**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ»**

Факультет физико-математических и естественных наук

Кафедра прикладной информатики и теории вероятностей

«Допустить к защите»

Заведующий кафедрой
прикладной информатики
и теории вероятностей
д.т.н., профессор
____________ К.Е. Самуйлов

«____»_____________ 20___г.

Выпускная квалификационная работа магистра

Направление 01.04.02 «Прикладная математика и информатика»

Магистерская программа «Теория вероятностей и математическая статистика»

ТЕМА «Моделирование стохастических систем с запаздыванием»

Выполнила студентка **Апреутесей Анна Мария Юрьевна**

(Фамилия, имя, отчество)

Группа НПМмд-02-19

Студ. билет № 1032193049

Руководитель выпускной квалификационной работы

Королькова А.В., к.ф.-м.н., доцент, доцент кафедры прикладной информатики и теории вероятностей

(Ф.И.О., степень, звание, должность)

____
(Подпись)

Автор ___________________________
(Подпись)

г. Москва
2021 г.

Федеральное государственное автономное образовательное учреждение высшего образования

«Российский университет дружбы народов»

АННОТАЦИЯ

выпускной квалификационной работы

Апреутесей Анны Марии Юрьевны

(фамилия, имя, отчество)

на тему: Моделирование стохастических систем с запаздыванием

Данная работа посвящена моделированию гибридных стохастических систем с запаздыванием, в качестве которой рассматривается система с управлением, состоящей из входящего Transmission Control Protocol (TCP) потока и маршрутизатора, обрабатывающего трафик по алгоритму Random Early Detection (RED). Целью данной работы является моделирование гибридных стохастических систем с запаздывание посредством Modelica и Julia.

В ходе данной работы было рассмотрено применение гибридного подхода к моделированию стохастических систем с запаздыванием на примере системы передачи данных по протоколу TCP с политикой активного управления очередью, в качестве которой выступает алгоритм RED. В результате моделирования представлена программная реализация данной системы и проведен сравнительный анализ возможностей языков Modelica и Julia по моделированию гибридных стохастических систем с запаздыванием.

Автор ВКР ____________ ________________________

(Подпись) (ФИО)

Оглавление

Оглавление 3

Список используемых сокращений 4

Введение 5

1. Методы и анализ моделирования стохастических систем 8

  1.1 Обзор исследований в области анализа стохастических систем 8

  1.2 Обзор исследований в области моделирования стохастических систем 11

2. Стохастическая система с запаздыванием 15

  2.1 Описание процесса передачи трафика с управлением динамической интенсивностью потока по алгоритму типа RED 15

  2.2 Математическая модель системы с управлением по алгоритму RED 17

3. Моделирование стохастической системы с запаздыванием 23

  3.1 Математическое моделирование алгоритма RED 25

  3.2 Анализ полученных в ходе моделирования результатов 34

Заключение 43

Список литературы 45

Приложение 1 51

Приложение 2 53

Список используемых сокращений

Англоязычные сокращения:

AQM	Active Queue Management
cwnd	Source Congestion Window
RED	Random Early Detection
RTT	Round Trip Time
ssthr	Slow Start Threshold
TCP	Transmission Control Protocol
UML	Unified Modeling Language

Русскоязычные сокращения:

СДУ	Система дифференциальных уравнений

Введение

Данная работа посвящена моделированию гибридных стохастических систем с запаздыванием, в качестве которой рассматривается система с управлением, состоящей из входящего Transmission Control Protocol (TCP) потока и маршрутизатора, обрабатывающего трафик по алгоритму Random Early Detection (RED).

Актуальность работы

Актуальность работы заключается в рассмотрении проблематики компьютерного моделирования стохастических систем, систем с запаздыванием и гибридных систем, сочетающих в себе работу непрерывных элементов и элементов управления с дискретным характером функционирования.

Цель работы

Целью данной работы является моделирование гибридных стохастических систем с запаздывание посредством Modelica и Julia.

Задачи работы

Основными задачами данной работы являются:

1. Применение гибридного подхода к моделированию стохастических систем с запаздыванием на примере системы передачи данных по протоколу TCP с политикой активного управления очередью, в качестве которого выступает алгоритм RED;
2. Программная реализация модели стохастической системы с запаздыванием на примере системы передачи трафика с регулируемой алгоритмом типа RED динамической интенсивностью потока в средствах Modelica и Julia;
3. Сравнительный анализ возможностей Modelica и Julia по моделированию гибридных стохастических систем с запаздыванием.

Методы исследования

В рамках данной работы применялись следующие методы исследования:

— метод стохастизации одношаговых процессов [43], примененный для описания математической модели передачи трафика с регулируемой алгоритмом типа RED динамической интенсивностью потока;

— решатель DASSL и решатели группы SUNDIALS языка Modelica для решения дифференциальных уравнений;

— библиотека DifferentialEquations.jl, основу которой составляют решатели группы SUNDIALS, для решения дифференциальных стохастических уравнений в Julia;

— метод Эйлера-Маруямы для численного решения СДУ.

Апробация работы

В ходе выполнения работы были получены результаты, представленные на следующих конференциях:

— Всероссийская конференция с международным участием «Информационно-телекоммуникационные технологии и математическое моделирование высокотехнологических систем (ITTMM)» (Москва, РУДН, 2020 г., 2021 г.);

— VI Международная научно-практическая конференция «Системы управления, технические системы: устойчивость, стабилизация, пути и методы исследования» (Елец, Елецкий государственный университет им. И. А. Бунина, 2020 г.);

— XXIII Международная научная конференция «Распределенные компьютерные и телекоммуникационные сети: управление, вычисление, связь (DCCN-2020)» (Москва, РУДН, 2020 г.);

— VII Международная конференция молодых ученых «Информационные технологии, телекоммуникации и системы управления (ITTCS2020)» (Иннополис, Университет Иннополис, 2020 г.);

— Международная конференция «Математика. Компьютер. Образование» (Дубна, Университет Дубны, 2020 г., 2021 г.)

— VII Всероссийская с международным участием студенческая научно-практическая конференция «New Technology» (Москва, РУДН, 2020 г.)

В рамках научной деятельности принимала участие в конкурсе научно-исследовательских работ и проектов студентов РУДН 2020 г., где заняла I место за научно-исследовательскую работу «Компьютерное моделирование кинетических уравнений в представлении стохастических дифференциальных уравнений».

Публикации

По теме выпускной квалификационной работы магистра были опубликованы работы [1-9]. Также были поданы документы для регистрации программ ЭВМ «Гибридное моделирование алгоритма управления трафиком RED» авторов А.М.Ю. Апреутесей и А.В. Корольковой и «Численное моделирование гибридной системы с запаздыванием» авторов А.М.Ю. Апреутесей и Д.С. Кулябова.

Структура работы

Работа состоит из введения, трех глав, заключения, списка литературы и приложения.

Во введении отражена проблематика выбранной темы, актуальность данного исследования в современном мире, поставлены цели и задачи данной работы.

В первой части выполнен обзор литературных источников, посвященных теме выпускной работы.

Во второй части представлено описание процесса передачи трафика с управлением по алгоритму RED динамической интенсивностью потока и описана математическая модель моделируемой системы.

Третья часть посвящена математическому моделированию гибридной стохастической системы с запаздыванием, в качестве которой рассматривается модель передачи трафика по протоколу TCP с регулируемой алгоритмом типа RED динамической интенсивностью потока. Проводится сравнение полученных реализаций с использованием языков Modelica и Julia, приводятся результаты численного эксперимента.

В Заключении описаны результаты и сделаны выводы по проделанной работе, предложены способы использования результатов данного исследования.

В Приложении представлены полные листинги программ, написанных в ходе подготовки Выпускной квалификационной работы.

1. Методы и анализ моделирования стохастических систем

1.1 Обзор исследований в области анализа стохастических систем

В приведенных ниже литературных источниках рассматриваются работы в области моделирования и анализа стохастических систем. Первыми работами, посвящёнными изучению стохастических моделей, считаются исследования, проведенные в XIX веке ученым Р. Броуном, в которых рассматривалось хаотическое тепловое движения атомов и молекул в жидкости или газе. Основные же задачи стохастической динамики были сформулированы Л.С. Понтрягиным, А.А. Андроновым и А.А. Витте в работе «О статистическом рассмотрении динамических систем» (1933 г.) [10]. Основополагающими работами, посвященными математической теории стохастических дифференциальных уравнений, являются работы С.Н. Бернштейна [11,12], И.И. Гихмана [13-15] и К. Ито [16,17]. Работа авторов И.И. Гихмана и А.В. Скорохода 1982 г. [21] посвящена теории стохастических дифференциальных уравнений. В работе проводятся исследования различных аспектов существования и единственности решений уравнений, рассматриваются проблемы устойчивости, приводятся предельные теоремы, а также изучается асимптотическое поведение решений некоторых классов СДУ.

С середины XX века ученые активно интересуются способами компьютерного решения задач стохастической природы. Так, в работе [18] автор одним из первых вводит понятие стохастического программирования при моделировании и анализе задач линейного программирования со случайными коэффициентами. В это же время активно развивается нелинейное программирование, авторы многих работ, посвященных решению задач стохастического характера, применяют методы нелинейного программирования и известных методов численного моделирования. Однако данный подход не является универсальным, со временем становится понятно, что стохастические задачи требуют своих специфических методов. Так, в работе [19] поднимается проблема необходимости развития стохастических методов поиска экстремума в задачах с ограничениями, при решении которых известные методы нелинейного программирования не применяются, особое внимание автор уделяет задачам стохастического программирования. На примере таких задач, как моделирование систем массового обслуживания, управление случайными процессами, планирование в условиях неопределенности и других задач математической статистики автор рассматривает стохастические квазиградиентные методы, в некотором смысле объединяющие методы стохастической аппроксимации и случайного поиска [20].

В работе [22] дается обзор численных методов решения стохастических дифференциальных уравнений с винеровским случайным процессом, излагается новый подход к устойчивости сильных и слабых приближенных решений. Авторы обсуждают такие вопросы математического моделирования, как соотношение между слабыми и сильными приближенными решениям стохастических дифференциальных уравнений, порядок точности приближения и вопросы устойчивости конечно-разностных методов в сильном и слабом случаях.

В работе [23] авторы анализирует различные способы построения моделей, описываемых стохастическими дифференциальными уравнениями Ито для случайных динамических систем. В качестве случайных процессов с дискретным временем авторы рассматривает модель взаимодействия молекул в процессе химической реакции, модель распределения длины хлопкового волокна при производстве хлопковой нити, а также влияние вакцинации на популяцию в эпидемиологической модели.

Авторами описаны два подхода к получению стохастического дифференциального уравнения для случайной динамической системы, состоящей из d компонент, где m различных независимых переходов системы из одного состояния в другое происходят в течение малого промежутка времени ($m \geq d$). В первом подходе рассматривается матрица диффузии B размером $d \times d$, во втором – матрица диффузии G размером $d \times m$. Системы стохастических дифференциальных уравнений Ито, полученных в рамках описываемых авторами подходов, дают сопоставимые результаты, решения данных систем обладают одинаковым распределением вероятностей, а пример решения одной системы является примером решения другой системы.

Авторы делают вывод о том, что каждая из рассмотренных процедур имеет как свои преимущества, так и недостатки, и выбор конкретного подхода зависит от поставленной задачи и имеющимися вычислительными мощностями. Первый подход авторы называют естественным продолжением процедуры моделирования, которая уже в течение многих лет применяется в физике и технике. Так, изменения в детерминированной системе рассматривается за небольшой интервал времени, дифференциальное уравнение получается, когда приращение времени стремится к нулю. Авторы обращают внимание, что несложно получить сходство между прямыми дифференциальными уравнениями Колмогорова, которым удовлетворяют вероятностные распределения стохастических моделей с дискретным и непрерывным временем, и эти сходства позволяют вывести модель СДУ Ито из дискретной стохастической модели. Однако, в данном подходе требуются вычисления квадратного корня из матрицы, в то время как система, полученная при втором подходе с большей матрицей диффузии, требует менее трудоемких вычислений и, соответственно, является более простой в вычислительном отношении системой при условии, что m не является много большей по отношению к d.

Таким образом, в данной работе авторы демонстрируют способы получения стохастических дифференциальных уравнений для случайных процессов с дискретным временем. С помощью разных подходов получены модели процессов, возникающих в химии, текстильной промышленности и эпидемиологии. В результате анализа полученных моделей и расчетов делается вывод об эквивалентности распределений вероятностей для двух моделей СДУ.

Работа [24] посвящена численному моделированию стохастических систем со скачками на примере некоторых биологических моделей. На основе численных методов Эйлера – Маруямы [25] и метода высшего порядка Милштейна, авторы предлагают усовершенствованный численный метод моделирования стохастических дифференциальных уравнений под управлением пуассоновского процесса. Методы, представленные в данной статье, могут быть использованы для приближенного моделирования СДУ с шумом Леви. Авторами был получен численный метод моделирования траекторий выборки для СДУ, управляемых стационарными пуассоновским процессом, проведено математическое моделирование биологической системы хищник-жертва, имеющий скачкообразный характер функционирования.

Таким образом, можно сделать вывод о том, что на сегодняшний день стоит задача усовершенствования существующих численных методов решения стохастических дифференциальных уравнений, в том числе стохастических дифференциальных уравнений с частными производными, т.к. большинство алгоритмов, используемых для решения обыкновенных дифференциальных уравнений, не применимы для стохастических дифференциальных уравнений различных типов и имеют плохую численную сходимость.

1.2 Обзор исследований в области моделирования стохастических систем

Современные исследователи активно занимаются поиском универсальных и доступных инструментов и языков программирования для моделирования стохастических систем, многими научными командами ведется разработка различного программного обеспечения, способного удовлетворить нужды быстро развивающейся сферы математического моделирования стохастических моделей, обсуждаются достоинства и недостатки использования различного программного для стохастического моделирования.

