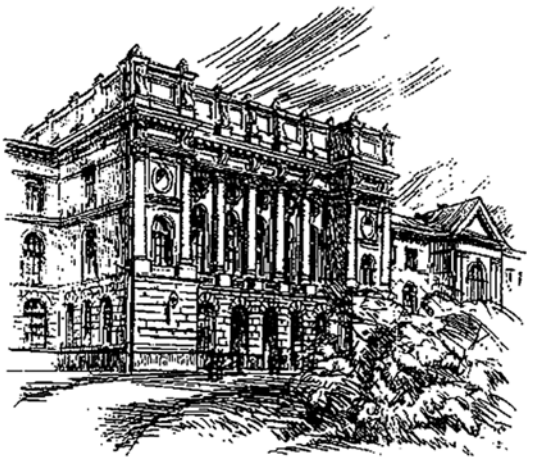
МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ



НАУЧНО-ТЕХНИЧЕСКИЕ ВЕДОМОСТИ

САНКТ-ПЕТЕРБУРГСКОГО ГОСУДАРСТВЕННОГО ПОЛИТЕХНИЧЕСКОГО УНИВЕРСИТЕТА

Информатика. Телекоммуникации. Управление

2(193) 2014

Издательство Политехнического университета
Санкт-Петербург
2014

НАУЧНО-ТЕХНИЧЕСКИЕ ВЕДОМОСТИ САНКТ-ПЕТЕРБУРГСКОГО ГОСУДАРСТВЕННОГО ПОЛИТЕХНИЧЕСКОГО УНИВЕРСИТЕТА
ИНФОРМАТИКА. ТЕЛЕКОММУНИКАЦИИ. УПРАВЛЕНИЕ

Журнал с 1995 года издается под научно-методическим руководством Российской академии наук. С 2008 года выпускается в составе сериального периодического издания «Научно-технические ведомости СПбГПУ» ISSN 1994-2354.

Журнал с 2002 года входит в Перечень ведущих рецензируемых научных журналов и изданий, в которых должны быть опубликованы основные результаты диссертаций на соискание ученой степени доктора и кандидата наук.

Сведения о публикациях представлены в Реферативном журнале ВИНИТИ РАН, в международной справочной системе «Ulrich`s Periodical Directory».

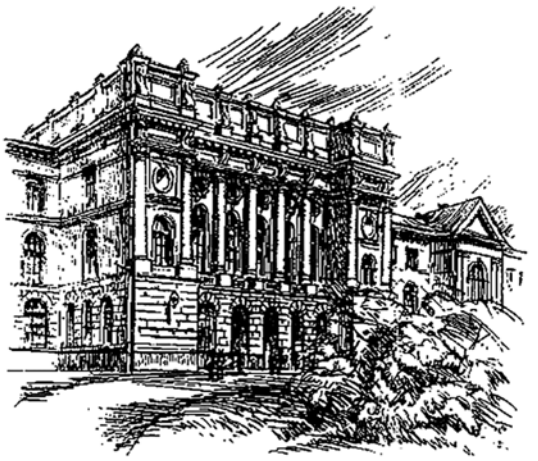
Журнал зарегистрирован Федеральной службой по надзору в сфере информационных технологий и массовых коммуникаций (Роскомнадзор). Свидетельство о регистрации ПИ № ФС77-51457 от 19.10.2012 г.

Подписной индекс **47517** в объединенном каталоге «Пресса России».

Журнал включен в базу данных «Российский индекс научного цитирования» (РИНЦ), размещенную на платформе Научной электронной библиотеки на сайте http://www.elibrary.ru

Адрес редакции и издательства: Россия, 195251, Санкт-Петербург, ул. Политехническая, д. 29.
Тел. редакции (812) 552-62-16.

THE MINISTRY OF EDUCATION AND SCIENCE OF THE RUSSIAN FEDERATION



ST. PETERSBURG STATE POLYTECHNICAL UNIVERSITY JOURNAL

Computer Science. Telecommunications and Control Systems

2(193) 2014

Polytechnical University Publishing House
Saint Petersburg
2014

ST. PETERSBURG STATE POLYTECHNICAL UNIVERSITY JOURNAL
COMPUTER SCIENCE. TELECOMMUNICATIONS AND CONTROL SYSTEMS

The journal is published under scientific and methodical guidance of the Russian Academy of Sciences since 1995. The journal is published since 2008 as part of the periodical edition «Nauchno-tekhnicheskie vedomosti SPbGPU» (ISSN 1994-2354).

The journal is included in the List of Leading Peer-Reviewed Scientific Journals and other editions to publish major findings of PhD theses for the research degrees of Doctor of Sciences and Candidate of Sciences.

The publications are presented in the VINITI RAS Abstract Journal and Ulrich's Periodical Directory International Database.

The journal is registered with the Federal Service for Supervision in the Sphere of Telecom, Information Technologies and Mass Communications (ROSKOMNADZOR). Certificate ПИ № ФС77-51457 issued Oct. 19, 2012.

Subscription index **47517** in the «Press of Russia» Joint Catalogue.

The journal is on the Russian Science Citation Index (RSCI) database
© Scientific Electronic Library (http://elibrary.ru/).

The address: 195251 Polytekhnicheskaya Str. 29, St. Petersburg, Russia.

Содержание

Contents

УДК 004.023, 004.383.8

Е.В. Агеев, Е.Н. Бендерская

ОБЗОР ПРИРОДНЫХ ВЫЧИСЛЕНИЙ: ОСНОВНЫЕ НАПРАВЛЕНИЯ И ТЕНДЕНЦИИ

E.V. Ageev, E.N. Benderskaya

REVIEW OF NATURAL COMPUTING: IMPORTANT TRENDS

Проведена классификация природных вычислений, представлены основные элементы природных вычислений и области применения. Предложены дополнительные классификационные признаки природных вычислений, а также схема общности природных вычислений. Показаны варианты комбинирования видов природных вычислений. Определены тенденции развития природных вычислений.

ПРИРОДНЫЕ ВЫЧИСЛЕНИЯ; БИОИНСПИРИРОВАННЫЕ ВЫЧИСЛЕНИЯ; ХАОТИЧЕСКИЕ ВЫЧИСЛЕНИЯ; НОВЫЕ ПАРАДИГМЫ ВЫЧИСЛЕНИЙ; ОБРАЗОВАНИЕ СТРУКТУРЫ.

The classification of natural computing and the basic elements of natural computing and natural computing applications are considered. Additional classification features of natural computing and natural computing interconnection scheme are proposed. Variants of hybrid natural computations are discussed. Tendencies of the development of natural computing are revealed.

NATURAL COMPUTING; BIO-INSPIRED COMPUTING; CHAOTIC COMPUTING; NEW COMPUTING PARADIGM; STRUCTURE FORMATION.

На сегодняшний день существует большое многообразие моделей природных вычислений. *Природными* (*естественными* или первоначально именуемыми *биоинспирированными*) вычислениями называются динамические модели, заимствованные так или иначе у природы и основанные на природных процессах, таких как информационные процессы, протекающие в клетках, эволюционные процессы естественного отбора, процессы взаимодействия простых живых организмов, процессы образования различных материалов и т. п. При этом формализованные в виде динамических моделей природные процессы используются для решения различных задач, и наиболее проработанными являются специализированные модели вычислений, хотя и универсальные разрабатываются тоже. Возрастающая популярность природных вычислений связана с необходимостью параллельной обработки данных, с возможностью создания искусственных биологических систем, с созданием новых парадигм вычислений, с исследованиями многообразия природы.

Классическими разделами природных вычислений на текущий момент являются клеточные автоматы, искусственные нейронные сети, генетические алгоритмы [3, 8, 22]. Клеточные автоматы появились в первой половине XX в. Клеточные автоматы представляют собой дискретную систему, изменяющую свое состояние, по шагам используя правила перехода. Вычис-

Виды природных вычислений

Вид вычислений	Природный аналог	Базовые элементы	Базовые правила
1	2	3	4
Мембранные вычисления (P системы)	Мембраны, клетки, бактерии, молекулы	Набор мембран, набор содержимого в мембранах, правила эволюции, среда	Правила эволюции: растворение мембран, разделение мембраны на две, перемещение содержимого мембран из одной мембраны в другую или в среду. На каждом шаге применяется максимальное мультимножество правил параллельным способом
Искусственные иммунные системы	Иммунная система человека	Антиген, антитело, обучение	Распознавание своих и чужих
Искусственная жизнь [18]	Естественная жизнь	Агент, среда, поведение	Взаимодействие агента в среде
ДНК-вычисления	ДНК	Основания: аденин, тимин, гуанин, цитозин, правила изменения, пробирка	Комплементарность Уотсона-Крика, правила соединения, разъединения, удлинения, укорочения, модификации
Искусственные нейронные сети	Биологические нейроны	Нейрон, нейронная сеть, состоящая из нескольких слоев, функция активации, обучение	Нейроны преобразуют входные данные, настраивая весовые коэффициенты. Нейроны способны обучаться
Роевой интеллект	Муравьи, пчелы, стаи птиц, светлячки	Агент, фермент	Уровень феромона, испарение феромона
Brain-вычисления	Мозг человека	Моделирование целого мозга человека или любого другого живого направления: составление карт целого мозга, моделирование целого	
Physarum вычисления [12]	Слизевик physarum polycephalum	Используются особенности организма physarum polycephalum для	
Аморфные вычисления	Развитие многоклеточных организмов из одной клетки, организация субклеточных отделов и внутриклеточная сигнализация	Множество идентичных вычислительных элементов, коммуникационный радиус	Конкуренция за лидерство. Лидеры набирают другие процессоры. Набранные лидером процессоры перестают конкурировать и становятся членами группы лидеров

Таблица 1

Результат	Подвиды вычислений, алгоритмы	Применение	Программные средства для моделирования
5	6	7	8
Содержимое мембраны выхода, последовательность выхода элементов в среду по шагам	С активными мембранами, симпорт-антипорт, с правилами перезаписи, тканевые, нейронные	Решение NP полных задач, моделирование биологических систем, алгоритмизация	Плагин к среде eclipse, язык программирования P-lingua [15]
Получение всех чужих элементов	Алгоритм отрицательного отбора	Распознавание образов, когнитивные модели, методы вычислений, методы обнаружения аномалий и неисправностей, мультиагентные системы, модели самоорганизации, модели искусственной жизни, системы компьютерной безопасности, модели обучающих систем, методы извлечения информации, выявление подделок, методы обработки сигналов и изображений	Jisys
Результат взаимодействия	Миры решетки, виртуальные среды, синтетическая наука о поведении [5]	Моделирование биологических систем	Avida
Находится в конечной пробирке	Вставка-удаление, соединение	Создание ДНК-программ	Sequencher 5.1
Результат считывается с выходного слоя нейронов	Самоорганизующиеся карты Кохонена, сети прямого распространения, рекуррентные нейронные сети	Распознавание образов, аппроксимация, прогнозирование, управление, классификация, кластеризация, принятие решений, сжатие данных, ассоциативная память	Matlab
Условия останова	Муравьиный алгоритм, пчелиный алгоритм, метод роя частиц, кошачий алгоритм	Распознавание образов, поиск кратчайшего пути	Matlab
существа. Так, полностью область «brain-вычислений» еще не сформировалась, отметим основные аспекты данного мозга, моделирование сознания, нейронные вычисления			
выполнения вычислений		Аппроксимация, поиск кратчайшего пути	–
Состояние, в котором все процессоры – лидеры или члены группы	Клубный алгоритм	Моделирование биологических систем	Симулятор Gray-Scott язык программирования gpl (growing point language)

1	2	3	4
Вычисления «реакция-диффузия»	Химические реакции	Входной концентрационный профиль, выходной концентрационный профиль	Взаимодействие волн
L-системы (системы Линденмайера)	Растения, деревья	Алфавит, аксиома, порождающие правила	Порождающие правила
Бактериальные вычисления	Кишечная палочка	Автономные, основанные	
Нечеткие вычисления (нечеткая логика)	Лингвистическая неопределенность	Мера принадлежности, лингвистическая переменная, терм	База правил
Клеточные автоматы	Клетки	Решетка клеток, клетка, соседние клетки, правила перехода	Правила перехода
Квантовые вычисления [7]	Фотон	Кубит	Математические операции
Эволюционные вычисления	Механизмы эволюции (выбор, скрещивание, отбор популяции)	Популяция, механизм скрещивания, фитнес-функция	Различные способы отбора
Мягкие вычисления	Комбинирование нечетких вычислений,		
Вычисления, основанные на столкновениях	Столкновения	Постоянные локализации, переменные локализации, вектор скорости, состояния	Наличие локализаций, отсутствие локализаций
Вычисления, основанные на динамических системах и хаосе [11, 17]	Время	Передаточная функция	Передаточная функция

ления на основе искусственных нейронных сетей появились также в первой половине XX в. после работы Мак-Каллока и Питца. Основой для них послужил биологический нейрон. Главной особенностью этого вида вычислений является обучение нейрона. Генетические алгоритмы были разработаны во второй половине XX в. Джоном Холландом и его студентами. Они основаны на принципах эволюции.

