 (C7) - Оптимальной загрузки;
- стратегия 8 (C8) - Оптимального использования узлов.

Для проверки данных моделирования на реальной системе программа «GridModel» была модифицирована, стратегии были адаптированы для данной задачи. Для соответствия с реальными тестами были добавлены четыре новые стратегии:

- стратегия 9 (C9) - Самого мощного сегмента;
- стратегия 10 (C10) - Балансировки текущей нагрузки между сегментами;
- стратегия 11 (C11) - Среднего времени выполнения подзадачи;

– стратегия 12 (С12) - Количества ждущих подзадач.

Всего тестирование проводилось для пяти стратегий:

1) стратегия случайного сегмента;

2) стратегия самого мощного сегмента;

3) стратегия балансировки текущей нагрузки между сегментами;

4) стратегия среднего времени выполнения подзадачи;

5) стратегия количества ждущих подзадач.

Сравнение стратегий дано в виде таблицы (табл. 3.3.1).

Таблица 3.3.1 – Сравнение стратегий

№	Стратегия	Учет кол-ва заданий в очереди	Учет сложности заданий	Учет производительности узла	Примечание
1	Случайного сегмента	Нет	Нет	Нет	Работает на однородных сетях
2	Самого мощного сегмента	Нет	Нет	Да	Минимизирует время выполнения задачи, но увеличивает очередь
3	Балансировки текущей нагрузки между сегментами	Да	Да	Нет	
4	Среднего времени выполнения подзадачи	Нет	Да	Да	Необходимо время на определение средних времен выполнения заданий
5	Количества ждущих подзадач	Да	Нет	Нет	

Качественное сравнение стратегий дано в виде диаграмм на рис. 3.3.1 (модель) и 3.3.2 (реальная система).

Рисунок 3.3.1 – Диаграмма результатов использования стратегий при моделировании

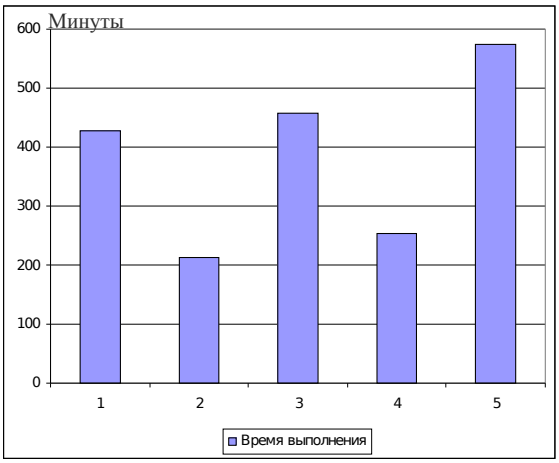


Рисунок 3.3.2 – Диаграмма результатов использования стратегий при распределении в реальной системе

Стратегия 1) подходит только для однородных сетей. В этом случае случайное равновероятное распределение заданий обеспечивает равномерную нагрузку при условии, что заданий достаточно много и они примерно равны по сложности. Для неоднородной сети стратегия может давать результат различного качества в зависимости от степени неоднородности сети.

Стратегия 5) учитывает только количество заданий в очереди, ни сложность заданий, ни производительность узлов не учитывается. Добавив сложность заданий, мы получим стратегию 3). При балансировке нагрузки между сегментами учитывается как длина очереди, так и сложность заданий в ней, поэтому среднее время выполнения задания меньше. Чтобы еще сократить время выполнения задания нужно учесть производительность сегмента.

Таким образом, мы приходим к стратегии 4), где мы учитываем и сложность заданий, и производительность узлов, т. к. время выполнения есть отношение производительности к сложности. Однако длина очереди здесь не учитывается. Это может стать фатальным при длинных очередях. Так же стратегия 4) неудобна тем, что требуется время на определение среднего времени выполнения заданий на каждом узле. До этих пор распределение по этой стратегии будет неопределенным.

Если предположить, что все задачи примерно одной сложности, то можно учитывать только производительность узлов; в этом случае не будет периода определения среднего времени выполнения.

Теперь остается попробовать учитывать только производительность сегмента. В этом случае приходим к стратегии 2). Время выполнения в этом случае получается минимальным. Однако необходимо обратить внимание, что сокращение времени выполнения, в том числе при переходе от стратегии 4) к 2), может привести к увеличению средней и пиковой длины очереди. Ни 2), ни 4) никак не учитывают длину очереди.

Как можно заметить из сопоставления рис. 3.3.1 и 3.3.2 программа правильно соотносит стратегии по производительности, за исключением стратегии 1), т. е. случайного распределения заданий по машинам. Это может быть объяснено, недостаточностью количества заданий при испытаниях, ведь моделирование осуществлялось на большом числе прогонок.

4 Модель надёжности

Надёжность – это свойство объекта сохранять во времени в установленных пределах все параметры, обеспечивающие выполнение требуемых функций в заданных условиях эксплуатации [19].

Модель надёжности включает в себя [8, 35]:

- резервирование;
- учёт способа восстановления отказавших узлов;
- учёт вероятности не обнаружения отказа;
- учёт времени подключения резерва.

В данной работе будет рассмотрена надёжность кластеров. Однако не будут анализироваться все параметры кластеров, а только параметры его узлов. Исправным будем считать кластер, у которого исправны все узлы.

Из всех параметров надёжности в имитационной модели нас будут интересовать следующие параметры узлов кластера:

- среднее время рабочего состояния;
- среднее время отказного состояния;
- коэффициент готовности;
- коэффициент использования (коэффициент готовности минус время технического обслуживания);
- средний и минимальный размер резерва;
- количество заданий, исполнение которых было прервано отказом.

Входными параметрами будут являться:

- законы распределения времени безотказной работы и времени отказного состояния;
- размер ненагруженного резерва (нагруженный резерв включён в рабочие узлы);
- вероятность необнаружения отказа [8].

Пусть имеется система из одного узла. Интенсивности потоков: λ - поток отказов этого узла, μ - поток восстановлений. Среднее время работы, соответственно, $t_{раб} = 1/\lambda$, а среднее время отказа (простоя) - $t_{простоя} = 1/\mu$. p_i - здесь и далее вероятность отказа i узлов, если не оговорено иное. Предположим, что в начальный момент времени система была исправна. Составив и решив систему дифференциальных уравнений Колмогорова, получим:

$$p_0(t) = \frac{\mu}{\mu+\lambda} + \frac{\lambda}{\mu+\lambda} e^{-(\mu+\lambda)t} \quad (4.1)$$

$$p_1(t) = \frac{\lambda}{\mu+\lambda}(1 - e^{-(\mu+\lambda)t}) \quad (4.2)$$

Финальные вероятности таковы:

$$p_0(\infty) = \frac{\mu}{\mu+\lambda} \quad (4.3)$$

$$p_1(\infty) = \frac{\lambda}{\mu+\lambda} \quad (4.4)$$

Эти вероятности имеют, физический смысл доли времени, которую в среднем система проводит в каждом из состояний в стационарном (установившемся) режиме. Логично, что доля времени рабочего состояния растет вместе с интенсивностью потока восстановлений, т. е. убывает с ростом среднего времени простоя. Аналогичные суждения верны и для доли времени простоя.

Пусть N – общее число узлов в системе, n – число узлов, которые должны работать, m – число резервных узлов (ненагруженный резерв), l – максимальное число параллельно восстанавливаемых узлов, S_i - состояние, когда неисправны i узлов. Пусть λ - пуассоновский поток отказов одного узла, а μ - пуассоновский поток восстановлений узла. Предположим, что в системе имеется резервирование узлов глубины m, причем резервные узлы не отказывают, а время подключения резерва пренебрежимо мало. Исправной

будем считать систему, в которой работает n узлов. Так же буде считать, что использовать более n мы не будем. Отказные состояния будем обозначать пунктиром.

Система отказывающих и восстанавливающихся узлов описывается схемой гибели и размножения (рис. 4.1, 4.2).

Из результатов теории систем массового обслуживания следует, что предельные (финальные) вероятности существуют и даются формулами Эрланга [20, 35, 45]:

$$a_k = \frac{\beta_k}{\gamma_k} \cdot \rho^k \quad ; \tag{4.5}$$

$$\rho = \lambda / \mu ; \tag{4.6}$$

$$b = \left(\sum_{i=0}^{N} a_i \right)^{-1} \quad ; \tag{4.7}$$

$$p_k = a_k \cdot b . \tag{4.8}$$

$$\rho = \lambda / \mu = t_{восст1} / t_{раб1} \tag{4.9}$$

(4.9) - соотношение времени восстановления одного узла и времени его безотказной работы.

Рассмотрим гомогенную (однородную) систему из N узлов в предположении, что каждый из узлов отказывает и восстанавливается вне зависимости от других (рис. 4.1). В данном случае

$$\beta_k = \frac{n!}{(n-k)!} \tag{4.10}$$

$$\gamma_k = k! \tag{4.11}$$

$$m = 0 \tag{4.12}$$

$$l \geq n + m \tag{4.13}$$

Можно заметить, что в соответствии с теорией комбинаторики

$$a_k = C_n^k \tag{4.14}$$

В случае если ремонт осуществляется одним рабочим (это абстрактное понятие, под которым может пониматься любой ресурс), решение имеет вид:

$$\beta_k = \frac{n!}{(n-k)!} \tag{4.15}$$

$$\gamma_k = 1 \tag{4.16}$$

$$m = 0 \tag{4.17}$$

$$l = 1 \tag{4.18}$$

Можно заметить, что в соответствии с теорией комбинаторики

$$a_k = A_n^k \tag{4.19}$$

Если ввести бригаду рабочих из нескольких человек, каждый из которых может обслуживать один из отказов, то поток восстановлений будет min(количество рабочих; количество отказавших узлов). Для большей адекватности модели мы введем в качестве параметра максимально возможное количество узлов, восстанавливающихся параллельно (т.е. количество рабочих, размер бригады). Как показала имитация, среднее число отказавших узлов может сильно зависеть от размера бригады. В этом случае:

$$\beta_k = \frac{n!}{(n-k)!} \tag{4.20}$$

$$\gamma_k = min(k, l)! \cdot l^{max(k,l)-l} \tag{4.21}$$

$$m = 0 \tag{4.22}$$

Для независимого восстановления узлов необходимо и достаточно, чтобы размер бригады был не меньше количества узлов. Это уже упоминалось ранее в одной из формул.

Все формулы можно получить, нарисовав размеченный граф состояний и применив решение схемы гибели и размножения. В силу большого размера эти вычисления здесь не приводятся.

Уже стало понятно, что знамениатель дроби γ отвечает за политику восстановления узлов. Вопрос : За что отвечает β ? За резервирование, которого пока еще нет в этой системе!

Данная модель не удобна тем, что не учитывается резервирование узлов. Дополним модель резервированием узлов [8, 19, 21]. Случай с резервированием также сводится к схеме гибели и размножения (рис. 4.2):

$$\beta_k = \frac{n^{min(m,k)} \cdot n!}{min(n+m-k, n)!} \quad (4.23)$$

Почему здесь нет γ ? Потому что мы уже определили его ранее.

Нам необходимо определить коэффициент готовности (по определению, это сумма вероятностей всех работоспособных состояний):

$$K = \sum_{i=0}^{m} p_i \quad (4.24)$$

Физический смысл предельных вероятностей в случае нескольких узлов такой же, как и в случае одного узла.

Среднее количество отказавших узлов можно рассчитать по формуле:

$$Broken = \sum_{i=0}^{N} p_i \cdot i = \sum_{i=1}^{N} p_i \cdot i \quad , \quad (4.25)$$

В случае если узлы отказывают и восстанавливаются независимо

$$Broken = \frac{N \cdot \lambda}{\mu + \lambda} \quad (4.26)$$

Эту формулу можно получить, зная, что все узлы независимы => $< \sum_k Broken_k >= \sum_k < Broken_k >$ (среднее суммы величин равно сумме средних).

Запишем полную формулу для модели с резервированием и параметризированным количеством линий ремонта:

$$a_k = \frac{n^{min(m,k)} \cdot n!}{min(n+m-k, n)! \cdot min(k, l)! \cdot l^{max(k,l)-l}} \cdot \rho^k \quad ; \quad (4.27)$$

$$\rho = \lambda / \mu ; \quad (4.28)$$

$$b = \left(\sum_{i=0}^{N} a_i \right)^{-1} \quad ; \quad (4.29)$$

$$p_k = a_k \cdot b . \quad (4.30)$$

$$\rho = \lambda / \mu = t_{восст1} / t_{раб1} \quad (4.31)$$

Поток переходов из состояния работы в состояние отказа для m узлов во всех случаях в m раз больше, чем в первом. Поэтому среднее время пребывания в состоянии работы становится в m раз меньше, т.е.

$t_{работ} = 1/m\lambda = t_{раб}/m$. Таким образом, можно сделать вывод, что для задачи, выполняющейся на m узлах, должно выполняться условие $t_{работ} >> t_{обслуж}$, где $t_{обслуж}$- время обслуживания задания. Иначе его можно записать так: $t_{раб}/m >> t_{обслуж}$. Если выполнять задание последовательно на одном узле ($t_{посл.обс} = mt_{обслуж}$), то вытекает условие $t_{раб} >> t_{посл.обс} = mt_{обслуж}$. Получается та же самая оценка.

Рассчитаем среднее время безотказной работы и среднее время отказного состояния в условиях стационарности. Давайте решим более общую задачу. Пусть t_k - среднее время пребывания в состоянии, когда отказало не более k узлов, $\bar{t}_k$ - среднее время пребывания в состоянии, когда отказало более k узлов. Это можно сделать по следующим формулам:

$$t_k = \frac{\sum_{i=0}^{k} p_i}{min(n, n+m-k)\cdot\lambda\cdot p_k} = \frac{\sum_{i=0}^{k} p_i}{min(n, n+m-k)\cdot p_k}\cdot t_{раб1} \qquad (4.32)$$

$$\bar{t}_k = \frac{\sum_{i=k+1}^{n+m} p_i}{min(l, k+1)\cdot\mu\cdot p_{k+1}} = \frac{\sum_{i=k+1}^{n+m} p_i}{min(l, k+1)\cdot p_{k+1}}\cdot t_{отк1} \qquad (4.33)$$

Эти формулы можно получить следующим образом. Определим по графу среднюю интенсивность перехода из одной группы состояний в другую группу и назад. В силу стационарного режима $min(n, n+m-k)\cdot\lambda\cdot p_k = min(l, k+1)\cdot\mu\cdot p_{k+1}$. Это знаменатели дробей. Среднее время между событиями этого потока есть среднее время цикла, включающее время пребывания в обоих состояниях. Доля времени в каждой группе состояний есть вероятность пребывания в этом состоянии.

Также давайте отметим, что $t_{раб} = t_m$ и $t_{отк} = \bar{t}_m$.

Дополним анализом неполноты контроля [8].

Пусть имеется система из 2 работающих узлов и 1 резервного (рис. 4.3). В системе действует пуассоновский поток отказов интенсивности λ, поток

восстановлений интенсивности μ, поток обнаружения необнаруженных отказов υ. При этом вероятность необнаружения отказа p.

Сразу же после обнаружения отказа отказавший узел заменяется резервным, если таковой имеется. Замена происходит мгновенно. Для краткости будем обозначать состояние так (i;j), где i — число обнаруженных отказавших узлов, j — число необнаруженных отказавших узлов. Также введем обозначения: поток обнаруживаемых отказов $\eta = (1-p)\cdot\lambda$ и поток необнаруживаемых отказов $\gamma = p\cdot\lambda$. Будем считать, что в любой момент времени могут восстанавливаться сразу все узлы.

Нас интересуют следующие параметры системы:

1) коэффициент готовности;

2) среднее время рабочего состояния;

3) среднее время отказного состояния.

Коэффициент готовности рассчитывается как сумма вероятностей рабочих состояний. В нашем случае, К = P(0;0) + P(1;0).

В большинстве источников вопрос резервирования рассматривается с допущением, что переключение отказавшей линии/узла на резерв происходит мгновенно. Также обычно делается допущение, что сами контрольные элементы, осуществляющие переключение, работают с абсолютной надёжность, т. е. без отказов. Это не всегда верно, потому что в некоторых случаях надёжность элемента управления может быть сопоставима с надёжность самого элемента.

Восстановленный узел либо ставится на подключение, либо ставится в резерв. Будем считать, что последнее происходит мгновенно. Введем для системы поток подключения резерва и поток тайм-аута. Тайм-аут – это время, за которое система должна подключить исправный канал вместо отказавшего. Если это не произойдет, система переходит в отказное состояние. Здесь мы сразу же сталкивается с непуассоновскими потоками переходов.

Пусть состояние (i;j) - это состояние, когда i функционирует, а j узлов зарезервировано. Звёздочкой обозначим состояние, когда истек тайм-аут. υ^+ - поток подключения узла до истечения тайм-аута, υ^- - поток истечения тайм-аута, υ^* - поток подключения после истечения тайм-аута. Эти потоки не являются пуассоновскими (рис. 4.4).

Аналитический расчёт становится невозможен или сильно затруднён.

Графы на рис. 4.1 и рис. 4.2 являются классическими схемами гибели и размножения. Они одномерны и описываются формулами Эрланга. Графы на рис. 4.3 и рис. 4.4, соответственно, двухмерен и трехмерен. Под n-мерностью графа подразумевается следующее. На каждое новое состояние узла появляется новое измерение (так проще представить себе граф).

Можно сделать вывод, добавление одного из следующих пунктов:

- неполнота контроля,
- конечное время подключения резерва,
- тайм-аут для подключения резерва

приводит к добавлению к графу одного измерения.

Итого получаем в обобщённом случае, заявленном в имитационной модели, четырёхмерный граф. Если отказаться от потока тайм-аута, то граф станет трёхмерным. Мы не только упростим граф, но и избавимся от непуассоновских потоков. Аналитический расчёт возможен с допущением пуассоновости всех остальных потоков.

Однако трёхмерный граф сложно даже представить. А ведь надо ещё составить систему уравнений Колмогорова и решить её.... не только составить и решить, но и составить и решить её правильно.

Именно поэтому мы отказываемся от расчёта полной аналитической модели. Но мы рассчитаем несколько частностей.

Имитационная модель включает в себя ВСЕ вышеописанные аспекты. Соответствующее аналитическое описание (даже с упрощениями) довольно трудно, но имитационная модели этого практически «не чувствует».

Каждое состояние будем описывать тремя числами из четырёх возможных (четвёртое число излишне, т. к. дается функцией от всех остальных и числа узлов в системе):

- кол-во отказавших и обнаруженных узлов;
- кол-во отказавших и необнаруженных узлов;
- кол-во рабочий узлов;
- размер резерва.

Так же необходим маркер, указывающий успел ли истечь тайм-аут, если он имеет смысл.

Будут следующие переходы:

- обнаруженные отказы;
- необнаруженные отказы;
- восстановление работоспособности;
- обнаружение отказа;
- подключение резерва;
- истечение тайм-аута.

Последний не всегда имеет смысл. Если все узлы независимы и используются для вычисления фрагментов задания, то отказ снижает производительные мощности. Когда нас интересует именно скорость выполнения заданий, нет нужды вводить границу отказ-работоспособность. Если же речь идёт о линии связи, то разрыв связи более, чем на некоторый период, имеет природу отказа.

А сейчас мы рассчитаем пример с помощью аналитической модели и с помощью имитационной, чтобы можно было сопоставить аналитическую

модель с имитационной. Все расчёты будем вести по рис. 4.2 и формулам (4.24) — (4.25) и (4.27) — (4.33).

Для сравнения рассмотрим систему с $n=256$, $m=7$, $\lambda=\frac{1}{100}час^{-1}$, $\mu=1\,час^{-1}$, $l\geq 256$.

Рассчитаем аналитическим методом значения K , $t_{раб}$, $t_{отк}$. Также проведем имитационный эксперимент при моделировании периода в 500 дн., совершив 200 испытаний модели. Это дает погрешность не более 5%.

Результаты расчётов представления в табл. 4.1. Обратите внимание, что, чтобы получить эти данные, не обязательно рассчитывать все параметры системы.

Таблица 4.1 – Сравнение аналитического и имитационного решения

Аналитическое решение			Имитационное решение		
K	$t_{раб}$	$t_{отк}$	K	$t_{раб}$	$t_{отк}$
0,995133	35,1708	0,172018	0,9951	35,5273	0,1731

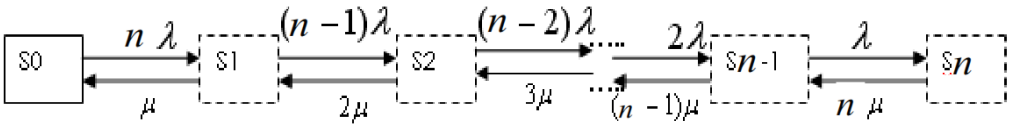


Рисунок 4.1 — Размеченный граф состояний системы без резервирования и с неограниченным восстановлением

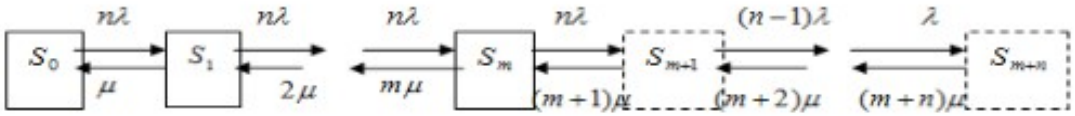


Рисунок 4.2 — Размеченный граф состояний системы с резервированием и неограниченным восстановлением

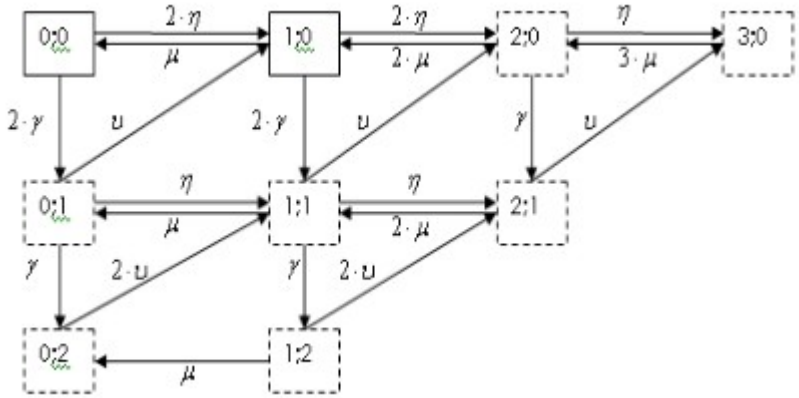


Рисунок 4.3 — Размеченный граф состояний системы с неполнотой контроля и неограниченным восстановлением

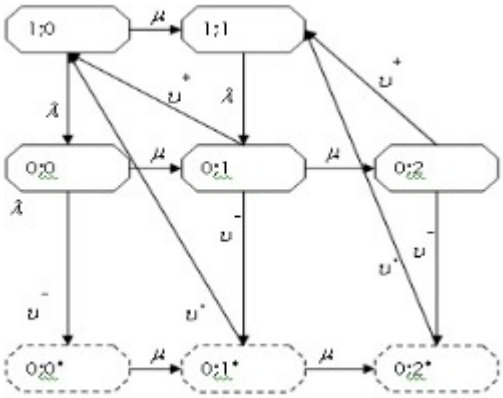


Рисунок 4.4 — Размеченный граф состояний системы с конечным временем подключения и тайм-аутом

5 Результат разработки

В результате проделанной работы нами было получено средство оценки стратегий распределения заданий по узлам грида.