Повсеместное использование дифференциальных уравнений в качестве инструмента для описания физических явлений, биологических явлений, химических реакций, процессов развития и др. привело к появлению различных наборов решателей, например для языка MATLAB [31], Fortran [32]. В статье [33] описывается популярный набор решателей SUNDIALS, компоненты которого используются в библиотеках языков Julia, Python, C, Fortran и др. для решения различных видов дифференциальных уравнений, приводится сравнительная характеристика таких решателей как CVODE, IDA, KINSOL с точки зрения реализации численных методов решения задач дифференциального моделирования.

Авторы книги [26] демонстрируют способы численного поиска решений систем стохастических дифференциальных уравнений с использованием мощностей суперкомпьютера, т.к. многопоточное моделирование подобных систем требует значительных вычислительных затрат. Авторы демонстрируют возможность использования суперкомпьютеров для распараллеливания алгоритмов решения стохастических дифференциальных уравнений, в частности методов Монте-Карло

[27], на примере библиотеки программ PARMONC, написанной на языке C и включающей блок AMIKS для параметрического анализа полученных решений. Авторами были описаны проводимые эксперименты в различных областях науки, приведены результаты численных расчетов стохастических уравнений, авторами также приведены листинги разработанных в ходе написания работы программ, дан анализ полученных результатов вычислительных экспериментов.

Авторы работы [28] на примере моделирования сложной комплексной системы имитации жилой зоны в здании с подогревом, которая принимает в качестве входных данных результаты стохастического моделирования прогноза погоды и прогнозируемую заполняемость, демонстрируют недостатки языка Modelica при моделировании стохастических систем, а также обращают внимание на сложности, с которыми они столкнулись, а именно отсутствие встроенной возможности стохастического моделирования и необходимость использовать дополнительные программные инструменты в данных целях, а также невозможность принимать во время моделирования входные данные, например от внешних датчиков и контроллеров.

Стремясь обойти ограничения Modelica при стохастическом моделировании, авторы описывают подход к расширению возможностей данного языка. Предложенный подход объединяет стохастическое моделирование в дискретном времени и моделирование, в котором стохастическая модель генерирует входные данные для каждого временного шага модели, однако авторы обращают внимание читателей и разработчиков на то, что описанные способы обойти ограничения языка Modelica не являются универсальными и язык требует доработки и усовершенствования в сфере стохастического моделирования.

Автор работы [29] применяет программное обеспечение, написанное на языке MatLab для демонстрации рассмотренного им подхода численного интегрирования. Данный подход к численному интегрированию дифференциальных уравнений основан на стохастических аналогах формулы Тейлора и специальных методах аппроксимации повторных стохастических интегралов, кроме того, в работе рассматривается среднеквадратическая аппроксимация повторных интегралов Стратоновича и Ито. Рассмотренные подходы численного интегрирования реализованы в программных комплексах в среде MatLab на примере решения широкого круга математических задач.

Работа [30] посвящена описанию и анализу созданного авторами инструмента моделирования BioSimulator.jl. Данное программное обеспечение применяется для моделирования основного кинетического уравнения, или как его называют в англоязычной литературе – Master equation. Авторы стремились создать быстрый и удобный инструмент для реализации алгоритма Гиллеспи, τ – прыжка и других алгоритмов стохастического моделирования, в результате чего был написал пакет BioSimulator.jl, основными преимуществами которого является простота и быстрая производительность.

Данный пакет написан на основе языка программирования для научных вычислений Julia, в том числе на базе библиотеки DifferentialEquations.jl. Помимо библиотеки Julia для решения дифференциальных уравнений, авторы также использовали инструменты языков C++, Python и R. BioSimulator.jl реализует набор алгоритмов стохастического моделирования, основанных на теории цепей Маркова. Данный программный инструмент дает возможность строить диаграммы сетей Петри, описывающие взаимодействия элементов системы, строить средние траектории и стандартные отклонения каждого участвующего вида с течением времени и создавать частотные распределения каждого вида в заданное время.

Для демонстрации широкой применимости разработанного программного обеспечения авторы моделируют с его помощью уравнение Михаэлиса-Ментен, стохастическую модель рождения – гибели и модель саморегулирующаяся генетической сети.

В работе [34] авторы демонстрируют преимущества и недостатки языка Julia и некоторых ее библиотек, в том числе DifferentialEquations.jl, для решения дифференциальных уравнений различных видов, в том числе стохастических, также авторами рассматривается применение в языке возможностей множественной диспетчеризации и метапрограммирования. Выбор языка Julia авторы обосновывают ограничениями, связанными с другими языками программирования. Многие решатели [31-33] стохастических дифференциальных уравнений, разработанных на раннем этапе развития C и Fortran, не имеют абстракций для обобщенных форматов массивов. В случаях, когда матрица или тензор более высокой размерности являются представлением дифференциального уравнения, для использования данных решателей пользователю требуется преобразовать моделируемое уравнение в векторное. Более того, эти решатели часто ограничены использованием 640-битных вычислений с плавающей точкой, что ограничивает их использование в приложениях, требующих высокой точности. Векторизованное кодирование, например в рамках NumPy или MATLAB, может не иметь оптимизаций компилятора, которые требуют вывода типа. Это увеличивает вычислительную нагрузку на пользовательские функции, что снижает эффективность решателя.

Julia – сценарный язык, который с точки зрения выполняемых операций может стать заменой таких языков, как R, Python, MATLAB, обеспечивая производительность, присущую низкоуровневым компилируемым языкам. Это позволяет пользователям запускать прототипы в Julia, а также выполнять моделирование сложных систем на одном языке вместо того, чтобы прибегать к использованию нескольких программных средств, зачастую, жертвуя производительностью. Помимо преимуществ языка Julia, авторы описывают текущие ограничения языка, указывают на необходимость более точной документации совместимости и несовместимости библиотек, пакетов и других языков, а также планы дальнейшего развития Julia.

Принимая во внимание вышеописанные работы, представляется интересным рассмотреть применимость для реализации гибридной парадигмы и моделирования стохастических дифференциальных уравнений с запаздыванием объектно-ориентированных языков Julia и Modelica, в основе которых для реализации численных методов решения задач дифференциального моделирования лежат такие решатели как CVODE и IDA.

2. Стохастическая система с запаздыванием

2.1 Описание процесса передачи трафика с управлением динамической интенсивностью потока по алгоритму типа RED

В качестве исследуемой стохастической системы рассмотрим модель процесса регулирования состояния потока при возникновении перегрузок маршрутизатора, в качестве которого рассматривается алгоритм Random Early Detection (RED), при передаче данных по протоколу Transmission Control Protocol (TCP) [35].

Механизм TCP функционирует как транспортный протокол с установлением логического соединения, осуществляет управление потоком данных и исправлением ошибок в сети, используя метод передачи с применением окон. Целость передаваемых данных обеспечивается благодаря механизму дублирования запросов в случае потери данных. В случае получения двух копий одного пакета, протокол устраняет дублирование и уведомляет отправителя о результатах передачи. Процесс работы данного алгоритма контроля и управления перегрузками, регулирующий интенсивность передаваемого потока, состоит из 4 этапов: медленный старт, предотвращение перегрузки, затем быстрая передача и восстановление [36-38]. Стоит отметить, что в разных версиях TCP могут быть использованы не все этапы, либо использованы дополнительные режимы.

Рассматриваемый в модели источник функционирует по протоколу TCP Reno [39]. Переменная $ssth$ (Slow Start Threshold, порог медленного старта) используется для перехода от этапа медленного старта к процессу предотвращения перегрузки. При установке нового соединения $ssth$ возрастает до максимально возможного размера окна. Режим медленного старта продолжается до тех пор, пока значение $cwnd$ (Source Congestion Window, окно перегрузки источника) не станет равным значению $ssth$, после этого происходит переход к состоянию предотвращения перегрузки. Когда происходит отбрасывание пакета, TCP переходит на этап быстрого восстановления или на этап тайм-аута. Основываясь на описании переходов TCP состояний, можно построить UML-диаграмму, демонстрирующую переходы между состояниями (рис. 2.1).

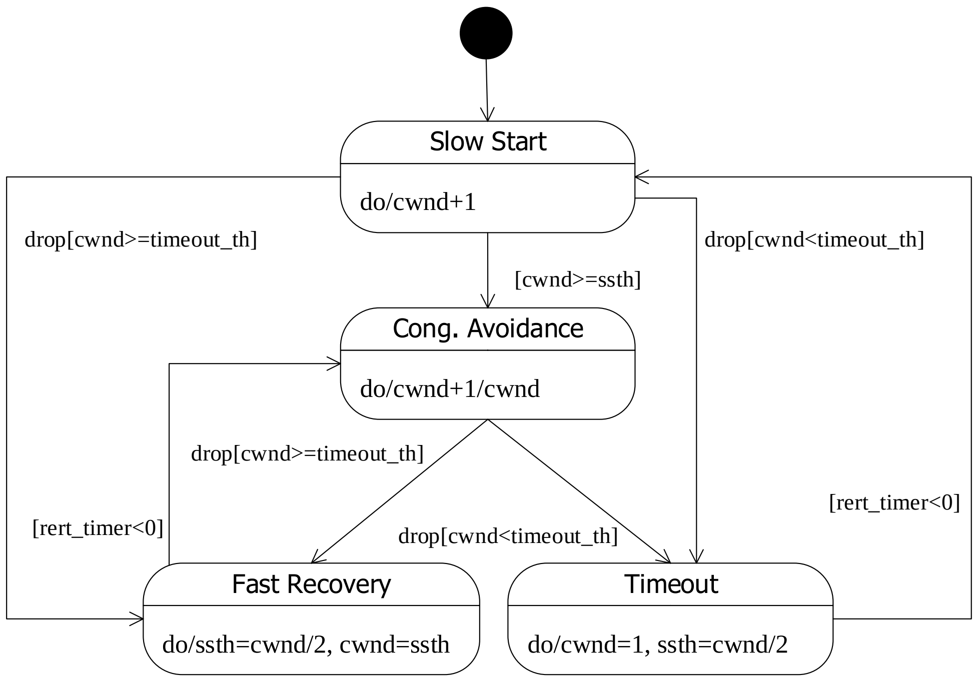

Рисунок 2.1 Диаграмма переходов между состояниями в гибридной модели в протоколе TCP Reno

Однако, в подобных сетях передачи данных имеет место глобальная синхронизация, при которой TCP источники синхронно отправляют пакеты и также синхронно прекращают передачу, при некоторых начальных параметрах системах и настройках маршрутизатора возникает устойчивый автоколебательный режим, который снижает скорость передачи и ухудшает показатели качества обслуживания в сети. Использование для управления трафиком алгоритмов активного управления очередью (AQM) [40], таких как RED, снижает вероятность возникновения глобальной синхронизации, однако не устраняет её полностью, при некоторых значениях начальных параметров в настройках маршрутизаторов в системе возникают автоколебания основных параметров.

Алгоритм RED был разработан для контроля трафика в сети и предотвращения перегрузок в очереди случае их возникновения. Классический алгоритм RED описан в статье [41]. Алгоритмы управления состоянием трафика могут быть представлены как модули управления в сетевом оборудовании, например в маршрутизаторах. При обнаружении затора алгоритмы типа RED начинают случайным образом отбрасывать пакеты прежде, чем весь допустимый размер очереди полностью заполнится. В процессе отбрасывания пакетов RED уведомляет источник о перегрузке и, тем самым, заставляет TCP подобные протоколы снижать скорость передачи, избегая повторной синхронизации.

Вероятность сброса или навешивания на пакет маркера сброса зависит от пороговых значений очереди маршрутизатора, учитываются минимальное и максимальное значения, т.е. значение, при превышении которого алгоритм начнет выборочно отбрасывать пакеты, избегая тем самым перегрузки и значение, ниже которого алгоритм RED будет пытаться удержать размер очереди. Подбор корректных начальных значений, в том числе пороговых, обеспечит эффективную работу алгоритма. Если минимальный предел окажется слишком маленьким, это может привести к падению пропускной способности сети, если слишком большим — к увеличению времени пребывания пакетов в очереди, что негативно отразится на качестве обслуживания.

Преимуществом алгоритма RED является не только эффективность его функционирования при корректных начальных параметрах, но и относительно простая реализация на сетевом оборудовании. В системах, где в качестве алгоритма активного управления очередью используется алгоритм RED, протокол TCP достаточно быстро находит подходящую скорость передачи информации, а также удерживает размер очереди и время задержки передачи пакетов на уровне, требуемом для предоставления достойного качества услуг связи [42].

2.2 Математическая модель системы с управлением по алгоритму RED

Опишем процесс передачи TCP трафика, где в качестве алгоритма активного управления очередью выступает RED. Первым элементом моделируемой системы является источник, который генерирует TCP пакеты, второй элемент – получатель, в качестве которого рассматривается очередь маршрутизатора, обрабатывающая поступающие пакеты в соответствии с алгоритмом активного управления очередью RED. В зависимости от длины очереди алгоритм сообщает TCP источнику о необходимости изменить параметры источника, например увеличить или уменьшить размер окна перегрузки TCP.

Для описания математической модели применим метод стохастизации одношаговых процессов, общие принципы которого подробно рассмотрены в работе

[43], для получения же основного кинетического уравнения применяется комбинаторный подход, подробно рассмотренный в работе [44].

В качестве непрерывного вектора состояний выступает $\left(W, Q, \hat{Q}\right)^T$, где $W \coloneqq W(t)$ – размер окна TCP перегрузки, $Q \coloneqq Q(t)$ – длина очереди маршрутизатора, $\hat{Q} \coloneqq \hat{Q}(t)$ – экспоненциально взвешенное скользящее среднее длины очереди, которое вводится для некоторого сглаживания выбросов мгновенного значения размера очереди $Q(t)$, функционируя как фильтр низких частот.