Последующее развитие работ в направлении заимствований у природных процессов привело к появлению новых разделов природных вычислений, например, таких как ДНК-вычисления, роевой интеллект, аморфные вычисления [6, 14, 20, 22]. В конце XX в. после опыта Эдлмана были созданы ДНК-вычисления. Они основываются на молекулах ДНК и используют комплементарность Уотсона–Крика и операции

Окончание таблицы 1

5	6	7	8
Конечный концентрационный профиль	Модель Грея–Скотта	Моделирование химических реакций	ReaDDy
Вычисленная функция на требуемом шаге	Контекстно-независимые, контекстно-зависимые, детерминированные, стохастические, параметрические, непараметрические	Моделирование роста растений	Расширение к открытой программе векторной, графики inkscape
на клетке машины Тьюринга		Создание машин Тьюринга на клетке	–
Выходные переменные	Четкие вычисления, нечеткие вычисления	Распознавание образов, управление	Matlab
Конечная конфигурация, в которой все клетки постоянны, периодическая конфигурация	Одномерные клеточные автоматы, многомерные клеточные автоматы	Распознавание языка, машина Тьюринга, моделирование физических систем	Matlab
Результат математических операций	Алгоритмы коррекции ошибок	Защита данных	Язык программирования QCL
Конечная популяция, которая получается на основании критерия останова	Генетические алгоритмы, эволюционные алгоритмы, генетическое программирование, эволюционное программирование, эволюционные стратегии	Решение NP полных задач, поиск экстремумов	Matlab
генетических алгоритмов и нейронных сетей			Matlab
Состояния локализаций	Биллиардный компьютер	Моделирование столкновений	Matlab
Результат передаточной функции	Хаотические вычисления, линейные системы, нелинейные системы	Обработка информации, управление	Matlab

над молекулами ДНК. Роевой интеллект основан на поведении групп, например, стаи птиц, роя пчел, колонии муравьев. Одним из используемых элементов роевого интеллекта является фермент, по которому другие элементы роя узнают необходимую информацию. Также к новым видам природных вычислений относятся аморфные вычисления. Они также основаны на процессах в клетке. Ключевыми особенностями аморфных вычислений являются идентичность, ограниченность ресурсов и наличие коммуникационного радиуса, в пределах которого могут происходить взаимодействия.

Кроме указанных выше природных вычислений к новым разделам относятся мембранные вычисления, искусственные иммунные системы, brain-вычисления [4, 13, 16, 19, 21]. Каждый из видов природных

вычислений использует некоторые особенности природных процессов, например ДНК-вычисления основаны на ДНК и ее свойствах, мембранные вычисления – на взаимодействии клеток и т. д., однако, что касается brain-вычислений, для них еще не сформировалась полная теоретическая база, тем не менее существуют достаточно проработанные отдельные модели.

Постоянно пополняющееся число различных неклассических моделей вычислений, разнообразие описаний и отсутствие систематизации в этой бурно развивающейся области знания обусловливает потребность в разработке классификаций природных вычислений, а также определении сходства и различий как по математическим моделям, так и по достоинствам и ограничениям, заложенным изначально в тот или иной вид природных вычислений.

Цель данной работы – проведение систематизации моделей природных вычислений и определение основных направлений развития и тенденций в новой и считающейся перспективной области науки решения сложных и ресурсоемких задач.

Основные виды природных вычислений

Природные (естественные) вычисления – область науки, занимающаяся решением задач методами, отличающимися от классических методов представления вычислительных процессов. Природные вычисления можно классифицировать по видам вычислений, например, нейронные сети, мембранные вычисления, L-системы, ДНК-вычисления. Виды природных вычислений, их природные аналоги, базовые элементы, правила, а также некоторые варианты применения и программные средства для моделирования приведены в табл. 1, где представлено 19 видов вычислений.

Из таблицы видно, что степень проработанности каждого из видов природных вычислений разная, и как один из показателей может использоваться показатель наличия разнообразных средств моделирования: для одних видов вычислений существуют универсальные библиотеки программ, а для других только разрозненные программы.

Кроме перечисленных выше видов к природным вычислениям также относятся: модели синтетической биологии, искусственные биологические системы, оптические вычисления, фрактальная геометрия, генная сборка, искусственная химия, нановычисления, эволюционирующие аппаратные средства.

Начало природным вычислениям было положено разработкой нейросетевого аппарата и генетических алгоритмов, а также созданием теории нечеткой логики. Необходимость в решении все более сложных задач подтолкнула исследователей к разработке гибридных методов, в результате появилась новая область науки – мягкие вычисления, которая объединила в себе подходы всех трех указанных областей. Были предложены также дополняющие технологии решения трудноформализуемых задач.

Дальнейшее развитие аппарата нейронных сетей шло по пути большего усложнения и большей детализации процессов, протекающих в нервной клетке, – биоинспирированные нейронные сети с биоподобными нейронами. Большая детализация процессов эволюции и генетического отбора и более подробное изучение работы клеток различного типа (не только нервных) привели к разработке ДНК-вычислений, клеточных, мембранных и иммунных вычислений. Первоначально указанные типы вычислений разрабатывались разрозненно, как и подходы к решению задач с помощью мультиагентных технологий. При этом основой для концепции мультиагентных систем стал децентрализованный распределенный подход, реализуемый на основе большого числа относительно простых элементов – агентов. В дальнейшем задачи разработки и исследования коллективного поведения и взаимодействия (коллективный интеллект) рассматривались как на уровне особей – агентов (роевой интеллект, интеллект колонии муравьев, стаи рыб и т. д.), так и на уровне агентов – отдельных клеток, антител, мембран, генов. Способы математической формализации работы целой системы из агентов оказались во многом схожими. Таким образом, сформировалось направле-

ние биоинспирированных вычислений [10], которое оказалось практически близким к мягким вычислениям, а по сути, только расширяющим методы решения задач дополнительными биоинспирированными подходами, которые сами по себе являются «мягкими». С другой стороны, процессы, протекающие в динамической многомерной системе с нелинейностями, изучаются в теории хаоса и теории динамических систем. Физические и химические процессы, рассматриваемые в том или ином типе биологических систем, выраженные на языке математических моделей, оказываются схожими с процессами, наблюдаемыми в неживой природе (образование облаков, цунами, торнадо, кластеризация в физике при изучении различных свойств материалов). Многие природные явления неживой материи также послужили толчком к созданию новых неклассических способов обработки информации [1, 2].

Классификации природных вычислений

Природные вычисления, рассмотренные в табл. 1, основаны на различных принципах, поэтому естественно возникает необходимость систематизации иного типа. Были предложены дополнительные классификационные признаки и на их основе разработаны дополнительные варианты классификации, представленные в табл. 2.

Общность, различия, ограничения и перспективы разных направлений в природных вычислениях представляется важным рассмотреть и оценить. Каждый классификационный признак раскрывает определенные возможности и области применения вычислений. Некоторые виды природных вычислений не могут быть отнесены только к одному значению классификационного признака. Например, вычисления, основанные на столкновениях, динамических системах и хаосе являются общими в классификации по уровню и имеющимся аналогам.

Родство видов природных вычислений

Из табл. 1 и 2 следует, что многие виды вычислений основаны на схожих принципах. Схема общности видов природных вычислений представлена на рисунке. Это согласуется и с логикой последовательного развития области знания «природные вычисления».

Схема общности позволяет сделать вывод, что природные вычисления основаны на вычислениях с признаками «общие» и «комбинирование». Другие вычисления можно рассматривать как частные случаи этих признаков. В данной работе исследуется признак «комбинирование».

Несмотря на некоторые сходства видов вычислений, каждый вид рассматривает тот или иной признак сходства по-своему. Признаки сходства и индивидуальные особенности природных вычислений представлены в табл. 3.

Развитие видов природных вычислений

Многие из природных вычислений могут быть формализованы в разной математической форме с привлечением разных математических моделей. Так, осцилляторные нейронные сети часто рассматриваются с позиций нелинейной динамики взаимодействия множества осцилляторных элементов и феномена кластеризации на их основе, а интерпретация результатов анализа может быть проведена как с привлечением аппарата нейронных сетей, так и без него. Поэтому можно говорить о наличии общности не только среди биоинспирированных подходов, но и среди природных в целом [9].

Многие физические, химические и биологические процессы стали объектом для изучения с последующей искусственной (технической) реализацией для решения задач обработки информации и управления. Для этого необходимо было найти связь между переменными, параметрами модели соответствующего процесса и обрабатываемой информацией. Так, например, абстрактный феномен кластеризации в природе и его простая модель связанных логистических решеток смогли послужить прототипом хаотической нейронной сети для решения задач кластеризации как только был предложен вариант обеспечения взаимосвязи между входными данными, подлежащими кластеризации и параметрами сети [2, 11].

Таблица 2

Дополнительные классификации природных вычислений

Классификационный признак	Уровень			
	нано	микро	макро	мета
По уровню	Нановычисления; ДНК-вычисления; квантовые вычисления; оптические вычисления; генная сборка; вычисления «реакция-диффузия»	Бактериальные вычисления; мембранные вычисления; аморфные вычисления; physarum вычисления	Роевой интеллект; искусственная жизнь; клеточные автоматы; искусственные нейронные сети; искусственные иммунные сети; эволюционные вычисления; нечеткие вычисления; мягкие вычисления; L-системы; синтетическая биология	brain-вычисления; вычисления, основанные на столкновениях; вычисления, основанные на динамических системах и хаосе; фрактальная геометрия
По природным аналогам	Аналоги			
	математические дополнения	биологические	физические	химические
	Фрактальная геометрия; L-системы; мягкие вычисления; нечеткие вычисления	Нейронные вычисления; иммунные вычисления; ДНК-вычисления; мембранные вычисления; генная сборка; клеточные автоматы; роевой интеллект; искусственная жизнь; бактериальные вычисления; аморфные вычисления; brain-вычисления; physarum вычисления; синтетическая биология; эволюционные вычисления	Оптические вычисления; квантовые вычисления; нановычисления; эволюционирующие аппаратные средства	Вычисления «реакция-диффузия»; искусственная химия
По видам природных аналогов	Виды природных аналогов			
	процесс	организм	свойство	
	Квантовые вычисления; оптические вычисления; эволюционные вычисления; генная сборка; искусственная химия; вычисления «реакция-диффузия»; роевой интеллект; искусственная жизнь; четкие-нечеткие вычисления; мягкие вычисления; вычисления, основанные на столкновениях; вычисления, основанные на динамических системах и хаосе; brain-вычисления	ДНК-вычисления; искусственные нейронные сети; искусственные иммунные сети; бактериальные вычисления; мембранные вычисления; аморфные вычисления; клеточные автоматы; physarum вычисления; роевой интеллект; искусственная жизнь; четкие-нечеткие вычисления; L-системы; синтетическая биология; биологические системы; мягкие вычисления; brain-вычисления	Нановычисления; ДНК-вычисления; квантовые вычисления; оптические вычисления; эволюционные вычисления; генная сборка; мембранные вычисления; аморфные вычисления; клеточные автоматы; искусственная химия; вычисления «реакция-диффузия»; эволюционирующие аппаратные средства; искусственная жизнь; четкие-нечеткие вычисления; L-системы; синтетическая биология; биологические системы; мягкие вычисления; фрактальная геометрия; brain-вычисления	