На данный момент существует программный продукт, осуществляющий сравнение различных стратегий распределения заданий и комбинаций этих стратегий. Новым в данном продукте является то, что он учитывает отказы элементов системы и неполноту контроля.

Была произведена качественная проверка работы программного продукта на реальной системе, использующей GridGain, которая дала соответствие между моделируемыми данными и полученными экспериментально.

Также необходимо сказать, что программа прошла государственную регистрацию [16], ей были посвящены статьи в журналах «ИТМУ» [12] и «Известия ВолгГТУ» [4] и выступление на международной научно-практической конференции [11]. Свидетельство о государственной регистрации представлено на рис. 5.1. По результатам исследований опубликована монография [35].

Модификации, заявленные в [12], выполнены на две трети. Сделано:

- Улучшение оценки надёжности системы: учёт неполноты контроля отказов узлов;
- Добавление новых стратегий распределения заданий.

От этой идеи пришлось отказаться из-за отказа от MPI.NET:

- Создание параллельной версии программы для распараллеливания вычислений на нашем университетском кластере.

Стоит отметить, что полученный алгоритм является почти универсальным. Его не обязательно использовать для моделирования распределения заданий в грид-системе. Также этот алгоритм может быть использован при распределении любого вида нагрузка между некоторыми центрами, например, при распределении товара между торговыми точками.

Аналогом задания является товар, аналогом расчетов – процесс продажи, очередь – количество товаров на складе магазина и т. д. С небольшими изменениями его можно обобщить на многие однотипные задания.

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2010610693

Имитационная модель грид-системы (GridModel)

Правообладатель(ли): *Государственное образовательное учреждение высшего профессионального образования Волгоградский государственный технический университет (RU)*

Автор(ы): *Лукьянов Виктор Сергеевич, Жариков Дмитрий Николаевич, Гаевой Сергей Владимирович, Шафран Юрий Владимирович (RU)*

Заявка № 2009616616

Дата поступления 23 ноября 2009 г.

Зарегистрировано в Реестре программ для ЭВМ

20 января 2010 г.

Руководитель Федеральной службы по интеллектуальной собственности, патентам и товарным знакам

Б.П. Симонов

Рисунок 5.1 – Свидетельство о Государственной регистрации программы «Имитационная модель грид-систем (GridModel)»

Впоследствии была разработана более подробная имитационная модель кластера Грид-системы [39] (рис. 5.2).

РОССИЙСКАЯ ФЕДЕРАЦИЯ

ЭВМ
22 пр/13
1983

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2013614201

Имитационная модель для оценки влияния параметров надежности и иных характеристик на производительность кластерной системы (SrvModel)

Правообладатель(ли): ***Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования «Волгоградский государственный технический университет» (RU)***

Автор(ы): ***Гаевой Сергей Владимирович (RU),***
Лукьянов Виктор Сергеевич (RU)

Заявка № **2013611975**
Дата поступления **15 марта 2013 г.**
Зарегистрировано в Реестре программ для ЭВМ
25 апреля 2013 г.

Руководитель Федеральной службы
по интеллектуальной собственности

Б.П. Симонов

ФЕДЕРАЛЬНАЯ СЛУЖБА ПО ИНТЕЛЛЕКТУАЛЬНОЙ СОБСТВЕННОСТИ (РОСПАТЕНТ) ОГРН 1047730015200 ИНН 7730176088 КПП 770901001

Рисунок 5.2 – Свидетельство о Государственной регистрации программы «Имитационная модель для оценки влияния параметров надёжности и иных характеристик на производительность кластерной системы (SrvModel) »

6 Лабораторная работа №1

6.1 Тема работы

Исследование надёжности гомогенной системы путём имитационного моделирования. Исследование аналитической модели.

6.2 Цель работы

Целью данной работы является получение навыков имитационного моделирования, знакомство со средой GridModel, улучшение понимания аналитического моделирования и его недостатков.

6.3 Рекомендации

Данная работа посвящена надёжности, поэтому рекомендуем ознакомиться с [2], [3], [9], [10], [19], [21], [35]. Для понимания основных принципов имитационного моделирования как такового можно обратиться к [20], [23] или [45].

Рекомендуем вспомнить:

- основы теории вероятности;
- неравенства и теорема Чебышёва, теорема Бернулли;
- центральная предельная теорема;
- коэффициенты Стьюдента;
- марковские процессы с дискретными состояниями и непрерывным временем;
- пуассоновские потоки событий;
- агрегативное представление систем;

- моделирование случайной величины, распределённой по определённому закону, в т. ч. по экспоненциальному, нормальному, равномерному.

6.4 Краткое описание работы

В данной работе студентам предлагается рассчитать несколько примеров моделей надёжности. Где это возможно, предлагается сравнить результаты имитационных экспериментов и аналитической модели.

Производится оценка таких показателей надёжности, как:

- коэффициент готовности;
- среднее время рабочего состояния;
- среднее время отказного состояния;
- среднее число отказавших узлов[21];
- среднее число отказавших и обнаруженных узлов;
- средний размер резерва;
- среднее число рабочих узлов;
- реальный коэффициент готовности, он же — коэффициент использования (меньше на долю времени допустимого технического обслуживания, это время, приходящееся на интервал допустимого простоя).

На основе этих параметров можно найти некоторые другие, например:

- среднее число отказавших, но необнаруженных узлов;
- доля времени, проводимая в состоянии допустимого простоя.

21 Обратите внимание, что в программе есть два показателя надёжности: «Среднее число отказавших узлов во всей системе» и «Среднее число отказавших узлов в блоке исполнения». Нас будет интересовать только первый параметр. Это число нерабочих блоков во всей системе. Второй параметр — это число нефункционирующих линий исполнения. Обратите внимание, что теоретически второй параметр МОЖЕТ быть больше первого, если восстановление производится быстрее, чем подключение. Всё сказанное выше справедливо и для «Среднего числа отказавших и обнаруженных узлов».

При определении параметра путём имитационного моделирования необходимо помнить, что необходимо определять параметр с погрешностью 5% с квантилем, обеспечивающим вероятность попадания не менее 99% (при большом числе прогонок это ≈ 3.0). Более подробно это описано в центральной предельной теореме и коэффициентах Стьюдента.

Появление в результатах бесконечности или NaN (стандарт IEEE 754) говорит о невозможности определения параметра на данном интервале моделирования — необходимо увеличить интервал. Допускается снижать требуемую точность (т. е. число прогонок), если время моделирования слишком велико.

6.5 Практическая часть

1) Получите у преподавателя вариант из таблицы 6.1. В большинстве практических случаев время безотказной работы распределено по экспоненциальному закону, по нормальному или закону Вейбула. Время восстановления и подключения резерва обычно распределено по нормальному закону. Также часто фигурирует закон Эрланга. В заданиях эти правила не выполняются. Также не выполняется реальное соотношение времени работы и времени восстановления. Это сделано по многим причинам. Во-первых, потому, что примеры подобраны так, чтобы время моделирования было не очень большим. Во-вторых, потому, что задания должны быть разнообразными.
2) Откройте программу GridModel и разберитесь, как осуществляются различные виды моделирования. Описание работы программы дано в руководстве пользователя. Чтобы открыть руководство пользователя нажмите F1.

3) Моделирование системы без неполноты контроля, времени подключения резерва и допустимого времени простоя (стандартная модель надёжности):
   a) Введите в среду имитационного моделирования параметры число элементов в системе (n), размер ненагруженного резерва (m), число линий восстановления (l), закон распределения времени рабочего состояния ($t_{раб1}$), закон распределения времени восстановления ($t_{восст1}$). Вероятность необнаружения ($p_{необн}$) и допустимое время (t_{out}) простоя отказа положите равными нулю. Закон подключения резерва ($t_{\text{подкл1}}$) назначьте К(0). Закон обнаружения необнаруженных отказов ($t_{\text{обн1}}$) пока не используется, но можете сразу перенести его из таблицы в программу.
   b) Проведите моделирование на малом числе прогонок, чтобы определить требуемое число прогонок. Учтите, что требуемое время растет примерно линейно с увеличением числа прогонок.
   c) Проведите имитационное моделирование с требуемой (или ослабленной) точностью и зафиксируйте результаты.
   d) Осуществите аналитический расчёт, используя прилагаемую аналитическую модель. Необходимо предполагать все потоки событий пуассоновскими, т. е. все законы распределения будут показательными. Программа принимает в stdin шесть чисел: количество работающих узлов, размер резерва, количество линий восстановления, среднее время работы одного узла, среднее время отказа (восстановления) одного узла, количество цифр после запятой в результате. Чтобы перенаправить потоки ввода/вывода в файлы, можно воспользоваться средствами операционной системы:

a.exe < stdin.txt > stdout.txt 2> stderr.txt

Можно перенаправить лишь некоторые потоки. Чтобы убрать файл, уберите его имя и команду перенаправления, т. е. <, > или 2>.

e) Проведите имитационный расчёт, полагая все законы ($t_{раб1}$ и $t_{восст1}$) показательными (сохранится только матожидание, а СКО и прочие параметры утрачиваются)[22]. Для такого изменения достаточно просто изменить типы законов в выпадающем списке. После окончания расчётов не забудьте (!) вернуть старые типы законов распределения.

f) Сравните три полученных результата и сделайте выводы.

g) Оформите результаты в виде таблицы и включите в отчёт.

4) Изменяйте допустимое время простоя 0%[23], 10%, 30%, 50%, 90%, 100%, 150%, 200% от времени простоя, указанного в таблице 6.1. Возможно, придётся нарастить интервал моделирования. Если он станет слишком большим, то сократите число прогонок. (Пусть Вас не смущает рост среднего времени отказа с ростом допустимого времени простоя, ведь в первую очередь исчезают самые короткие отказы). Оформите данные в таблицу. Сделайте выводы. Как соотносятся коэффициент готовности и реальный коэффициент готовности (он же коэффициент использования), как они изменяются? Постройте график зависимости одного-трех параметров по выбору преподавателя от допустимого времени простоя[24].

22 Зачем это? Предположим, что есть некоторое число — реальный результат. Имитационное моделирование дает его с некоторой погрешностью, описываемой коэффициентами Стьюдента и центральной предельной теоремой. Аналитическая модель дает погрешность из-за пренебрежения «непуассоновостью» потоков событий. Т. е. результаты разнятся из-за двух видов погрешностей. Объявив всех потоки в имитационной модели пуассоновскими, мы сможем наблюдать только погрешность самой имитационной модели. Вообще-то, есть ещё третий вид погрешности — погрешность стандарта IEEE 754, но она присутствует всегда, и мы её касаться не будем. Только упомянем, что везде в подсчётах используется формат double.

23 Этот результат уже получен в 3c).

24 В качестве таких параметров рекомендую брать среднее время рабочего состояния и среднее время отказного состояния.

5) Установите допустимое время простоя в нуль и закон подключения резерва — К(0). Установите закон времени обнаружения отказа из табл. 6.1. Промоделируйте для вероятности необнаружения отказа 0%[25], 1%, 5%, 10%, 20%, 50%, 70%, 100%. Учтите, что программа принимает вероятность необнаржунеия отказа уже в процентах и никакие преобразования не требуются. Постройте график зависимости одного-трех параметров по выбору преподавателя от вероятности необнаружения.
6) Установите вероятность необнаружения отказа в 0%. Возьмите закон распределения времени подключения резерва и допустимое время простоя из табл. 6.1, введите их в программу. Осуществите моделирование, зафиксируйте результаты.
7) Осуществите полное моделирование, установив ВСЕ параметры из табл. 6.1, получите результаты и зафиксируйте их.

6.6 Содержание отчёта

- Тема и цель работы;
- Номер варианта и данные для индивидуального задания;
- Сводная таблица результатов имитационной модели, аналитической модели и имитационной модели с пуассоновскими потоками заданий;
- Таблица зависимости параметров от допустимого времени простоя;
- Графики зависимости нескольких параметров от допустимого времени простоя;
- Таблица зависимости параметров от вероятности необнаружения отказа;
- Графики зависимости нескольких параметров от вероятности необнаружения отказа;

25 И этот результат уже получен в 3с).

– Данные из задания 6);

– Данные из задания 7).

Таблица 6.1 — Варианты заданий для лабораторной работы

Вариант	n	m	l	$t_{раб1}$	$t_{восст1}$	$p_{необн}$	$t_{обн1}$	$t_{подкл1}$	t_{out}
1	20	3	20	П(700)	Н(10;3)	5%	П(1)	К(4)	5
2	10	1	4	Т(700;300)	Р(10;4,5)	15%	Л(1)	П(5)	5
3	15	4	3	Р(700;300)	П(12)	1%	Н(1;0,1)	Р(3;2)	4
4	12	1	7	Н(900;300)	Н(12;4)	20%	Э(1,1;2)	Н(3;1)	7
5	11	2	3	Э(800;5)	Р(12;7)	17%	В(0,9; 0,9)	Л(4)	3
6	10	2	3	Л(750)	Т(10;5)	11%	Р(0,9;1,1)	Э(3;3)	4,5
7	15	4	4	Э(750;3,3)	Л(11)	9%	Т(1;0,1)	П(4)	6
8	7	2	1	В(800;4)	Э(14;5)	10%	К(1)	К(5)	5
9	19	1	5	Л(900)	Л(12)	8%	Л(1)	П(6)	7
10	22	3	2	Н(900;300)	В(13;2)	2%	П(1,1)	Л(5)	4
11	10	3	1	Т(800;800)	Т(12;12)	13%	Т(1;0,2)	Т(5;5)	7
12	12	1	7	В(900;0,4)	Э(10;4)	9%	Н(1; 0,07)	Н(6;2)	6
13	10	5	1	В(900;4)	В(10;0,1)	11%	Э(1;2)	Э(5;10)	8
14	15	1	3	П(700)	Н(15;5)	7%	Р(1;0,9)	П(7)	7
15	11	2	3	Р(800;100)	Р(12;3)	8%	П(0,9)	Р(10;2)	9
16	9	5	2	Н(700;200)	Р(10;7)	13%	Л(1,1)	В(11;0,2)	11
17	10	2	3	Р(700;300)	Н(11;3)	11%	П(0,8)	П(12)	10
18	7	3	2	В(800;8)	В(12;0,8)	9%	Н(0,9;0,3)	Э(11; 1)	8
19	14	1	2	Э(750;5)	Э(15;2)	14%	Э(1;1)	В(12; 3,3)	14
20	15	5	1	Т(800;700)	К(12)	12%	П(1,2)	П(11)	10
21	11	4	2	Н(700;150)	Т(15;12)	10%	Т(1;1)	Р(5;2)	5
22	12	4	3	В(800;0,5)	Т(12;12)	7%	Л(0,9)	П(7)	4
23	9	5	1	Т(700;550)	Э(12; 4)	8%	П(0,9)	Н(9;3)	10
24	11	1	5	Р(700;550)	В(12; 4)	9%	Л(1,2)	Р(9;3)	10
25	10	2	2	Н(800;250)	Р(10;7)	12%	В(1;3)	Л(10)	4
26	9	3	2	Л(900)	Л(11)	13%	П(1)	Н(15;5)	10

Вариант	n	m	l	$t_{раб1}$	$t_{восст1}$	$p_{необн}$	$t_{обн1}$	$t_{подкл1}$	t_{out}
27	12	1	7	Р(900;100)	Н(10;3)	10%	Л(1)	П(12)	11
28	10	5	1	В(800;0,8)	Э(11;3)	11%	Н(1;0,01)	Т(11;1)	11,5
29	8	1	1	Э(700;7)	Л(12)	9%	Р(1;0,1)	В(10; 1,1)	10,1
30	10	2	1	В(800;7)	Э(12;1)	12%	Э(1;4)	Л(11)	15
31	20	1	3	Н(900;300)	Р(12;7)	11%	Т(1;0,1)	К(5)	7
32	10	4	7	Э(800;5)	Т(10;5)	9%	К(1)	П(6)	4
33	15	1	3	Л(750)	Л(11)	10%	Л(1)	Л(5)	7
34	12	2	3	Э(750;3,3)	Э(14;5)	8%	П(1,1)	Т(5;5)	6
35	11	2	4	В(800;4)	Л(12)	2%	Т(1;0,2)	Н(6;2)	8
36	10	4	1	Л(900)	В(13;2)	13%	Н(1; 0,07)	Э(5;10)	7
37	15	2	5	Н(900;300)	Т(12;12)	9%	Э(1;2)	П(7)	9
38	7	1	2	Т(800;800)	Э(10;4)	11%	Р(1;0,9)	Р(10;2)	11
39	19	3	1	В(900;0,4)	В(10;0,1)	7%	П(0,9)	В(11;0,2)	10
40	22	3	7	В(900;4)	Н(15;5)	8%	Л(1,1)	П(12)	8
41	10	1	1	П(700)	Р(12;3)	13%	П(0,8)	Э(11; 1)	14
42	12	5	3	Р(800;100)	Р(10;7)	11%	Н(0,9;0,3)	В(12; 3,3)	10
43	10	1	3	Н(700;200)	Н(11;3)	9%	Э(1;1)	П(11)	5
44	15	2	2	Р(700;300)	В(12;0,8)	14%	П(1,2)	Р(5;2)	4
45	11	5	3	В(800;8)	Э(15;2)	12%	Т(1;1)	П(7)	10
46	9	2	2	Э(750;5)	К(12)	10%	Л(0,9)	Н(9;3)	10
47	10	3	2	Т(800;700)	Т(15;12)	7%	П(0,9)	Р(9;3)	4
48	7	1	1	Н(700;150)	Т(12;12)	8%	Л(1,2)	Л(10)	10
49	14	5	2	В(800;0,5)	Э(12; 4)	9%	В(1;3)	Н(15;5)	11
50	15	4	3	Т(700;550)	В(12; 4)	12%	П(1)	П(12)	11,5
51	11	4	1	Р(700;550)	Р(10;7)	13%	Л(1)	Т(11;1)	10,1
52	12	5	5	Н(800;250)	Л(11)	10%	Н(1;0,01)	В(10; 1,1)	15
53	9	1	2	Л(900)	Н(10;3)	11%	Р(1;0,1)	Л(11)	5
54	11	2	2	Р(900;100)	Э(11;3)	9%	Э(1;4)	К(4)	5
55	10	3	7	В(800;0,8)	Л(12)	12%	П(1)	П(5)	4
56	9	1	1	Э(700;7)	Э(12;1)	5%	Л(1)	Р(3;2)	7
57	12	5	1	В(800;7)	Н(10;3)	15%	Н(1;0,1)	Н(3;1)	3

Вариант	n	m	l	$t_{раб1}$	$t_{восст1}$	$p_{необн}$	$t_{обн1}$	$t_{подкл1}$	t_{out}
58	10	1	1	Н(900;300)	Р(10;4,5)	1%	Э(1,1;2)	Л(4)	4,5
59	8	2	3	Э(800;5)	П(12)	20%	В(0,9; 0,9)	Э(3;3)	6
60	10	3	7	Л(750)	Н(12;4)	17%	Р(0,9;1,1)	П(4)	5
61	20	4	3	Э(750;3,3)	Л(12)	13%	Э(1;2)	Р(10;2)	10
62	10	1	3	В(800;4)	В(13;2)	9%	Р(1;0,9)	В(11;0,2)	8
63	15	2	4	Л(900)	Т(12;12)	11%	П(0,9)	П(12)	14
64	12	2	1	Н(900;300)	Э(10;4)	7%	Л(1,1)	Э(11; 1)	10
65	11	4	5	Т(800;800)	В(10;0,1)	8%	П(0,8)	В(12; 3,3)	5
66	10	2	2	В(900;0,4)	Н(15;5)	13%	Н(0,9;0,3)	П(11)	4
67	15	1	1	В(900;4)	Р(12;3)	11%	Э(1;1)	Р(5;2)	10
68	7	3	7	П(700)	Р(10;7)	9%	П(1,2)	П(7)	10
69	19	3	1	Р(800;100)	Н(11;3)	14%	Т(1;1)	Н(9;3)	4
70	22	1	3	Н(700;200)	В(12;0,8)	12%	Л(0,9)	Р(9;3)	10
71	10	5	3	Р(700;300)	Э(15;2)	10%	П(0,9)	Л(10)	11
72	12	1	2	В(800;8)	К(12)	7%	Л(1,2)	Н(15;5)	11,5
73	10	2	3	Э(750;5)	Т(15;12)	8%	В(1;3)	П(12)	10,1
74	15	5	2	Т(800;700)	Т(12;12)	9%	П(1)	Т(11;1)	15
75	11	2	2	Н(700;150)	Э(12; 4)	12%	Л(1)	В(10; 1,1)	5
76	9	3	1	В(800;0,5)	В(12; 4)	13%	Н(1;0,01)	Л(11)	5
77	10	1	2	Т(700;550)	Р(10;7)	10%	Р(1;0,1)	К(4)	4
78	7	5	3	Р(700;550)	Л(11)	11%	Э(1;4)	П(5)	7
79	14	4	1	Н(800;250)	Н(10;3)	9%	П(1)	Р(3;2)	3
80	15	4	5	Л(900)	Э(11;3)	12%	Л(1)	Н(3;1)	4,5
81	11	5	2	Р(900;100)	Л(12)	5%	Н(1;0,1)	Л(4)	6
82	12	1	2	В(800;0,8)	Э(12;1)	15%	Э(1,1;2)	Э(3;3)	5
83	9	2	7	Э(700;7)	Н(10;3)	1%	В(0,9; 0,9)	П(4)	7
84	11	3	1	В(800;7)	Р(10;4,5)	20%	Р(0,9;1,1)	К(5)	4
85	10	1	1	Н(900;300)	П(12)	17%	Т(1;0,1)	П(6)	7
86	9	5	1	Э(800;5)	Н(12;4)	11%	К(1)	Л(5)	6
87	12	1	3	Л(750)	Р(12;7)	9%	Л(1)	Т(5;5)	8
88	10	2	7	Н(900;300)	Т(10;5)	10%	П(1,1)	Н(6;2)	7