Пусть пакеты поступают в систему из источника бесконечной емкости с некоторой интенсивностью k_1^1, с интенсивностью k_2^1 – покидают систему. Изменение числа пакетов в системе может быть записано в виде кинетических уравнений:

$$\begin{cases} 0 \overset{k_1^1}{\rightarrow} W \\ W \overset{k_2^1}{\rightarrow} 0. \end{cases} \tag{2.1}$$

Пусть $N_{\underline{i}}^{\alpha}$ и $M_{\underline{i}}^{\alpha}$ – операторы, описывающие число пакетов до и после взаимодействия, тогда оператор перехода $r_{\underline{i}}^{\alpha} = M_{\underline{i}}^{\alpha} - N_{\underline{i}}^{\alpha}$. Получим

$$N_{\underline{i}}^{\alpha} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \quad M_{\underline{i}}^{\alpha} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad r_{\underline{i}}^{\alpha} = \begin{pmatrix} 1 \\ -1 \end{pmatrix}. \tag{2.2}$$

Интенсивности перехода $S_\alpha^+\left(x_{\underline{i}}\right)$ – в прямом направлении, $S_\alpha^-\left(x_{\underline{i}}\right)$ – в обратном, рассчитываемые по формулам

$$S_\alpha^+\left(x_{\underline{i}}\right) = k_\alpha^+ \prod_{\underline{i}=1}^{n} \frac{x_{\underline{i}}!}{\left(x_{\underline{i}} - N_{\underline{i}}^{\alpha}\right)!},$$

$$S_\alpha^-\left(x_{\underline{i}}\right) = k_\alpha^- \prod_{\underline{i}=1}^{n} \frac{x_{\underline{i}}!}{\left(x_{\underline{i}} - M_{\underline{i}}^{\alpha}\right)!},$$

для представленных кинетических реакций имеют вид

$$S_1^+ = k_1^1, \quad S_2^+ = k_2^1 W. \tag{2.3}$$

Отсюда выпишем получившиеся коэффициенты уравнения Фоккера-Планка:

$$\begin{gathered} A^i = S_\alpha^+\left(x_{\underline{i}}\right) r^{\alpha i} = k_1^1 - k_2^1 W, \\ B^{ij} = S_\alpha^+\left(x_{\underline{i}}\right) r^{\alpha i} r^{\alpha j} = k_1^1 + k_2^1 W, \end{gathered} \tag{2.4}$$

где согласно спецификации TCP, $k_1^1 = 1/W$, $k_2^1 = \frac{1}{2}\frac{\mathrm{d}N}{\mathrm{d}t}$, где $\mathrm{d}N$ – винеровский процесс.

Уравнение Фоккера-Планка имеет следующий вид

$$\frac{\partial w}{\partial t} = -\frac{\partial}{\partial W}\left[\left(\frac{1}{W} - \frac{W}{2}\frac{\mathrm{d}N}{\mathrm{d}t}\right)w\right] + \frac{1}{2}\frac{\partial^2}{\partial W^2}\left[\left(\frac{1}{W} + \frac{W}{2}\frac{\mathrm{d}N}{\mathrm{d}t}\right)w\right], \tag{2.5}$$

где $w \coloneqq w(t)$ – плотность распределения случайного процесса $W(t)$.

Следовательно, уравнение Ланжевена выражено следующим образом:

$$\mathrm{d}W = \frac{1}{W}\mathrm{d}t - \frac{W}{2}\mathrm{d}N + \sqrt{\frac{1}{T} + \frac{W}{2}\frac{\mathrm{d}N}{\mathrm{d}t}}\,\mathrm{d}V^1, \tag{2.6}$$

где $\mathrm{d}V^1$ – винеровский процесс, соответствующий случайному процессу $W(t)$.

Аналогичным образом опишем процесс изменения размера очереди маршрутизатора, куда пакеты поступают с интенсивностью k_1^2, с интенсивностью k_2^2 – покидают систему.

Кинетическое уравнение имеет вид:

$$\begin{cases} 0 \overset{k_1^2}{\rightarrow} Q \\ 0 \overset{k_2^2}{\rightarrow} Q. \end{cases} \tag{2.7}$$

Опишем операторы состояний:

$$N_{\underline{i}}^{\underline{\alpha}} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \quad M_{\underline{i}}^{\underline{\alpha}} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \quad r_{\underline{i}}^{\underline{\alpha}} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}. \tag{2.8}$$

Тогда интенсивности переходов имеют вид:

$$S_1^+ = k_1^2, \quad S_2^+ = k_2^2. \tag{2.9}$$

Из этого следует, что коэффициенты уравнения Фоккера-Планка равны

$$\begin{aligned} A^i &= S_\alpha^+(x_{\underline{i}})r^{\alpha i} = k_1^2 - k_2^2, \\ B^{ij} &= S_\alpha^+(x_{\underline{i}})r^{\alpha i}r^{\alpha j} = k_1^1 + k_2^2. \end{aligned} \tag{2.10}$$

Количество поступающих пакетов за единицу времени и есть интенсивность входящего потока $k_1^2 = \frac{W}{T}$. Интенсивность обслуживания есть C, следовательно $k_2^2 = C$.

Тогда, уравнение Фоккера-Планка для размера длины очереди имеет следующий вид:

$$\frac{\partial q}{\partial t} = -\frac{\partial}{\partial Q}\left[\left(\frac{W}{T} - C\right) q\right] + \frac{1}{2}\frac{\partial^2}{\partial Q^2}\left[\left(\frac{W}{T} - C\right) q\right], \tag{2.11}$$

где $q \coloneqq q(t)$ – плотность распределения случайного процесса $Q(t)$.

Следовательно, уравнение Ланжевена выражено следующим образом:

$$\mathrm{d}Q = \left(\frac{W}{T} - \mathrm{C}\right) \mathrm{d}t + \sqrt{\frac{W}{T} - C}\, \mathrm{d}V^2, \tag{2.12}$$

где $\mathrm{d}V^2$ – винеровский процесс, соответствующий случайному процессу $Q(t)$.

Винеровские процессы $\mathrm{d}V^1$ и $\mathrm{d}V^2$ можно интерпретировать как случайное отклонение размера пакета от своего среднего значения.

Рассматриваемая в данной работе система содержит C приборов, что соответствует скорости обработки C штук пакетов в единицу времени. Также в системе учитываются параметры $N(t)$ – число TCP-сессий и $T(t)$ – время приема-передачи (Round Trip Time, RTT, сек).

Время $T(t)$ рассчитывается следующим образом

$$T(t) = \begin{cases} T_b + \dfrac{q(t)}{C(t)}, & Q(t) > 0, \\ T_b \end{cases}$$

$$C(t) = \begin{cases} C, & Q(t) > C \\ Q(t), & Q(t) \le C, \end{cases} \tag{2.13}$$

где T_b – время приема-передачи одного пакета (Round Trip Time, сек), $C(t)$ – интенсивность обслуженной нагрузки [45].

Коэффициент w_q рассчитывается как

$$w_q = 1 - e^{-1/C}.$$

Управление алгоритмом RED осуществляется согласно вероятностной функции маркировки на отбрасывание пакетов $P\left(\hat{Q}(t)\right)$. В статьях [46,47] получена формула для вычисления вероятности сброса пакета:

$$P\left(\hat{Q}(t)\right) = \begin{cases} 0, & 0 \le \hat{Q}(t) < Q_{\min}, \\ \dfrac{\hat{Q}(t) - Q_{\min}}{Q_{\max} - Q_{\min}} p_{\max}, & Q_{\min} \le \hat{Q}(t) \le Q_{\max}, \\ 1, & \hat{Q}(t) > Q_{\max}. \end{cases} \tag{2.14}$$

Данная функция зависит от пороговых значений размера очереди $Q_{\min}$ и $Q_{\max}$, а также параметра $p_{\max}$, задающего часть пакетов, которые будут отброшены в случае, если $\hat{Q}(t)$ достигнет максимального значения.

На рисунке 2.2 представлен вид вероятностной функции сброса пакетов в алгоритме RED.

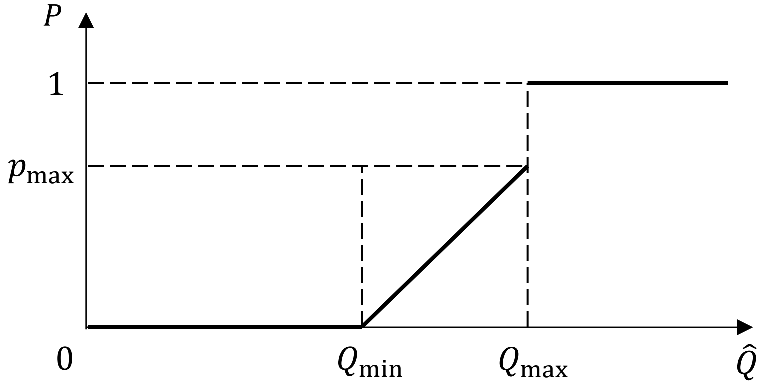

Рисунок 2.2 Вид функции сброса в алгоритме RED

Таким образом, принимая во внимание полученные уравнения (2.6), (2.12) и [47], выпишем стохастическую модель, которая представляет собой систему стохастических дифференциальных уравнений Ито с запаздыванием:

$$\begin{cases} \mathrm{d}W(t) = \left(\dfrac{1}{T(t)} - \dfrac{W(t)W\big(t-T(t)\big)P\big(t-T(t)\big)}{2T\big(t-T(t)\big)}\right)\mathrm{d}t + \\ \quad + \sqrt{\left(\dfrac{1}{T(t)} - \dfrac{W(t)W\big(t-T(t)\big)P(t-T(t))}{2T(t-T(t))}\right)}\,\mathrm{d}V^1, \\ \mathrm{d}Q(t) = \left(\dfrac{W(t)}{T(t)}N(t) - C(t)\right)\mathrm{d}t + \sqrt{\dfrac{W(t)}{T(t)}N(t) - C(t)}\mathrm{d}V^2, \\ \mathrm{d}\hat{Q}(t) = w_q C(t)\left(Q(t) - \hat{Q}(t)\right). \end{cases} \tag{2.15}$$

Таким образом, для моделировании гибридной системы процесса передачи данных по протоколу Transmission Control Protocol (TCP) и регулирования состояния потока при возникновении перегрузок маршрутизатора, в качестве которого рассматривается алгоритм Random Early Detection (RED), требуется использование языка, позволяющего реализовать стохастическую часть модели, запаздывание некоторых элементов системы и использовать гибридный подход, т.к. требуется учитывать особенности непрерывных параметров, таких как изменения длины очереди маршрутизатора $Q(t)$, экспоненциально взвешенного скользящего среднего длины очереди $\hat{Q}(t)$ и размера TCP-окна $W(t)$, а также дискретные переходы между TCP состояниями, запаздывания некоторых переменных и вероятностную функцию сброса пакетов $P\left(\hat{Q}(t)\right)$.

3. Моделирование стохастической системы с запаздыванием

Гибридные системы могут быть достаточно просто реализованы в специализированных языках моделирования динамических систем, например Modelica [48]. Этот язык хорошо подходит для моделирования комплексных систем, состоящих из элементов различной природы, а также систем с элементами управления. Так, в работах [4, 5, 49, 50] на языке Modelica построена имитационная детерминированная модель системы с управлением по алгоритму типа RED. Гибридная модель состоит из классов, соединенных коннекторами и описывающих поведение следующих элементов системы: TCP-источника, получателя, двух маршрутизаторов, каналов передачи данных и обработку очередей в сети в соответствии с алгоритмом RED. Модель была сформулирована в рамках гибридного подхода, следовательно на языке Modelica реализация вышла достаточно естественной, а именно при помощи высокоуровневых средств была записана система дифференциальных уравнений с запаздывающим аргументом, функция управления реализована как контроллер. Однако данный программный комплекс имеет ряд ограничений, подробно описанных в статье [47], более того Modelica не имеет встроенных универсальных инструментов для моделирования стохастических систем, где важно учитывать случайный характер поведения основных параметров.

В качестве альтернативного языка реализации был выбран язык высокого уровня Julia, обладающий встроенной многопоточностью и разработанный специально для научных и инженерных расчётов [51-53]. Благодаря активному использованию множественной диспетчеризации, метапрограммированию [54], интерфейсов внешних функций (FFI) и перегрузки вызовов DifferentialEquations.jl предлагает производительный унифицированный интерфейс для решения и анализа различных форм дифференциальных уравнений [55]. Множественная диспетчеризация в языке Julia используется для объединения функций, необходимых пользователю для моделирования, например, `solve` и `plot`, гибкость языка обеспечивается высокой совместимостью библиотек и возможностью объединения решателей компонентов и дополнительных пакетов. Также множественная диспетчеризация используется для написания единого универсального метода, который компилируется в специализированные функции в зависимости от заданных в программе числовых типов. Julia обладает возможностями применения многопоточного расчета внутри методов, которая используется для повышения производительности, и многоузлового параллелизма, применяемого для моделирования стохастических уравнений, например методом Монте-Карло.

Стоит отметить, что язык Julia — это относительно новый язык научных расчетов, что является как его преимуществом, так и недостатком. Проблема заключается в отсутствии полной документации языка, пользователям приходится обходиться лишь примерами запросов и реализаций простых моделей, что значительно усложняет моделирование с использованием Julia. Одни из ведущих разработчиков языка и авторы работы [34] обращают внимание пользователей на то, что в силу гибкости Julia и совместимости с большим объемом программного обеспечения задача создания полной технической документации с подробно описанными методами моделирования не выглядит тривиальной. Данный язык требует высокого профессионализма разработчика и обладание большим объемом знаний в различных аспектах прикладной математики и программирования для разработки сложных моделей, требующих корректировки и переопределения используемых в Julia методов и алгоритмов. С другой стороны, данная проблема порождает множество перспективных научных задач усовершенствования библиотек для сложных, комплексных моделей, создание собственных пакетов, реализующих различные аспекты моделирования, создание технического описания используемых модулей, библиотек и деталей совместимости их с другими программными средствами практически любых языков.