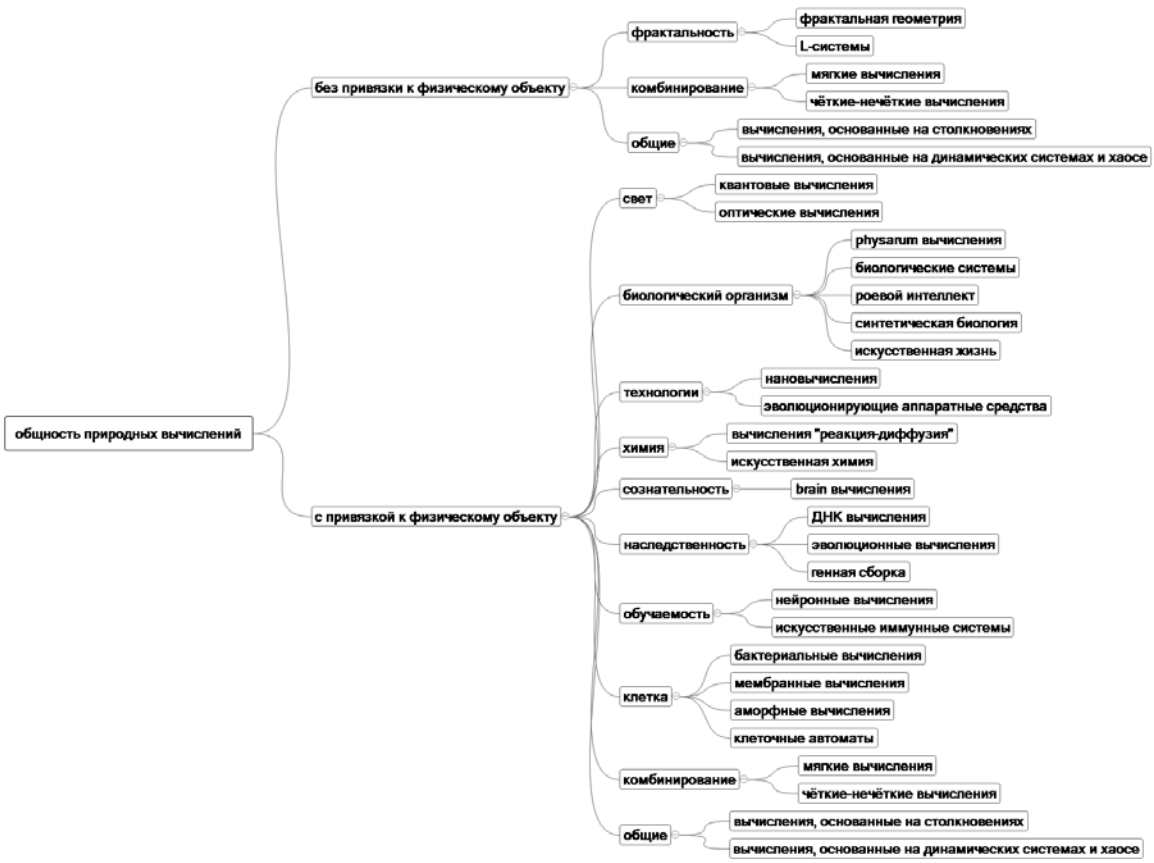


Схема общности природных вычислений

При все большей детализации процессов, протекающих в природе, наблюдается процесс поиска и разработки универсального инструмента решения разноплановых задач. Как следствие появился новый вид природных вычислений – brain-вычисления, – один из самых новых.

Комбинирование вычислений необходимо для достижения определенных целей, например, для составления комбинированных систем, для уменьшения времени вычислений.

Комбинирование по различным классификационным признакам может привести к улучшению того или иного результата. Например, если используется:

классификация по типу решаемых задач, то комбинирование позволит добиться лучших результатов по сравнению с использованием одного вида вычислений;

классификация по уровню, то комбинирование может развивать, микро, нано или любой другой уровень;

классификация по изменениям числа элементов, то комбинирование позволит использовать более динамичные системы;

комбинирование по достоинствам или недостаткам, то это позволит улучшить соответствующие достоинства или недостатки.

В результате комбинирование сводится к следующим вариантам.

1. Один вид вычислений используется в качестве элемента другого вида вычислений.

2. Создается новый вид вычислений, содержащий свойства нескольких видов вычислений.

3. Используются виды природных вычислений как независимые. Например, определенные части задачи решают различные виды вычислений, а в итоге все части соединяются в единое целое.

4. Комбинирование первых трех вариантов.

Приведем несколько примеров комбинирования видов природных вычислений.

Таблица 3

Различия видов природных вычислений

Признак сходства	Вид природных вычислений	Индивидуальные особенности
Клетка	Бактериальные вычисления	Клетка рассматривается как автономная машина Тьюринга
	Мембранные вычисления	Рассматривается только одна клетка («скин мембрана») и все взаимодействие в ней, а также ее взаимодействие со средой, но среда без других клеток
	Аморфные вычисления	Ограниченность ресурсов клетки
	Клеточные автоматы	Не рассматривается внутренняя структура клетки. Рассматривается взаимодействие клетки с соседями
Фрактальность	Фрактальная геометрия	Рассматриваются все фракталы
	L-системы	Рассматриваются только определенные правила
Свет	Квантовые вычисления	Параллельная обработка данных. Рассматривается состояние суперпозиции
	Оптические вычисления	Рассматривается высокая производительность
Обучаемость	Нейронные вычисления	Положение нейронной сети фиксировано
	Искусственные иммунные системы	Положение иммунной системы не фиксировано
Наследственность	ДНК-вычисления	Программы, комплементарность Уотсона–Крика
	Эволюционные вычисления	Рассматриваются механизмы скрещивания
	Генная сборка	Рассматриваются только механизмы сборки генов
Биологический организм	Physarum вычисления	Рассматривается поведение только организма Physarum polycephalum
	Биологические системы	Исследование биологических систем
	Роевой интеллект	Рассматривается только поведение роя пчел, колонии муравьев, стаи птиц и т. д.
	Синтетическая биология	Моделирование биологических систем
	Искусственная жизнь	Рассматривается агент, среда и поведение агента в среде
Сознательность	Brain-вычисления	Предположительно, мозг рассматривается как сознательная единица
Комбинирование	Мягкие вычисления	Используются эволюционные, нечеткие и нейронные вычисления
	Четкие-нечеткие вычисления	Используются четкие и нечеткие вычисления
Химия	Вычисления «реакция-диффузия»	Моделирование химических реакций
	Искусственная химия	Моделирование химических реакций
Общие	Вычисления, основанные на столкновениях	Столкновения
	Вычисления, основанные на динамических системах и хаосе	Динамика
Технологии	Нановычисления	Используется взаимодействие материи на наноуровне
	Эволюционирующие аппаратные средства	Восстановление

Таблица 4

Основные свойства природных вычислений

Вид вычислений	Основные свойства вычислений
Вычисления, основанные на динамических системах и хаосе	Динамика
Клеточные автоматы	Дискретность
Искусственные нейронные сети	Параллельная обработка данных, обучаемость, самоорганизация
Эволюционные вычисления	Выделение определенных свойств
Нановычисления	Выполнение алгоритмов на наномасштабе
Четкие-нечеткие вычисления	Объединение свойств четких и нечетких вычислений
L-системы (системы Линденмайера)	Моделирование роста растений и деревьев
Фрактальная геометрия	Моделирование самоподобных структур
Квантовые вычисления	Состояние суперпозиции
Оптические вычисления	Высокая производительность
Вычисления, основанные на столкновениях	Столкновения
Искусственная жизнь	Программируемая среда, программируемые агенты
Искусственная химия	Моделирование химических реакций
Роевой интеллект	Параллельная обработка данных
ДНК-вычисления	ДНК-программа, комплементарность Уотсона–Крика
Мягкие вычисления	Объединение свойств эволюционных вычислений, нечетких вычислений и нейронных вычислений
Аморфные вычисления	Параллельная обработка данных, ограниченность
Мембранные вычисления (Р системы)	Параллельная обработка данных
Вычисления «реакция-диффузия»	Моделирование химических реакций
Бактериальные вычисления	Автономность, машина Тьюринга
Искусственные иммунные системы	Обучаемость, распознавание «своих» и «чужих»
Physarum вычисления	Параллельная обработка данных
Генная сборка	Моделирование сборки генов
Эволюционирующие аппаратные средства	Самовосстановление
Синтетическая биология	Моделирование биологических систем
Биологические системы	-*
Brain-вычисления	Сознательность

* Является областью исследований

Комбинирование ДНК-вычислений и эволюционных вычислений позволяет рассматривать элементы, реализующие ДНК-алгоритмы как популяции и выявлять те ДНК-алгоритмы, которые справляются с поставленной задачей лучше остальных.

Комбинирование роевого интеллекта и нейронных вычислений позволяет рассматривать элемент роя как обучающийся элемент.

Комбинирование нейронных вычислений и клеточных автоматов позволяет рассматривать обучающие клеточные автоматы, т. е. каждая клетка в клеточных автоматах является выходом нейрона.

Комбинирование квантовых вычислений и клеточных автоматов позволяет получать клеточные автоматы с квантовыми свойствами.

В табл. 4 представлены свойства, которые добавляются в систему, если она использует виды природных вычислений и один из них является элементом другого. В качестве основного свойства brain-вычислений используем наличие сознания.

В качестве перспектив развития природных вычислений можно отметить направление, связанное с объединением различных видов природных вычислений в одной системе. Комбинирование природных вычислений позволит более гибко настраивать систему. Исходя из родства природных вычислений следует, что комбинируя вычисления, например, с признаком «клетка», можно получить более реалистичные системы, т. к. в них будут учитываться признаки всех комбинируемых видов вычислений.

Одним из направлений в области природных вычислений по-прежнему остается поиск аналогов в природе для последующей реализации в виде алгоритмов. Однако самоорганизация условных алгоритмов принятия решений возможна только при наличии большого числа степеней свободы у системы и, как правило, при наличии первоначальной хаотической динамики, из которой под действием внешних воздействий образуется структура.

Проведенная разноплановая классификация природных вычислений с учетом разных признаков общности подходов и уровней детализации позволяет провести как сравнение различных природных вычислений, так и их систематизацию. По результатам анализа существующих на сегодняшний день природных вычислений можно сделать вывод о перспективности хаотических вычислений и вычислений, базирующихся на фундаментальном заделе в области нелинейных динамических систем, как ввиду возможности описания наиболее сложных явлений, так и по универсальности применения. В этом же ряду стоят brain-вычисления, в значительной мере использующие хаотические вычисления и придающие им иерархическую организацию.

Работа выполнена при частичной финансовой поддержке правительства Санкт-Петербурга.

СПИСОК ЛИТЕРАТУРЫ

1. **Бендерская Е.Н.** Перспективные концепции разработки интеллектуальных систем // Научно-технические ведомости СПбГПУ. Информатика. Телекоммуникации. Управление. СПб.: Изд-во СПбГПУ, 2011. № 6.1 (138). С. 173–181.

2. **Бендерская Е.Н., Жукова С.В.** Анализ процессов самоорганизации в хаотических физико-технических системах и их использование для обработки информации // Научно-технические ведомости СПбГПУ. СПб.: Изд-во СПбГПУ, 2008. № 4(63). С. 131–138.

3. **Беркович С.Я.** Клеточные автоматы как модели реальности: поиски новых представлений физических и информационных процессов. Пер. с англ. М.: Изд-во МГУ, 1993. 112 с.

4. **Дасгупта Д.** Искусственные иммунные системы и их применение. М.: Физматлит, 2006. 341 с.

5. **Джонс М.Т.** Программирование искусственного интеллекта в приложениях. Пер. с англ. М.: ДМК Пресс, 2004. 312 с.

6. **Паун Г., Розенберг Г., Саломаа А.** ДНК-компьютер. Новая парадигма вычислений. Пер. с англ. М.: Мир, 2003. 528 с.

7. **Прескилл Дж.** Квантовая информация и квантовые вычисления. 2008. Т. 1. 464 с.

8. **Хайкин С.** Нейронные сети: полный курс. 2-е изд. Пер. с англ. М.: ИД «Вильямс», 2006. 1104 с.