Вариант	n	m	l	$t_{раб1}$	$t_{восст1}$	$p_{необн}$	$t_{обн1}$	$t_{подкл1}$	t_{out}
89	8	3	1	Э(800;5)	Л(11)	8%	Т(1;0,2)	Э(5;10)	9
90	10	1	4	Л(750)	Э(14;5)	2%	Н(1; 0,07)	П(7)	11
91	20	1	4	Н(900;300)	В(10;0,1)	13%	Э(1;1)	П(7)	4
92	10	2	1	Т(800;800)	Н(15;5)	11%	П(1,2)	Н(9;3)	10
93	15	2	5	В(900;0,4)	Р(12;3)	9%	Т(1;1)	Р(9;3)	11
94	12	4	2	В(900;4)	Р(10;7)	14%	Л(0,9)	Л(10)	11,5
95	11	2	1	П(700)	Н(11;3)	12%	П(0,9)	Н(15;5)	10,1
96	10	1	7	Р(800;100)	В(12;0,8)	10%	Л(1,2)	П(12)	15
97	15	3	1	Н(700;200)	Э(15;2)	7%	В(1;3)	Т(11;1)	5
98	7	3	3	Р(700;300)	К(12)	8%	П(1)	В(10; 1,1)	5
99	19	1	3	В(800;8)	Т(15;12)	9%	Л(1)	Л(11)	4
100	22	5	2	Э(750;5)	Т(12;12)	12%	Н(1;0,01)	К(4)	7
101	10	1	3	Т(800;700)	Э(12; 4)	13%	Р(1;0,1)	П(5)	3
102	12	2	2	Н(700;150)	В(12; 4)	10%	Э(1;4)	Р(3;2)	4,5
103	10	5	2	В(800;0,5)	Р(10;7)	11%	П(1)	Н(3;1)	6
104	15	2	1	Т(700;550)	Л(11)	9%	Л(1)	Л(4)	5
105	11	3	2	Р(700;550)	Н(10;3)	12%	Н(1;0,1)	Э(3;3)	7
106	9	1	3	Н(800;250)	Э(11;3)	5%	Э(1,1;2)	П(4)	4
107	10	5	1	Л(900)	Л(12)	15%	В(0,9; 0,9)	К(5)	7
108	7	4	5	Р(900;100)	Э(12;1)	1%	Р(0,9;1,1)	П(6)	6
109	14	4	2	В(800;0,8)	Н(10;3)	20%	Т(1;0,1)	Л(5)	8
110	15	5	2	Э(700;7)	Р(10;4,5)	17%	К(1)	Т(5;5)	7
111	11	1	7	В(800;7)	П(12)	11%	Л(1)	Н(6;2)	9
112	12	2	1	Н(900;300)	Н(12;4)	9%	П(1,1)	Э(5;10)	11
113	9	3	1	Э(800;5)	Р(12;7)	10%	Т(1;0,2)	П(7)	10
114	11	1	1	Л(750)	Т(10;5)	8%	Н(1; 0,07)	Р(10;2)	8
115	10	5	3	Н(900;300)	Л(11)	2%	Э(1;2)	В(11;0,2)	14
116	9	1	7	Э(800;5)	Э(14;5)	13%	Р(1;0,9)	П(12)	10
117	12	2	1	Л(750)	Л(12)	9%	П(0,9)	Э(11; 1)	5
118	10	3	4	Э(750;3,3)	В(13;2)	11%	Л(1,1)	В(12; 3,3)	4
119	8	1	3	В(800;4)	Т(12;12)	7%	П(0,8)	П(11)	10

Вариант	n	m	l	$t_{раб1}$	$t_{восст1}$	$p_{необн}$	$t_{обн1}$	$t_{подкл1}$	t_{out}
120	10	4	3	Л(900)	Э(10;4)	8%	Н(0,9;0,3)	Р(5;2)	10
121	8	4	3	Т(700;550)	Н(10;3)	8%	П(0,8)	Н(9;3)	5
122	10	2	2	Р(700;550)	Р(10;4,5)	2%	Н(0,9;0,3)	Р(9;3)	5
123	20	1	3	Н(800;250)	П(12)	13%	Т(1;0,2)	Л(10)	4
124	10	3	2	Л(900)	Н(12;4)	9%	Н(1; 0,07)	Н(15;5)	7
125	15	3	2	Р(900;100)	Р(12;7)	11%	Э(1;1)	П(12)	3

6.7 Контрольные вопросы

1 Сформулируйте цель работы.
2 Какие параметры надёжности мы определяли в данной работе?
3 В чем отличие коэффициента готовности от реального коэффициента готовности?
4 Как связаны среднее время работы системы, среднее время отказного состояния и коэффициент готовности?
5 В чем недостатки и достоинства аналитической модели?
6 Насколько сильно коррелируют результаты аналитический и имитационной модели?
7 В чем недостатки и достоинства имитационной модели?
8 О чем говорит центральная предельная теорема?
9 Как это используется в данной работе?
10 Почему аналитический расчёт присутствует только в первом задании?
11 Каким законом распределения даются все величины в аналитической модели?
12 Что такое пуассоновские потоки событий и марковские процессы?

7 Лабораторная работа №2

7.1 Тема работы

Исследование различных стратегий распределения задач в гетерогенной среде путём имитационного моделирования. Поиск оптимальной стратегии. Анализ полученных результатов.

7.2 Цель работы

Целью данной работы является получение навыков имитационного моделирования, знакомство со средой GridModel, улучшение понимания имитационного моделирования, понимания механизмов распределения заданий в гетерогенной среде, качественная оценка стратегий.

7.3 Рекомендации

Данная работа посвящена моделированию распределения задач, поэтому рекомендуем ознакомиться с [12], [14], [24], [35], [43], [44]. Для понимания основных принципов имитационного моделирования как такового можно обратиться к [20], [23] или [45].

Рекомендуем вспомнить:

- основы теории вероятности;
- неравенства и теорема Чебышёва, теорема Бернулли;
- центральная предельная теорема;
- коэффициенты Стьюдента;
- пуассоновские потоки событий;
- системы массового обслуживания (одноканальные, многоканальные, без очереди, с ограниченной очередью, с неограниченной очередью);

- параметры систем массового обслуживание (среднее время пребывания в системе, среднее время ожидания в очереди, средняя длина очереди, среднее число заявок в системе, среднее число обслуживаемых заявок, среднее число занятых каналов[26], среднее время обслуживания и т. д.);
- формулы Литтла;
- агрегативное представление систем;
- моделирование случайной величины, распределённой по определённому закону, в т. ч. по экспоненциальному, нормальному, равномерному.

7.4 Краткое описание работы

Данная работа представляет собой моделирование распределения заданий в системе с абсолютной надёжностью. ***Так что всем узлам и линиям связи в качестве закона распределения времени безотказной работы (или времени рабочего состояния) надо указывать Б()!*** Предлагается изучить несколько топологий: звезда, кольцо, разомкнутое кольцо.

Производится измерения таких показателей производительности всей системы, как:

- количество созданных заданий;
- количество выполненных заданий;
- среднее время выполнения задания;
- средняя длина очереди;
- среднее время задания в системе;
- среднее время ожидания задания;
- интервал моделирования.

26Если заявка может требовать несколько каналов, то среднее число обслуживаемых заявок <= среднее число занятых каналов. Если каждая заявка требуется только один канал, то среднее число обслуживаемых заявок = среднее число занятых каналов.

Производится измерения таких показателей производительности всей системы, как:

- созданное число заданий;
- количество полученных по сети заданий;
- количество отправленных по сети заданий;
- количество заданий, приступивших к исполнению;
- количество завершённых исполнений;
- среднее время выполнения задания на этом кластере;
- средняя длина очереди;
- среднее число используемых узлов.

Обратите внимание, что нам НЕ НУЖНЫ ВСЕ параметры моделирования.

При определении параметра путём имитационного моделирования необходимо помнить, что необходимо определять параметр с погрешностью 5% с квантилем, обеспечивающим вероятность попадания не менее 99% (при большом числе прогонок это ≈ 3.0). Более подробно это описано в центральной предельной теореме и коэффициентах Стьюдента. Учитывая длительность процесса моделирования мы будем использовать погрешность 10-20%.

Появление в результатах бесконечности или NaN говорит о невозможности определения параметра на данном интервале моделирования — необходимо увеличить интервал. Допускается снижать требуемую точность (т. е. число прогонок), если время моделирования слишком велико.

7.5 Практическая часть

В силу большого объёма вычислений допускается объединение студентов в группы с распараллеливанием задач внутри, т. е. сначала все вместе ставим на расчёт первую задачу, затем вторую и т. д.

1) Откройте программу GridModel и изучите, как осуществляется моделирование. Описание работы программы дано в руководстве пользователя. Чтобы открыть руководство пользователя, нажмите F1.
2) Пробное моделирование:
    a) Добавьте рабочую область один кластер (рис. 7.1), введите ему параметры из таблицы 7.1. Не забудьте отключить отказы.
    b) Откройте «Виды испытаний» и уберите все стратегии, добавьте стратегию «Создатель» с весом 1.0 .
    c) Осуществите пробное моделирование. Определите требуемое число прогонок, установив требуемую точность в 1%. Учтите, что с ростом интервала моделирования требуемое число прогонок сокращается, происходит изменение условий задачи. Для стационарного режима это почти не играет роли[27].
    d) Зафиксируйте результат и поместите его в протокол.
3) Топология «Звезда»:
    a) Создайте схему, как на рис. 7.2. Возьмите параметры из табл. 7.2. Установите центральному кластеру нулевое число узлов (отключите исполнение) и поток задач П(1), а всем остальным бесконечное время появление задач (отключите поток задач). Не забудьте отключить ВСЕ отказы.
    b) Откройте «Виды испытаний» и добавьте анализ всех стратегий, кроме «Создатель» (неразумно) и «Минимального риска» (она для отказов). В каждом испытании используйте только одну стратегию с весом 1.0[28].
    c) Зафиксируйте полученный результат и поместите его в протокол. Посмотрите 1-2 истории моделирования.
    d) Повторите для потока задач П(2) для центрального узла.

27 А мы исследуем именно его!
28 Т. е. ставьте только одну галочку в «Параметрах стратегии».

4) Повторите исследование топологии «Звезда», установив ВСЕ параметры из табл. 7.2 и все стратегии, кроме «Минимального риска». Занесите данные в протокол.
5) Топология «Кольцо»:
    a) Создайте схему, как на рис. 7.3 (попробуйте сделать её из предыдущей). Возьмите параметры из табл. 7.3 (параметры узлов почти не изменились, изменились только параметры третьего узла). Не забудьте отключить ВСЕ отказы.
    b) Проведите моделирование, используя тот же набор из 11 стратегий из пункта 4.
    c) Зафиксируйте полученный результат и поместите его в протокол. Посмотрите 1-2 истории моделирования.
6) Топология «Полукольцо»:
    a) В топологии «Кольцо» удалите связь между «1» и «2» узлами (рис. 7.4) и проведите испытания как и в предыдущем случае на 11 стратегиях.
    b) Сопоставьте результаты моделирование «Кольца» и «Полукольца».
7) Придумайте какую-нибудь свою топологию (3-5 узла, 2-7 линии связи. Но топология должна быть связной) и рассчитайте её как предыдущие.

7.6 Содержание отчёта

1) Тема и цель работы;
2) Выходные данные пробного моделирования;
3) Выходные данные моделирования «Звезды» с централизованным генератором заданий для двух случаев;
4) Выходные данные моделирования «Звезды» с распределёнными генераторами заданий;

5) Выходные данные моделирования «Кольца»;
6) Выходные данные моделирования «Полукольца»;
7) Входные и выходные данные моделирования своей собственной топологии вместе с рисунком этой топологии.

Таблица 7.1 — параметры для пробного моделирования

Параметр	**Cluster**
Количество узлов*	32
Ядер в узле*	4
Частота, GHz	2,33
Flops/Hz	4
Время безотказной работы, ч	Б()
Появление задач, ч	П(0,1)
Приоритеты задач*	Н*(4;1)
Память для задачи, Gb	Н(1;0,2)
Память для результатов, Gb	Н(0,7;0,15)
Сложность задания, TFlops[29]	Н(300;100)
Требуемые задаче узлы*	Р*(8;7)
Интервал моделирования, дн.	100

29 Триллионов операций с плавающей запятой

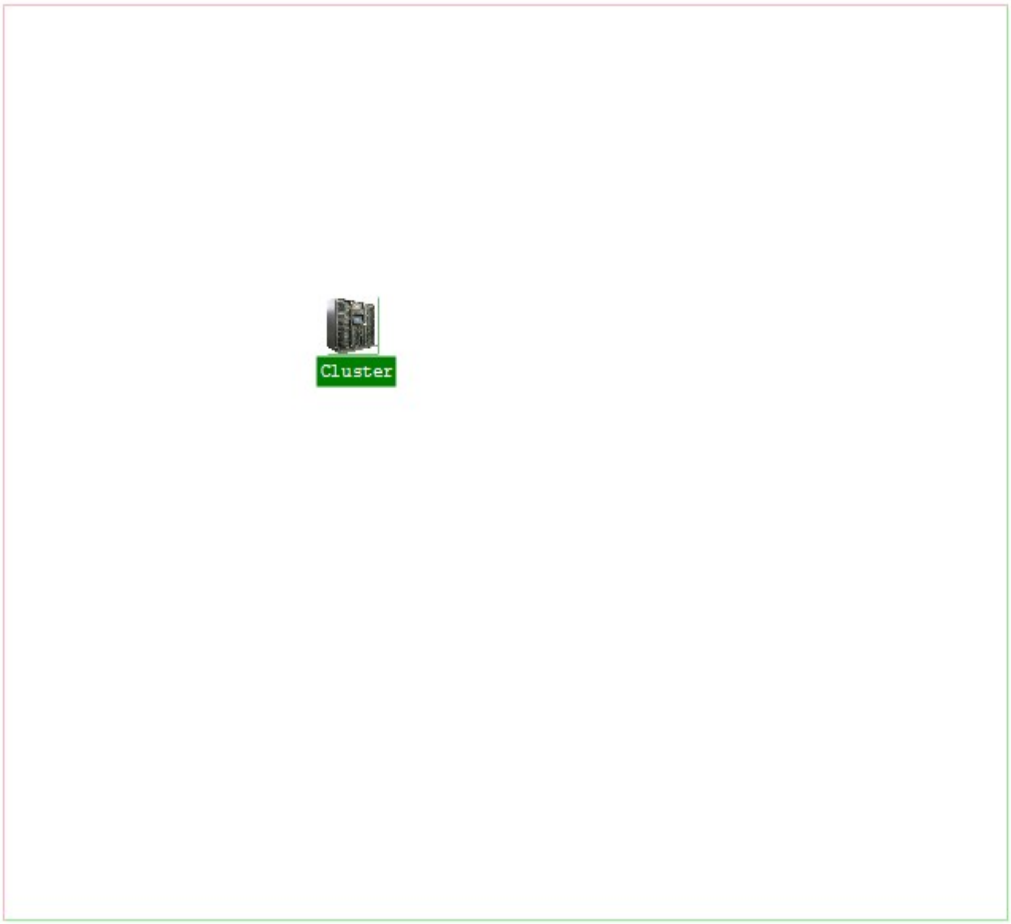


Рис. 7.1 — Пример топологии из одного кластера

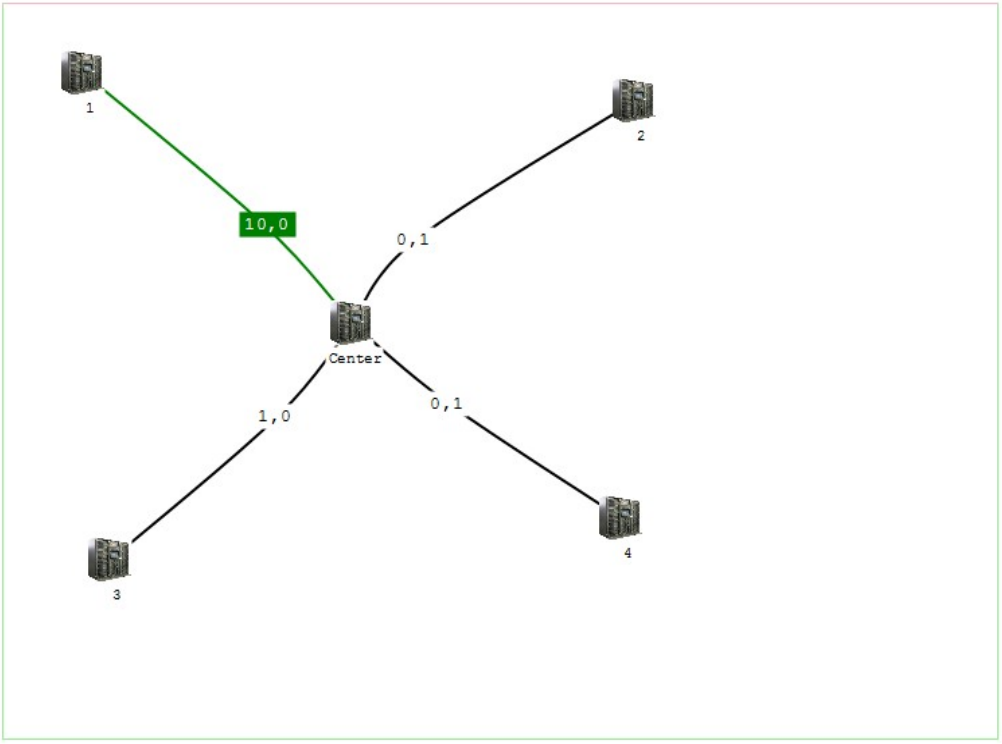


Рис. 7.2 — топология «Звезда»

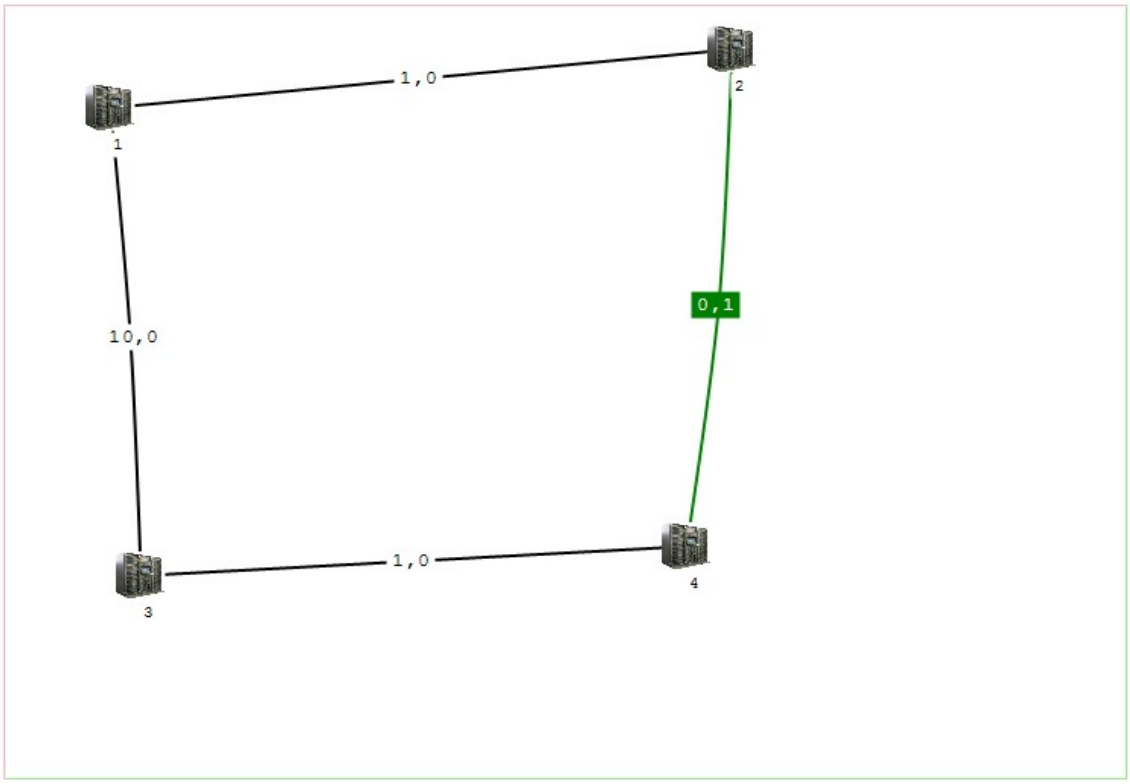


Рис. 7.3 — топология «Кольцо»