В данной работе активно использовалась библиотека DifferentialEquations.jl [55], предназначенная для эффективного решения дифференциальных уравнений различных видов, таких как обыкновенные дифференциальные уравнения, гибридные уравнения, стохастические дифференциальные уравнения Ито, в том числе с запаздывающими аргументами. Данная библиотека наследует возможности нескольких библиотек для решения дифференциальных уравнений, например DiffEqBase.jl, Sundials.jl [], StochasticDiffEq.jl, DelayDiffEq.jl, DiffEqCallbacks.jl, ссылки на которые будут автоматически установлены в `Pkg.add("DifferentialEquations")`.

Библиотека DifferentialEquations.jl имеет встроенные инструменты моделирования стохастических дифференциальных уравнений с винеровским процессом, которые традиционно записываются в форме уравнения Ланжевена. Подобные уравнения состоят из детерминированной и стохастической частей и имеют следующий вид:

$$\mathrm{d}x = f\mathrm{d}t + g\mathrm{dW}, \qquad ()$$

где $x := x(t) \in \mathbb{R}$ – некоторый случайный процесс; $f := f(x,t) \in \mathbb{R}, g := g(x,t) \in \mathbb{R}$; $W := W(t) \in \mathbb{R}$ – винеровский процесс.

Для многомерного случая

$$\mathrm{d}x^i = f^i\mathrm{d}t + b^i_f g W^f, \qquad (2)$$

где $x^i := x^i(t) \in \mathbb{R}^n$ – n-мерный случайный процесс; $f^i := f^i(x^k, t) \in \mathbb{R}^n \to \mathbb{R}^n, g^i_f := g^i_f(x^k, t) \in \mathbb{R}^n \to \mathbb{R}^s$; $W^f := W^f(t) \in \mathbb{R}^s$ – s-мерный винеровский процесс.

3.1 Математическое моделирование алгоритма RED

В качестве численно моделируемой стохастической системы с управлением выступает система, функционирующая по алгоритму RED. Полная реализация вычислительной модели системы с управлением по алгоритму типа RED на языках Julia и Modelica представлена в разделах Приложение 1 и Приложение 2.

Рассмотрим псевдокод работы дисциплины RED:

Алгоритм 3.1: Алгоритм RED

$\hat{Q} \leftarrow 0$
$pkgcount \leftarrow -1$
while $packet$ **do** *// Пакеты поступают в систему*
  $\hat{Q} \leftarrow QueueAvg()$ *// Расчет переменной $\hat{Q}$*
  if $Q > 0$ **then**
    $\hat{Q} \leftarrow (1 - w_q)\hat{Q} + w_q Q$
  else
    $m \leftarrow f(time - Q_{\text{time}})$
    $\hat{Q} \leftarrow (1 - w_q)^m \hat{Q}$
  end if

  if $Q_{\min} < \hat{Q} < Q_{\max}$ **then**
    $pkgcount \leftarrow pkgcount + 1$
    $p \leftarrow p_{\max}(\hat{Q} - Q_{\min})/(Q_{\max} - Q_{\min})$
    $p \leftarrow p / (1 - p \cdot pkgcount)$

$Mark(packet, p)$ // *Вычисление вероятности сброса пакета*
$count \leftarrow 0$
else if $\hat{Q} > Q_{\max}$ **then**
$Mark(packet, 1)$
$pkgcount \leftarrow 0$
else
$pkgcount \leftarrow -1$
end if

if $Q = 0$ **then** $Q_{\text{time}} \leftarrow time$
end if
end while

Переменные:

– Q — текущая средняя длина очереди;

– $\hat{Q}$ — экспоненциально взвешенное скользящее среднее длины очереди;

– Q_{time} — время, начиная с которого очередь пуста;

– $pkgcount$ — количество пакетов;

Константы:

– w_q — весовой коэффициент;

– $Q_{\min}$ — минимальный порог очереди;

– $Q_{\max}$— максимальная порог очереди;

– $p_{\max}$ — максимальное значение вероятности сброса p;

Остальные параметры:

– p — вероятность маркировки пакета как отброшенного;

– $time$ — текущее время;

– $f(t)$ — линейная функция от времени t.

Перейдем к моделированию системы передачи данных по TCP протоколу и обработки очереди в соответствии с алгоритмом RED и сравнению реализаций данной системы на языках Modelica и Julia с упором на последний, так как именно Julia позволяет эффективно моделировать стохастические системы, в Modelica же приведем лишь моделирование детерминированной части вышеописанной системы.

Для установки в Julia пакета DifferentialEquations.jl используем следующую команду в Julia REPL:

```
using Pkg
Pkg.add("DifferentialEquations")
```

Подключим пакет, используя команду:

```
using DifferentialEquations
```

Зададим вектор начальных параметров системы `p = (T, N, C, wq, q_min, q_max, R, p_max, w_max)`.

Зададим вероятностную функцию сброса пакетов согласно уравнению (2.14). В Julia реализуем алгоритм управления как отдельную функцию `p(q_avg)` (листинг 3.1).

Листинг 3.1: Вероятностная функция сброса пакетов в Julia

```
function p(q_avg)
    global q_min, q_max
    if (q_avg < q_min * R)
        p = 0.0
    elseif (q_avg > q_max * R)
        p = 1.0
    else
        p = p_max * (q_avg / R - q_min) / (q_max - q_min)
    end
end
```

На языке Modelica дискретная функция задается не как функция, а как уравнение класса RED (листинг 3.2).

Листинг 3.2: Вероятностная функция сброса пакетов в Modelica

```
class Red
equation
p = if q_avg < thmin * R
    then 0.0
else if q_avg > thmax * R
    then 1.0
else (q_avg / R - thmin) * pmax / (thmax - thmin);
```

Для рассматриваемой нами задачи динамическая функция `RED`, описывающая детерминированную часть системы дифференциальных уравнений (2.15) в Julia будет иметь следующий вид (листинг 3.3):

Листинг 3.3: Функция RED, задающая детерминированную часть СДУ в Julia

```
function RED(du, u, param, t)
    w, q, q_avg = u
    if w < w_max
        du[1] = 1.0 / T(q) - ( w / 2 )* w * p(q_avg) /
T(q)
    else
        du[1] = - ( w / 2 )* w * p(q_avg) / T(q)
    end
    du[2] = N * w / T(q) - C(q)
    du[3] = wq * C(q) * (q - q_avg)
end
```

В Modelica математически описывается система дифференциальных уравнений в классе RED в разделе `equation` (листинг 3.4). С помощью оператора `der` задается производная переменной по времени. При моделировании в Modelica стоит помнить, что количество уравнений в разделе `equation` должно совпадать с количеством выходных параметров системы.

Листинг 3.4: Уравнения класса RED, задающие детерминированную часть СДУ в Modelica

```
class Red
equation
der(w) = w_add(w, wmax, T) + (-0.5) * w * delay(w, T,
T) * delay(p, T, T) / delay(T, T, T);
der(q) = q_add(pre(q), w, T, C, N, R);
der(q_avg) = wq * C * (q - q_avg);
```

Запаздывание на Modelica реализуется достаточно просто с использованием оператора `delay`, в то время как на Julia требуется вводить функцию истории (листинг 3.5), зависящую от вектора параметров `p = (T, N, C, wq, q_min, q_max, R, p_max, w_max)`:

Листинг 3.5: Пример реализации запаздывания непрерывной переменной в Julia

```
h(p, t) = zeros(1)
```

```
tau = T
lags = [tau]
```

Далее в Julia определим стохастическую часть системы уравнений (2.15), которая задаётся в функции `REDst` (листинг 3.6) :

Листинг 3.6: Функция REDst, задающая стохастическую часть СДУ в Julia

```
function REDst(du, u, param,t)
    w, q, q_avg = u
    du[1] = sqrt(1.0 / T(q) + ((w / 2) * w * p(q_avg)/
 T(q)))
    if ((N * w / T(q) - C(q)) < 0)
        du[2] = 0.0
    else
        du[2] = sqrt(N * w / T(q) - C(q))
    end
    du[3] = 0.0
end
```

При моделировании гибридных систем на языке Modelica не возникает сложностей с моделированием как элементов с непрерывным характером функционирования, так элементов управления с дискретным поведением. При использовании языка Julia требуются дополнительные инструменты для реализации гибридной парадигмы, а именно использование функции обратных вызовов.

Пакет DifferentialEquations.jl позволяет использовать механизмы внедрения пользовательского кода в алгоритмы решателя с помощью функции обратных вызовов для моделирования сложных функций с условиями и разрывами [34]. Для работы с обратными вызовами определяются две функции – функция условия, или condition function, проверяющая, произошло ли некоторое событие, а также воздействующая функция, или affect function, которая выполняется, если событие произошло.

Задав функцию условия и воздействующую функцию, ограничим рост размера TCP-окна максимальным значением (листинг 3.7, 3.8).

Листинг 3.7: Функция условия для параметра $W(t)$

```
function condition_w_max(u,t,integrator)
    global w_max
    u[1] >= w_max
end
```

Листинг 3.8: Воздействующая функция для параметра $W(t)$

```
function affect_w_max!(integrator)
    global w_max
    for c in full_cache(integrator)
        c.u[1] = w_max
    end
end
```

Также добавим обратные вызовы для контроля роста переменной $Q(t)$ (листинг 3.9, 3.10).

Листинг 3.9: Функция условия для параметра $Q(t)$

```
function condition_Rq(u,t,integrator)
    global R
    u[2] >= R
end
```

Листинг 3.10: Воздействующая функция для параметра $Q(t)$

```
function affect_Rq!(integrator)
    global R
    for c in full_cache(integrator)
        c.u[2] = R
    end
end
```

Опция обратного вызова для экспоненциально взвешенного скользящего среднего длины очереди задаётся аналогичным образом, т.е. ограничим размер переменной $\hat{Q}(t)$ размером очереди (листинг 3.11, 3.12).

Листинг 3.11: Функция условия для параметра $\hat{Q}(t)$

```
function condition_Rq(u,t,integrator)
    global R
    u[2] >= R
end
```

Листинг 3.12: Воздействующая функция для параметра $\hat{Q}(t)$

```
function affect_Rq_avg!(integrator)
    global R
    for c in full_cache(integrator)
        c.u[3] = R
    end
end
```

В данном случае использовалось несколько функций обратных вызовов дискретного типа `DiscreteCallback`, где функция условия реализует обнаружение событий на каждом шаге решения `dt`, а воздействующая функция выполнятся в случае, если функция условия возвращает значение `true`. В качестве альтернативного способа можно использовать опцию обратного вызова непрерывного типа `ContinuousCallback`, где функция условия является непрерывной, а вызов опции обратного вызова инициируется в случае, если функция условия оценивается как 0.

Объявим вызовы заданных функций условия и воздействующих функций для переменных $W(t)$, $Q(t)$, $\hat{Q}(t)$ (листинг 3.13).

Листинг 3.13: Вызов опции обратных вызовов для непрерывных переменных системы

```
cb_w_max = DiscreteCallback(condition_w_max, affect_w_max!, save_positions = save_positions)
cb_Rq = DiscreteCallback(condition_Rq, affect_Rq!, save_positions = save_positions)
cb_Rq_avg = DiscreteCallback(condition_Rq_avg, affect_Rq_avg!, save_positions = save_positions)
```

С помощью инструмента `CallbackSet()` несколько обратных вызовов, в том числе разных типов, могут быть объединены в одну группу (листинг 3.14).

Листинг 3.14. Объединение обратных вызовов

```
Clbsset = CallbackSet(PositiveDomain(), cb_w_max,
cb_Rq, cb_Rq_avg)
```

На языке Modelica контроль над переменными может быть легко осуществлен с помощью объявления отдельной функции (листинг 3.15).

Листинг 3.15. Функция изменения среднего размера очереди в Modelica

```
function q_add
input Real q, w, T, C, N, R;
output Real q_out;
q1 := N * w / T - C;
q2 := q + q1;
q_out := if q2 > R then R - q
         else if q2 > 0.0 then q1
         else -q;
end q_add;
```

Для решения системы и запуска моделирования в Julia после объявления вектора начального состояния и параметров моделирования вызовем решатель `SDEProblem` пакета DifferentialEquations.jl, в аргументы которого передаётся функция `RED`, задающая детерминированную часть уравнений, и функция `REDst`, задающая стохастическую часть системы. Также в аргументах к решателю указывается вектор начальных состояний системы и время моделирования (листинг 3.16).

Листинг 3.16. Вызов решателя СДУ

```
prob_sde_RED = SDEProblem(RED, REDst, u0, tspan)
```

Вызовем метод `solve` библиотеки DifferentialEquations.jl для решения вышеописанной системы уравнений (листинг 3.17). В качестве численного метода используется метод Эйлера-Маруямы, имеющий сильный порядок детерминированной части $p_d = 1{,}0$ и порядок приближения стохастической части $p_s = 0{,}5$:

Листинг 3.17. Вызов метода solve

```
sol = solve(prob_sde_RED, EM(), dt=dt, callback = Clbsset)
```

Переведем вышеописанную задачу в тип `EnsembleProblem`, используя одноименный конструктор (листинг 3.18). В параметре командной строки `-t/--threads` можно указать количество используемых потов для много поточных вычислений, по умолчанию в операционных системах Linux и macOS Julia использует 4 потока. Таким образом, определив функцию `EnsembleThreads()`, подключим использование встроенного параллелизма Julia и смоделируем `trajectories = 200` число траекторий:

Листинг 3.18. Моделирование множества траекторий

```
ensembleprob = EnsembleProblem(prob_sde_RED)
sim = solve(ensembleprob, EnsembleThreads(), EM(), dt=dt, callback = Clbsset, trajectories = 200)
```

Метод `EnsembleSummary()` дает возможность объединять смоделированное множество траекторий, как по временным шагам `dt`, которые использовались интегратором `solve`, так и по временным точкам, что требует интерполяции решения. Также метод `EnsembleSummary()` позволяет визуализировать статистические параметры, например среднее и дисперсию:

```
summ = EnsembleSummary(sim, 0:dt:tf)
```

По умолчанию используются 5% и 95% значения квантилей.