9. **Benderskaya E.N., Zhukova S.V.** Multidisciplinary trends in modern artificial intelligence: Turing's way // Studies in Computational Intelligence. Springer, 2013. Vol. 427. Pp. 319–343.

10. **Benderskaya E.N., Zhukova S.V.** Dynamic data mining: synergy of bio-inspired clustering methods // Knowledge-Oriented Applications in Data Mining. InTech, 2011. Pp. 398–410.

11. **Benderskaya E.N., Zhukova S.V.** Nonlinear approaches to automatic elicitation of distributed oscillatory clusters in adaptive self-organized sys-

tem // Advances in Intelligent and Soft Computing. Springer, 2012. Vol. 151. Pp. 733–741.

12. **Calude C.S., Costa J.F.** (eds.) Pre-proceedings of the Workshop Physics and Computation. 2008. 353 p.

13. **Ciobanu G., Paun G., Perez-Jimenez M.J.** Applications of Membrane Computing: Natural Computing Series. Springer, 2006. 439 p.

14. **D'Hondt E.** Exploring the amorphous computing paradigm. 2000. 101 p.

15. **Dıaz-Pernil D., Perez-Hurtado I., Perez-Jimenez M.J., Riscos-Nunez A.** P-Lingua: A programming language for membrane computing // 6th Brainstorming Week on Membrane Computing. Sevilla: University of Sevilla, 2008. Pp. 135–155.

16. **Grossberg S.** Foundations and New Paradigms of Brain Computing: Past, Present, and Future Artificial Intelligence Around Man and Beyond // Lecture Notes in Computer Science. 2011. Vol. 6934. Pp. 1–7.

17. **Kia B., Murali K., Jahed Motlagh M.R., Sinha S., Ditto W.L.** Synthetic Computation: Chaos Computing, Logical Stochastic Resonance, and Adaptive Computing // Internat. Conf. on Theory and Application in Nonlinear Dynamics Understanding Complex Systems. Springer, 2014. Pp. 51–65.

18. **Komosinski M., Adamatzky A.** Artificial Life Models in Software. Springer, 2009. 464 p.

19. **Matsumoto G.** Brain computing Artificial Life and Robotics. 1999. Vol. 3. Iss. 1. Pp. 24–26.

20. **Panigrahi B.K., Shi Y., Lim M.H.** Handbook of swarm intelligence. Springer, 2011. 556 p.

21. **Popovici-Vlad O., Curaj A.** Robot Control System Structure from Classical to «Evolutionary Brainway» Approach Acta // Electrotehnica. 2001. Vol. 42. No. 1. Pp. 87–94.

22. **Rozenberg G.** Handbook of natural computing. Springer, 2012. 2097 p.

REFERENCES

1. **Benderskaya E.N.** Perspektivnyye kontseptsii razrabotki intellektualnykh system, *Nauchno-tekhnicheskiye vedomosti SPbGPU. Informatika. Telekommunikatsii. Upravleniye*, St. Petersburg: SPbGPU Publ., 2011, No. 6.1 (138), Pp. 173–181. (rus)

2. **Benderskaya E.N., Zhukova S.V.** Analiz protsessov samoorganizatsii v khaoticheskikh fiziko-tekhnicheskikh sistemakh i ikh ispolzovaniye dlya obrabotki informatsii, *Nauchno-tekhnicheskiye vedomosti SPbGPU,* St. Petersburg: SPbGPU Publ., 2008, No. 4(63), Pp. 131–138. (rus)

3. **Berkovich S. Ya.** *Kletochnyye avtomaty kak modeli realnosti: poiski novykh predstavleniy fizicheskih i informatsionnykh protsessov,* Per. s angl., Moscow: MGU Publ., 1993, 112 p. (rus)

4. **Dasgupta D.** *Iskusstvennyye immunnyye sistemy i ikh primeneniye,* Moscow: Fizmatlit Publ., 2006, 341 p. (rus)

5. **Dzhons M.T.** *Programmirovaniye iskusstvennogo intellekta v prilozheniyakh*, Per. s angl., Moscow: DMK Press Publ., 2004, 312 p. (rus)

6. **Paun G., Rozenberg G., Salomaa A.** *DNK-kompyuter. Novaya paradigma vychis-leniy,* Per. s angl., Moscow: Mir Publ., 2003, 528 p. (rus)

7. **Preskill Dzh.** *Kvantovaya informatsiya i kvantovyye vychisleniya*, 2008, Vol. 1, 464 p. (rus)

8. **Khaykin S.** *Neyronnyye seti: polnyy kurs,* 2-ye izd., Per. s angl., Moscow: ID «Vilyams» Publ., 2006, 1104 p. (rus)

9. **Benderskaya E.N., Zhukova S.V.** Multidisciplinary trends in modern artificial intelligence: Turing's way, *Studies in Computational Intelligence*, Springer, 2013, Vol. 427, Pp. 319–343.

10. **Benderskaya E.N., Zhukova S.V.** Dynamic Data Mining: Synergy of Bio-inspired Clustering Methods, *Knowledge-Oriented Applications in Data Mining*, InTech Publ., 2011, Pp. 398–410.

11. **Benderskaya E.N., Zhukova S.V.** Nonlinear approaches to automatic elicitation of distributed oscillatory clusters in adaptive self-organized system, *Advances in Intelligent and Soft Computing*, Springer, 2012, Vol. 151, Pp. 733–741.

12. **Calude C.S., Costa J.F.** (eds.) *Pre-proceedings of the Workshop Physics and Computation*, 2008, 353 p.

13. **Ciobanu G., Paun G., Perez-Jimenez M.J.** *Applications of Membrane Computing: Natural Computing Series*, Springer, 2006, 439 p.

14. **D'Hondt E.** *Exploring the amorphous computing paradigm*, 2000, 101 p.

15. **Dıaz-Pernil D., Perez-Hurtado I., Perez-Jimenez M.J., Riscos-Nunez A.** P-Lingua: A programming language for membrane computing, *6th Brainstorming Week on Membrane Computing,* Sevilla, University of Sevilla, 2008, Pp. 135–155.

16. **Grossberg S.** Foundations and New Paradigms of Brain Computing: Past, Present, and Future Artificial Intelligence Around Man and Beyond, *Lecture Notes in Computer Science,* 2011, Vol. 6934, Pp. 1–7.

17. **Kia B., Murali K., Jahed Motlagh M.R., Sinha S., Ditto W.L.** Synthetic Computation: Chaos Computing, Logical Stochastic Resonance and Adaptive Computing, *International Conference on Theory and Application in Nonlinear Dynamics Understanding Complex Systems*, Springer, 2014, Pp. 51–65.

18. **Komosinski M., Adamatzky A.** *Artificial Life Models in Software*, Springer, 2009, 464 p.

19. **Matsumoto G**. *Brain computing Artificial Life and Robotics*, 1999, Vol. 3, Iss. 1, Pp. 24–26.

20. **Panigrahi B.K., Shi Y., Lim M.H.** *Handbook of swarm intelligence*, Springer, 2011, 556 p.

21. **Popovici-Vlad O., Curaj A.** Robot Control System Structure from Classical to «Evolutionary Brainway» Approach Acta, *Electrotekhnika*, 2001, Vol. 42, No. 1, Pp. 87–94.

22. **Rozenberg G.** *Handbook of natural computing*, Springer, 2012, 2097 p.

АГЕЕВ Евгений Владимирович – *студент 6-го курса кафедры компьютерных систем и программных технологий Санкт-Петербургского государственного политехнического университета.*
195251, Россия, Санкт-Петербург, ул. Политехническая, д. 29.
E-mail: agevg@yandex.ru

AGEEV, Evgeniy V. *St. Petersburg State Polytechnical University.*
195251, Politekhnicheskaya Str. 29, St. Petersburg, Russia.
E-mail: agevg@yandex.ru

БЕНДЕРСКАЯ Елена Николаевна – *доцент кафедры компьютерных систем и программных технологий Санкт-Петербургского государственного политехнического университета, кандидат технических наук.*
195251, Россия, Санкт-Петербург, ул. Политехническая, д. 29.
E-mail: helen.bend@gmail.com

BENDERSKAYA, Elena N. *St. Petersburg State Polytechnical University.*
195251, Politekhnicheskaya Str. 29, St. Petersburg, Russia.
E-mail: helen.bend@gmail.com

УДК 004.932

С.Ю. Беляев, В.Г. Шубников

ОПРЕДЕЛЕНИЕ РАЗМЕРОВ СТОПЫ ЧЕЛОВЕКА ПО ФОТОГРАФИЯМ

S.Yu. Belyaev, V.G. Shubnikov

ESTIMATING A HUMAN FOOT SIZE FROM IMAGES

Интернет-магазины обуви заинтересованы в снижении процента отказов от покупки (по причине несоответствия размера) после доставки обуви клиенту. Этого можно достичь с помощью виртуальной примерки. Для ее выполнения клиент (частное лицо, не владеющее специальным измерительным оборудованием) должен выполнить измерения характерных размеров своей стопы и передать их через on-line сервис в интернет-магазин.

Недостатком такого подхода является недостаточная точность измерений с помощью доступных клиенту измерительных инструментов. Использование фотографий стопы для определения ее характерных размеров решает эту проблему.

Предложен метод, восстанавливающий по трем фотографиям горизонтальный и вертикальный контуры стопы и определяющий их физические размеры с достаточной для указанного применения точностью.

ИЗМЕРЕНИЕ СТОПЫ ЧЕЛОВЕКА; ОБРАБОТКА ИЗОБРАЖЕНИЙ; РАСПОЗНАВАНИЕ ОБРАЗОВ; ПОДБОР ОБУВНЫХ ИЗДЕЛИЙ; ОЦЕНКА УДОБСТВА НОШЕНИЯ ОБУВНЫХ ИЗДЕЛИЙ.

Most of the internet shoe shops are interested in selling rate increasing. Main problem is inside too high percentage of delivery returns due to wrong shoe fit. Minimization of wrong shoe size selection can be achieved using virtual shoe fitting. In this scheme customer (private person without specific measuring equipment) should perform a measurement of some important (non-trivial) dimensions of his(her) feet and pass them through on-line service to the online store.

The disadvantage of this approach is the lack of accuracy of measurements with the measuring instruments available to the client. Using digital foot images it is possible to detect characteristic foot sizes automatically, with enough high level of accuracy.

This paper proposes a method for horizontal and vertical foot contours restoration from digital images, and detection physical foot size with sufficient accuracy for shoe fitting applications.

HUMAN FOOT MEASUREMENT; IMAGE PROCESSING; IMAGE RECOGNITION; SHOE FIT; SHOE COMFORT ESTIMATION.

Для оценки размеров стопы на сегодняшний день не существует простых и одновременно надежных методов. Обычно используется старый метод, которым великолепно владеют профессиональные мастера по изготовлению обуви. С помощью линейки и гибкого метра мастер замеряет стопу клиента [1] и на основании мерок либо подбирает подходящее обувное изделие, либо создает индивидуальную обувную колодку, которая будет использоваться при фабричном или индивидуальном изготовлении обувного изделия для конкретного клиента. К сожалению, этот метод не может применяться в виртуальной примерочной (рассчитанной на обыкновенного потребителя), т. к. выполнить такие измерения с достаточной точностью могут только опытные мастера. Какая точность измерений является «достаточной» для изготовления подходящего обувного изделия? На этот вопрос нет однозначного ответа, однако известно, что стопа в свободном и нагруженном состоянии отличается по длине приблизительно на 5 мм. При изготовлении обувных изделий многие мастера закладывают порядка 10 мм, чтобы компенсировать увеличение длины стопы, связанное с ходьбой.

Более современный подход состоит в построении 3D модели стопы с помощью 3D сканера (например, этот метод описан в [2]). Результат работы 3D сканера обеспечивает высокую точность (менее 3 мм), но он также не может применяться для виртуальной примерки по причине отсутствия у обыкновенного потребителя (клиента) соответствующего оборудования.