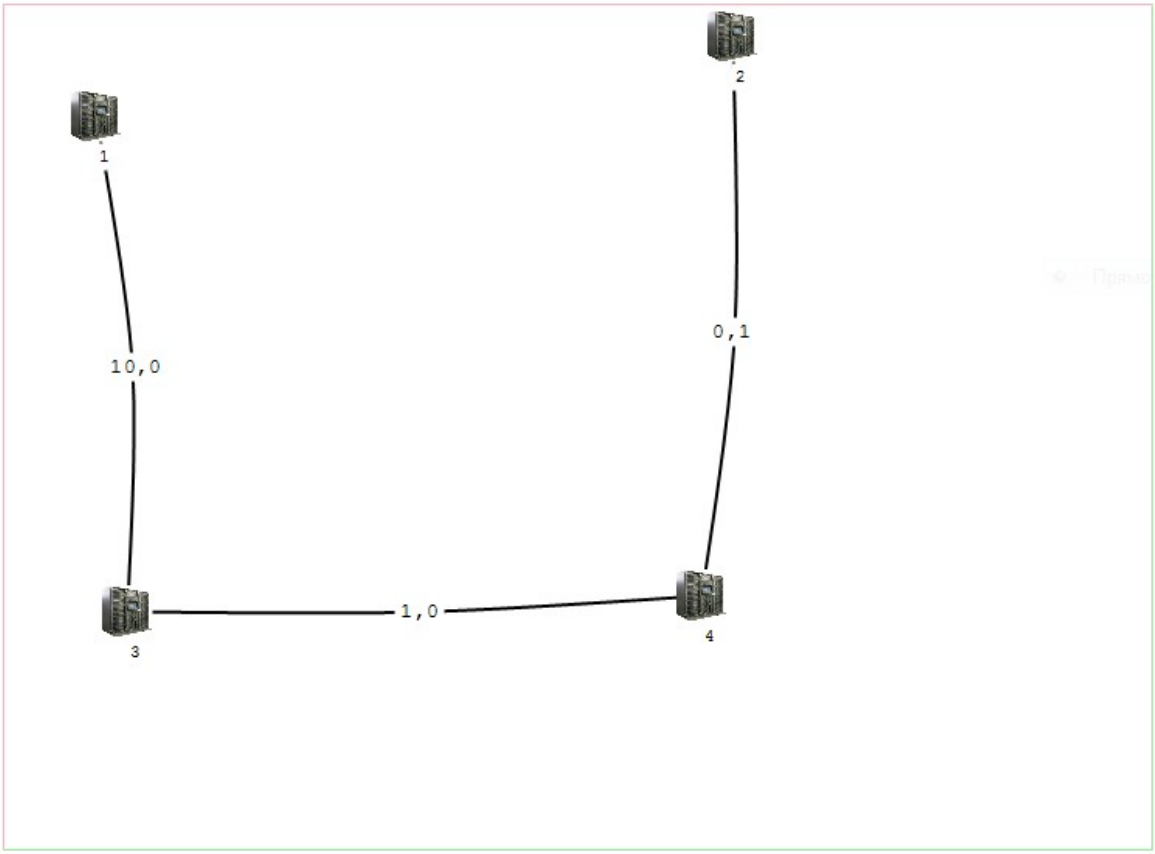


Рис. 7.4 — топология «Полукольцо»

Таблица 7.2 — параметры для моделирования «Звезды»

Параметр	Center	1	2	3	4
Количество узлов*	128	16	64	8	128
Ядер в узле*	8	8	2	8	1
Частота, GHz	2,33	2,33	2,33	2,33	2,33
Flops/Hz	4	4	4	4	4
Время безотказной работы, ч	Б()	Б()	Б()	Б()	Б()
Появление задач, ч	Л(4)	П(2)	Т(3;1)	Э(2;3)	В(3;0,5)
Приоритеты задач*	Н*(4;1)	Н*(4;1)	Н*(4;1)	Н*(4;1)	Н*(4;1)
Память для задачи, Gb	Н(1;0,2)	Н(0,5;0,1)	Н(0,5;0,1)	Н(0,5;0,1)	Н(0,5;0,1)
Память для результатов, Gb	Н(0,7;0,15)	Н(0,4;0,1)	Н(0,4;0,1)	Н(0,4;0,1)	Н(0,4;0,1)
Сложность задания, TFlops[30]	Н(15000;4500)	Н(10000;1700)	Н(10000;1700)	Н(10000;1700)	Н(10000;1700)
Требуемые задаче узлы*	Р*(8;7)	Р*(5;3)	Р*(5;3)	Р*(5;3)	Р*(5;3)
Интервал моделирования, дн.	100				

Параметр	Center - 1	Center - 2	Center - 3	Center - 4
Пропускная способность	10 Gbit/sec	100 Mbit/sec	100 Mbit/sec	1 Gbit/sec
Время безотказной работы, ч	Б()	Б()	Б()	Б()

30 Триллионов операций с плавающей запятой

Таблица 7.3 — параметры для моделирования «Кольца» и «Полукольца»

Параметр	**1**	**2**	**3**	**4**
Количество узлов*	16	64	8	128
Ядер в узле*	8	2	8	1
Частота, GHz	2,33	2,33	2,33	2,33
Flops/Hz	4	4	4	4
Время безотказной работы, ч	Б()	Б()	Б()	Б()
Появление задач, ч	П(2)	Т(3;1)	Э(3;3)	В(3;0,5)
Приоритеты задач*	Н*(4;1)	Н*(4;1)	Н*(4;1)	Н*(4;1)
Память для задачи, Gb	Н(0,5;0,1)	Н(0,5;0,1)	Н(0,7;0,2)	Н(0,5;0,1)
Память для результатов, Gb	Н(0,4;0,1)	Н(0,4;0,1)	Н(0,5;0,15)	Н(0,4;0,1)
Сложность задания, TFlops[31]	Н(10000;1700)	Н(10000;1700)	Н(12000;3500)	Н(10000;1700)
Требуемые задаче узлы*	Р*(5;3)	Р*(5;3)	Р*(7;4)	Р*(5;3)
Интервал моделирования, дн.	100			

Параметр	**1 - 2**	**1 - 3**	**3 - 4**	**2 - 4**
Пропускная способность	1 Gbit/sec	10 Gbit/sec	1 Gbit/sec	100 Mbit/sec
Время безотказной работы, ч	Б()	Б()	Б()	Б()

31 Триллионов операций с плавающей запятой

7.7 Контрольные вопросы

1 Сформулируйте цель работы.
2 Какие параметры мы определяли в данной работе?
3 Какие стратегии дали самые хорошие результаты? Почему?
4 В каком случае получился более хороший результат: в случае кольца или полукольца? Почему?
5 Как можно использовать формулы Литтла?
6 Можно ли для длинного интервала времени предсказать количество созданных заданий?
7 Что такое стационарный режим работы?

8 Лабораторная работа №3

8.1 Тема работы

Исследование влияния показателей надёжности на производительность системы путём имитационного моделирования. Анализ полученных результатов.

8.2 Цель работы

Целью данной работы является получение навыков имитационного моделирования, знакомство со средой GridModel, улучшение понимания имитационного моделирования, понимания надёжности и СМО, сопрокосновения этих понятий.

8.3 Рекомендации

Данная работа посвящена надёжности, поэтому рекомендуем ознакомиться с [2], [3], [9], [10], [19], [21], и моделированию распределения задач, поэтому рекомендуем ознакомиться с [12], [14], [24], а также ознакомится с [35]. Для понимания основных принципов имитационного моделирования как такового можно обратиться к [20], [23] или [45].

Рекомендуем вспомнить:

- основы теории вероятности;
- неравенства и теорема Чебышёва, теорема Бернулли;
- центральная предельная теорема;
- коэффициенты Стьюдента;
- марковские процессы с дискретными состояниями и непрерывным временем;
- пуассоновские потоки событий;

- системы массового обслуживания (одноканальные, многоканальные, без очереди, с ограниченной очередью, с неограниченной очередью);
- параметры систем массового обслуживание (среднее время пребывания в системе, среднее время ожидания в очереди, средняя длина очереди, среднее число заявок в системе, среднее число обслуживаемых заявок, среднее число занятых каналов, среднее время обслуживания и т. д.);
- формулы Литтла;
- агрегативное представление систем;
- моделирование случайной величины, распределённой по определённому закону, в т. ч. по экспоненциальному, нормальному, равномерному;
- материал лабораторных работ №1 и №2.

При определении параметра путём имитационного моделирования необходимо помнить, что необходимо определять параметр с погрешностью 5% с квантилем, обеспечивающим вероятность попадания не менее 99% (при большом числе прогонок это ≈ 3.0). Более подробно это описано в центральной предельной теореме и коэффициентах Стьюдента. Учитывая длительность процесса моделирования мы будем использовать погрешность 10-20%.

Появление в результатах бесконечности или NaN говорит о невозможности определения параметра на данном интервале моделирования — необходимо увеличить интервал. Допускается снижать требуемую точность (т. е. число прогонок), если время моделирования слишком велико.

8.4 Краткое описание работы

В данной работе мы попытаемся изучить влияние показателей надёжности на производительность системы, поэтому мы будем измерять такие параметры, как:

- количество выполненных заданий;
- доля выполненных заданий;
- средняя длина очереди;
- количество отказов, послуживших причиной прерывания исполнения задания;
- среднее время выполнения задания;
- среднее время выполнения задания с учётом неудач;
- коэффициент готовности;
- коэффициент использование (реальный коэффициент готовности).

8.5 Практическая часть

При выполнении нижеуказанных заданий просим обратить внимание, что в разных заданиях мы может рассчитывать одни и те же параметры. Поэтому нет смысла моделировать их несколько раз. Просто перепишите их из предыдущих заданий.

В силу большого объёма вычислений допускается объединение студентов в группы с распараллеливанием задач внутри, т. е. сначала все вместе ставим на расчёт первую задачу, затем вторую и т. д.

Моделирования в заданиях 2-8, осуществляется так же, как в задании 1.

1) Исследование влияния показателей надёжности на производительность узла:
    a) Создайте топологию, как на рис. 8.1. Возьмите параметры из табл. 8.1.
    b) Установите стратегию распределения «Создатель».
    c) Промоделируйте работу системы.
    d) Зафиксируйте результаты работы и занесите их в протокол.

2) Промоделируйте работу системы из табл. 8.1 при мгновенном подключении резерва, т. е. К(0), и нулевом допустимом времени простоя. Зафиксируйте данные в протокол.
3) Промоделируйте работу системы из табл. 8.1 при вероятностях необнаружения отказа 0%, 5%, 10%, 25%, 50%, 70%, 100%. Сведите данные в таблицу. Для 1-2 величин по выбору преподавателя постройте график.
4) Промоделируйте работу системы из табл. 8.1 при размерах резерва 0, 1, 2, 3, 4. Сведите данные в таблицу. Для 1-2 величин по выбору преподавателя постройте график.
5) Попробуйте заменить в системе из табл. 8.1 ненагруженный резерв нагруженным, т. е. сначала моделируем n рабочих узлов, m резервных, затем (n + 1) рабочих узлов, (m - 1) резервных, потом (n + 2) рабочих узлов, (m - 2) резервных... и так до (n + m) рабочих узлов, 0 резервных. Для 1-2 величин по выбору преподавателя постройте график. Сведите данные в таблицу. Для 1-2 величин по выбору преподавателя постройте график.
6) Промоделируйте работу системы из табл. 8.1 при количестве ремонтных линий 1, 2, 3, 4. Сведите данные в таблицу. Для 1-2 величин по выбору преподавателя постройте график.
7) Промоделируйте работу системы при времени безотказной работы одного компьютера в кластере П(100), П(700), П(1000), П(7000), Б(). Сведите данные в таблицу. Для 1-2 величин по выбору преподавателя постройте график.
8) Промоделируйте работу системы при времени обнаружения отказа работы П(1), П(7), П(10), П(17). Сведите данные в таблицу. Для 1-2 величин по выбору преподавателя постройте график.

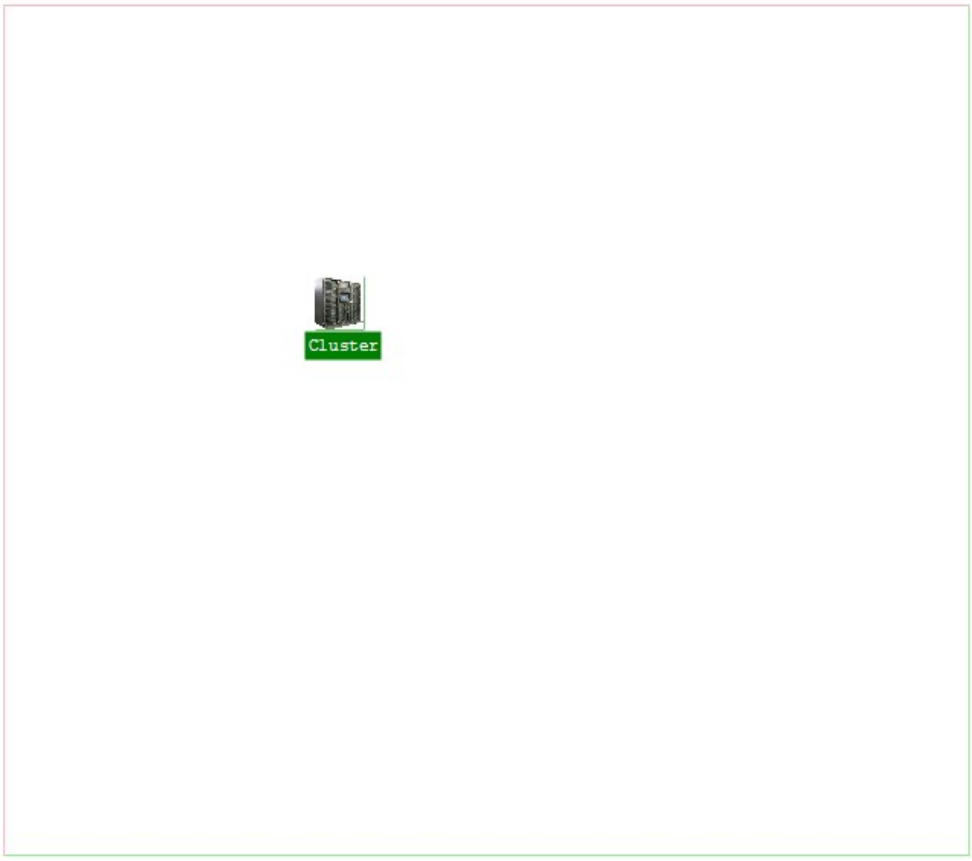


Рис. 8.1 — Пример топологии из одного кластера

Таблица 8.1 — параметры для пробного моделирования

Параметр	Cluster
Количество узлов*	32
Ядер в узле*	4
Резерв*	2
Ремонт*	1
Вероятность необнаружения отказа сразу	5%
Время рабочего состояния одного компьютера, ч	П(7000)
Время обнаружения необнаруженного сразу отказа, ч	П(17)
Время восстановления одного компьютера, ч	Н(100;17)
Время подключения резерва	Н(1;0,25)
Допустимое время простоя	1,2
Частота, GHz	2,33
Flops/Hz	4
Появление задач, ч	П(0,1)
Приоритеты задач*	Н*(4;1)
Память для задачи, Gb	Н(1;0,2)
Память для результатов, Gb	Н(0,7;0,15)
Сложность задания, TFlops[32]	Н(300;100)
Требуемые задаче узлы*	Р*(8;7)
Интервал моделирования, дн.	100

32 Триллионов операций с плавающей запятой

8.6 Содержание отчёта

1) Тема и цель работы;
2) Выходные данные моделирования системы из таблицы 8.1;
3) Выходные данные моделирования системы из таблицы 8.1 при условии мгновенного подключения резерва;
4) Таблица зависимости величин от вероятности необнаружения отказа с графиками изменения двух величин по выбору преподавателя;
5) Таблица зависимости величин от размера резерва с графиками изменения двух величин по выбору преподавателя;
6) Таблица зависимости величин от использования нагруженного резерва с графиками изменения двух величин по выбору преподавателя;
7) Таблица зависимости величин от количества ремонтных линий с графиками изменения двух величин по выбору преподавателя;
8) Таблица зависимости величин от среднего времени безотказной работы одного компьютера с графиками изменения двух величин по выбору преподавателя;
9) Таблица зависимости величин от среднего времени обнаружения отказа с графиками изменения двух величин по выбору преподавателя.

8.7 Контрольные вопросы

1 Сформулируйте цель работы.

2 Какие параметры мы определяли в данной работе?

3 Как вероятность необнаружения отказа влияет на показатели производительности?

4 Как размер резерва влияет на показатели производительности? Каков оптимальный размер резерва?

5 Имеет ли смысл использовать нагруженный резерв?

6 Каково оптимальное число ремонтных линий для системы? Сильно ли оно влияет на производительность системы?

7 Зависимость от каких параметров надёжности не изучалась в данной работе?

9 Лабораторная работа №4*

9.1 Тема работы

Исследование характеристик системы массового обслуживания (СМО), обладающей определёнными показателями надёжности, при различных политиках обработки сообщений. Анализ полученных результатов.

9.2 Цель работы

Целью данной работы является получение навыков имитационного моделирования, знакомство со СМО, улучшение понимания имитационного моделирования, понимания влияния показателей надёжности на производительность.

9.3 Рекомендации

Данная работа посвящена моделированию распределения задач, поэтому рекомендуем ознакомиться с [8], [9], [10], [12], [19], [20], [21], [23], [45]. Примерное описание реализации модели может быть найдено в [35], [38], [40].

Рекомендуем вспомнить:

- основы теории вероятности;
- неравенства и теорема Чебышёва, теорема Бернулли;
- центральная предельная теорема;
- коэффициенты Стьюдента;
- пуассоновские потоки событий;
- марковские процессы с дискретными состояниями и непрерывным временем;
- системы массового обслуживания (одноканальные, многоканальные, без очереди, с ограниченной очередью, с неограниченной очередью);

- параметры систем массового обслуживание (среднее время пребывания в системе, среднее время ожидания в очереди, средняя длина очереди, среднее число заявок в системе, среднее число обслуживаемых заявок, среднее число занятых каналов, среднее время обслуживания и т. д.);
- формулы Литтла;
- показатели надёжности элементов технической аппаратуры;
- агрегативное представление систем;
- моделирование случайной величины, распределённой по определённому закону, в т. ч. по экспоненциальному, нормальному, равномерному;
- гистограммы для определения средних значений параметров моделируемой системы во времени;
- технологии OpenMP, MPI, MS Parallel Extensions .NET, система параллельных вычислений F#;
- материал лабораторных работ №№ 1, 2 и 3.

9.4 Краткое описание работы

Данная работа требует написания студентами своей программы для определения показателей производительности однаканальной СМО с ограниченной очередью и отказывающим каналом обслуживания. Также у каждого задание есть некоторый критический интервал времени от времени создания, по истечению которого исполнение задания нецелесообразно.

Производится измерения таких показателей производительности всей системы, как:

- количество созданных заданий;
- количество выполненных заданий;

- количество отброшенных заданий;
- средняя длина очереди;
- использование канала (среднее число заявок под обслуживанием);
- среднее число заданий в системе;
- среднее время пребывание задания в системе;
- среднее время ожидания задания в очереди (сумма времён всех пребываний в очереди);
- среднее время, которое проводит на исполнении каждое задание (с учётом прерванных исполнений и отброшенных заданий);
- вероятность выполнения задания, она же доля выполненных заданий (обратите внимание, что при её расчёте необходимо НЕ учитывать задания все ещё пребывающие в системе, т. е. $= \frac{completed}{completed + dropped}$).

Средние времена пребывания измеряются как есть. Для оценки средней длины очереди и подобных параметров рекомендуем воспользоваться следующей техникой. Момент изменения параметра и начала/окончания процесса моделирования назовем особыми моментами параметра. Между особыми моментами находятся интервалы, на которых параметр не изменяется.

Необходимо найти $\frac{\sum_i s_i t_i}{T}$, $T = \sum_i t_i$, где s_i - значение параметра, t_i - длина временного интервала, T - все время моделирования.

При определении параметра путём имитационного моделирования необходимо помнить, что необходимо определять параметр с погрешностью как максимум 1% с квантилем, обеспечивающим вероятность попадания не менее 99,73% (при большом числе прогонок это ≈ 3.0). Более подробно это описано в центральной предельной теореме и коэффициентах Стьюдента. Не забывайте для каждого испытания использовать новый Seed для Random`a!

Для получения оценки математического ожидания как самой величины, так и среднего значения выборки используйте формулу:

$$M*(X)=M*(<X>)=<X>=\frac{1}{N}\sum_{i=1}^{N}X_i$$

Для получения оценки дисперсии среднего случайной величины можно использовать одну из формул:

$$D*(<X>)=\frac{1}{N(N-1)}\sum_{i=1}^{N}(X_i-<X>)^2 \quad ,$$

$$D*(<X>)=\frac{1}{N-1}(<X^2>-<X>^2)$$ (выражение в скобках означает «среднее значение квадрата минус квадрат среднего»).

$$<X^2>=\frac{1}{N}\sum_{i=1}^{N}X_i^2$$

Для получения оценки дисперсии самой случайной величины можно использовать одну из формул:

$$D*(X)=\frac{1}{N-1}\sum_{i=1}^{N}(X_i-<X>)^2 \quad ,$$

$$D*(X)=\frac{N}{N-1}(<X^2>-<X>^2)$$

Во обоих случаях СКО дается формулой:

$\sigma*=\sqrt{D*}$.

Абсолютная погрешность оценки математического ожидания рассчитывается по формуле:

$\delta=Z_{p,N}\cdot\sigma*(<X>)=\frac{Z_{p,N}\cdot\sigma*(X)}{\sqrt{N}}$, где $Z_{p,N}$ - квантиль, соответствующей вероятности попадания и числу испытаний. Возьмем $N\geq 30$. Тогда квантиль почти не зависит от N . Мы будем брать $Z=3.0$.

Относительная погрешность рассчитывается по формуле:

$$\epsilon=\frac{\delta}{M*(X)}=\frac{Z_p\cdot\sigma*(<X>)}{M*(X)}=\frac{Z_p\cdot\sigma*(X)}{\sqrt{N}M*(X)} \quad .$$

Тогда требуемое число прогонок можно определить по формуле:

$$N_{min} \geq \left(\frac{Z_p \cdot \sigma*(X)}{\epsilon \cdot M*(X)} \right)^2 = \left(\frac{Z_p \cdot \sigma*(X)}{\delta} \right)^2 = \frac{Z_p^2 \cdot D*(X)}{(\epsilon \cdot M*(X))^2} = \frac{Z_p^2 \cdot D*(X)}{\delta^2}, N_{min} \geq 30 \quad .$$

Сама величин X распределена неизвестным нам образом, поэтому оценить её отклонение от математического ожидания таким методом нельзя. Конечно, нам может повести и критерий Пирсона покажет, что распределение является нормальным. Ну или хотя бы каким-то из известным нам. Естественно, мы говорим обо всем об этом с определенной долей вероятности, что вытекает из самой природы критерия Пирсона.

Мы не будем решать эту проблему. Также мы не будем оценивать доверительные интервалы для дисперсии и СКО. Все, что от Вас требуется, это определить погрешность полученного значения и предсказать требуемое число прогонок.

Для сбора статистики рекомендуется использовать следующий подход. Для каждого параметра отдельно суммируются его значения и значения его квадратов. Число прогонок известно, поэтому получить среднее не представляет труда.

Но есть одна тонкость. Она возникает, когда программа начинает выполнять прогонки параллельно (а так и рекомендовано по заданию!) . Что будет, если два потока обратятся с попыткой суммирования к общей переменной? Ничего хорошего... Что будет, если два процесса обратятся с попыткой суммирования к общей переменной? Такой переменной не окажется в природе, если, конечно, не играться с разделяемой памятью... Но это уже совсем другая история...