Таким образом, была представлена реализация детерминированной модели с запаздыванием алгоритма RED на языке Modelica и стохастической системы с запаздыванием — на языке Julia. Оба языка являются объектно-ориентированными, но применяемыми для различных целей. На специализированной для моделирования комплексных систем с управлением реализация модели, сформулированная в рамках гибридного подхода, оказалась достаточно простой, элементы с непрерывным характером функционирования хорошо взаимодействовали с дискретной функцией управления в рамках класса `RED` раздела `equation`. Ограничения для параметров системы задавались с помощью стандартной функции `when`, запаздывание для элементов любого характера функционирования может быть реализовано с помощью функции `delay`. Однако, данный язык не обладает встроенными инструментами стохастического моделирования. Язык Julia, который был разработан для научных расчетов, располагает инструментами, такими как библиотека DifferentialEquations.jl, для моделирования стохастических систем различных видов, в том числе с запаздыванием. Однако, в обладающей большим количеством синтаксического сахара Julia реализация гибридной парадигмы требует от программиста использования дополнительных инструментов, например опцию обратных вызовов, с помощью которой реализован дополнительный контроль над переменными и взаимодействие с функцией управления.

В следующем разделе рассмотрим результаты, полученные в ходе численного эксперимента, проведенного в рамках программных комплексов на Modelica и Julia.

3.2 Анализ полученных в ходе моделирования результатов

В результате моделирования были получены графики изменения основных параметров системы с модулем управления очередью по алгоритму RED при различных начальных значениях модели.

Как уже отмечалось в [28], в Modelica нет встроенных инструментов для моделирования стохастических уравнений, в качестве результатов моделирования сначала приведем графики поведения основных параметров системы лишь детерминированной части уравнения (2.15). На рисунке 3.1 представлен график, отражающий поведение основных параметров системы $W(t)$, $Q(t)$, $\hat{Q}(t)$ по результатам моделирования на языке Julia детерминированной части уравнения (2.15) при пороговых значениях очереди $Q_{\min} = 0{,}25$ и $Q_{\max} = 0{,}5$, на рисунке 3.2 – результаты моделирования на Modelica математической модели сети передачи данных с модулем активного управления трафиком, работающим по алгоритму RED, при тех же пороговых значениях. Из графиков видно, что при данных начальных параметрах в системе присутствую автоколебания, что негативно влияет на качество обслуживания в сети. Численное моделирование, проведённое в рамках обоих программных комплексов, даёт сопоставимые результаты.

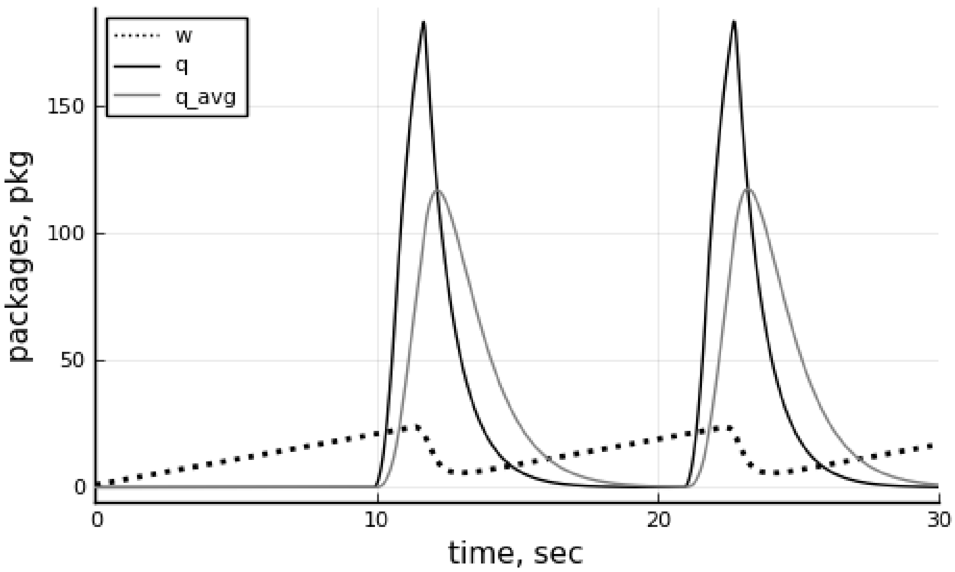

Рисунок 3.1. Поведение параметров $W(t), Q(t), \hat{Q}(t)$ по результатам моделирования на языке Julia $\left(Q_{min} = 0{,}25, Q_{max} = 0{,}5,\ g\big(t, x(t)\big) = 0\right)$

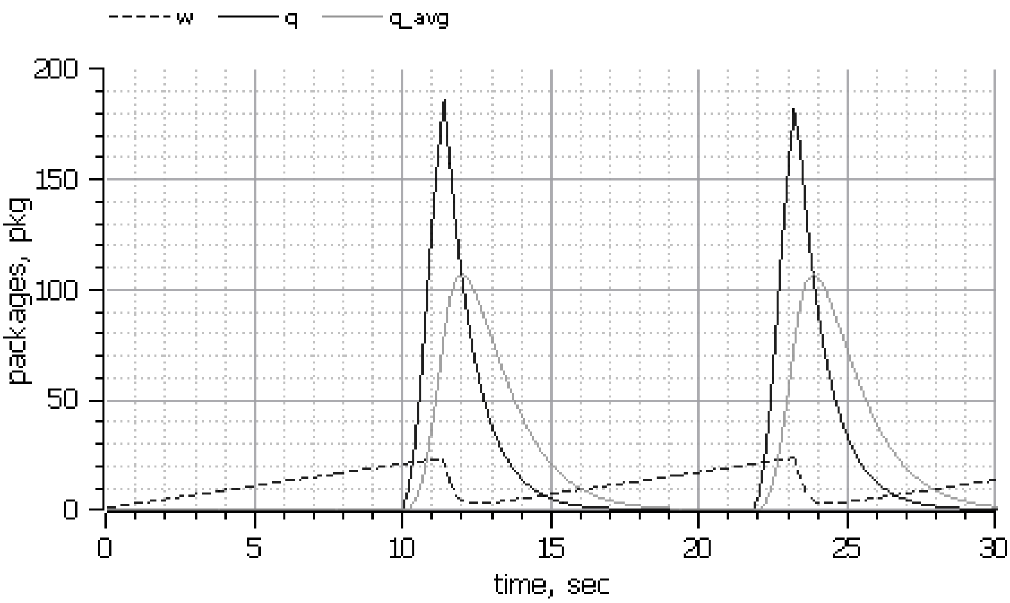

Рисунок 3.2. Поведение параметров $W(t), Q(t), \hat{Q}(t)$ по результатам моделирования на языке Modelica $\left(Q_{min} = 0{,}25, Q_{max} = 0{,}5,\ g\big(t, x(t)\big) = 0\right)$

Далее приведем результаты моделирования на языке Julia. При некоторых начальных параметрах система быстро находит подходящие значения переменных и не изменяются, в детерминированном случае, или же изменяются в пределах своих средних значений при стохастической модели. На рис. 3.3, 3.4 демонстрируется поведение параметров $W(t)$ и $Q(t)$ при обнулении стохастической части системы уравнений (2.15). На рис. 3.5, 3.6 приводится поведение данных параметров стохастической системы уравнений (2.15).

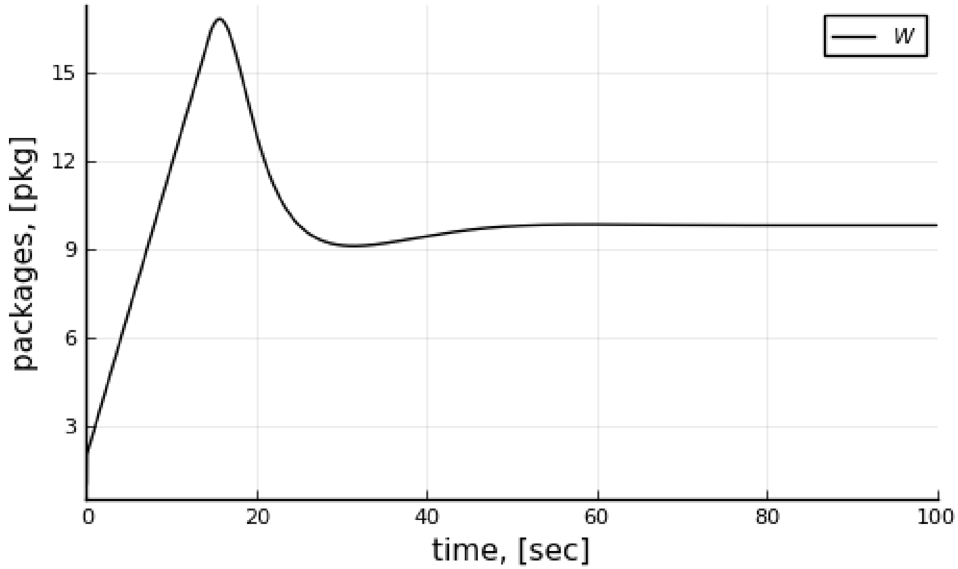

Рисунок 3.3. График изменения размера TCP-окна $W(t)$

$\left(Q_{min} = 0{,}2, Q_{max} = 0{,}8,\ g\big(t, x(t)\big) = 0\right)$

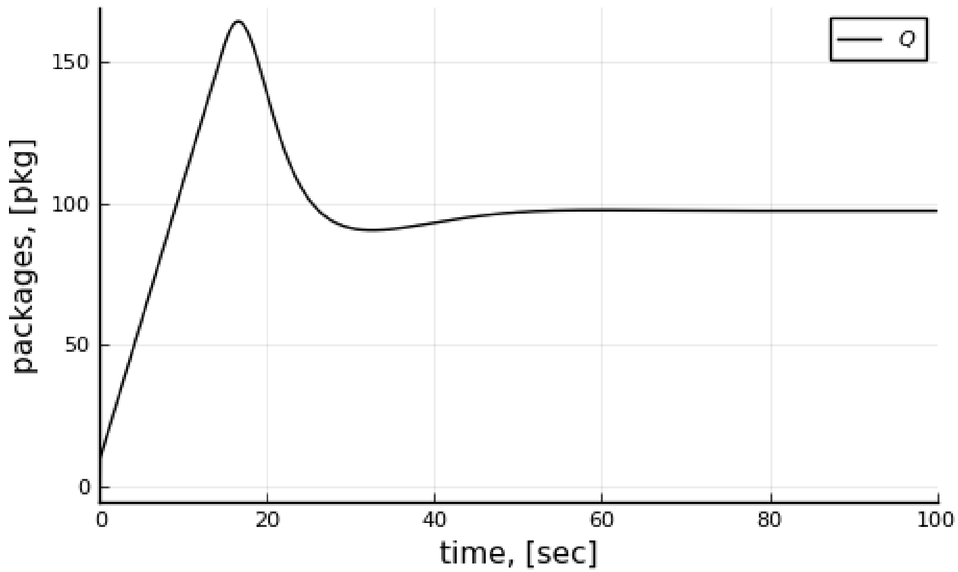

Рисунок 3.4. График изменения средней длины очереди $Q(t)$

$\left(Q_{min} = 0{,}2, Q_{max} = 0{,}8,\ g\big(t, x(t)\big) = 0\right)$

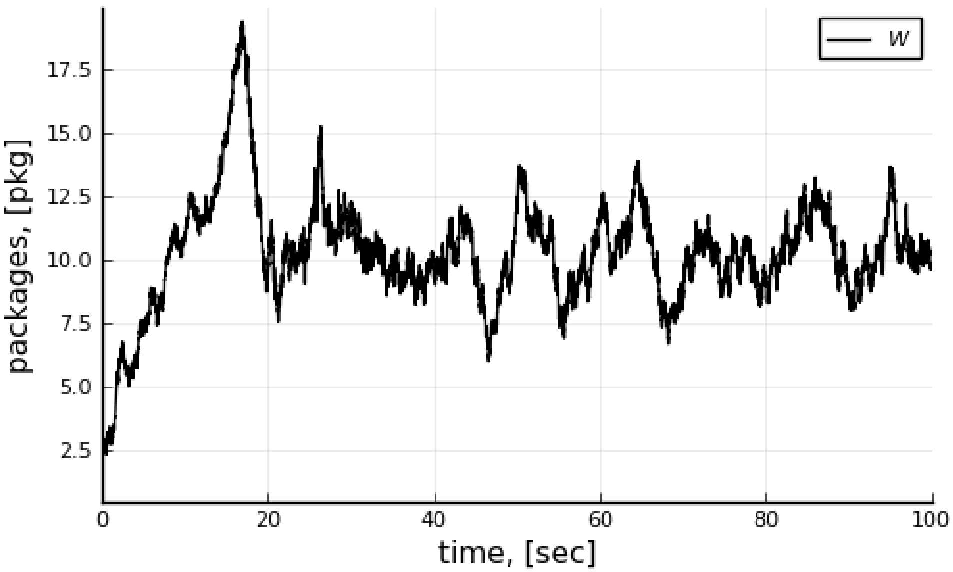

Рисунок 3.5. График изменения размера TCP-окна $W(t)$

$(Q_{min} = 0{,}2, Q_{max} = 0{,}8)$

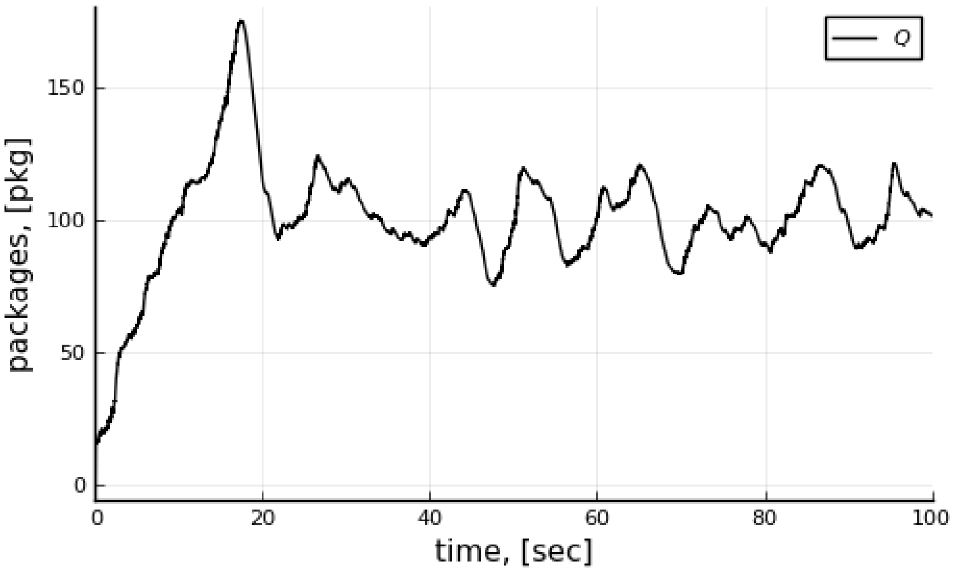

Рисунок 3.6. График изменения средней длины очереди $Q(t)$

$(Q_{min} = 0{,}2, Q_{max} = 0{,}8)$

Смоделируем большое число траекторий и построим среднее и дисперсию (при 5% и 95% перцентилях) на основе полученных результатов в ходе моделирования 200 траекторий.