Недавно был предложен способ детектирования силуэта человека путем вычисления «особенных» точек (feature points) и применения цепных кодов для поиска совпадения с искомыми формами [3]. Подобная техника имеет высокий научный потенциал, т. к. существует множество способов вычисления этих самых особенных точек, однако на практике такой прием может привести к большим затратам на производительность из-за высокой вычислительной сложности. Один из широко известных подходов к вычислению особенных точек приведен в [4]. Работа [5] посвящена идентификации походки по изображениям силуэта человека. Авторы предложили разделить контур человека на шесть частей и в каждой применить трансформацию Хофа для выявления линий на сгибах и, тем самым, определения характерных точек. Далее предлагается использовать метод опорных векторов (SVM) [6] для разделения пространства характерных точек и идентификации частей тела. В [7] поставлена задача извлечения из изображения геометрических размеров ладони человека, что, по сути, напоминает задачу, поставленную в нашем исследовании. Авторы предложили провести цветовую сегментацию, а затем бинаризацию изображения ладони. Далее по двухцветному изображению предложено рассчитать некоторые статистические характеристики, такие как среднюю ширину, среднюю высоту, центроид, расстояние до центроида от центра системы координат. По статистическим характеристикам предлагается классифицировать изображение ладони в одном из 24 возможных положений ладони. В очень известной работе [8] предложено использовать гистограммы ориентированных градиентов в качестве описателей свойств изображения для поиска контуров людей в изображении.

В большинстве существующих подходов перед применением алгоритмов распознавания используются различные методы предварительной обработки изображения, цель которых – улучшение визуального качества изображения и уменьшение влияния шумовых помех. Существует огромное количество способов предобработки изображения, мы используем [9] модификацию двустороннего фильтра для уменьшения шумовых помех с целью улучшения качества выделения реберной карты на последующих этапах.

Постановка задачи

Пусть заданы три фотографии, снятые в соответствии с перечисленными ниже требованиями. Первые две фотографии содержат стандартный белый лист бумаги формата А4 (размер которой 210 * 297 мм) и правую стопу в черном носке. На одной из этих фотографий видна внутренняя часть стопы, а на другой – внешняя. Будем называть внутренней часть стопы, направленную к противоположной ноге, когда человек стоит прямо. Эти фотографии сняты камерой в положении «вид сверху». В этом положении плоскость линзы параллельна полу. В дизайнерских пакетах трехмерной графики (например, 3D Studio Max) для описания ориентации камеры используют термины «вид сверху», «вид слева» и т. д. Мы будем придерживаться той же терминологии. Третья фотография содержит внутреннюю сторону ступни, снятую камерой в положении «вид сбоку». Один из краев листа бумаги и ступня должны быть ориентированы параллельно верхнему краю фотографий.

Пример таких фотографий приведен на рис. 1. Далее в тексте эти фотографии будут обозначены «фото 1», «фото 2», «фото 3». Требуется построить горизонтальный и вертикальный контуры стопы и определить ее длину. Почему именно три фотографии, а не две или одна? Если пытаться решить задачу поиска физических размеров стопы только по фото 1, то у нас не будет никакой информации о виде стопы «сбоку».

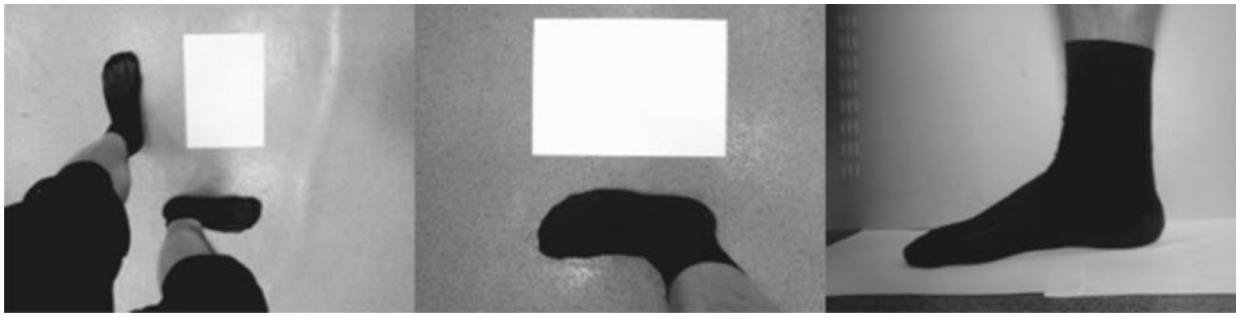


Рис. 1. Пример исходных фотографий. Слева направо – фото 1, 2, 3

Этот вид нам обеспечивает фото 3. Фото 2 необходимо для более точного распознавания внешнего контура стопы, т. к. на фото 1 этот внешний контур частично загорожен ногой и не может быть точно описан.

Пример результирующих контуров приведен на рис. 2.

Функциональная схема метода

Функциональная схема метода построения контуров стопы и определения ее размеров показана на рис. 3.

В соответствии с рис. 3 первые две фотографии обрабатываются в блоках 1, 2 и 4. Блок 1 служит для выделения контура листа бумаги, присутствующего на фотографиях. Эти контуры используются в блоке 3 для вычисления линейных размеров пикселей в рассматриваемых фотографиях по известным линейным размерам листа бумаги. Знание линейных размеров пикселей используется для выделения внутреннего и внешнего контуров стопы в блоках 2 и 4. Эти контуры объединяются в единый контур в блоке 6. В этом же блоке с помощью одного из контуров листа бумаги вычисляется физическая длина горизонтального контура стопы, которая используется в блоке 7 для определения физических размеров вертикального контура стопы, выделенного в блоке 5 из фото 3.

Рис. 2. Пример построенных контуров стопы

Алгоритм построения контуров

Блок-схема алгоритма построения контуров приведена на рис. 4.

В блоке 8 с помощью алгоритма Канни [10] строится реберная карта (edge map). Параметры алгоритма выбираются так, чтобы гарантировать сохранение значимых деталей. Значимыми деталями являются, собственно, части контура стопы. Из-за различных условий освещения, при которых могут быть получены исходные три фотографии, на изображениях разные части стопы могут иметь разный контраст с полом. В результате вычисления градиентов изображения напрямую мы получим не одинаковые по значению градиенты для разных частей стоп (важных для нас в конечном итоге). С другой стороны, выполняя простейшую бинаризацию реберной карты с небольшим порогом, мы получим избыточные ребра. Данная проблема частично решается алгоритмом Канни за счет применения рекурсивного обхода соседних пикселей. Следствием такой стратегии (не потерять никаких важных деталей) является порождение большого количества коротких ребер и шумовой информации, удаление которой выполняется в блоке 9 с помощью следующей морфологической операции. Каждый пиксель изображения окружается прямоугольником, и для всех пикселей периметра прямоугольника сравниваются их значения с нулем. Если все пиксели периметра имеют нулевое значение, то внутри такого прямоугольника нет пикселей,

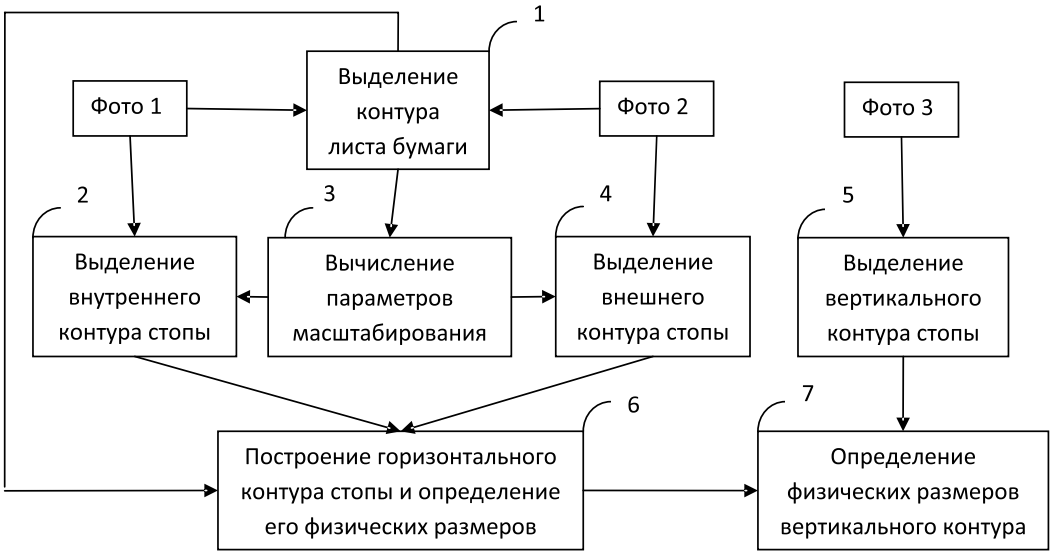


Рис. 3. Функциональная схема метода

относящихся к значимым деталям изображения, и всем внутренним пикселям прямоугольника могут быть присвоены нулевые значения. Более подробно различные морфологические операции для обработки изображений описаны в [11].

Цель специального масштабирования изображения, которое выполняется в блоке 10, – исключение разрывов в искомых контурах. Это достигается путем замены каждого окна размером 8×8 исходного изображения одним пикселем со значением, равным максимальному значению среди пикселей внутри этого окна. Результирующее изображение используется в блоке 11 для нахождения множества замкнутых контуров, из которого в блоке 12 выделяется искомый контур путем проверки каждого элемента множества на соответствие определенным критериям. Эти критерии зависят от формы искомого контура.

При поиске контура листа бумаги (блок 1) применяются следующие критерии:

- контур должен иметь форму четырехугольника;
- длина наибольшей стороны должна превышать ¼ высоты фотографии;

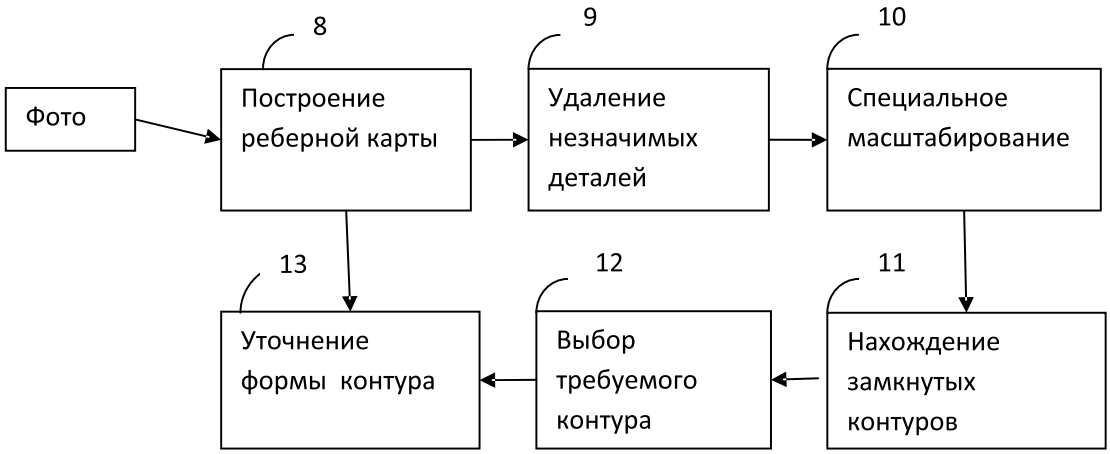


Рис. 4. Блок-схема алгоритма построения контуров

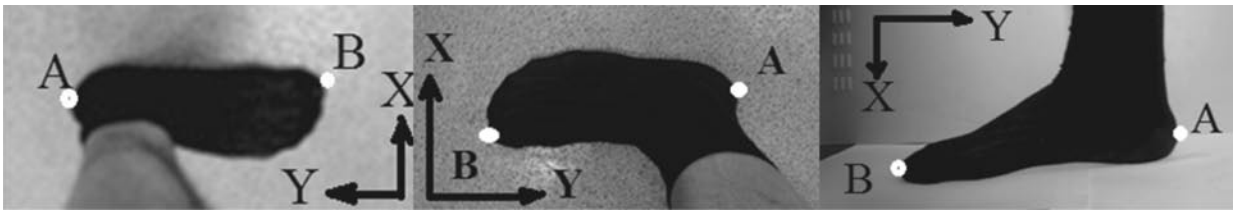


Рис. 5. Обозначения, используемые при распознавании контуров

• пропорции четырехугольника приблизительно соответствуют пропорциям листа бумаги.