Идея проста: каждый процесс или поток формирует свои частичные суммы, а затем они сводят свои частичные результаты. В случае OpenMP это проще всего сделать с помощью директив «reduction» или «atomic». «reduction» проще в использовании, однако её функционал не позволяет суммировать массивы значений, поэтому потребуется вручную объявить частичные результаты и в цикле вызывать «atomic». Можно не использовать атомарный

доступ к памяти, а разнести обращения во времени. Например, можно обращаться к общей переменной из критической секции или использовать барьерную синхронизацию (пример для двух потоков: сначала первый обращается к чётным элементам списка, второй к нечётным, затем барьер, затем наоборот). В MPI сведение результатов можно осуществить вызовом MPI_Reduce() или MPI_Allreduce() . Примеры даны в приложении В. В других технологиях может потребоваться иная реализация, но идея остается прежней.

Проблемы начнутся, если данных станет так много, что выделение локальной копии на каждый поток станет критичным для всей программы, но это не наш случай. Даже если нам надо сохранить 100 параметров, это 200 чисел с плавающей запятой (сумма величин и сумма их квадратов). Если взять числа двойной точности, получим 1600 байт, что на 64 байта больше, чем 1,5 Кбайт. Гораздо больше памяти может потребоваться на создание модели в каждом потоке.

Есть ещё один момент, который уже вскользь подразумевался, но не был озвучен. По причинам параллельного исполнения ГСЧ должен быть своим на каждой прогонке. Возможно, кому-то покажется, что проще создать ГСЧ на весь поток/процесс, но тогда результат работы программы будет зависеть от числа потоков/процессов, а это — нехорошо. Принцип прост. Есть входной параметр моделирования Seed. На каждой прогонке для создания ГСЧ используется число (Seed + i), где i — номер прогонки. Материал по написанию своего собственного ГСЧ дан в приложении Б и подразделе 2.3.

9.5 Практическая часть

Необходимо создать следующую СМО:

- одноканальная СМО;
- имеется очередь, ограниченная по размеру (при попытке добавления задания в полную очередь оно отбрасывается);

- задания приходят под воздействие пуассоновского потока событий[33];
- задания имеют время выполнения, распределённое по равномерному закону ($m > w > 0$);
- так же задания имеет тайм-аут, по истечению которого задания не могут начинать своё исполнение (в том числе и повторное) и больше ***не способны находится в очереди***. Если в момент истечения тайм-аута задание находится в очереди, это необходимо ***немедленно изъять оттуда и отбросить***. Если на момент истечения тайм-аута задание находится на исполнении, оно ***продолжает исполняться***. Тайм-аут распределён по линейному закону[34];
- канал СМО отказывает по экспоненциальному закону (пуассоновский поток) и восстанавливает по нормальному закону ($m \geq 3\sigma, \sigma > 0, m > 0$)[35];
- на время отказного состояния выполнение заданий прекращается и возобновляется при восстановлении канала.

Существует пять вариантов того, что может происходить с заданиями, прерванными отказом (***BreakPolicy***):

1 Отбрасывать эти задания (CorrectDrop);
2 Ставить в начало очереди (CorrectFirst);
3 Ставить в конец очереди (CorrectLast);
4 Начинать исполнение задания после восстановления канала (CorrectWait).

При постановке в очередь (CorrectFirst, CorrectLast) ***необходимо проверять ограничения*** на длину очереди и тайм-аут задания (задание отбрасывается, если очередь полна и/или тайм-аут задания истек за время исполнения). Если задание не попадает в очередь (CorrectWait), то эти

33 Для генерации всех случайных значение обратитесь к подразделу 2.3 и Приложениям А-Б.
34 Просто ведено для разнообразия.
35 Не забудьте поставить защиту от отрицательных значений, о которой говорилось в 2.3.

ограничения не проверяются (задания, единожды попавшие на исполнение, уже не могут быть отброшены).

После прерывания исполнения часть выполненной работы может сохранится. Это определяется ***BackUpPolicy***:

1 Вся работа утрачивается (RecovNone);

2 Вся работа сохраняется и задание продолжает исполняться с того момента, где было оборвано (RecovWhole);

3 Половина выполненной работы сохраняется (RecovHalf);

4 С равной вероятностью работа сохраняется целиком или полностью утрачивается (RecovProb);

5 Сохраняется доля работы, равномерно распределенная от нуля до размера выполненной части (RecovRand).

Выбор следующего задания из очереди можно осуществлять по одному из следующих алгоритмов (***QueuePolicy***):

1 Первое пришедшее (FirstTask);

2 Задание с раньше всего истекающим тайм-аутом (MinTimeOutTask);

3 Наиболее быстро выполняющиеся задания (FastTask);

4 Случайное задание при равной вероятности извлечения каждого (RandomTask).

Также студентам предлагается проверить контрпродуктивные стратегии:

5 Последнее пришедшее (LastTask);

6 Задание с позже всего истекающим тайм-аутом (MaxTimeOutTask);

7 Наименее быстро выполняющиеся задания (SlowTask);

Следующие стратегии QueuePolicy используют сверточный критерий $T_{death}+T_{exe}$, где T_{death} - время, когда истечет тайма-аут, T_{exe} - время исполнения задания:

8 Минимальное значение критерия (продуктивная MinCriterion);

9 Максимальное значение критерия (контрпродуктивная MaxCriterion);

И еще одна нестандартная стратегия:

10 Извлечение из середины очереди: номер извлекаемого задания из очереди определяется как $\left[\frac{Q-1}{2}\right]$, где $[]$ - изъятие целой части числа, Q - текущая (до извлечения) длина очереди (MiddleTask). Нумерация в очереди ведется с нуля (при использовании средств, где нумерация очереди ведется с единицы, необходимо использовать формулу $\left[\frac{Q+1}{2}\right]$).

Вероятность совпадения оценок 2-3 и 6-9 в реальной системе равна нулю, а в модели близка к нулю, поэтому при совпадении оценок 2-3 и 8 будем выбирать задание, первое стоящее в очереди, а при совпадении 6-7 и 9 — последнее. Очевидно, что при политиках QueuePolicy 2-4 и 6-9 политики CorrectFirst и CorrectLast практически не различимы.

Обратите внимание, что при повторном попадании задания в очередь его время выполнения может быть меньше, чем при предыдущем из-за сохранения некоторого количества ранее полученного результата. При выборке по QueuePolicy необходимо учитывать именно новое значение параметра!

Популярные ошибки при выполнении работы:

1) Исполнение задания может быть прервано несколько раз, то есть слагаемых во времени исполнения может быть больше двух.

2) При CorrectWait среднее задание, прерванные отказом, будут некоторое время пребывать вне исполнения и вне очереди, поэтому сумма среднего времени ожидания и среднего времени на исполнении будет меньше среднего времени в системе, а сумма средней длины очереди и использования канала будет меньше среднего числа заявок в системе. При других BreakPolicy сумма среднего времени ожидания и среднего времени на исполнении будет равна среднему времени в системе, а сумма средней длины очереди и использования канала будет — среднему числа заявок в системе, но ***эти соотношения не надо использовать для расчета выходных***, лишь для проверки правильности.

3) Время на исполнении включает только ***время, реально проведенное заданиями на исполнении***. Оно может быть как больше, так и меньше времени, требуемого заданиями для исполнения.

4) Для проверки правильности расчетных данных используйте формулы Литтла.

5) Число пришедших заявок в данной задаче является величиной, распределенной по закону Пуассона, поэтому его дисперсия равна математическому ожиданию (при моделировании она должна быть примерно равна).

Рекомендуемая структура задания:

- Время генерации;
- Тайм-аут (либо сама величина, либо время истекания);
- Требуемое время выполнения;
- Время, проведённое на исполнениях[36];
- Время, проведённое в очереди[37];
- Время последней постановки в очередь/на исполнение;
- Флаг Обрабатывается/Исполнено/Отброшено.

По окончания выполнение работы студенты должны сравнить свои разные модели на одних и тех параметрах системы. Варианты работы приведены в табл. 9.1. Данные для моделирования даны в табл. 9.2.

При выполнении заданий настоятельно рекомендуется использовать OpenMP и/или MPI. В качестве языка в этом случае можно выбрать C/C++ или Fortran. Допускается использование языков платформы .NET и Microsoft Parallel Extensions .NET или языка F#.

9.6 Содержание отчёта

1) Тема и цель работы;

36 Обновляется при окончании исполнения.
37 Обновляется при выходе из очереди.

2) Код программы;

3) Входные данные моделирования;

4) Выходные данные моделирования;

5) Оценка погрешности выходных значений;

6) Входные данные для сравнения моделей;

7) Сравнение результатов моделирования нескольких студентов.

Таблица 9.1 — Варианты заданий для студентов

Вариант	**QueuePolicy**	**BreakPolicy**	**BackUpPolicy**
1	FirstTask	CorrectDrop	–
2	MinTimeOutTask	CorrectDrop	–
3	FastTask	CorrectDrop	–
4	RandomTask	CorrectDrop	–
5	LastTask	CorrectDrop	–
6	MaxTimeOutTask	CorrectDrop	–
7	SlowTask	CorrectDrop	–
8	MinCriterion	CorrectDrop	–
9	MaxCriterion	CorrectDrop	–
10	MiddleTask	CorrectDrop	–
11	FirstTask	CorrectFirst	RecovNone
12	MinTimeOutTask	CorrectFirst	RecovNone
13	FastTask	CorrectFirst	RecovNone
14	RandomTask	CorrectFirst	RecovNone
15	LastTask	CorrectFirst	RecovNone
16	MaxTimeOutTask	CorrectFirst	RecovNone
17	SlowTask	CorrectFirst	RecovNone
18	MinCriterion	CorrectFirst	RecovNone
19	MaxCriterion	CorrectFirst	RecovNone
20	MiddleTask	CorrectFirst	RecovNone
21	FirstTask	CorrectLast	RecovNone
22	LastTask	CorrectLast	RecovNone

Вариант	**QueuePolicy**	**BreakPolicy**	**BackUpPolicy**
23	MiddleTask	CorrectLast	RecovNone
24	FirstTask	CorrectWait	RecovNone
25	MinTimeOutTask	CorrectWait	RecovNone
26	FastTask	CorrectWait	RecovNone
27	RandomTask	CorrectWait	RecovNone
28	LastTask	CorrectWait	RecovNone
29	MaxTimeOutTask	CorrectWait	RecovNone
30	SlowTask	CorrectWait	RecovNone
31	MinCriterion	CorrectWait	RecovNone
32	MaxCriterion	CorrectWait	RecovNone
33	MiddleTask	CorrectWait	RecovNone
34	FirstTask	CorrectFirst	RecovWhole
35	MinTimeOutTask	CorrectFirst	RecovWhole
36	FastTask	CorrectFirst	RecovWhole
37	RandomTask	CorrectFirst	RecovWhole
38	LastTask	CorrectFirst	RecovWhole
39	MaxTimeOutTask	CorrectFirst	RecovWhole
40	SlowTask	CorrectFirst	RecovWhole
41	MinCriterion	CorrectFirst	RecovWhole
42	MaxCriterion	CorrectFirst	RecovWhole
43	MiddleTask	CorrectFirst	RecovWhole
44	FirstTask	CorrectLast	RecovWhole
45	LastTask	CorrectLast	RecovWhole
46	MiddleTask	CorrectLast	RecovWhole
47	FirstTask	CorrectWait	RecovWhole
48	MinTimeOutTask	CorrectWait	RecovWhole
49	FastTask	CorrectWait	RecovWhole
50	RandomTask	CorrectWait	RecovWhole
51	LastTask	CorrectWait	RecovWhole
52	MaxTimeOutTask	CorrectWait	RecovWhole
53	SlowTask	CorrectWait	RecovWhole

Вариант	**QueuePolicy**	**BreakPolicy**	**BackUpPolicy**
54	MinCriterion	CorrectWait	RecovWhole
55	MaxCriterion	CorrectWait	RecovWhole
56	MiddleTask	CorrectWait	RecovWhole
57	FirstTask	CorrectFirst	RecovHalf
58	MinTimeOutTask	CorrectFirst	RecovHalf
59	FastTask	CorrectFirst	RecovHalf
60	RandomTask	CorrectFirst	RecovHalf
61	LastTask	CorrectFirst	RecovHalf
62	MaxTimeOutTask	CorrectFirst	RecovHalf
63	SlowTask	CorrectFirst	RecovHalf
64	MinCriterion	CorrectFirst	RecovHalf
65	MaxCriterion	CorrectFirst	RecovHalf
66	MiddleTask	CorrectFirst	RecovHalf
67	FirstTask	CorrectLast	RecovHalf
68	LastTask	CorrectLast	RecovHalf
69	MiddleTask	CorrectLast	RecovHalf
70	FirstTask	CorrectWait	RecovHalf
71	MinTimeOutTask	CorrectWait	RecovHalf
72	FastTask	CorrectWait	RecovHalf
73	RandomTask	CorrectWait	RecovHalf
74	LastTask	CorrectWait	RecovHalf
75	MaxTimeOutTask	CorrectWait	RecovHalf
76	SlowTask	CorrectWait	RecovHalf
77	MinCriterion	CorrectWait	RecovHalf
78	MaxCriterion	CorrectWait	RecovHalf
79	MiddleTask	CorrectWait	RecovHalf
80	FirstTask	CorrectFirst	RecovProb
81	MinTimeOutTask	CorrectFirst	RecovProb
82	FastTask	CorrectFirst	RecovProb
83	RandomTask	CorrectFirst	RecovProb
84	LastTask	CorrectFirst	RecovProb

Вариант	**QueuePolicy**	**BreakPolicy**	**BackUpPolicy**
85	MaxTimeOutTask	CorrectFirst	RecovProb
86	SlowTask	CorrectFirst	RecovProb
87	MinCriterion	CorrectFirst	RecovProb
88	MaxCriterion	CorrectFirst	RecovProb
89	MiddleTask	CorrectFirst	RecovProb
90	FirstTask	CorrectLast	RecovProb
91	LastTask	CorrectLast	RecovProb
92	MiddleTask	CorrectLast	RecovProb
93	FirstTask	CorrectWait	RecovProb
94	MinTimeOutTask	CorrectWait	RecovProb
95	FastTask	CorrectWait	RecovProb
96	RandomTask	CorrectWait	RecovProb
97	LastTask	CorrectWait	RecovProb
98	MaxTimeOutTask	CorrectWait	RecovProb
99	SlowTask	CorrectWait	RecovProb
100	MinCriterion	CorrectWait	RecovProb
101	MaxCriterion	CorrectWait	RecovProb
102	MiddleTask	CorrectWait	RecovProb
103	FirstTask	CorrectFirst	RecovRand
104	MinTimeOutTask	CorrectFirst	RecovRand
105	FastTask	CorrectFirst	RecovRand
106	RandomTask	CorrectFirst	RecovRand
107	LastTask	CorrectFirst	RecovRand
108	MaxTimeOutTask	CorrectFirst	RecovRand
109	SlowTask	CorrectFirst	RecovRand
110	MinCriterion	CorrectFirst	RecovRand
111	MaxCriterion	CorrectFirst	RecovRand
112	MiddleTask	CorrectFirst	RecovRand
113	FirstTask	CorrectLast	RecovRand
114	LastTask	CorrectLast	RecovRand
115	MiddleTask	CorrectLast	RecovRand

Вариант	**QueuePolicy**	**BreakPolicy**	**BackUpPolicy**
116	FirstTask	CorrectWait	RecovRand
117	MinTimeOutTask	CorrectWait	RecovRand
118	FastTask	CorrectWait	RecovRand
119	RandomTask	CorrectWait	RecovRand
120	LastTask	CorrectWait	RecovRand
121	MaxTimeOutTask	CorrectWait	RecovRand
122	SlowTask	CorrectWait	RecovRand
123	MinCriterion	CorrectWait	RecovRand
124	MaxCriterion	CorrectWait	RecovRand
125	MiddleTask	CorrectWait	RecovRand

Таблица 9.2 — Пример данных для исследования работы системы

Параметр	**Значение**
Максимальный размер очереди*	8
Интервалы между приходами заданий	П(7)
Время выполнения	Р(5;4)
Тайм-аут	Л(24)
Время работы канала	П(128)
Время восстановления канала	Н(8;2)
Интервал моделирования	100000
Рекомендуемое число прогонок	>= 1000

9.7 Контрольные вопросы

1 Сформулируйте цель работы.

2 Какие параметры мы определяли в данной работе?

3 Как мы оценивали погрешность?

4 Что именно было распараллелено в данной работе?

5 В каком из сделанных вами вариантов система получилась наиболее производительной?

6 Можно ли для длинного интервала времени предсказать количество созданных заданий?

7 Что такое стационарный режим работы?

10 Примеры выполнения лабораторных работ

Обратите внимание, что данном разделе дан пример выполнения лабораторной работы. Правильность данных, указанных в экранных формах не гарантируется. Также возможны некоторые незначительные изменения во внешнем виде программы. Например, был расширен список определяемых показателей надёжности и их имена были расшифрованы без сокращений. Также изменились некоторые внутренние особенности моделирования. Были исправлены мелкие ошибки.

10.1 Лабораторная работа №1. Нулевой вариант

Вариант 0.

n	m	l	t(раб1)	t(отк1)	P	t(обн1)	t(подкл1)	timeOut
20	1	3	Н(800;75)	Л(14)	8,00%	Н(1;0,3)	Л(5)	6

1 **Введите в среду имитационного моделирования параметры n, m, l,** $t_{раб1}$, $t_{восст1}$. **Вероятность необнаружения и допустимое время простоя отказа положите равными нулю. Закон подключения резерва назначьте К(0).**

Для ввода параметров, соответствующих вашему варианту откройте главное окно программы. Добавьте вершину, для возможности моделирования. После добавления вершины появляется возможность задания величин для моделирования. В главном окне, справа зададим соответствующие значения.

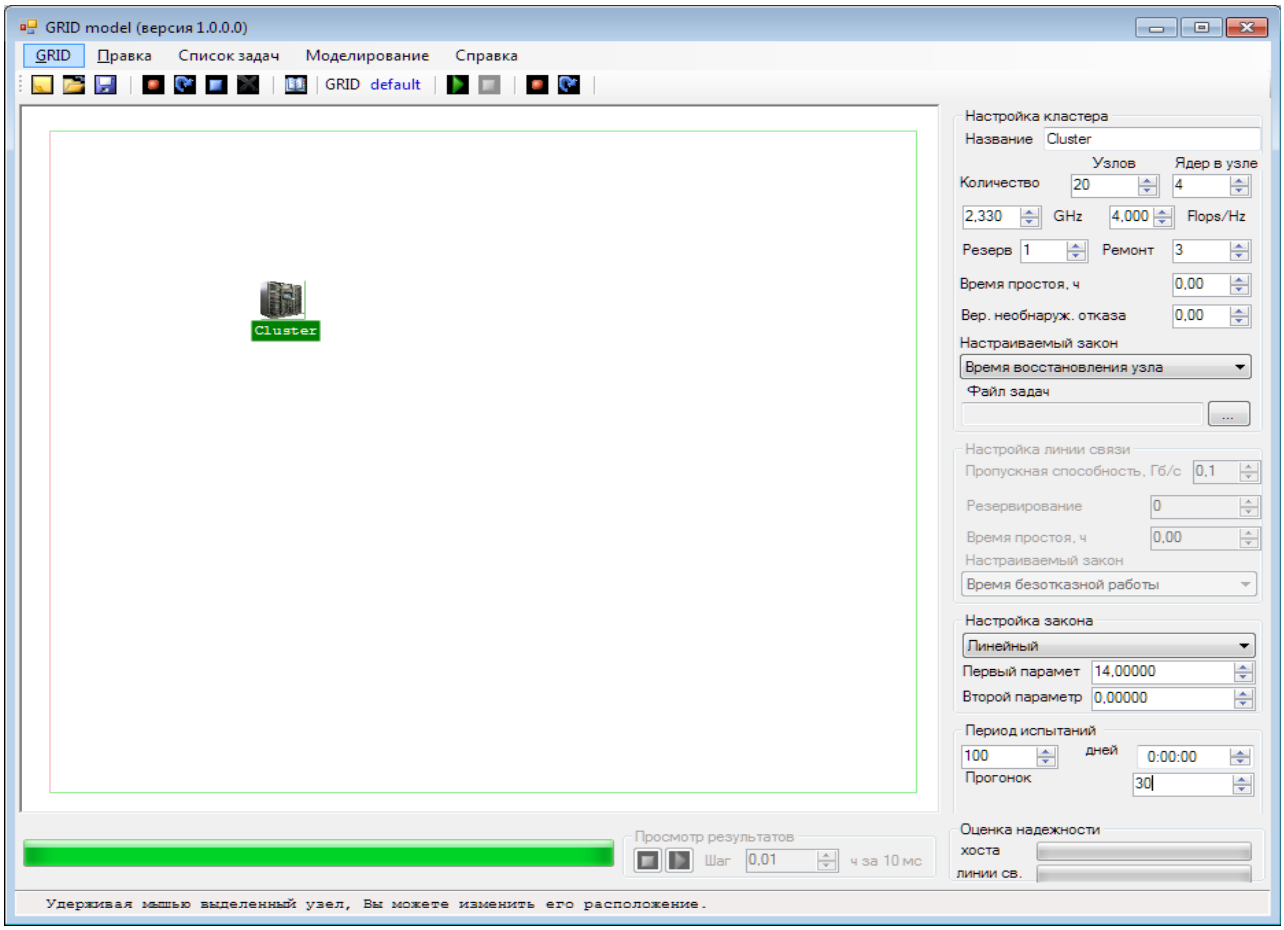


Рис. 1 Добавление вершины

Задайте количество узлов (n).