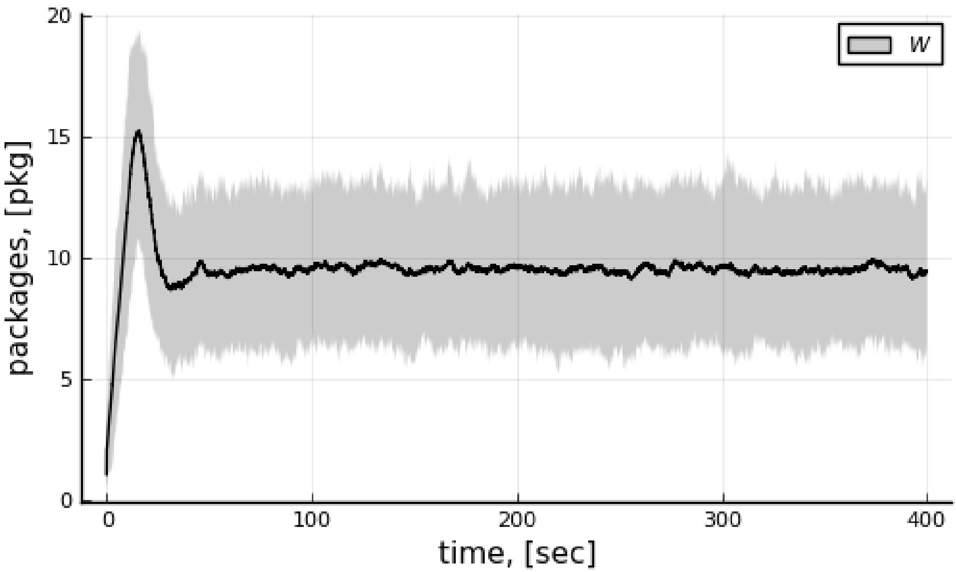

Рисунок 3.7. График среднего по ансамблю (200 траекторий) размера TCP-окна $W(t)$ ($Q_{min} = 0{,}2, Q_{max} = 0{,}8$)

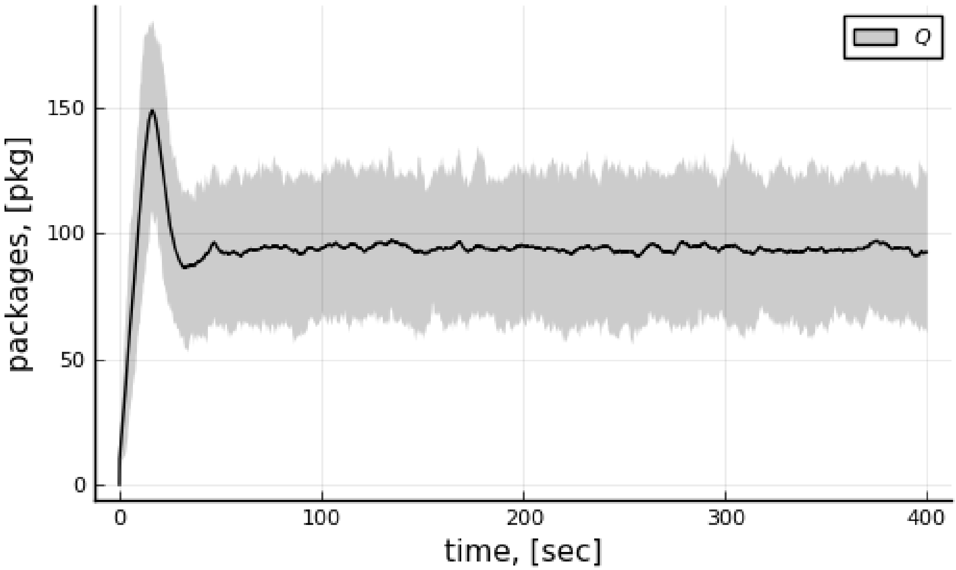

Рисунок 3.8. График среднего по ансамблю (200 траекторий) средней длины очереди $Q(t)$ ($Q_{min} = 0{,}2, Q_{max} = 0{,}8$)

При относительно небольшом промежутке допустимых значений очереди, система переходит в автоколебательный режим функционирования. Так, на рис. 3.9, 3.10 видно присутствие автоколебаний основных параметров системы. На рис. 3.11, 3.12 при добавлении стохастики в модель, параметры так же сильно колеблются.

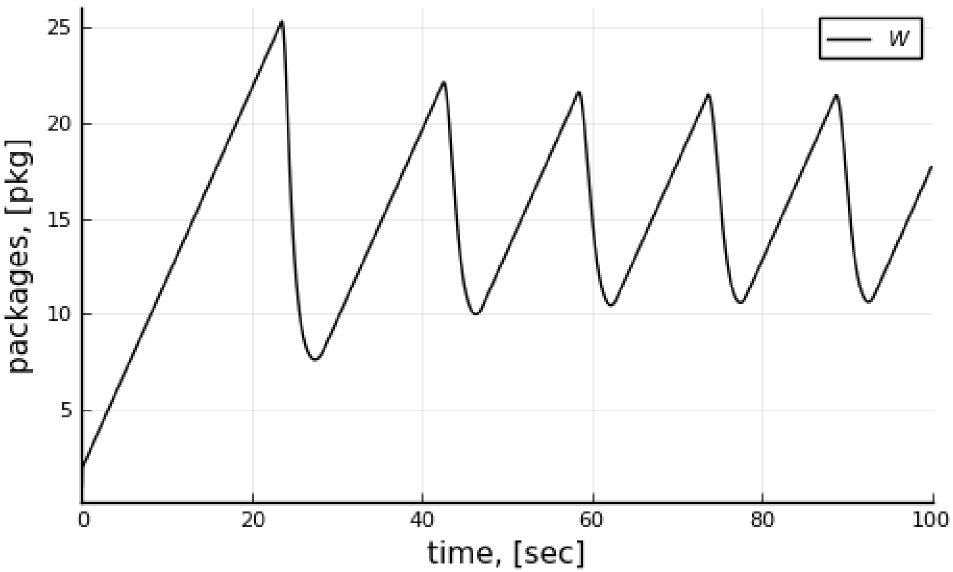

Рисунок 3.9. График изменения размера TCP-окна $W(t)$

$\left(Q_{min} = 0{,}55, Q_{max} = 0{,}6,\ g\big(t, x(t)\big) = 0\right)$

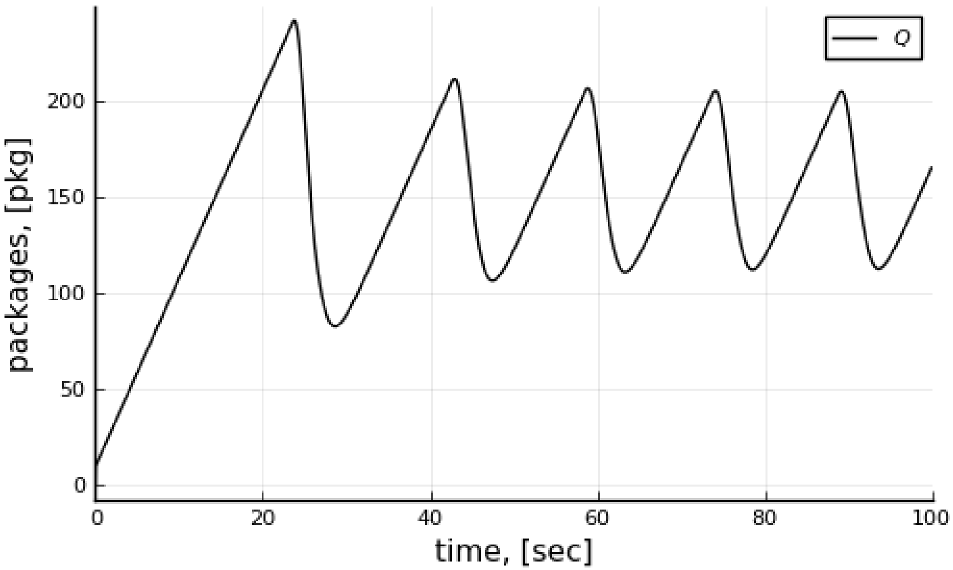

Рисунок 3.10. График изменения средней длины очереди $Q(t)$

$\left(Q_{min} = 0{,}55, Q_{max} = 0{,}6,\ g\big(t, x(t)\big) = 0\right)$

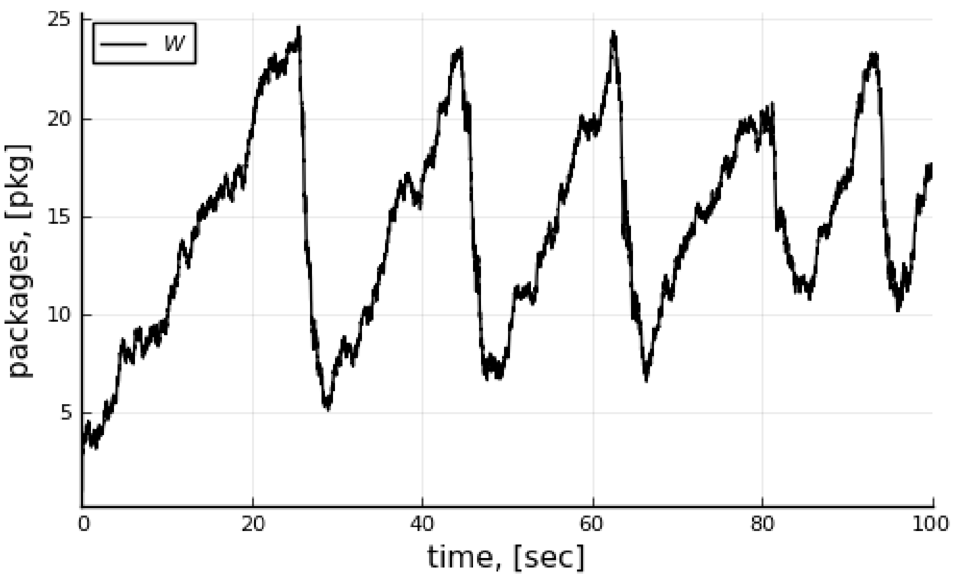

Рисунок 3.11. График изменения размера TCP-окна $W(t)$

$(Q_{min} = 0{,}55, Q_{max} = 0{,}6)$

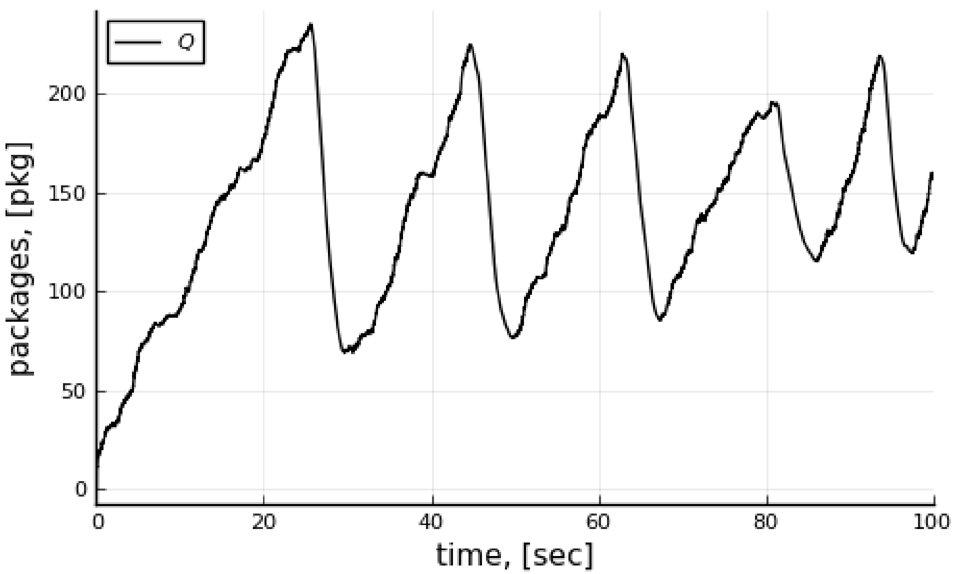

Рисунок 3.12. График изменения средней длины очереди $Q(t)$

$\left(Q_{min} = 0{,}55, Q_{max} = 0{,}6,\ g\big(t, x(t)\big) = 0\right)$

Пользуясь возможностями параллельных вычислений в Julia, построим ансамбль траекторий (200 штук) и покажем на графике среднее и дисперсию по ансамблю.