Основным критерием при поиске контуров стопы является наличие двух точек A и B, являющихся концами пятки и носка соответственно (рис. 5).

При анализе контуров в системах координат *XY*, показанных на рисунке, точка A обнаруживается как точка, в которой достигается первый локальный максимум при обходе верхней половины контура против часовой стрелки. Точка B определяется как точка контура, наиболее удаленная от A в секторе с углом приблизительно 60° (рассматривается небольшой диапазон углов), направленном против оси *Y*. Дополнительными критериями являются следующие:

• отрезок AB не должен иметь внутренних пересечений с контуром;

• длина отрезка AB должна находиться в пределах возможных длин стоп.

Для проверки последнего критерия длина отрезка AB в пикселях переводится в миллиметры путем умножения на масштаб, вычисленный в блоке 3.

Уточнение формы контура в блоке 13 выполняется с помощью алгоритма активного контура [12]. В соответствии с этим алгоритмом контур приблизительно описывается многоугольником и переносится на реберную карту, полученную в блоке 10. Многоугольник расширяется путем перемещения каждой вершины по направлению ее внешней нормали, пока не будет детектировано пересечение одного из отрезков, связанного с вершиной, с ребром в реберной карте. Для обеспечения гладкости результирующего контура в процессе расширения многоугольника могут порождаться новые вершины, если угол между соседними отрезками меньше заданного барьера. Формальное описание алгоритма приведено ниже:

Создать вписанный в контур многоугольник;
While (хоть одна вершина многоугольника перемещена)
{
For (все вершины)
{
While (нет пересечения с ребрами *edge map*)
Переместить вершину;
If (угол между связанными с вершиной отрезками меньше барьера)
Создать новые вершины в многоугольнике;
}
}

Построение горизонтального контура стопы в блоке 6 выполняется путем объединения внешнего и внутреннего контуров в соответствии со следующим алгоритмом:

1. Выполнить масштабирование внешнего контура так, чтобы длина отреза AB этого контура совпала с длиной такого же отрезка на внутреннем контуре.

2. Совместить эти отрезки.

3. Минимизировать скачки производных в точках совмещения контуров путем вращения внешнего контура относительно центра отрезка AB.

4. Удалить часть внешнего контура, расположенную выше точек совмещения.

5. Удалить часть внутреннего контура, расположенную ниже точек совмещения.

Определение длины стопы

Физические размеры контуров определяются масштабами их изображений. Для определения масштаба горизонтального контура в блоке 6 используется контур ли-

ста бумаги, полученный из первой фотографии. Длина горизонтального контура стопы L вычисляется по формуле:

$$L = w * \frac{|AB|}{|CD| * \cos(\alpha)}, \quad (1)$$

где w – ширина листа бумаги, мм; α – угол между отрезками AB и CD (показаны на рис. 6),

|AB| и |CD| длины отрезков AB и CD соответственно в пикселях. Если контур листа бумаги имеет прямоугольную форму, то CD – нижний отрезок этого контура (рис. 6, слева). Если конур не имеет прямоугольной формы (вследствие перспективных искажений, которые возникают, если камера позиционирована не строго горизонтально), то CD – это отрезок прямой, проходящей через точку O1 (рис. 6, справа) и центр отрезка AB, заключенный между прямыми, являющимися продолжением боковых ребер листа бумаги. Масштаб изображения m дается соотношением:

$$m = \frac{L}{|AB|}. \quad (2)$$

Данная формула используется также в блоке 7 для определения масштаба вертикального контура. Значение L передается в этот блок из блока 6, а значение |AB| из блока 5.

Если в процессе фотосъемки требования по позиционированию камеры не выдержаны, то возникают перспективные искажения контура стопы. Формула (1) учитывает такие искажения при определении длины стопы, однако сам контур остается искаженным. Это порождает ошибки при вычислении ширины и высоты контура. Величина максимальной относительной ошибки δ может быть вычислена по следующей приблизительной формуле:

$$\delta = 1 - \cos\varphi, \quad (3)$$

где φ – угол отклонения камеры от требуемого положения.

Максимальная абсолютная ошибка, например, для стопы шириной 100 мм при наклоне камеры в 10° составляет 1,5 мм. Это значение получается при подстановке значения φ, равного 10°, в формулу (3) и дальнейшего умножения результата на 100.

Абсолютные ошибки, связанные с дискретностью фотографий, составляют менее 1 мм при разрешении фотокамеры более двух мегапикселей. При таком разрешении фотокамеры длина фотографии в пикселях будет составлять не мене 1200 ед. Лист бумаги на фотографии занимает не менее ¼ длины. Длина листа бумаги 295 мм. Следовательно, на 1 мм приходится не менее 1200/4/295 = 1 пиксель.

Представленный метод позволяет измерять стопу человека с весьма жесткими требованиями к погрешности измерения (1,5 мм). Тесты проводились с исполь-

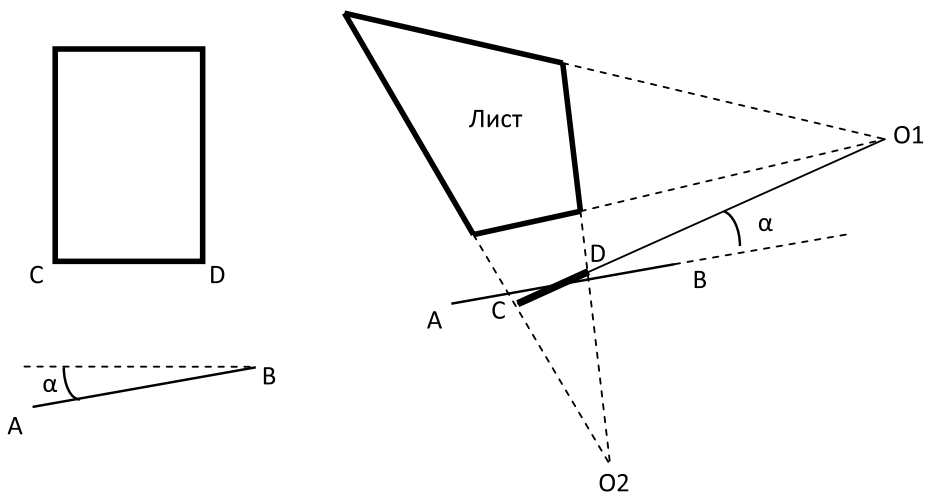


Рис. 6. Обозначения, используемые при вычислении масштабов контуров

зованием бытовых портативных фотокамер и фотокамер, встроенных в современные смартфоны. Показаны серьезные перспективы для исследования проблемы надежного получения размеров стопы по фотографиям. Следующие научные поиски могут быть посвящены снятию многочисленных ограничений, предъявляемых к фотографиям, которые были приняты в данной работе.

СПИСОК ЛИТЕРАТУРЫ

1. **Леденева И.Н.** Индивидуальное изготовление и ремонт обуви. М.: Изд. центр «Академия», 2004.

2. **Lee H., Lee K., Choi T.** Development of a low cost foot-scanner for a custom shoe tailoring system. South Korea: Seoul National University, 2007.

3. **Ling Y-L.** Automatic Feature Extraction from Front and Side Images // Industrial Engineering and Engineering Management. 2008.

4. **Lowe D.G.** Distinctive Image Features from Scale-Invariant Keypoints. 2004.

5. **Ng H., Tong H-L., Tan W-H., Yap T., Chong P-F., Abdulah J.** Human Identification Based on Extracted Gait Features // Internat. J. of New Computer Architectures and Their Applications. 2011.

6. **Burges C.** A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery. 1998.

7. **Mohammed F., Mohhamed W., Kayes A.S.M., Hasan J.** Geometrical Feature Extraction of Human Hand // Internat. J. of Computer and Information Technology. 2013.

8. **Dalal N., Triggs B.** Histogram of Oriented Gradients for Human Detection, Computer Vision and Pattern Recognition. 2005.

9. **Шубников В.Г., Беляев С.Ю.** Подавление шума и оценка различий в изображениях // Научно-технические ведомости СПбГПУ. Информатика. Телекоммуникации. Управление. СПб.: Изд-во СПбГПУ, 2013. № 3(174). С. 58–66.

10. **Canny J.** A computational approach to edge detection // IEEE Trans. Pattern Analysis and Machine Intellegence. 1986. Vol. 8(6). Pp. 679–698.

11. **Gonsalez R., Woods R.** Digital Image Processing. Prentice Hall, 2007.

12. **Kass M., Witkin A., Terzopoulos D.** Snakes: Active Contour Models // Internat. J. of Computer Vision. 1988.

REFERENCES

1. **Ledeneva I.N.** *Individual manufacturing and shoe repairing*, Moscow: Academia Publ., 2004. (rus)

2. **Lee H., Lee K., Choi T.** *Development of a low cost foot-scanner for a custom shoe tailoring system*, Seoul National University, South Korea, 2007.

3. **Ling Y-L.** Automatic Feature Extraction from Front and Side Images, *Industrial Engineering and Engineering Management*, 2008.

4. **Lowe D.G.** *Distinctive Image Features from Scale-Invariant Keypoints*, 2004.

5. **Ng H., Tong H-L., Tan W-H., Yap T., Chong P-F., Abdulah J.** Human Identification Based on Extracted Gait Features, *International Journal of New Computer Architectures and Their Applications*, 2011.

6. **Burges C.** *A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery*, 1998.

7. **Mohammed F., Mohhamed W., Kayes A.S.M., Hasan J.** Geometrical Feature Extraction of Human Hand, *International Journal of Computer and Information Technology*, 2013.

8. **Dalal N., Triggs B.** *Histogram of Oriented Gradients for Human Detection, Computer Vision and Pattern Recognition*, 2005.

9. **Shubnikov V.G., Belyayev S.Yu.** Podavlenie shuma i otsenka razlichiy v izobrazheniyah [Noise removal and difference estimation in images], *Nauchno-tehnicheskie vedomosti SPbGPU. Informatika. Telekommunikatsii. Upravlenie*, St. Petersburg: SPbGPU Publ., 2013, No. 3(174), Pp. 58–66. (rus)

10. **Canny J.** A computational approach to edge detection, *IEEE Trans. Pattern Analysis and Machine Intellegence*, 1986, Vol. 8(6), Pp. 679–698.

11. **Gonsalez R., Woods R.** *Digital Image Processing*, Prentice Hall, 2007.

12. **Kass M., Witkin A., Terzopoulos D.** Snakes: Active Contour Models, *International Journal of Computer Vision*, 1988.

БЕЛЯЕВ Сергей Юрьевич – *профессор кафедры прикладной математики Института прикладной математики и механики Санкт-Петербургского политехнического университета.*

195251, Россия, Санкт-Петербург, ул. Политехническая, д. 29.

E-mail: sergey.belyaev@d-inter.ru

BELYAEV, Sergey Yu. *St. Petersburg State Polytechnical University.*
195251, Politekhnicheskaya Str. 29, St. Petersburg, Russia.
E-mail: sergey.belyaev@d-inter.ru

ШУБНИКОВ Владислав Германович – *доцент кафедры прикладной математики Института прикладной математики и механики Санкт-Петербургского политехнического университета.*
195251, Россия, Санкт-Петербург, ул. Политехническая, д. 29.
E-mail: vlad.shubnikov@gmail.com

SHUBNIKOV, Vladislav G. *St. Petersburg State Polytechnical University.*
195251, Politekhnicheskaya Str. 29, St. Petersburg, Russia.
E-mail: vlad.shubnikov@gmail.com

УДК 04.004

В.Е. Мухин, Я.И. Корнага, В.В. Стешин

АДАПТИВНЫЕ СРЕДСТВА ЗАЩИТЫ КОМПЬЮТЕРНЫХ СИСТЕМ НА ОСНОВЕ МОДИФИЦИРОВАННЫХ НЕЙРОННЫХ СЕТЕЙ КОХОНЕНА

V.Ye. Mukhin, Ya.I. Kornaga, V.V. Steshyn

ADAPTIVE SAFETY MECHANISMS FOR COMPUTER SYSTEMS BASED ON THE MODIFIED KOHONEN NEURAL NETWORKS

Предложен новый подход к обеспечению безопасности распределенных компьютерных систем на основе механизма нейронных сетей. Представлена комплексная адаптивная система защиты с интеллектуальным агентом на основе модифицированных нейронных сетей. Данная система позволяет реализовать адаптивное управление системой защиты информационной системы, обеспечить своевременное реагирование на угрозы безопасности, и оперативное автоматическое принятие решений по управлению параметрами системы защиты. Проведены экспериментальные исследования по определению уровня правильности обнаружения и распознавания угроз безопасности.