Рис. 2 Редактирование количество узлов

Задайте размер резерва(m)

Рис. 3 Редактирование резерва

Задайте максимальное число параллельно восстанавливающихся узлов (l)

Рис. 4 Редактирование резерва

Задайте время безотказной работы узла

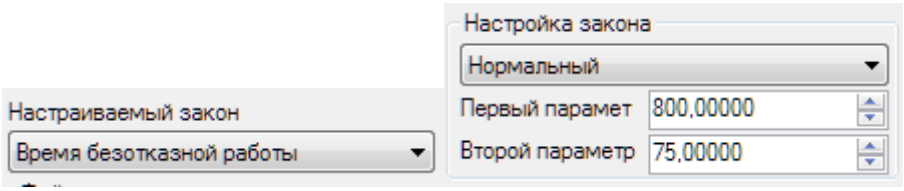


Рис. 5 Редактирование времени безотказной работы

Задайте время восстановления

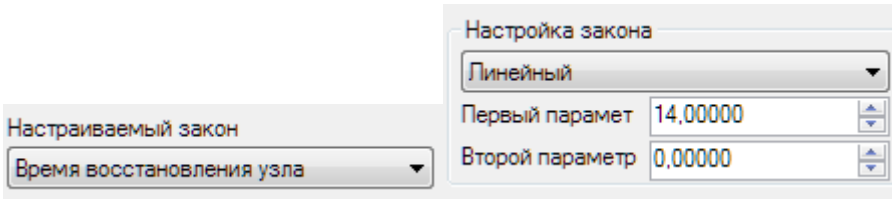


Рис. 6 Редактирование времени безотказной работы

Вероятность необнаружения и допустимое время простоя отказа положите равными нулю.

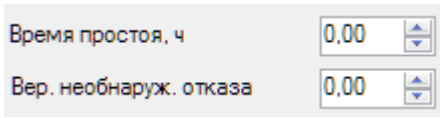


Рис. 7 Редактирование вероятности необнаружения и допустимого времени простоя

Закон подключения резерва подключения резерва по аналогии установите К(0).

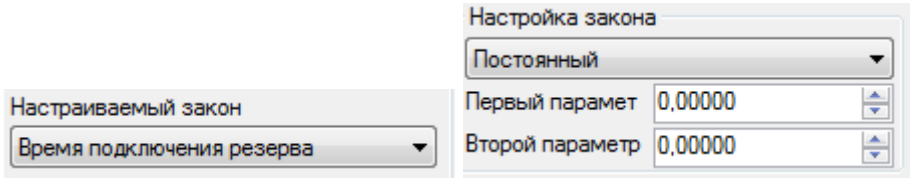


Рис. 8 Редактирование закона подключения резерва

2 **Проведите моделирование на малом числе прогонок, чтобы определить требуемое число прогонок. Учтите, что требуемое время растет примерно линейно с увеличением числа прогонок. Интервал моделирования следует выбирать та, что бы успело случиться достаточное число событий. Для данного примера это что-то в районе 1 тыс. дн. (успевают пройти сотни отказов) За начальное количество прогонок возьмём 30.**

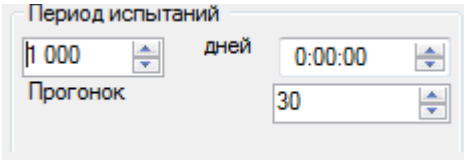


Рис. 9 Временные параметры моделирования

Промоделируйте систему с заданными параметрами. На вкладке статистике перейдите на Анализ результатов. Определите оптимальное значение прогонок.

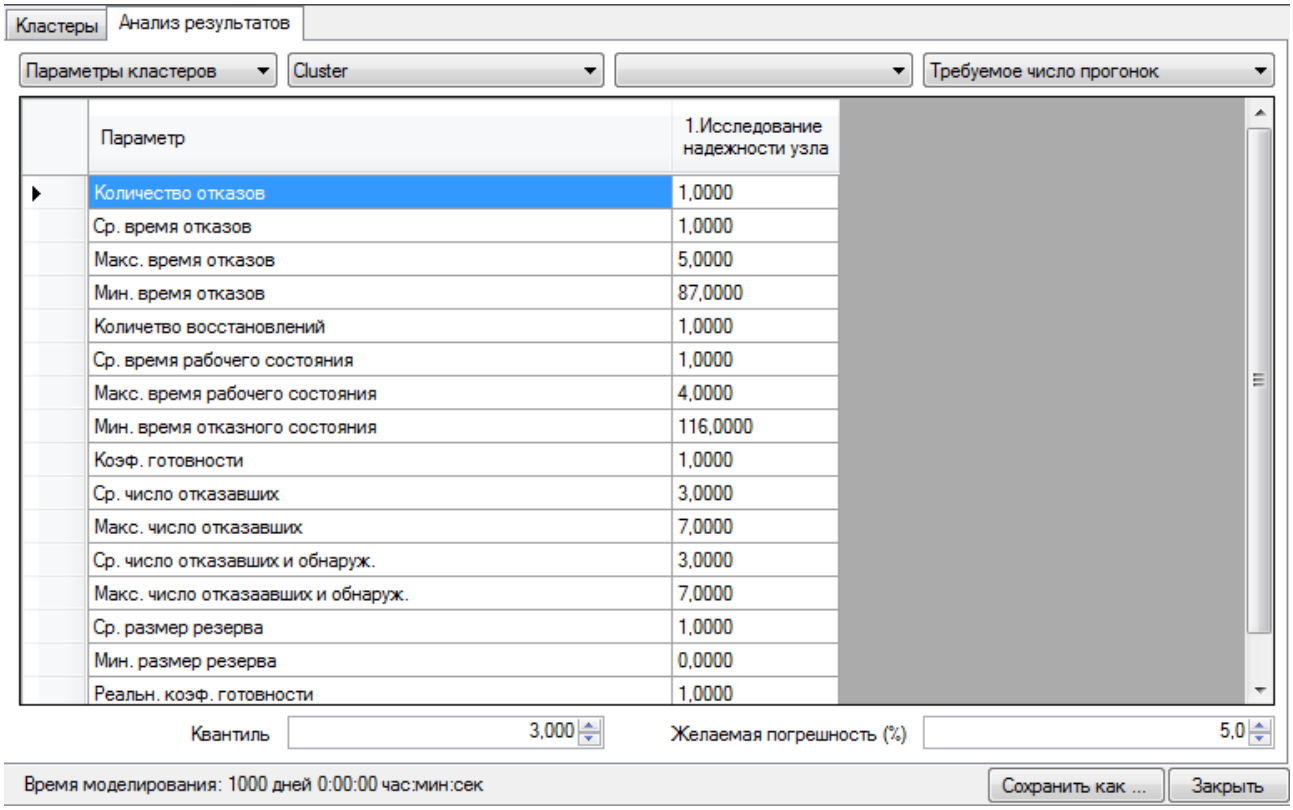


Рис. 10 Требуемое число прогонок

Отсюда видно, что хватает 3 прогонок, но, вспомнив, что коэффициенты Стьюдента (т. е. квантили) меняются для числа прогонок меньше 20, останавливаемся на числе в 20 прогонок.

3 Проведите имитационное моделирование с требуемой (или ослабленной) точностью и зафиксируйте результаты

Закройте окно статистики и проведите требуемое число прогонок.

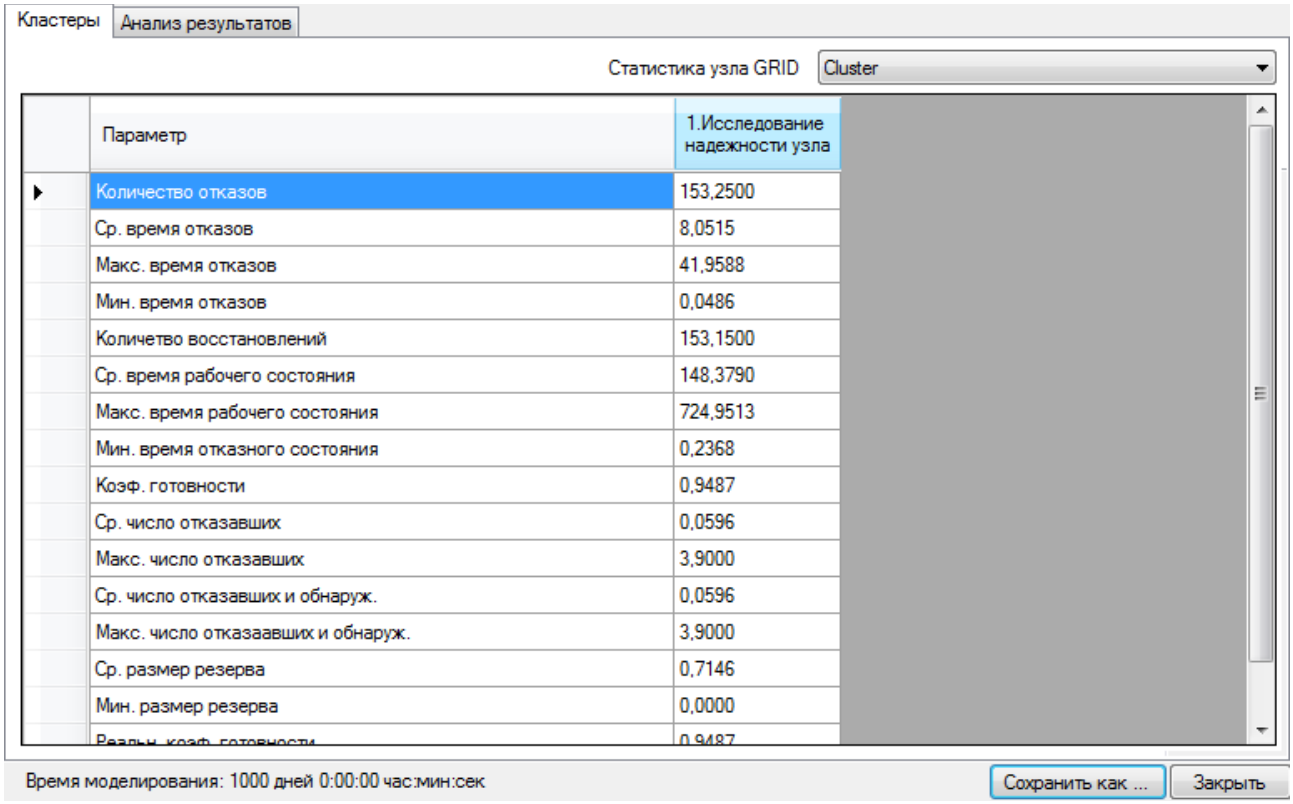


Рис. 11 Требуемый результат

4 Осуществите аналитический расчёт, используя прилагаемую аналитическую модель. Необходимо предполагать все потоки событий пуассоновскими, т. е. все законы распределения будут показательными.

Теперь нужно просчитать Ваш вариант аналитически. Для аналитического расчёта используйте программу, находящуюся (путь).

Расчёт под MinGW/gcc:

1. Подключите в файле arch.mak компилятор файлы gcc.mak и win.mak.

2. Проверьте переменные среды. В переменной PATH должна быть папка, где лежат gcc.exe, g++.exe, make.exe и т.д.

Если Вы изменяли переменные среды, то убедись, что все используемые процессы получили новые переменные среды.

3. Откройте консоль в паке с этой программой.

4. Введите в файл in.txt требуемые данные.

5. Выполните команду "make all" в папке с кодами программы и нажмите ввод.

6. В файле out.txt должны быть результаты расчёта, а в файле err.txt - отчёт об ошибках.

7. Повторяйте пункты 4 - 6, пока не рассчитайте все необходимые данные.

8. По окончанию работы выполните команду "make clean" в папке программы для удаления файлов результата и временных файлов.

Расчёт под MS Visual Studio:

1. Подключите в файле arch.mak компилятор файлы cl.mak и win.mak.

2. Запустите консоль Visual Studio x86 (или x64).

3. Перейдите в папку с этой программой.

4. Введите в файл in.txt требуемые данные.

5. Наберите в консоле "nmake all" в папке с кодами программы и нажмите ввод.

6. В файле out.txt должны быть результаты расчета, а в файле err.txt - отчет об ошибках.

7. Повторяйте пункты 4 - 6, пока не рассчитайте все необходимые данные.

8. По окончанию работы выполните команду "nmake clean" в папке программы для удаления файлов результата и временных файлов.

Расчёт с предварительной компиляцией:

1. Соберите main.cpp каким-либо компилятором.

2. Запустите бинарный файл.

3. Введите входные данные.

4. Получите выходные данные и отчёт об ошибках.

Можно перенастроить потоки ввода/вывода следующим образом:

```
a.exe < in.txt > out.txt 2> err.txt
```

Это УНИЧТОЖИТ старые файлы out.txt и err.txt. Чтобы дописать данные в конец, используйте:

a.exe < in.txt >> out.txt 2>> err.txt

Файл in.txt должен содержать

```
n m l tраб1 tотк1 точность
```

траб1 – вводится для показательного закона, то есть только первый параметр. В нашем случае 800.

Точность - это количество знаков после запятой в ответе

После выполнения программы, программа выдаст нам:

- вероятности пребывания в состояниях;
- среднее время пребывания в группе состояний;
- значения средних параметров системы;
- сводные данные в конце.

Для нашего случая данные представлены в лист. 1 и 2.

Листинг 1. Входные данные

```
20 1 3 800 14 12
```

Листинг 2. Результат аналитического расчёта

```
P(brk == 0)=0.704806445265
P(brk == 1)=0.246682255843
P(brk == 2)*=0.0431693947725
P(brk == 3)*=0.00478460792062
P(brk == 4)*=0.000502383831665
P(brk == 5)*=4.98197299734e-005
P(brk == 6)*=4.64984146419e-006
P(brk == 7)*=4.06861128116e-007
P(brk == 8)*=3.32269921295e-008
P(brk == 9)*=2.51971356982e-009
P(brk == 10)*=1.76379949887e-010
P(brk == 11)*=1.13177134511e-011
P(brk == 12)*=6.60199951315e-013
P(brk == 13)*=3.4660497444e-014
P(brk == 14)*=1.61748988072e-015
P(brk == 15)*=6.60475034628e-017
P(brk == 16)*=2.3116626212e-018
P(brk == 17)*=6.74234931182e-020
P(brk == 18)*=1.57321483943e-021
```

```
P(brk == 19)*=2.75312596899e-023
P(brk == 20)*=3.21198029716e-025
P(brk == 21)*=1.87365517334e-027

P(wrk == 20)=0.951488701108
P(wrk >= 19)*=0.99465809588
P(wrk >= 18)*=0.999442703801
```

```
P(wrk >= 17)*=0.999945087632
P(wrk >= 16)*=0.999994907362
P(wrk >= 15)*=0.999999557204
P(wrk >= 14)*=0.999999964065
P(wrk >= 13)*=0.999999997292
P(wrk >= 12)*=0.999999999812
P(wrk >= 11)*=0.999999999988
P(wrk >= 10)*=0.999999999999
P(wrk >= 9)*=1
P(wrk >= 8)*=1
P(wrk >= 7)*=1
P(wrk >= 6)*=1
P(wrk >= 5)*=1
P(wrk >= 4)*=1
P(wrk >= 3)*=1
P(wrk >= 2)*=1
P(wrk >= 1)*=1

P(rsr >= 1)=0.704806445265
P(rsr == 0)=0.295193554735

avgDuration[wrk >= 20]= 154.285714286
avgDuration[wrk <  20]= 7.86619997887
avgDuration[wrk >= 19]= 970.139634801
avgDuration[wrk <  19]= 5.21022543679
avgDuration[wrk >= 18]= 9283.86953779
avgDuration[wrk <  18]= 5.17675019169
avgDuration[wrk >= 17]= 93665.9113054
avgDuration[wrk <  17]= 5.14369941203
avgDuration[wrk >= 16]= 1003613.33542
avgDuration[wrk <  16]= 5.11106512893
avgDuration[wrk >= 15]= 11469920.0238
avgDuration[wrk <  15]= 5.07883956854
avgDuration[wrk >= 14]= 140448057.434
avgDuration[wrk <  14]= 5.04701512424
avgDuration[wrk >= 13]= 1852062357.37
avgDuration[wrk <  13]= 5.01558400372
avgDuration[wrk >= 12]= 26458033743.4
avgDuration[wrk <  12]= 4.98453393935
avgDuration[wrk >= 11]= 412332993477
avgDuration[wrk <  11]= 4.95379184603
avgDuration[wrk >= 10]= 7.06856560254e+012
avgDuration[wrk <  10]= 4.92206758231
avgDuration[wrk >= 9]= 1.3463934481e+014
avgDuration[wrk <  9]= 4.85809027044
avgDuration[wrk >= 8]= 2.88512881737e+015
avgDuration[wrk <  8]= 4.16407737467
avgDuration[wrk >= 7]= 7.06562159355e+016
avgDuration[wrk <  7]= -15.6888315529
avgDuration[wrk >= 6]= 2.01874902673e+018
avgDuration[wrk <  6]= -448.252330083
avgDuration[wrk >= 5]= 6.9214252345e+019
avgDuration[wrk <  5]= -15368.6513171
avgDuration[wrk >= 4]= 2.9663251005e+021
avgDuration[wrk <  4]= -658656.48502
avgDuration[wrk >= 3]= 1.69504291457e+023
avgDuration[wrk <  3]= -37637513.4297
avgDuration[wrk >= 2]= 1.45289392678e+025
avgDuration[wrk <  2]= -3226072579.69
avgDuration[wrk >= 1]= 2.49067530304e+027
avgDuration[wrk <  1]= -553041013661
```

```
avgDuration[rsr >= 1]= 40
avgDuration[rsr <  1]= 16.7531699926

avgDuration[brk <= 0]= 40
avgDuration[brk >  0]= 16.7531699926
avgDuration[brk <= 1]= 154.285714286
avgDuration[brk >  1]= 7.86619997887
avgDuration[brk <= 2]= 970.139634801
avgDuration[brk >  2]= 5.21022543679
avgDuration[brk <= 3]= 9283.86953779
avgDuration[brk >  3]= 5.17675019169
avgDuration[brk <= 4]= 93665.9113054
avgDuration[brk >  4]= 5.14369941203
avgDuration[brk <= 5]= 1003613.33542
avgDuration[brk >  5]= 5.11106512893
avgDuration[brk <= 6]= 11469920.0238
avgDuration[brk >  6]= 5.07883956854
avgDuration[brk <= 7]= 140448057.434
avgDuration[brk >  7]= 5.04701512424
avgDuration[brk <= 8]= 1852062357.37
avgDuration[brk >  8]= 5.01558400372
avgDuration[brk <= 9]= 26458033743.4
avgDuration[brk >  9]= 4.98453393935
avgDuration[brk <= 10]= 412332993477
avgDuration[brk >  10]= 4.95379184603
avgDuration[brk <= 11]= 7.06856560254e+012
avgDuration[brk >  11]= 4.92206758231
avgDuration[brk <= 12]= 1.3463934481e+014
avgDuration[brk >  12]= 4.85809027044
avgDuration[brk <= 13]= 2.88512881737e+015
avgDuration[brk >  13]= 4.16407737467
avgDuration[brk <= 14]= 7.06562159355e+016
avgDuration[brk >  14]= -15.6888315529
avgDuration[brk <= 15]= 2.01874902673e+018
avgDuration[brk >  15]= -448.252330083
avgDuration[brk <= 16]= 6.9214252345e+019
avgDuration[brk >  16]= -15368.6513171
avgDuration[brk <= 17]= 2.9663251005e+021
avgDuration[brk >  17]= -658656.48502
avgDuration[brk <= 18]= 1.69504291457e+023
avgDuration[brk >  18]= -37637513.4297
avgDuration[brk <= 19]= 1.45289392678e+025
avgDuration[brk >  19]= -3226072579.69
avgDuration[brk <= 20]= 2.49067530304e+027
avgDuration[brk >  20]= -553041013661
K=0.951488701108
T(work)=154.285714286
T(idle)=7.86619997887
AvgWorkCount=19.9455290141
AvgBrokenCount=0.349664540593
AvgReservedCount=0.704806445265
```

Отрицательные времена пребывания в состоянии? Откуда они появились? Они появляются, когда степень 10 в avgDuration противоположного по смыслу времени стала больше 15. Совпадение? Число двойной точности по стандарту

IEEE 754 хранит 15-16 десятичных знаков. Если степень равна, например, 27, как в последнем примере avgDuration, то допустимая для неё погрешность $\sim 10^{12}$, а $|-553041013661| < 10^{12}$. Точнее считать не позволяет аппаратура... так что не стоит уделять внимание практически невозможным событиям...

5 Проведите имитационный расчёт, полагая все законы показательными (сохранится только матожидание, СКО и прочие такие вещи утрачиваются).

Вернемся к имитационному моделированию, изменив законы времени рабочего состояния и времени восстановления на показательные. Просчитаем.

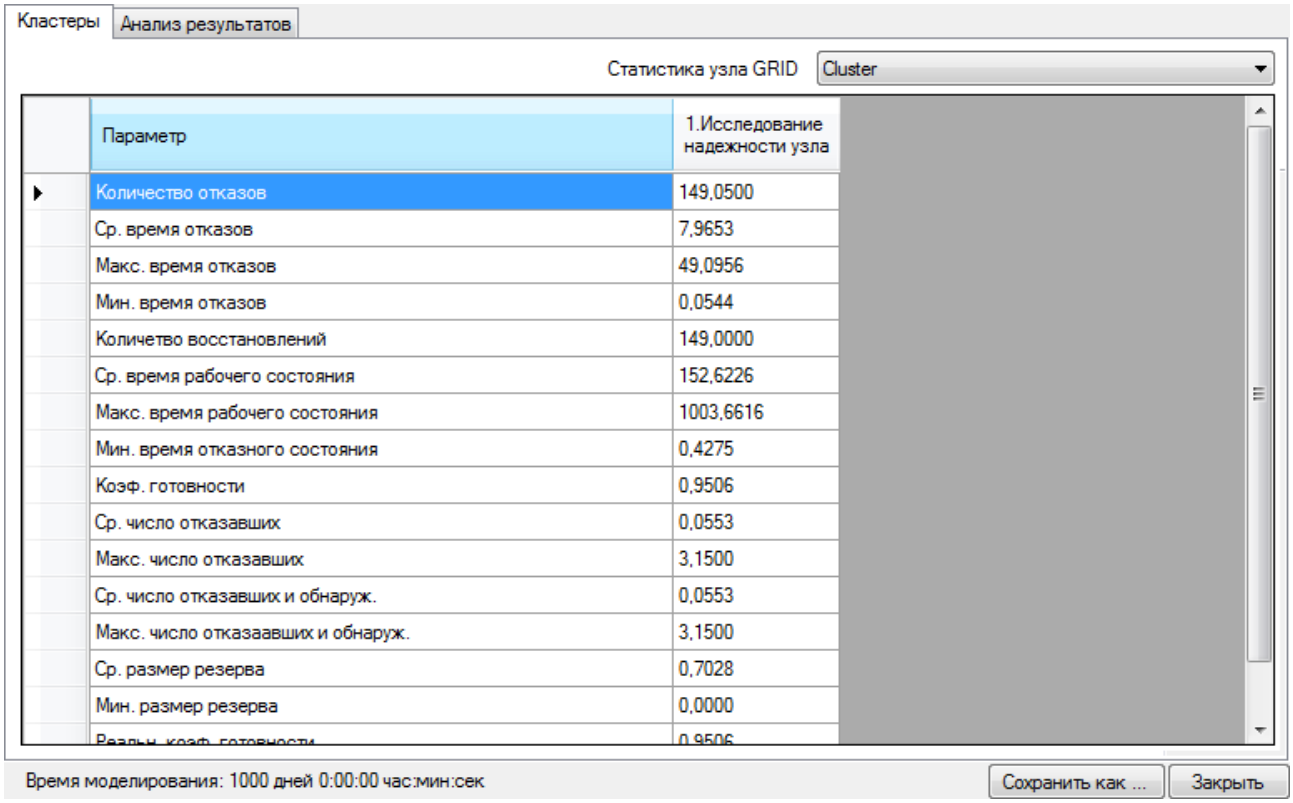


Рис. 12 Результаты расчёта имитационного моделирования при показательном законе

После окончания не забываем присвоить этим двум законам распределения из истинное значение из таблицы.