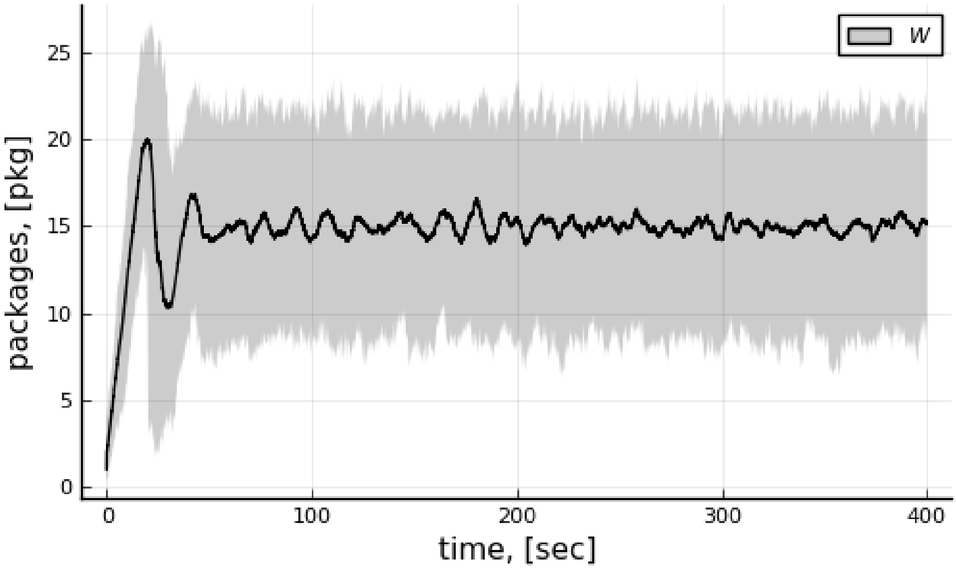

Рисунок 3.13. График среднего по ансамблю (200 траекторий) размера TCP-окна $W(t)$ ($Q_{min} = 0{,}55, Q_{max} = 0{,}6$)

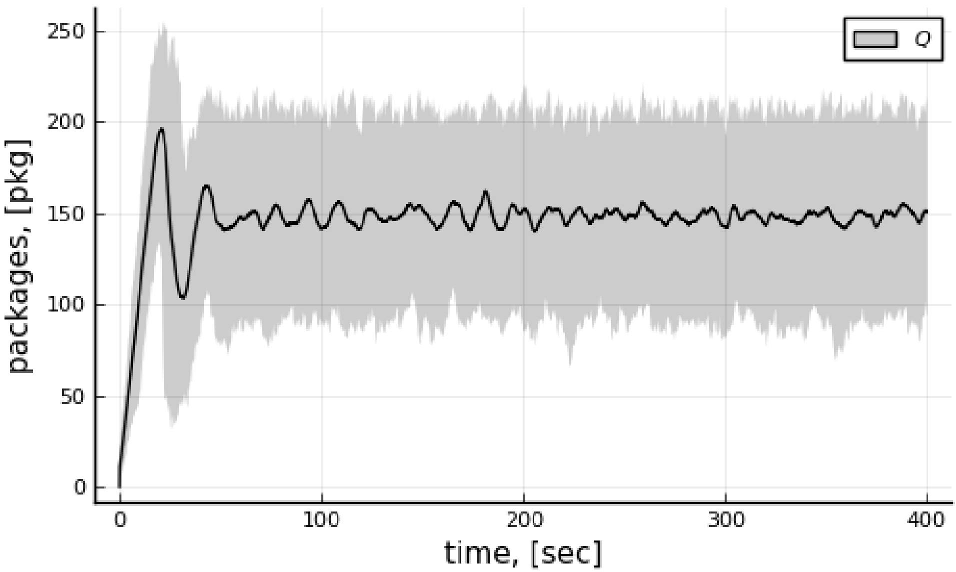

Рисунок 3.14. График среднего по ансамблю (200 траекторий) средней длины очереди $Q(t)$ ($Q_{min} = 0{,}55, Q_{max} = 0{,}6$)

Таким образом, были представлены графики изменения размера окна TCP и среднего размера очереди, полученные в результате численного эксперимента при различных пороговых значениях очереди. Наглядно продемонстрировано, что под управлением алгоритм RED TCP-подобные системы достаточно быстро находят подходящие значения переменных для эффективной работы. Алгоритм RED снижает вероятность возникновения перегрузок в очередях маршрутизатора, однако несмотря на очевидные преимущества его алгоритма, например простота реализации в сетевом оборудовании и эффективное функционирование, в данном алгоритме возникает устойчивый автоколебательный режим функционирования системы при некоторых начальных параметрах системы, что делает ее поведение менее стабильным, и в свою очередь, отрицательно влияет на показатели качества обслуживания сети.

Заключение

В ходе данной работы был дан литературный обзор, посвященный обзору исследований в области анализа и моделирования стохастических систем, описан принцип взаимодействия передачи данных по протоколу TCP и алгоритмом регулирования состояния потока при возникновении перегрузок, в качестве которого рассматривался RED, представлена математическая модель данной стохастической системы с запаздыванием, полученная в рамках комбинаторного подхода метода стохастизации одношаговых процессов.

В среде OpenModelica на языке программирования Modelica и в среде Atom на языке Julia было проведено математическое моделирование описанной системы, в результате исследования получены два программных комплекса, дающие сопоставимые результаты. В результате численного эксперимента были получены графики изменений основных параметров системы, отражающие факт наличия автоколебательного режима функционирования данная в данной стохастической системе с управлением при некоторых начальных параметрах.

Оба рассматриваемых языка относятся к объектно-ориентированным языкам, однако языка Modelica — специализированный язык моделирования динамических систем, как гибридных, так и непрерывных. Julia же позиционируется как язык, разработанный для реализации научных расчетов, что сказалось на сложности и быстроте реализации модели. Система была описана в рамках гибридного подхода, и реализация детерминированной модели на специализированной Modelica была достаточно простой, дискретный контроллер и непрерывные элементы системы хорошо взаимодействовали друг с другом в рамках записанных уравнений и не требовали дополнительного контроля. В языке же Julia для контроля переменных потребовалось использовать дополнительный инструмент, а именно функцию обратных вызовов. Также реализация запаздывающих как непрерывных, так и дискретных переменных осуществлялась с помощью функции `delay`, на языке же Julia требовалось вводить функцию истории `hist`, реализация которой требует от программиста более высокой квалификации, чем при работе с языком Modelica.

Несмотря на преимущества языка Modelica при моделировании детерминированного алгоритма RED, данный язык не пригоден для моделирования случайных процессов, в стандартной библиотеке OpenModelica нет функций или подпрограмм для генерации псевдослучайных чисел. Julia же направлена на решение более широкого круга задач по сравнению с Modelica. Этот гибкий язык имеет возможности объединения библиотек и решателей, переопределения методов и обладает большим количеством инструментов для научных расчетов, в том числе для стохастического моделирования. Так, в рамках данной работы использовала библиотека DifferentialEquations.jl, позволяющая решать различные виды стохастических уравнений. Еще одним преимуществом Julia является встроенная многопоточность расчетов внутри методов, что улучает производительность программы при моделировании стохастических уравнений.

Таким образом, в рамках данной работы

1. Рассмотрено применение гибридного подхода к моделированию стохастических систем с запаздыванием на примере системы передачи данных по протоколу TCP с политикой активного управления очередью, в качестве которой выступает алгоритм RED;

2. Представлена программная реализация модели стохастической системы с запаздыванием на примере системы передачи трафика с регулируемой алгоритмом типа RED динамической интенсивностью потока в средствах Modelica и Julia;

3. Проведен сравнительный анализ возможностей Modelica и Julia по моделированию гибридных стохастических систем с запаздыванием.

Список литературы

1. Апреутесей А.М.Ю., Федоров А.В., Королькова А.В., Кулябов Д.С. Диаграммное описание для закрытой и открытой модели Шлёгеля // Системы управления, технические системы: устойчивость, стабилизация, пути и методы исследования. Материалы молодежной секции в рамках VI Международной научно-практической конференции, посвященной 100-летию со дня рождения профессора А. А. Шестакова (16–17 сентября 2020 г.). – Елец: Елецкий государственный университет им. И. А. Бунина, 2020. — С. 156–162.
2. Апреутесей А.М.Ю., Королькова А.В., Кулябов Д.С. Возможности гибридного моделирования систем с управлением на языках Modelica и Julia // Распределенные компьютерные и телекоммуникационные сети: управление, вычисление, связь (DCCN-2020): материалы XXIII Международной научной конференции 14–18 сент. 2020 г. – Москва. – С. 434-440.
3. Apreutesey A.M.Y., Korolkova A.V., Kulyabov D.S. Hybrid modelling of the RED algorithm in the Julia language // Journal of Physics: Conference Series. – 2020. – Vol. 1694. – doi:10.1088/1742-6596/1694/1/012025.
4. Apreutesey A.M.Y., Korolkova, A.V., Kulyabov D.S. Modeling RED algorithm modifications in the OpenModelica // CEUR Workshop Proceedings. – 2019. – Vol. 2407. – Pp. 5-14.
5. Apreutesey A.M.Y., Korolkova A.V., Kulyabov D.S. Languages for modeling the RED active queue management algorithms: Modelica vs. Julia // CEUR Workshop Proceedings. – 2020. – Vol. 2639. – Pp. 130-140.
6. Апреутесей А.Ю., Федоров А.В. Применение комбинаторного и операторного подходов к закрытой и открытой модели Шлёгеля // XXVII международная конференция «Математика. Компьютер. Образование»: Сборник трудов международной конференции «Математика. Компьютер. Образование», 2020. – С. 162.
7. Апреутесей А.М.Ю. Математическое моделирование системы с управлением по алгоритму типа RED на языке Julia // Информационно-телекоммуникационные технологии и математическое моделирование высокотехнологичных систем: материалы Всероссийской конференции с

международным участием. Москва, РУДН, 13–17 апреля 2020 г. – М.: РУДН, 2020. – С. 240-243.

8. Апреутесей А.Ю. Гибридное моделирование нелинейных систем с управлением на языке Julia // XXVIII международная конференция «Математика. Компьютер. Образование»: Сборник трудов международной конференции «Математика. Компьютер. Образование», 2021.
9. Апреутесей А.М.Ю. Моделирование стохастического алгоритма управления очередями RED на языке Julia // Информационно-телекоммуникационные технологии и математическое моделирование высокотехнологичных систем: материалы Всероссийской конференции с международным участием. Москва, РУДН, 19–23 апреля 2021 г. – М.: РУДН, 2021.
10. Понтрягин Л.С., Андронов А.А., Витт А.А. О статистическом рассмотрении динамических систем // ЖЭТФ. – 1933. – Т.3, № 3. – С. 165-180.
11. Бернштейн С.Н. Принципы теории стохастических дифференциальных уравнений // Труды физико-математического института им. В.А. Стеклова, 1933. – Т. 5. – С. 95-124.
12. Bernstein S.N. Equations diffe'rentielles stochastiques // Act. Sci. et Ind. 738 Conf. Intern. Sci. Math. Univ. Gene've. – Paris: Herman, 1938. – Pp.5-31.
13. Гихман И.И. Об одной схеме образования случайных процессов // Докл. АН СССР. – 1947. – Т. 58, № 6. – С. 961-964.
14. Гихман И.И. О некоторых дифференциальных уравнениях со случайными функциями // Укр. мат. журн. – 1950. – Т. 2, № 4. – С. 45-69.
15. Гихман И.И. К теории дифференциальных уравнений случайных процессов // Укр. мат. журн. – 1950. – Т. 2, № 4. – С. 37-63.
16. Ito K. Differential equations determining Markov processes // Zenkoku Shijo Sugaka Danwakai, – 1942, – Vol.244, № 1077, – P.1352-1400.
17. Ito K. Stochastic integral // Proc. Imperial Acad., Tokyo, 1944, – Vol. 20, – Pp. 519-524
18. Dantzig G.B. Linear Programming under Uncertainty // Management Science. – 1955. – Vol. 1, № 3-4. – Pp. 197-206. – doi:10.1287/mnsc.1.3-4.197
19. Ермольев Ю.М. Методы стохастического программирования. – М.: Наука, 1976.

20. Растригин Л.А. Теория статистических методов поиска. – М.: Наука, 1968.
21. Гихман И.И., Скороход А.В. Стохастические дифференциальные уравнения. – Киев: Наукова думка, 1982.
22. Лукшин А.В., Смирнов С.Н. Численные методы решения стохастических дифференциальных уравнений // Математическое моделирование. – 1990. – Т. 2, № 11. – С. 108-121.
23. Allen E.J., Allen L.J.S., Arciniega A., Greenwood P. Construction of equivalent stochastic differential equation models // Stochastic Analysis and Applications. – 2008. – Vol. 26. – Pp. 274-297.
24. Zou X., Wang K. Numerical simulations and modeling for stochastic biological systems with jumps // Communications in Nonlinear Science and Numerical Simulation. – 2014. – Vol.19, №5. – Pp. 1557–1568. – doi:10.1016/j.cnsns.2013.09.010
25. Marujma G. Continuous Markov process and stochastic equations // Rendiconti del Circolo Matematico di Palermo. – 1955. – Vol. 2, № 4. – Pp. 48.
26. Артемьев С.С., Марченко М.А., Корнеев В.Д., Якунин М.А., Иванов А.А., Смирнов Д.Д. Анализ стохастических колебаний методом Монте-Карло на суперкомпьютерах // Рос. акад. наук, Сиб. отд., ИВМиМГ. – Новосибирск: СО РАН, 2016.
27. Wagner W. Unbiased Monte Carlo Evaluation of Certain Functional Integrals // Journal of Computational Physics. – 1987. – Vol. 71, № 1. – Pp. 21-33. – doi:10.1016/0021-9991(87)90017-9.
28. Zhe Wang, Tianzhen Hong, Ruoxi Jia Buildings.Occupants: a Modelica package for modelling occupant behaviour in buildings // Journal of Building Performance Simulation. – 2019. – Vol. 12, № 4. – Pp. 433-444. – doi:10.1080/19401493.2018.1543352.
29. Кузнецов Д.Ф. Стохастические дифференциальные уравнения: теория и практика численного решения. С программами в среде MatLab // Дифференциальные уравнения и процессы управления. – 2017. – № 2. – С. 1001-2000.
30. Landeros A., Stutz T., Keys K., Alekseyenko A., Sinsheimer J.S., Lange K., Sehl M. BioSimulator.jl: Stochastic simulation in Julia // Computer Methods and

Programs in Biomedicine. – 2018. – Vol. 167. – Pp. 23-25. – doi:10.1016/j.cmpb.2018.09.009.