РАСПРЕДЕЛЕННЫЕ КОМПЬЮТЕРНЫЕ СИСТЕМЫ; НЕЙРОННАЯ СЕТЬ; ИНТЕЛЛЕКТУАЛЬНЫЙ АГЕНТ; КОМПЛЕКСНЫЕ АДАПТИВНЫЕ СИСТЕМЫ; ОБЕСПЕЧЕНИЕ БЕЗОПАСНОСТИ.

In the paper we propose a new approach to the safety ensurance for the distributed computer systems based on the neural network mechanism. There is described a complex adaptive safety support system with the intelligent agent based on the modified neural networks. This system allows to implement the adaptive control for the security mechanisms and to provide the in-time reaction to the security threats, and also to support the fast decisions making on the security mechanisms parameters modification. Also, we perform the experimental researches to evaluate the correctness level of the threats detection and recognition.

DISTRIBUTED COMPUTER SYSTEM; NEURAL NETWORK; INTELLECTUAL AGENT; COMPLEX ADAPTIVE SYSTEM; SAFETY ENSHURANCE.

Для работы с большими объемами данных и в случаях, когда требуется механизм быстрой обработки информации, используются распределенные компьютерные системы (РКС), структура которых во многом определяется отсутствием центра управления и распределения функций и ресурсов между всеми узлами системы. Типичный пример такой системы – сеть Интернет, обеспечивающая открытую и масштабируемую коммуникационную среду. Данный принцип является базовым для всех распределенных систем, что в целом снижает общий уровень их безопасности. В сети любой субъект может отправить пакет данных любому приемнику, при этом получатель должен обработать соответствующим образом полученный пакет данных.

Снижение уровня безопасности заключается в том, что злоумышленник может сформировать фальшивую учетную запись и практически безнаказанно генерировать и передавать вредоносный трафик. Таким образом, все соединенные узлы системы находятся в состоянии потенциальной опасности, поскольку присущее им свойство открытости делает их доступными для атакующей стороны. Методы обеспечения безопасности распределенных компьютерных систем, базирующиеся на стандартных методах проверки авторизационных данных, не могут в полной мере противодействовать этим угрозам.

Как свидетельствует статистика «Лаборатории Касперского» большинство атак в последние годы направлены на получение доступа к распределенной системе и запуск процессов с правами зарегистрированного пользователя, а также на специальную модификацию данных, генерацию DDoS-атак для отказа системных ресурсов или для создания эффекта перегрузки в системе. Также распространены атаки на уязвимости, позволяющие изменять данные, обходить систему защиты и проводить XSS-атаки [1].

Особенностями современных атак является увеличение сложности комплекса атакующих действий и соответствующего технологического уровня атак. Чаще всего атаки реализуют многоуровневый алгоритм и имеют распределенную структуру, что существенно увеличивает их опасность и последствия реализации. В результате особой задачей современных систем обеспечения безопасности распределенных систем является обработка и анализ большого объема данных, в частности, с использованием интеллектуального подхода. Системы защиты должны поддерживать распознавание и классификацию угрозы, а в случае отсутствия информации о типе атаки, адаптироваться к новому типу атакующих действий. Таким образом, ввиду приведенных выше требований к построению механизмов защиты распределенных систем, для повышения их защищенности предлагается использовать механизм нейронных сетей на основе интеллектуальных агентов.

Особенности и основные принципы построения адаптивных систем обеспечения защиты РКС

Обеспечение защиты распределенной компьютерной системы реализуется двумя путями. Первый путь заключается в попытке построения т. н. абсолютно защищенной системы, в которой постоянно усложняются процессы аутентификации, механизмов разграничения прав доступа и т. д. Однако данный путь имеет ряд недостатков. Во-первых, возникает проблема защиты от локальных легальных субъектов; во-вторых, сами протоколы аутентификации имеют уязвимости, а пароли могут быть похищены либо подобраны. Второй путь состоит в усложнении механизмов выявления аномалий в действиях субъектов по отношению к ресурсам и другим субъектам.

Среди существующих методов защиты распределенных компьютерных систем, предложенных в других работах, выделяются следующие: деревья принятия решения, Байесовская сеть, скрытая Марковская сеть, нечеткая логика, метод опорных векторов. Все данные методы подразумевают выполнение анализа исходных данных и реализуют алгоритм реагирования на угрозы, при этом наилучшие результаты по распознаванию новых и модифицированных угроз показывают системы на основе нейронных сетей, т. е. такие, в которых используется интеллектуальный подход к выявлению угроз в РКС [2, 3].

Одним из базовых компонентов таких систем является прототип биологического нейрона. На основе полученных входных данных и предыдущего опыта, полученного в виде заданных весов связей между входным и промежуточным слоями, генерируется результат, являющийся следствием активации нейрона [4, 5].

На рис. 1 изображен формальный нейрон как элемент нейронной сети. Объединение данной структуры в единую систему позволяет проанализировать входные параметры и с некоторой вероятностью ответить на вопрос, к какому классу или типу относится входная информация [6].

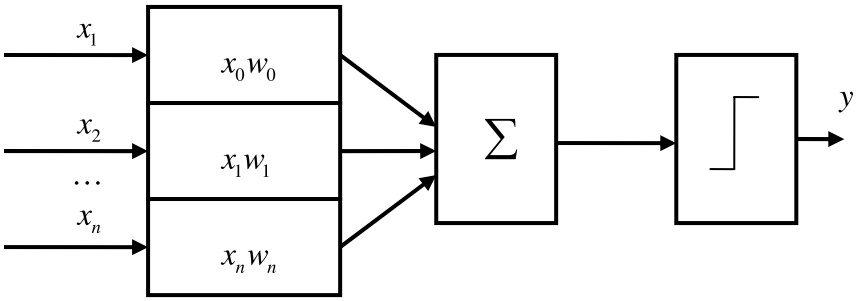


Рис. 1. Нейрон как элемент нейронной сети

Механизм обнаружения вторжений на основе модифицированной нейронной сети Кохонена

В целом нейронные сети характеризуются возможностью обучения. В отличие от обычных механизмов, они не имеют стандартного алгоритма выполнения и могут обучаться путем изменения коэффициентов связей между нейронами. В процессе обучения нейронная сеть способна выявлять сложные зависимости между входными и выходными данными и выполнять обобщение полученного образа [7]. По методу обучения нейронные сети делятся на требующие предварительного обучения и самообучающиеся сети. К числу последних относятся сети Кохонена, характеризующиеся структурно распределенной памятью, что позволяет избежать отказа сети в целом в случае отказа одного из нейронов. Данный эффект достигается за счет того, что классификацию входных данных выполняет не один нейрон, а целый кластер нейронов. Каждый входной вектор сигналов $X = \{x_1, x_2, ..., x_n\}$ поступает на вход каждого нейрона двумерной матрицы нейронов, а множество весов связей представляется в виде матрицы [8]:

$$W = \begin{pmatrix} w^{11} & w^{12} & w^{1j} \\ w^{21} & w^{22} & w^{2j} \\ w^{i1} & w^{i2} & w^{ij} \end{pmatrix}. \qquad (1)$$

Элементами данной матрицы являются векторы весовых коэффициентов связей $w^{ij} = \{w_1^{ij}, w_2^{ij}, ..., w_n^{ij}\}$. В начале обучения весовые коэффициенты нейронной сети задаются случайным образом из некоторого диапазона, далее вычисляется расстояние между входным вектором сигналов и множеством нейронов сети:

$$d_{ij} = \sum_{p=1}^{n} (x(t) - w_p^{ij}(t))^2, \qquad (2)$$

где $x(t)$ – входной вектор данных в момент времени t; $w_p^{ij}(t)$ – вектор весовых коэффициентов связей в момент времени t.

На третьем этапе выполняется поиск нейрона с координатами (i, j), для которого это расстояние является наименьшим. Далее выполняется изменение весов связей для обучения сети:

$$w^{ij}(t+1) = w^{ij}(t) + k(t)(x(t) - w^{ij}(t)), \qquad (3)$$

где $k(t)$ – коэффициент обучения (или скорость обучения), который уменьшается со временем.

Таким образом, сеть Кохонена обучается методом последовательных приближений. В процессе обучения на ее входы подаются данные, но при этом сеть подстраивается не под эталонное значение выхода, а под определенные закономерности во входных данных. Обучение начинается с выбранного случайным образом исходного расположения центров.

В процессе последовательной подачи на вход нейронной сети обучающих наборов определяется наиболее похожий нейрон, т. е. тот, у которого скалярное произведение весов и поданного на вход вектора является минимальным. Этот нейрон объявляется победителем и становится центром при подстройке весов соседних нейронов. Такое правило обучения предполагает «соревновательное» обучение с учетом расстояния нейронов от «нейрона-победителя». Суть

обучение при этом состоит не в минимизации ошибок распознавания, а в подстройке весов – внутренних параметров нейронной сети – для наиболее полного совпадения с входными данными.

Основой сети является скрытый слой Кохонена. Однако для улучшения результата анализа состояния системы и обнаружения вторжений предложена модификация нейронной сети Кохонена. При этом, как показано на рис. 2, скрытый слой нейронной сети Кохонена предложено разделить на две части или набора. Первый набор нейронов [1...f] отвечает за определение типа и класса атак, при этом изменение входных весов на выходном слое вызывает активацию линейной функции Y_1, значение нуль соответствует разрешенному состоянию системы, а единица – соответственно, атаке. Второй набор нейронов [n...m] анализирует нормальное состояние системы Y_2, что позволяет дополнить и уточнить результат выходного слоя класса атаки Y_1.

Модель данного механизма изображена на рис. 2.

Основой данной сети является скрытый слой Кохонена. Параметрами, которые используются в качестве входных данных для распознавания системных атак, являются:

сетевые записи и события;

данные о времени входа субъектов в систему и выхода из нее;

количество процессов;

индикаторы доступа к файлам;

временные интервалы доступа к ресурсам;

запросы к ресурсам и объектам компьютерной системы.

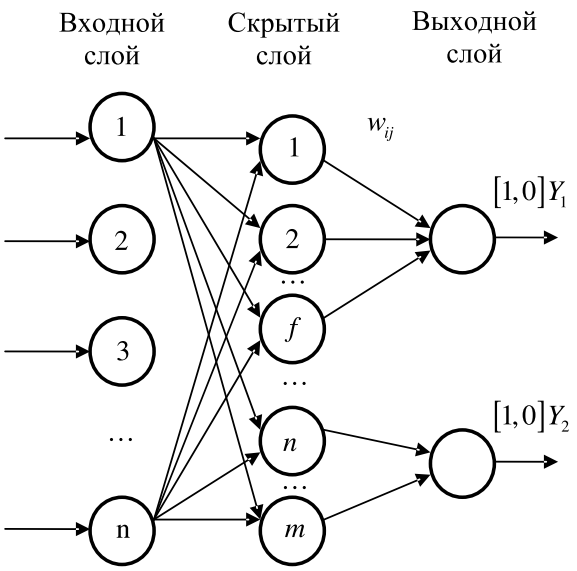


Рис. 2. Нейронная сеть для определения атаки

Далее интеллектуальный агент комплексной адаптивной системы защиты компьютерных систем использует механизм модифицированной нейронной сети для обнаружения и классификации потенциальных угроз безопасности. Входные параметры системы попадают на сенсор агента, который непрерывно считывает данные из соответствующей среды и передает их на входы нейронной сети. Кроме данных системы, интеллектуальный агент в качестве входной информации также может подавать данные о самом субъекте, его активностях и др.

После получения результата анализа нейронной сети по типу и классу атаки или по констатации нормального состояния системы данная информация передается через связи агента другим агентам, а также главному агенту администратора безопасности системы. Таким образом, информация о потенциальных угрозах анализируется всей структурой механизма обеспечения безопасности, что в целом повышает точность анализа возможности реализации угроз.