6 **Включите допустимое время простоя и изменяйте его 0%, 10%, 30%, 50%, 90%, 100%, 150%, 200% от времени простоя, указанного в Вашем варианте.**

Требуется изменять допустимое время простоя.

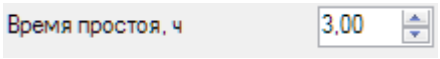


Рис. 13 Изменение времени простоя

Сведем данные в таблицу 1.

Таблица 1.

Допустимое время простоя, ч	0[38]	0.6	1.8	3	5.4	6	9	12
Коэффициент готовности	….	….	….	….	….	….	….	….
Среднее время рабочего состояния	….	….	….	….	….	….	….	….
Среднее время отказного состония	….	….	….	….	….	….	….	….
….	….	….	….	….	….	….	….	….
….	….	….	….	….	….	….	….	….
Реальный коэффициент готовности (коэффициент использования)	….	….	….	….	….	….	….	….

7 **Установите допустимое время простоя в нуль и закон подключения резерва — К(0). Промоделируйте для вероятности необнаружения отказа 0%, 1%, 5%, 10%, 20%, 50%, 70%, 100%. Установите закон распределения времени обнаружения отказа по варианту.**

38 Этот результат мы получили ранее.

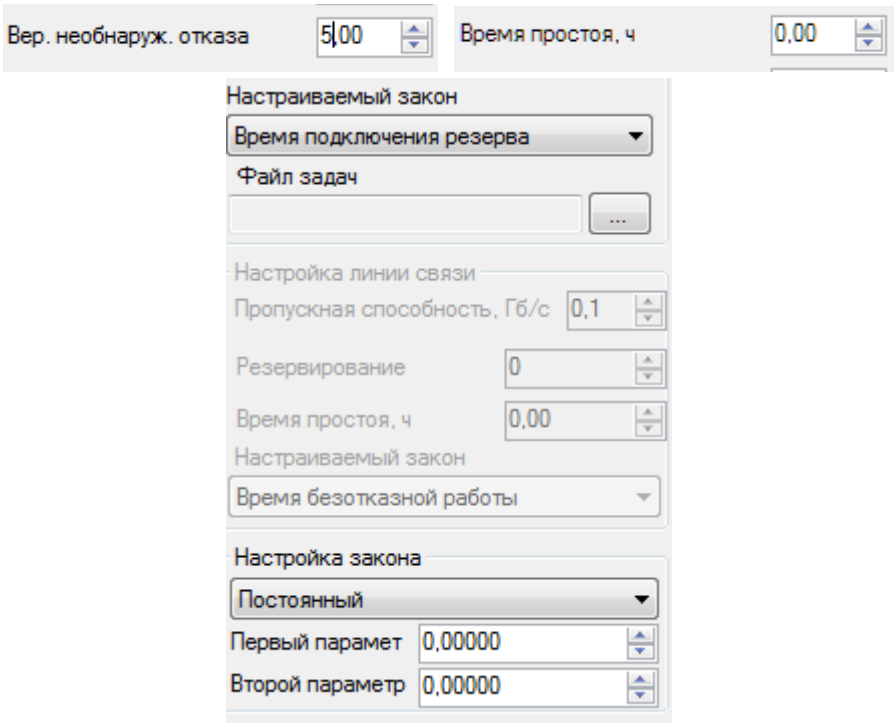


Рис. 14 Установим время простоя равным нулю и Время подключения резерва К(0).

Сведем данные в таблицу 2.

Таблица 2.

Вероятность необнаружения отказа, %	0[39]	1	5	10	20	50	70	100
Коэффициент готовности	….	….	….	….	….	….	….	….
Среднее время рабочего состояния	….	….	….	….	….	….	….	….
Среднее время отказного состония	….	….	….	….	….	….	….	….
….	….	….	….	….	….	….	….	….
….	….	….	….	….	….	….	….	….
Реальный коэффициент готовности (коэффициент использования)	….	….	….	….	….	….	….	….

8 Установите вероятность необнаружения отказа в 0%. Возьмите закон распределения времени подключения резерва и допустимое

39 Этот результат мы получили ранее.

время простоя по варианту, введите их в программу. Осуществите моделирование, зафиксируйте результаты.

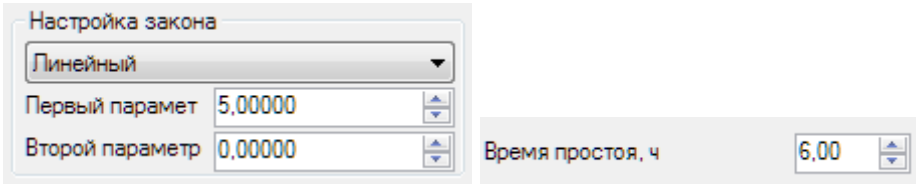


Рис. 15 Установим время простоя и Время подключения резерва по варинту

Выполнив моделирование на достаточном числе прогонок получис данные, отображенные на рис. 16.

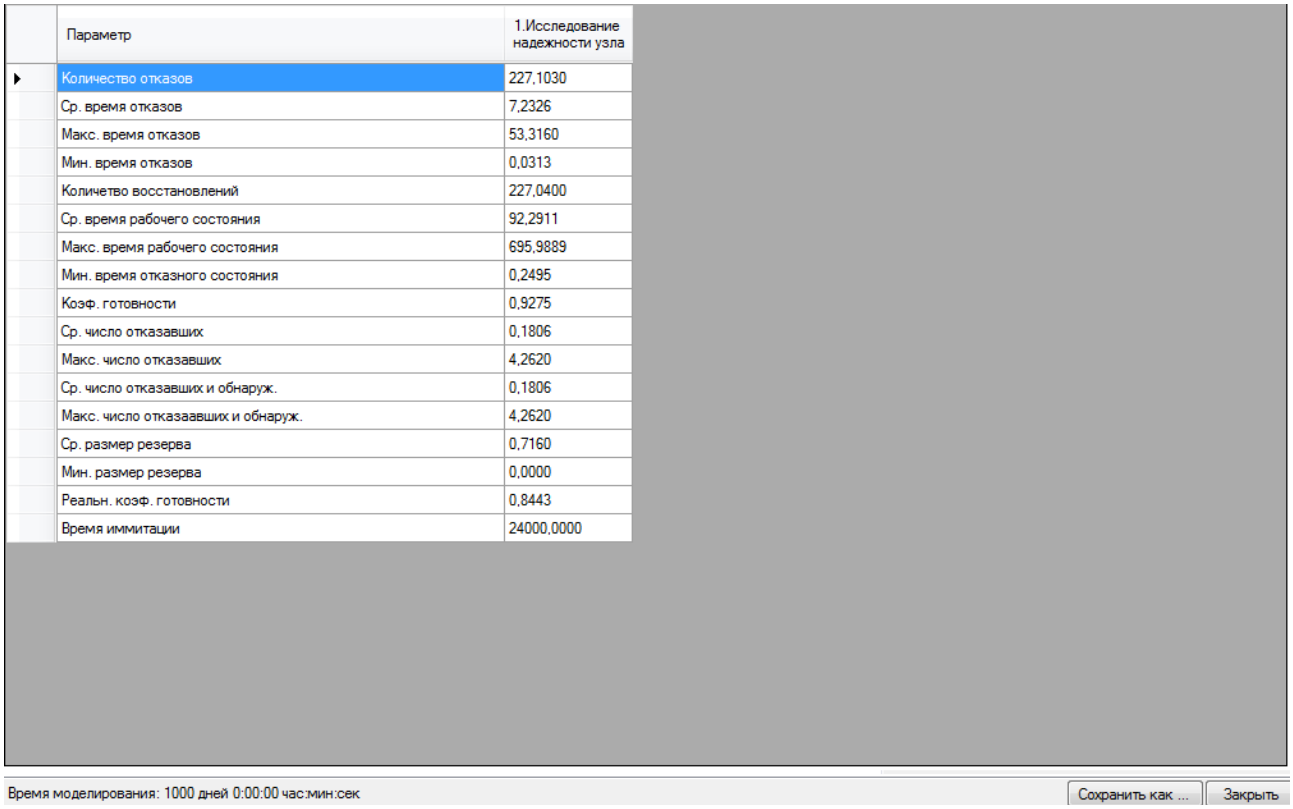
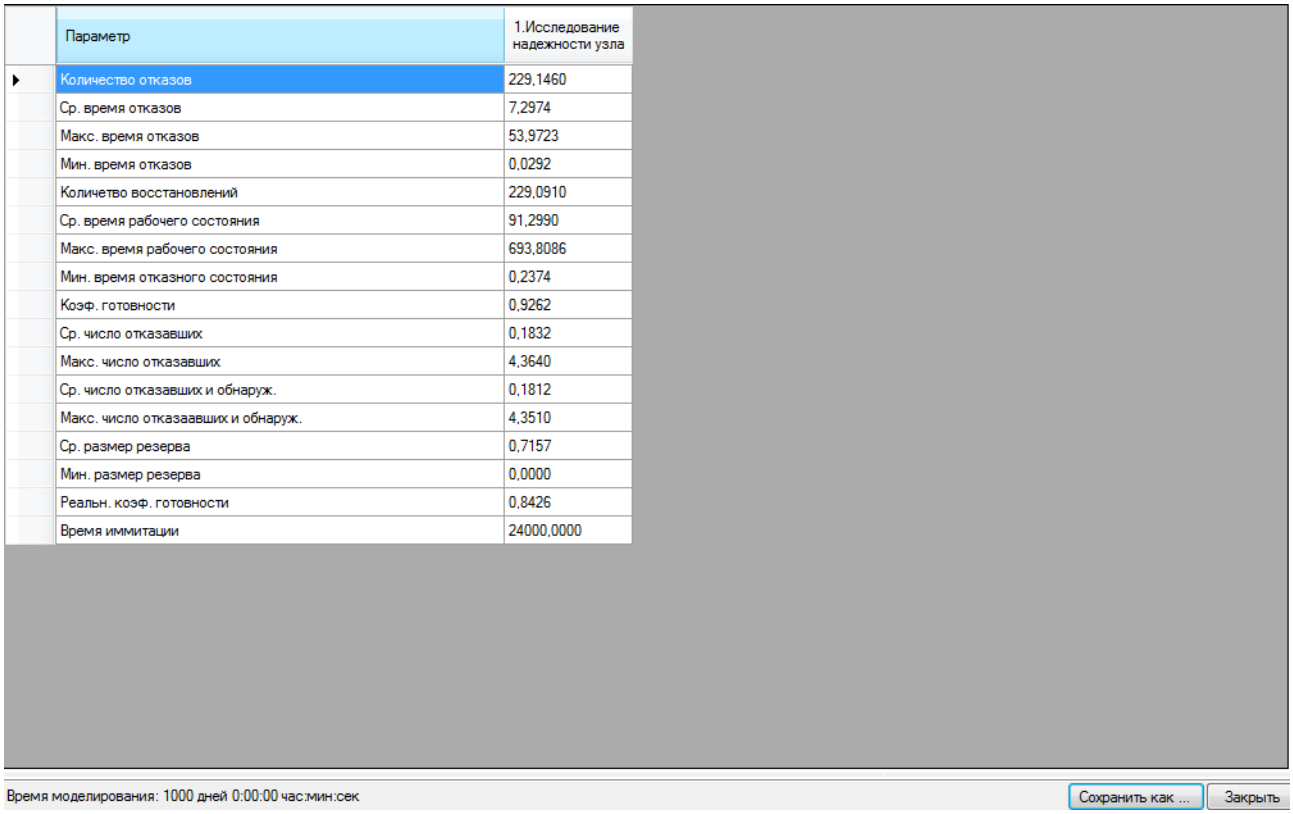


Рис. 16 Результаты моделирования при полном контроле

9 **Осуществите полное моделирование, установив все параметры по варианту.**

Полагаю комментарии излишни: все делается так же, как и в предыдущих пунтах.



Параметр	1.Исследование надежности узла
Количество отказов	229,1460
Ср. время отказов	7,2974
Макс. время отказов	53,9723
Мин. время отказов	0,0292
Количетво восстановлений	229,0910
Ср. время рабочего состояния	91,2990
Макс. время рабочего состояния	693,8086
Мин. время отказного состояния	0,2374
Коэф. готовности	0,9262
Ср. число отказавших	0,1832
Макс. число отказавших	4,3640
Ср. число отказавших и обнаруж.	0,1812
Макс. число отказаавших и обнаруж.	4,3510
Ср. размер резерва	0,7157
Мин. размер резерва	0,0000
Реальн. коэф. готовности	0,8426
Время иммитации	24000,0000

Рис. 17 Результаты моделирования всего задания

10.2 Лабораторная работа №2. Пример топологии для индивидуального задания

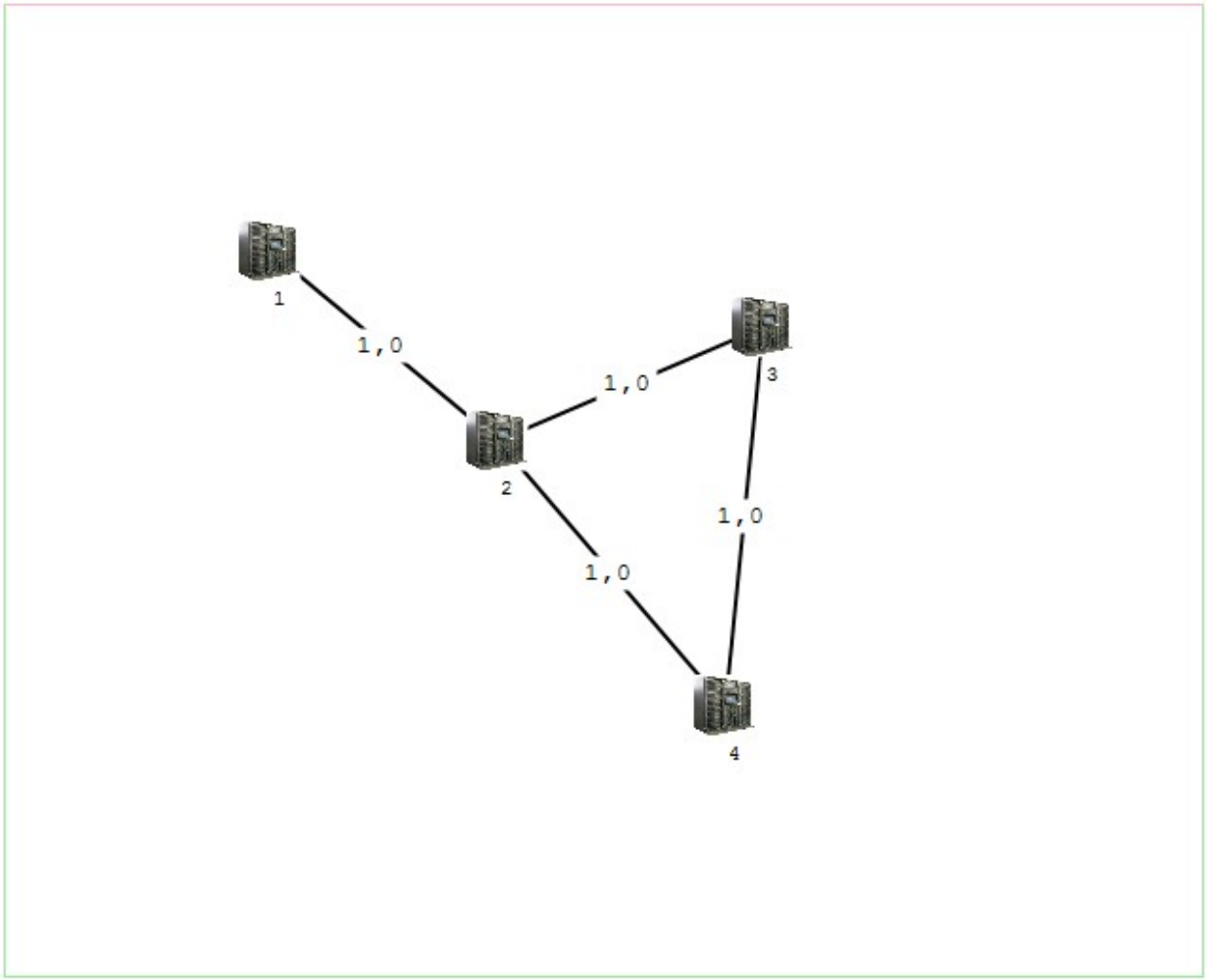


Рис. 10.2.1 Пример топологии[40]

Ещё раз напомним, что все моделирования ведётся при абсолютной надёжности. Моделирование осуществляется на всех стратегиях, кроме Минимального риска. Порядок указания стратегий такой же, как и в окне «Параметры стратегии».

40 Данный раздел создан, чтобы показать ход решения. Правильность конкретных данных не гарантируется!

Таблица 10.2.1 — Параметры кластеров топологии

Параметр	Кластер			
	1	2	3	4
Узлов	64	128	512	128
Ядер	2	4	4	4
Частота	2,000	2,330	2,330	3,000
Flops/Hz	4	4	4	4
Появление заданий, ч	П(2)	∞	∞	∞
Требуемые заданию узлы	Р*(50;49)	—	—	—
Приоритеты задач	К(4)	—	—	—
Память для задачи, Гб	Н(3;1)	—	—	—
Память для результата, Гб	Н(2;0,5)	—	—	—
Сложность заданий, TFlop	Н(1000;170)	—	—	—

Таблица 10.2.2 — Параметры линий связи

Параметр	Линии связи			
	1 — 2	2 — 3	2 — 4	3 — 4
Пропускная способность Гб/с	1,0	1,0	1,0	1,0

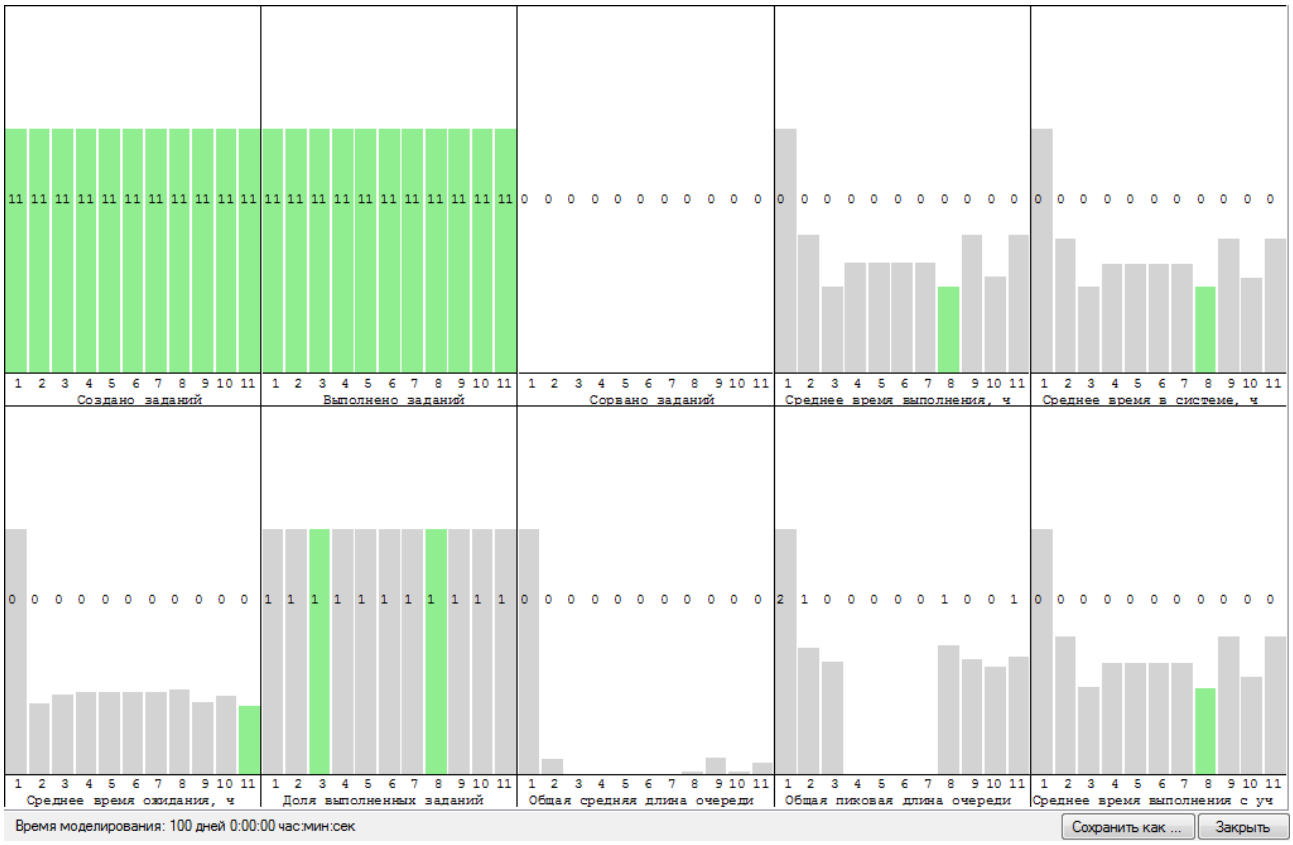


Рис. 10.2.2 — Диаграммы параметров системы

Таблица 10.2.3 — Параметры системы

Параметр	11 стратегий										
Ср. время выполнения задания	0,7812	0,4404	0,2777	0,3552	0,3552	0,3552	0,3552	0,276	0,4415	0,3095	0,4409
Ср. время выполнения задания с учетом прерываний	0,7812	0,4404	0,2777	0,3552	0,3552	0,3552	0,3552	0,276	0,4415	0,3095	0,4409
Ср. длина очереди	0,0192	0,0012	0,0001	0	0	0	0	0,0002	0,0013	0,0002	0,001
Пиковая длина очереди	2,08	1,08	0,96	0	0	0	0	1,1	0,98	0,92	1
Заданий создано	1195,78	1195,78	1195,78	1195,78	1195,78	1195,78	1195,78	1195,78	1195,78	1195,78	1195,78
Выполнений прервано	0	0	0	0	0	0	0	0	0	0	0
ВЫполнений завершено	1195,3	1195,5	1195,58	1195,52	1195,52	1195,52	1195,52	1195,58	1195,52	1195,54	1195,5
Ср. время задания в системе	0,8245	0,453	0,2916	0,3697	0,3697	0,3697	0,3697	0,2909	0,4542	0,3232	0,4531
Ср. время ожид. задания	0,0427	0,0124	0,0139	0,0143	0,0143	0,0143	0,0143	0,0148	0,0126	0,0136	0,0119
Доля выполненных заданий	0,9996	0,9998	0,9998	0,9998	0,9998	0,9998	0,9998	0,9998	0,9998	0,9998	0,9998

ПРИЛОЖЕНИЕ А. Доказательство формулы генерации двух независимых чисел, распределённых по нормальному закону, без использования суммирования (ЦПТ)[41] и спецфункций

Рассмотрим точку (x;y), координаты которой распределены по H(0;1).

$$\rho(x)=\frac{1}{\sqrt{2\pi}}\cdot e^{-\frac{x^2}{2}}$$

$$\rho(y)=\frac{1}{\sqrt{2\pi}}\cdot e^{-\frac{y^2}{2}}$$

Найдем закон распределения пары значений:

$$dP(x,y)=\rho(x,y)\,dx\,dy=\rho(x)\,dx\cdot\rho(y)\,dy=\frac{1}{\sqrt{2\pi}}\cdot e^{-\frac{x^2}{2}}\,dx\cdot\frac{1}{\sqrt{2\pi}}\cdot e^{-\frac{y^2}{2}}\,dy=\frac{1}{2\pi}\cdot e^{-\frac{x^2+y^2}{2}}\,dxdy$$

Перейдем от декартовым координат к полярным:

$$\rho^2=x^2+y^2$$

В соответствии с якобианом:

$$dxdy=\rho\,d\rho\,d\varphi$$

А, значит:

$$d\tilde{P}(\rho,\varphi)=\tilde{\rho}(\rho,\varphi)d\rho\,d\varphi=\rho(x(\rho,\varphi),y(\rho,\varphi))\rho\,d\rho\,d\varphi=\frac{1}{2\pi}\cdot e^{-\frac{\rho^2}{2}}\rho\,d\rho\,d\varphi=e^{-\frac{\rho^2}{2}}d\left(\frac{\rho^2}{2}\right)\cdot\frac{d\varphi}{2\pi}$$

Плотность распределения равна произведению плотностей распределения. Значит, полярный радиус и полярный угол распределены независимо, причём $\frac{\rho^2}{2}$ распределено по П(1), а φ — равномерно.