31. Shampine L.F. Reichelt M.W. The MATLAB ODE suite // SIAM Journal on Scientific Computing. – 1997. – Vol. 18, № 1. – Pp.1-22. – doi:10.1137/S1064827594276424.
32. Hairer E., Nørsett S.P., Wanner G. Solving Ordinary Differential Equations I: Nonstiff Problems. – Berlin: Springer–Verlag, 1993.
33. Hindmarsh A.C., Brown P.N., Grant K.E., Lee S.L., Serban R., Shumaker D.E., Woodward C.S. Sundials: Suite of nonlinear and differential/algebraic equation solvers // ACM Transactions on Mathematical Software. – 2005. – Vol. 31, № 3. – Pp. 363-396. – doi:10.1145/1089014.1089020.
34. Rackauckas C., Nie Q. DifferentialEquations.jl – A Performant and Feature-Rich Ecosystem for Solving Differential Equations in Julia // Journal of Open Research Software. – 2017. – Vol. 5, № 1. – Pp. 15. – doi:10.5334/jors.151.
35. TCP Congestion Control. RFC-2581 / Ed. by M. Allman, V. Paxson,W. Stevens. — United States: RFC Editor, 1999.
36. Jacobson V. Congestion Avoidance and Control // SIGCOMM ’88: Symposium Proceedings on Communications Architectures and Protocols. – New York, NY, USA: ACM, 1988. – Pp. 314–329.
37. Requirements for Internet Hosts – Communication Layers. RFC-1122 / Ed. by R. Braden. – United States: RFC Editor, 1989.
38. Floyd S. Explicit Congestion Notification (ECN) Mechanism in the TCP/IP Protocol // ACM Computer Communications Review. — 1994. — Vol. 24.
39. The NewReno Modification to TCP's Fast Recovery Algorithm. RFC-6582 / Ed. by T. Henderson, S. Floyd, A. Gurtov, Y. Nishida. – United States: RFC Editor, 2012.
40. Breitenecker F. Classification and evaluation of features in advanced simulators / F. Breitenecker, N. Popper // Proceedings MATHMOD-09 Vienna, Full papers CD Volume. – 2009. – Pp. 1445-1467.
41. Floyd S., Jacobson V. Random Early Detection gateways for congestion avoidance // IEEE / ACM Transactions on Networking. – 1993. – Vol. 1, №4, – Pp. 397-413.

42. Башарин Г.П., Гайдамака Ю.В., Самуйлов К.Е., Яркина Н.В. Управление качеством и вероятностные модели функционирования сетей связи следующего поколения: Учеб. пособие. – М.: РУДН, 2008.
43. Korolkova A.V., Eferina E.G., Laneev E.B. et al. Stochastization of one-step processes in the occupations number representation // Proceedings - 30th European Conference on Modelling and Simulation, ECMS 2016. – 2016. – Pp. 698-704.
44. Demidova A.V., Korolkova A.V., Kulyabov D.S., Sevastianov L.A. The method of stochastization of one-step processes // Mathematical Modeling and Computational Physics. – JINR, 2013. – P. 67.
45. Королькова А.В., Кулябов Д.С., Черноиванов А.И. К вопросу о классификации алгоритмов RED // Вестник РУДН, серия «Математика. Информатика. Физика». — 2009. — № 3. — С. 34-46.
46. Королькова А.В., Кулябов Д.С. Математическая модель динамики поведения параметров систем типа RED // Вестник РУДН. Серия «Математика. Информатика. Физика». – 2010. – № 2(1). – С. 54-64.
47. Korolkova A.V., Kulyabov D.S., Velieva T.R. and Zaryadov I.S. Essay on the study of the self-oscillating regime in the control system // Proceedings - European Council for Modelling and Simulation 33rd International ECMS Conference on Modelling and Simulation. – 2019. – Vol. 33, № 1. – Pp. 473-480.
48. Modelica Language Specification, Version 3.3. Modelica Association (May 9, 2012) // Режим доступа: URL: https://www.modelica.org/documents/ModelicaSpec33.pdf. — (дата обращения: 08.05.2021).
49. Апреутесей А.Ю., Завозина А.В., Королькова А.В., Кулябов Д.С. Вычислительная и имитационная модели системы с управлением на Modelica // Вестник РУДН. Серия «Математика. Информатика. Физика». – 2018. – № 4. – С. 357-370.
50. Апреутесей А.М.Ю., Королькова А.В., Кулябов Д.С. Моделирование модификаций алгоритма RED в среде OpenModelica // Информационно-телекоммуникационные технологии и математическое моделирование высокотехнологичных систем: материалы Всероссийской конференции с международным участием. Москва, РУДН, 15–19 апреля 2019 г. – Москва: РУДН, 2019. – С. 398-406.

51. Bezanson J., Karpinski S., Shah V.B., Edelman A. Julia: A Fast Dynamic Language for Technical Computing. — 2012. — P. 1–27. — arXiv:1209.5145.
52. Bezanson J., Edelman A., Karpinski S., Shah V.B. Julia: A fresh approach to numerical computing // SIAM Review. —2017. — 1. — Vol. 59, no. 1. — P. 65-98. — arXiv:1411.1607.
53. Joshi A., Lakhanpal R. Learning Julia. – Birmingham-Mumbai: Packt, 2017.
54. Elmqvist H., Henningsson T., Otter M. Systems Modeling and Programming in a Unified Environment Based on Julia // Leveraging Applications of Formal Methods, Verification and Validation: Discussion, Dissemination, Applications. ISoLA 2016. Lecture Notes in Computer Science. 2019. –Vol. 9953. – Pp. 198-217. – doi:10.1007/978-3-319-47169-3_15
55. Rackauckas C., Nie Q. DifferentialEquations.jl – A Performant and Feature-Rich Ecosystem for Solving Differential Equations in Julia // Journal of Open Research Software. — 2017. — Vol. 5. – Pp. 15-25. – doi:10.5334/jors.151

Приложение 1

Программа для ЭВМ «Гибридное моделирование алгоритма управления трафиком RED»

В разработке программы участвовали А.М.Ю. Апреутесей (реализация алгоритма), А.В. Королькова (постановка задачи).

Полный листинг программы:

```
function wAddFunc
input Real wIn;
input Real wmax;
input Real T;
output Real wOut;
algorithm
wOut := if noEvent(wIn >= wmax) then 0.0 else 1.0 /
T;
end wAddFunc;

function qAddFunc
input Real N;
input Real R;
input Real q;
input Real w;
input Real T;
input Real C;
output Real qOut;
protected Real q_add;
protected Real q_new;
algorithm
q_add := N * w / T - C;
q_new := q + q_add;
qOut := if noEvent(q + q_add > 0.0) then q_add else -
q;
end qAddFunc;

class Red
parameter Real N(start = 80.0) «Количество сессий»;
parameter Real c(start = 10.0) «Интенсивность
обслуживания, Mbps»;
parameter Real packet_size(start = 600.0) «Размер
пакета, bit»;
parameter Real T(start = 0.5) «Время двойного
оборота»;
parameter Real thmin(start = 0.25) «Нормализованный
нижний порог»;
parameter Real thmax(start = 0.8) «Нормализованный
```

```
верхний порог»;
parameter Real R(start = 300.0) «Размер очереди»;
parameter Real w_q(start = 0.0004) «Параметр EWMS»;
parameter Real pmax(start = 0.1) «Максимальная
вероятность сброса»;
parameter Real wmax(start = 32.0) «Максимальный
размер окна»;
Real p(start = 1.0) «Вероятность сброса»;
Real w(min = 1.0, max = wmax, start = 1.0, fixed = true)
«Окно»;
Real q(max = R, start = 0.0, fixed = true) «Мгновенная
длина очереди»;
Real q_avg(start = 0.0) «EWMS длины очереди»;
Real C = 125000.0 * c / packet_size «Интенсивность
обслуживания, packets»;

equation
der(w) = wAddFunc(w,wmax,T) - w * delay(w,T) * delay(p,T)
/ (2 * delay(T,T));
der(q) = qAddFunc(pre(q), w, T, C, N, R);
der(q_avg) = w_q * C * (q - q_avg);

p = if ( q_avg < thmin*R ) then 0.0
elseif ( q_avg > thmax*R ) then 1.0
else (q_avg/(R - thmin)) * pmax / (thmax - thmin);

when q >= R then
reinit(q, R);
end when;

when w <= 1.0 then
reinit(w, 1.0);
end when;

end Red;
```

Приложение 2

Программа для ЭВМ «Численное моделирование стохастического алгоритма управления трафиком RED»

В разработке программы участвовали А.М.Ю. Апреутесей (реализация алгоритма), Д.С. Кулябов (постановка задачи).

Полный листинг программы:

```
# подключение библиотек

using DifferentialEquations
using Plots
using LaTeXStrings

# параметры системы

T0 = 0.01 # T
N = 10.0 # N
C0 = 1400 # C
wq = 0.0007 # wq
q_min = 0.55 # q_min
q_max = 0.6 # q_max
R = 300.0   # R
p_max = 0.1 #p_max
w_max = 32.0 #w_max
# вектор начального состояния
param = (T0, N, C0, wq, q_min, q_max, R, p_max, w_max)

# параметры моделирования

tf = 100.0
tspan = (0.0, tf)
dt = 0.01 # dt для метода EM()

# вектор начального состояния системы

u0 = [1.0, 0.0, 0.0]

function C(q)
    if q > C0
        return C0
    else
        return q
    end
end
```

```
function T(q)
    if q > 0
        return T0 + q / C(q)
    else # q == 0
        return T0
    end
end

function p(q_avg)
    global q_min, q_max

    if (q_avg < q_min * R)
        p = 0.0
    elseif (q_avg > q_max * R)
        p = 1.0
    else
        p = p_max * (q_avg / R - q_min) / (q_max - q_min)
    end
end

function RED(du, u, param, t)
    w, q, q_avg = u
    if w < w_max
        du[1] = 1.0 / T(q) - ( w / 2 )* w * p(q_avg) /
T(q)
    else
        du[1] = - ( w / 2 )* w * p(q_avg) / T(q)
    end
    du[2] = N * w / T(q) - C(q)
    du[3] = wq * C(q) * (q - q_avg)
end

function σ_RED(du,u,param,t)
  w, q, q_avg = u

   du[1] = sqrt(1.0 / T(q) + ((w / 2) * w * p(q_avg)/
T(q)))
   if ((N * w / T(q) - C(q)) < 0)
     du[2] = 0.0
   else
     du[2] = sqrt(N * w / T(q) - C(q))
   end
   du[3] = 0.0
end

function condition_w_max(u,t,integrator)
    global w_max
    u[1] >= w_max
end
```

```
function affect_w_max!(integrator)
    global w_max
    for c in full_cache(integrator)
        c.u[1] = w_max
    end
end

function condition_Rq(u,t,integrator)
    global R
    u[2] >= R
end

function affect_Rq!(integrator)
    global R
    for c in full_cache(integrator)
        c.u[2] = R
    end
end

function condition_Rq_avg(u,t,integrator)
    global R
    u[3] >= R
end

function affect_Rq_avg!(integrator)
    global R
    for c in full_cache(integrator)
        c.u[3] = R
    end
end

# объявление callback'ов
save_positions = (true,true)
cb_w_max      =      DiscreteCallback(condition_w_max,
affect_w_max!, save_positions=save_positions)
cb_Rq   =   DiscreteCallback(condition_Rq,   affect_Rq!,
save_positions=save_positions)
cb_Rq_avg      =      DiscreteCallback(condition_Rq_avg,
affect_Rq_avg!, save_positions=save_positions)

# контейнер для callback'ов
Clbsset = CallbackSet(PositiveDomain(), cb_w_max, cb_Rq,
cb_Rq_avg)

prob_sde_RED = SDEProblem(RED, σ_RED, u0, tspan)

# вызов решателя
```

```
sol = solve(prob_sde_RED, EM(), dt=dt, callback = Clbsset)

# построение графиков
pyplot(size = (500, 300))
plot(sol, vars = (1),
    labels  = L"W",
    xlabel = "time, [sec]",
    ylabel = "packages, [pkg]",
     color = :black
    )

plot(sol, vars = (2),
    labels  = L"Q",
    xlabel = "time, [sec]",
    ylabel = "packages, [pkg]",
    color = :black
    )

# ансамблевое моделирование
ensembleprob = EnsembleProblem(prob_sde_RED)
sim = solve(ensembleprob,EM(), dt=dt, callback = Clbsset, trajectories=100)

summ = EnsembleSummary(sim, 0:dt:tf)

# построение среднего и дисперсии по ансамблю
pyplot(size = (500, 300))
plot(summ,
    idxs = (1,), # w(t)
    fillalpha=0.2,
    labels  = L"W",
    legend = :topright,
    xlabel = "time, [sec]",
    ylabel = "packages, [pkg]",
    color = :black,
    line = (:line,1)
    )

plot(summ,
    idxs = (2,), # q(t)
    fillalpha=0.2,
    labels  = L"Q",
    legend = :topright,
    xlabel = "time, [sec]",
    ylabel = "packages, [pkg]",
    color = :black,
    line = (:line,1))
```