Т. е. эти величины можно сгенерировать, используя:

$$\frac{\rho^2}{2}=-\ln(1-R_0)$$

$$\varphi=2\pi R_1$$

Учитывая, что полярный радиус всегда неотрицателен:

$$\rho=\sqrt{-2\ln(1-R_0)}$$

Возвращаемся к декартовым координатам:

41 ЦПТ = Центральная предельная теорема

$$x=\sqrt{-2\ln(1-R_0)}\cos(2\pi R_1)$$

$$y=\sqrt{-2\ln(1-R_0)}\sin(2\pi R_1)$$

Эти значения независимы! Т. е. мы взяли две независимые равномерно распределённые величины и получили две независимые нормированные нормально распределённые величины.

Примечание. Величины x^2 и y^2 распределены независимо по χ^2 с одной степенью свободы. Величина x^2+y^2 распределена по χ^2 с двумя степенями свободы. Можно доказать, что χ^2 с μ степенями свободы есть гамма-распределение порядка $\nu=\frac{\mu}{2}$ и $\lambda=\frac{1}{2}$.

Параметры гамма-распределения таковы:

$$m=\frac{\nu}{\lambda}$$

$$t\in[0;\infty)$$

$$f(t)=\lambda^{\nu}\cdot\frac{t^{\nu-1}}{\Gamma(\nu)}\cdot e^{-\lambda t}$$

$$F(t)=\frac{1}{\Gamma(\nu)}\cdot\int_0^{\lambda t}\tau^{\nu-1}e^{-\tau}d\tau=\frac{\gamma(\nu,\lambda t)}{\Gamma(\nu)}=P(\nu,\lambda t)$$

При $\nu\in\mathbb{N}$ гамма-распределение является распределением Эрланга порядка $K=\nu-1$ [42]. При $\nu=1$ оно переходит в экспоненциальное (показательное) распределение.

Распределение x^2+y^2 , оно же — распределение χ^2 с двумя степенями свободы $\mu=2$, есть именно экспоненциальное (показательное) распределение с $\lambda=1/2$. Тогда $\frac{x^2+y^2}{2}$ - есть экспоненциальное (показательное) распределение с $\lambda=1$. В этом — основа расчётных формул. Использование тригонометрических функций очевидно.

---

42 Если перенумеровать порядки распределения Эрланга, как говорилось выше, распределение Эрланга становится частным случаем гамма-распределения. Свойство «порядок суммы гамма распределений с одинаковым λ равен сумме порядков с тем же λ » работает и для вещественного порядка. Это можно легко доказать, обратившись к характеристической функции гамма-распределения.

Примечание 2. Если Вам нужное только одно значение нормально распределённой величины, действуйте по одной из схем:

1) Генерируйте и используйте только одно значение:

$$t = m + \sigma\sqrt{-2\ln(1-R_0)}\cos(2\pi R_1)$$

2) Генерируйте два значения, используйте первое, а второе сохраняйте до следующего вызова процедуры[43]:

$$t_{current} = m_0 + \sigma_0\sqrt{-2\ln(1-R_0)}\cos(2\pi R_1)$$

$$t_{next} = m_1 + \sigma_1\sqrt{-2\ln(1-R_0)}\sin(2\pi R_1)$$

43 Не используйте статистические переменные процедур! Это делает функции не реентерабильными! А, значит, нормальное распараллеливание станет не возможным.

ПРИЛОЖЕНИЕ Б. Написание генератора случайных чисел, равномерно распределённых по интервалу от нуля до единицы

Рассмотрим следующий пример, взятый из [9, с. 14]. Он переписан на Java с Fortran`a. Обращу внимание, что тип int/Integer знаковый и имеет разрядность 32-бита, как и в Fortran. Значит, $Integer.\mathrm{MAX\_VALUE}=2^{31}-1$.

```
package myrandom;

public class MyRandom {

	private int last = 65539;

	public MyRandom()
	{	}

	public MyRandom(int seed)
	{
		last = ((seed << 1) | 1) & Integer.MAX_VALUE;
	}


	protected int NextInternal()
	{
		last *= -65539;
		last &= Integer.MAX_VALUE;
		return last;
	}

	public final int INCLUDE_ZERO = -1, EXCLUDE_ALL = 0, INCLUDE_ONE =+1;

	public double Next()
	{
		return Next(INCLUDE_ZERO);
	}

	public double Next(int inc)
	{
		if(inc != INCLUDE_ZERO && inc != INCLUDE_ONE && inc != EXCLUDE_ALL)
			throw new IllegalArgumentException("Incorrect \"inc\" -
value!");
		return (NextInternal() * 1.0 + inc) * .46566128730773926E-9;
	}
}
```

Вызовы Next() и Next(Myrandom.INCLUDE_ZERO) возвращают $R\in[0;1)$, а Next(Myrandom.EXLUDE_ALL) — $R\in(0;1)$ и Next(Myrandom.INCLUDE_ONE) — $R\in(0;1]$ [44]. Все остальные функции реализуются из этих[45]. Период такого генератора 2^{29} . Т. е. через такое количество запросов значения начинают повторяться.

44 Используя это число, можно сразу получать $1-R\in(0;1]$ для логарифмирования и т. д.
45 См. подраздел 2.3.

ПРИЛОЖЕНИЕ В. Пример сборки результата при параллельном исполнении

Вот пример для технологии OpenMP.

```
double global_data[DATA_COUNT] = {0};

#pragma omp parallel
{
      double local_data[DATA_COUNT] = {0};

#pragma omp for nowait
      for(int i = 0; i < ITER_COUNT; i++)
      {
            ...

            for(int j = 0; j < DATA_COUNT; j++)
                  local_data[j] += ...
      }

      for(int j = 0; j < DATA_COUNT; j++)
#pragma omp atomic
            global_data[j] += local_data[j];

}
```

А вот пример для MPI:

```
int rank = 0, size = 1;

MPI_Init(&argc, &argv);
MPI_Comm_rank(MPI_COMM_WORLD, &rank);
MPI_Comm_size(MPI_COMM_WORLD, &size);

struct indata; //This must be POD struct!!!
//That means this struct CANNOT contain objects or pointers
//but it CAN contain other POD structs

if(rank == 0)
{
      indata = ...
}

MPI_Bcast(&indata, sizeof(indata), MPI_BYTE, 0, MPI_COMM_WORLD);

double global_data[DATA_COUNT] = {0};

double local_data[DATA_COUNT] = {0};

for(int i = rank; i < ITER_COUNT; i+=size)
{
      ...

      for(int j = 0; j < DATA_COUNT; j++)
            local_data[j] += ...
}
```

```
      MPI_Reduce(local_data, global_data, DATA_COUNT, MPI_DOUBLE, MPI_SUM, 0,
MPI_COMM_WORLD);
      //Or  :  MPI_Allreduce(local_data,  global_data,  DATA_COUNT,  MPI_DOUBLE,
MPI_SUM, MPI_COMM_WORLD); if you REALLY need this!

      if(rank == 0)
      {
            ...
      }

      MPI_Finalize();
```

Список использованных источников

1 Грид // Википедия. – [2009]. – Режим доступа : http://ru.wikipedia.org/wiki/GRID

2 Дзеба, В. В. Моделирование GRID систем [Электронный ресурс] / В. В. Дзеба. – [2010] . – Режим доступа : http://masters.donntu.edu.ua/2008/fvti/dzeba/diss/referat.htm

3 Жариков, Д.Н. Моделирование высоконадёжных гетерогенных вычислительных систем / Д.Н. Жариков, В.С. Лукьянов // Успехи современного естествознания. - 2010. - № 5. - С. 101-104.

4 Имитационная модель гетерогенной вычислительной системы / В.С. Лукьянов, Д.Н. Жариков, С.В. Гаевой, Д.С. Попов // Изв. ВолгГТУ. Серия "Актуальные проблемы управления, вычислительной техники и информатики в технических системах". Вып. 11 : межвуз. сб. науч. ст. / ВолгГТУ. - Волгоград, 2011. - № 9. - С. 85-88.

5 Интернет-портал по грид-технологиям :: GRIDCLUB.RU [Электронный ресурс]. – [2011]. – Режим доступа : http://gridclub.ru/

6 Каменщиков, М. А. Сервисы GRID, как объекты стандартизации [Электронный ресурс] / М. А. Каменщиков // Журнал радиоэлектроники. – 2003. – № 12. – Режим доступа : http://jre.cplire.ru/jre/dec03/4/text.html

7 Коваленко, В. Эволюция и проблемы Grid [Электронный ресурс] / В. Коваленко, Д. Корягин // Открытые системы. – [2009]. – Режим доступа : http://www.osp.ru/os/2003/01/182396/_p2.html

8 Лукьянов, В. С. Влияние встроенных контрольных устройств на надежность резервируемой восстанавливаемой аппаратуры / В. С. Лукьянов // Вопросы радиоэлектроники. Сер. 12. Общетехническая. – 1966. – Вып. 25. – С. 65 – 73.

9 Лукьянов, В. С. Модели анализа вероятностно-временных характеристик сетей передачи данных : монография / В. С. Лукьянов, А. В.

Старовойтов, И. В. Черковский ; ВолгГТУ. – Волгоград : Политехник, 2006. – 176 с.

10 Лукьянов, В. С. Проектирование компьютерных сетей методами имитационного моделирования: учеб. пособие / В. С. Лукьянов, Г. В. Слесарев ; ВолгГТУ. – Волгоград : Политехник, 2001. - 74 с.

11 Моделирование отказоустойчивых GRID-систем / В. С. Лукьянов, Д. Н. Жариков, С. В. Гаевой, О. В. Шаповалов // Инновации на основе информационных и коммуникационных технологий : матер. междунар. науч.-практ. конф. (Россия, г. Сочи, 1-10 окт. 2010 г.) / Московский гос. ин-т электроники и математики МИЭМ (ТУ) [и др.]. - М., 2010. - С. 253-254.

12 Моделирование GRID-систем / В. С. Лукьянов, Д. Н. Жариков, С. В. Гаевой, Ю. В. Шафран // Информационные технологии моделирования и управления. – 2009. – № 5. – С. 669 – 677.

13 Научная электронная библиотека eLibrary.ru [Электронный ресурс]. – [2011]. – Режим доступа : http://elibrary.ru/

14 Проблемы моделирования GRID-систем и их реализация [Электронный ресурс] / О. И. Самоваров [и др.] // Портал «Информационно-коммуникационные технологии в образовании». – [2010]. – Режим доступа : http://www.ict.edu.ru/vconf/files/9451.pdf

15 Родин, А. В. Классификации распределенных систем [Электронный ресурс] / А. В. Родин, В. Л. Бурцев. – [2010] . – Режим доступа : http://gridclub.ru/library/publication.2006-02-07.1818516730/publ_file/

16 Свид. о гос. Регистрации программы для ЭВМ № 2010610693 от 20 янв. 2010 г. РФ, МПК (нет). Имитационная модель грид-системы (GridModel) / В. С. Лукьянов, Д. Н. Жариков, С. В. Гаевой, Ю. В. Шафран; ВолгГТУ. - 2010.

17 Системы Grid-вычислений — перспектива для научных исследований / А. Е. Дорошенко [и др.] // Проблеми програмування. – 2005. – №1. – С. 14 – 38.

18 Строганов, Д. В. Модели стохастической аппроксимации в управляемом имитационном эксперименте / Д. В. Строганов, С. О. Польшин, О. Л. Снеткова // Известия ВолгГТУ. Серия «Актуальные проблемы управления, вычислительной техники и информатики в технических системах». – Волгоград. Изд-во ВолгГТУ, №2, 2008. – С. 23-25.

19 Труханов, В. М. Краткий курс теории и практики надежности сложных систем : учеб. пособие / В. М. Труханов ; ВолгГТУ. – 2-е изд., перераб и доп. – Волгоград : Политехник, 2008. – 162 с.

20 Фоменков, С. А. Моделирование систем [Электронный ресурс] / С. А. Фоменков. – Волгоград, [2004]. – 1CD-ROM

21 Чернышов, К. В. Показатели надежности технических систем : наработка до отказа, ресурс, срок службы : учеб. пособие / К. В. Чернышов ; Волгоград. гос. ун-т. – Волгоград : Политехник, 2007. – 80 с.

22 Шелестов, А. Ю. Подходы и средства моделирования GRID -систем обработки спутниковых данных [Электронный ресурс] / А. Ю. Шелестов. – Бібліотека ІПС, [2009] . – Режим доступа : http://eprints.isofts.kiev.ua/447/

23 Шеннон, Р. Имитационное моделирование систем – искусство и наука / Р. Шеннон ; пер. с англ. под ред. Е. К. Масловского. – М. : Мир, 1978. – [418 с.]

24 Эвристики распределения задач для брокера ресурсов Grid [Электронный ресурс] / А.И. Аветисян [и др.] . – [2009]. – Режим доступа : http://www.citforum.ru/nets/digest/grid/index.shtml

25 A toolkit for modelling and simulating Data Grids: An extension to GridSim [Электронный ресурс] / A. Sulistio [et al.]. – [2009]. – Режим доступа : http://www.gridbus.org/reports/datagrid_fgcs.pdf

26 Buyya, R. GridSim: a toolkit for the modeling and simulation of distributed resource management and scheduling for Grid computing [Электронный ресурс] / R. Buyya, M. Murshed. – [2010]. – Режим доступа : http://www.buyya.com/papers/gridsim.pdf

27 Dobre, C. Monarc simulation framework [Электронный ресурс] / C. Dobre, C. Stratan. – [2009]. – Режим доступа : http://monarc.cacr.caltech.edu:8081/Papers/CorinaStratan_CiprianDobre_MONARC.pdf

28 MicroGrid 2.4.6 User Guide [Электронный ресурс] / Concurrent Systems Architecture Group. – [2010]. – Режим доступа : http://www-csag.ucsd.edu/projects/grid/mgrid2-user.html

29 Microsoft Developer Network (MSDN) [Электронный ресурс]. – [2010]. – Режим доступа : http://msdn.microsoft.com

30 OptorSim: A simulation tool for scheduling and replica optimisation in data grids [Электронный ресурс] / D. G. Cameron [et al.]. – [2009]. – Режим доступа : http://www.gridpp.ac.uk/papers/chep04_optorsim.pdf

31 Overview of a performance evaluation system for global computing scheduling algorithms [Электронный ресурс] / Takefusa, A. [et al.]. – [2010]. – Режим доступа : http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.101.1306

32 Quinson, M. The SimGrid Framework for Research on Large-Scale Distributed Systems - Second Part [Электронный ресурс] / M. Quinson, A. Legrand, H. Casanova. – [2010]. – Режим доступа : http://mescal.imag.fr/membres/pedro.velho/_workshop/simgrid/simgrid-tutorial-part-2.pdf

33 Ranganathan, K. Decoupling Computation and Data Scheduling in Distributed Data-Intensive Applications [Электронный ресурс] / K. Ranganathan, I. Foster. – [2009]. – Режим доступа : http://www.globus.org/alliance/publications/papers/decouple.pdf

34 The MicroGrid: Using Online Simulation to Predict Application Performance in Diverse Grid Network Environments [Электронный ресурс] / H. Xia [et al.]. – [2009]. – Режим доступа : http://www-csag.ucsd.edu/papers/clade04xia.pdf

35 Имитационное моделирование грид-систем : монография / Лукьянов В.С., Андреев А.Е., Жариков Д.Н., Островский А.А., Гаевой С.В.; ВолгГТУ. - Волгоград, 2012. - 215 с.

36 Гаевой, С.В. Аппроксимация стохастических параметров вычислительного кластера на примере LANL CM5 / Гаевой С.В., Аль-Хадша Ф.А.Х. // Perspektywiczne opracowania sa nauka i technikami – 2013 : mater. IX miedzynarod. nauk.-prakt. konf., 7–15 listopada 2013 r. Vol. 33. Matematyka. – Przemysl, 2013. – S. 67–70.

37 Гаевой, С.В. Вероятностно-временные характеристики обслуживания отдельно взятой заявки в СМО / Гаевой С.В., Аль-Хадша Ф.А.Х. // Инновационные информационные технологии : матер. междунар. науч.-практ. конф., г. Прага, Чехия, 22-26 апр. 2013 г. В 4 т. Т. 2 / МИЭМ НИУ ВШЭ [и др.]. - М., 2013. - С. 152-154.

38 Гаевой, С.В. Детерминированная имитационная модель кластеров грид-системы, обслуживающих задания / Гаевой С.В., Аль-Хадша Ф.А.Х., Лукьянов В.С. // Вестник компьютерных и информационных технологий. - 2014. - № 6. - С. 39-43.

39 Свид. о гос. регистрации программы для ЭВМ № 2013614201 от 25 апреля 2013 г. РФ, МПК (нет). Имитационная модель для оценки влияния параметров надёжности и иных характеристик на производительность кластерной системы (SrvModel) / Гаевой С.В., Лукьянов В.С.; ВолгГТУ. - 2013.

40 Гаевой, С.В. Моделирование работы вычислительного кластера на примере LANL CM5 [Электронный ресурс] / Гаевой С.В., Аль-Хадша Ф.А.Х. // SCI-ARTICLE.RU : электронный периодический научный журнал. - 2013. - № 3 (ноябрь). - С. 304-313. - Режим доступа : http://sci-article.ru/stat.php?i=modelirovanie_rabotY_vYchislitelnogo_klastera_na_primere_LANL_CM5.

41 Гаевой, С.В. Оценка вероятности обслуживания заявки в СМО путём имитационного моделирования / Гаевой С.В. // Перспективы развития информационных технологий : сб. матер. XI междунар. науч.-практ. конф.

(Новосибирск, 28 февр. 2013 г.) / Центр развития научного сотрудничества (ЦРНС). - Новосибирск, 2013. - С. 13-18.

42 Гаевой, С.В. СМО с заявками, исполняемыми несколькими каналами / Гаевой С.В., Аль-Хадша Ф.А.Х., Лукьянов В.С. // России – творческую молодёжь : матер. I всерос. науч.-практ. студ. конф., г. Камышин, 22-23 мая 2013 г. В 4 т. Т. 2 / ВолгГТУ, КТИ (филиал) ВолгГТУ. - Волгоград, 2013. - С. 26-27.

43 Гаевой, С.В. Эвристики распределения заданий в системах обслуживания / Гаевой С.В., Лукьянов В.С. // Инновации в технологиях и образовании : сб. ст. участников VI междунар. науч.-практ. конф. (17-18 мая 2013 г.). В 4 ч. Ч. 2 / Филиал Кузбасского гос. техн. ун-та в г. Белово, Великотырновский ун-т им. Святых Кирилла и Мефодия (Болгария). - Белово, 2013. - С. 187-191.

44 Гаевой, С.В. Эвристики распределения заявок в Грид-системах (Grid) / Гаевой С.В., Аль-Хадша Ф.А.Х., Лукьянов В.С. // Perspektywiczne opracowania sa nauka i technikami – 2013 : mater. IX miedzynarod. nauk.-prakt. konf., 7–15 listopada 2013 r. Vol. 33. Matematyka. – Przemysl, 2013. – S. 63-66.

45 Фоменков, С. А. Математическое моделирование системных объектов: учеб. Пособ. (гриф). Доп. УМО вузов по университетскому политехн. образованию / С. А. Фоменков, Д. А. Давыдов, В. А. Камаев; ВолгГТУ. - Волгоград : РПК "Политехник", 2006. - 180 с.