Поскольку мы используем нейронную сеть Кохонена в качестве скрытого слоя, все входные параметры подаются на матрицу нейронов, формируя полносвязную нейронную сеть. Важно отметить, что данный тип нейронной сети характеризуется значительным числом связей, следовательно, целесообразно выполнить некоторое упрощение и вербализацию нейронной сети за счет того, что элементы (входные параметры, нейроны, оказывающие минимальное влияние на корректность распознавания состояний) могут быть исключены из сети без существенного снижения качества распознавания [9].

Для выявления уязвимостей данный механизм может использовать настройки компьютерной сети, число ее пользователей, права пользователей, параметры их до-

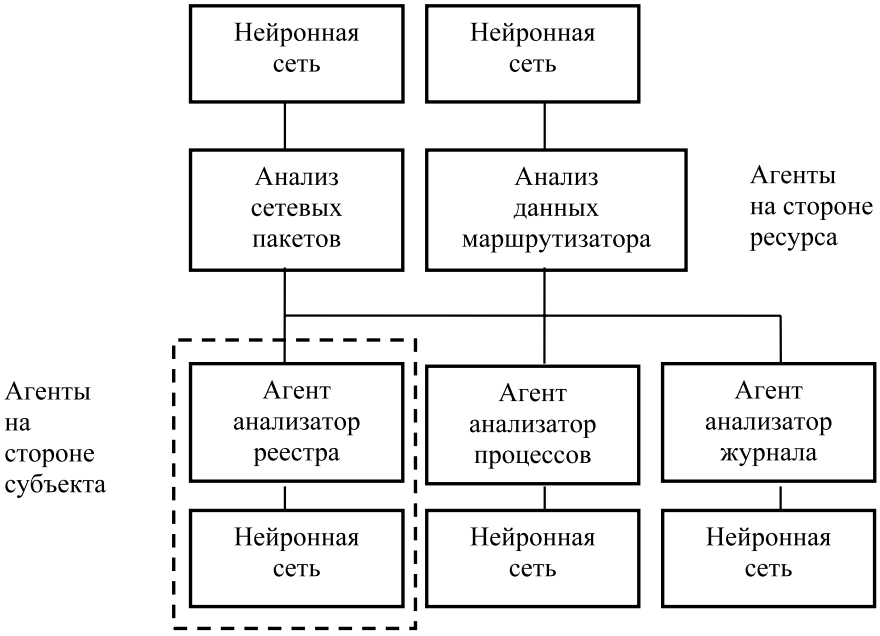


Рис. 3. Общая архитектура системы выявления атаки

ступа, число и типы портов, сетевые службы и настройки администратора.

Для выявления атакующих действий анализ и сбор данных требуется выполнять на нескольких уровнях распределенной компьютерной системы. Таким образом, использованы следующие исходные данные: данные о сетевых пакетах; данные журнала маршрутизатора; данные журнала безопасности операционной системы; данные реестра операционной системы и данные о процессах операционной системы.

Далее, на каждом уровне сбора информации прикрепляется выделенный интеллектуальный агент, как показано на рис. 3.

На рисунке изображена общая архитектура системы выявления атаки на основе механизма нейронной сети и интеллектуальных агентов. Каждый из прикрепленных интеллектуальных агентов включает свою собственную нейронную сеть, которая в рамках своего анализа выполняет классификацию угроз и атак. Соответственно, для повышения корректности распознавания

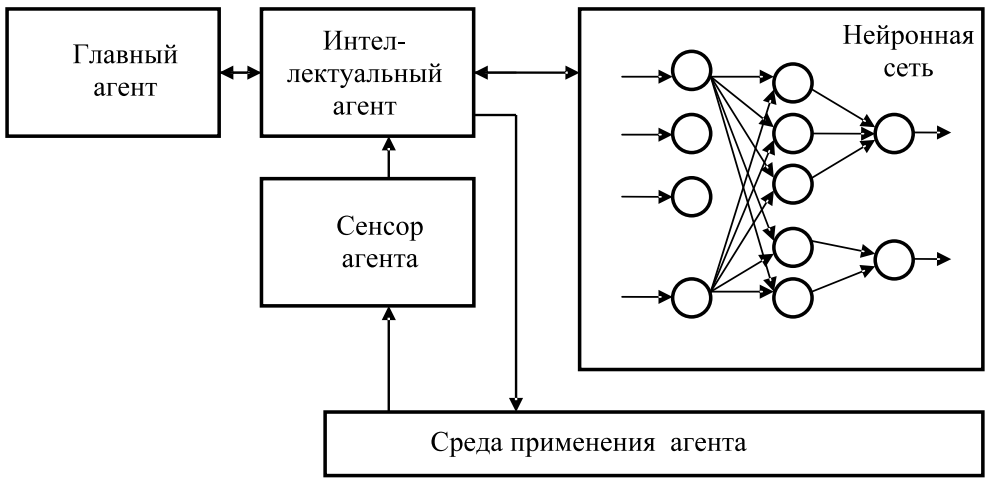


Рис. 4. Архитектура комплексной адаптивной системы защиты с интеллектуальным агентом на основе модифицированной нейронной сети

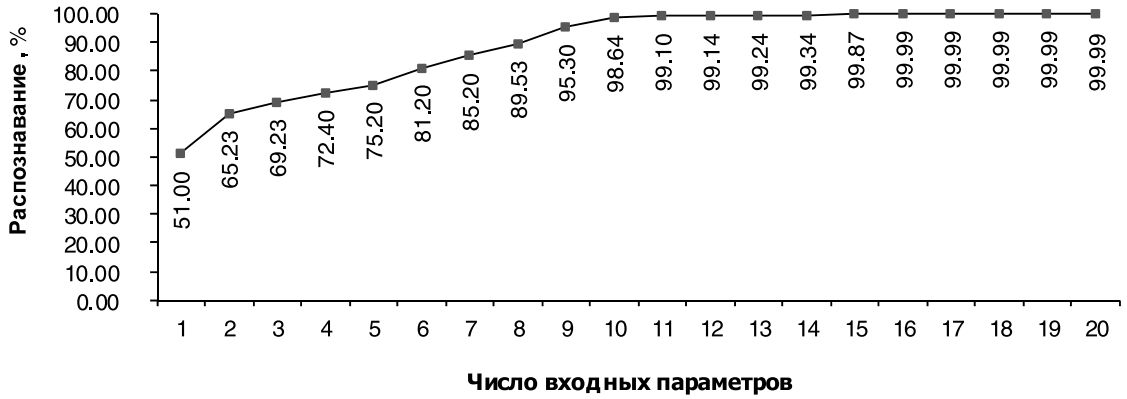


Рис. 5. Анализ результатов распознавания состояния РКС в зависимости от входных параметров

атак предлагается использование взаимодействующих между собой и с нейронной сетью интеллектуальных агентов. Схема взаимодействия интеллектуальных агентов и нейронной сети показана на рис. 4.

Экспериментальные исследования параметров адаптивной системы защиты

Проведены экспериментальные исследования корректности распознавания ситуации типа атака как числа выявленных вторжений со стороны предложенной адаптивной системы защиты на основе нейронной сети с интеллектуальным агентом. Для этого на вход нейронной сети, моделируемой с использованием специализированной среды, подавались различные наборы входных параметров, отражающих факт нарушения безопасности РКС. Число параметров в наборе изменялось от 1 до 20.

В ходе экспериментальных исследований установлено, что, в частности, при анализе выбранных DDoS-атак некоторые из параметров не имеют существенной роли в процессе распознавания. Как показывает рис. 5, при наличии более 11 входных параметров нейронной сети результаты распознавания незначительно отличаются друг от друга и, следовательно, их общее количество может быть снижено до этого числа.

Однако при появлении новых входных параметров для обучения при использовании данного метода могут измениться функции нейронной сети, и она может лишиться свойства обобщения. Таким образом, данный алгоритм снижения входных параметров целесообразно использовать именно в статических задачах, а не в задачах с динамически изменяющимися условиями.

В ситуации, когда число атак и их сложность постоянно увеличивается, организация безопасности распределенных компьютерных систем требует специального интеллектуального подхода, при котором система защиты способна обучаться и принимать решения по обеспечению безопасности автономно, при этом выявляя угрозы, классифицируя их по определенным признакам.

Разработан специальный программный комплекс на основе механизма нейронных сетей с использованием интеллектуальных агентов, позволяющий выявлять и классифицировать атаки на РКС. Предложенный механизм основан на модифицированных нейронных сетях Кохонена и характеризуется повышенной корректностью обнаружения атак, в т. ч. в условиях, когда отказал один из нейронов нейронной сети, ввиду того что классификацию атаки выполняет несколько нейронов – кластер нейронов. Кроме того, способность предложенного механизма анализировать состояние РКС без наличия полного набора параметров системы, как показали экспериментальные исследования, позволяет избежать некорректного результата в случае отсутствия ряда параметров или некорректного изменения их значений.

СПИСОК ЛИТЕРАТУРЫ

1. [Электронный ресурс] / URL: http://www.securelist.com/ru/analysis/208050810/DDoS_ataki_pervogo_polugodiya_2013_goda (дата обращения 2013)

2. **Андон Ф.И., Игнатенко А.П.** Атаки на отказ в сети Интернет: описание проблемы и подходов к ее решению. Киев, Препринт. НАН Украина. Ин-т программных систем, 2008. 50 с.

3. **Уланов А.В., Котенко И.В.** Защита от DDoS-атак: механизмы предупреждения, обнаружения, отслеживания источника и противодействия // Защита информации INSIDE. 2007. № 1–3.

4. **Pushnoi G.S., Bonser G.L.** Method of Systems Potential as «Top-Bottom» Technique of the Complex Adaptive Systems Modelling // Intelligent Complex Adaptive Systems. Hershey-London, IGI-Publishing, 2008. Pp. 26–73.

5. Artificial Neural Networks for Misuse Detection. StudyMode.com [электронный ресурс] / URL: http://www.studymode.com/essays/Artificial-Neural-Networks-For-Misuse-Detection-63716.html (дата обращения Aug. 2005)

6. **Нестеренко Б.Б., Новотарский М.А.** Дискретные клеточные нейронные сети с обобщенным нейроном // Искусственный интеллект. 2007. № 4.

7. **Хайкин С.** Нейронные сети. 2-е изд. М.: ИД «Вильямс», 2008. 1103 с.

8. **Kohonen T.** Self-Organized Formation of Topologically Correct Feature Maps // Biological Cybernetics. 1982. No. 43(1). Pp. 59–69.

9. **Pham D.T., Ghanbarzadeh A., Koc E., Otri S., Rahim S., Zaidi M.** The Bees Algorithm – A Novel Tool for Complex Optimisation Problems Cardiff: Cardiff University, 2006. Pp. 454–459.

REFERENCES

1. Available: http://www.securelist.com/ru/analysis/208050810/DDoS_ataki_pervogo_polugodiya_2013_goda (Accessed 2013)

2. **Andon F.I., Ignatenko A.P.** *Ataki na otkaz v seti Internet: opisanie problemy i podhodov k ee resheniyu*. Kiev, Preprint, NAN Ukrainy, Institute porgrammnykh system, 2008, 50 p. (rus)

3. **Ulanov A.V., Kotenko I.V.** Zaschita ot DDoS-atak: mechanismy preduprezdenia, obnaruhenia, otslezhivania istochnika i protivodeistvia. *Zaschita informacii INSIDE*, 2007, No. 1–3. (rus)

4. **Pushnoi G.S., Bonser G.L.** Method of Systems Potential as «Top-Bottom» Technique of the Complex Adaptive Systems Modelling, *Intelligent Complex Adaptive Systems*, IGI-Publishing, Hershey-London, 2008, Pp. 26–73.

5. *Artificial Neural Networks for Misuse Detection. StudyMode.com* Available: http://www.studymode.com/essays/Artificial-Neural-Networks-For-Misuse-Detection-63716.html (Accessed Aug. 2005)

6. **Nesterenko B.B., Novotarsky M.A.** Discretnye kletochnye neironnye seti s obobschenym neironom, *Iskusstvennyi intellekt*, 2007, No. 4. (rus)

7. **Haykin S.** *Neironnye seti*, 2-e izd, ID «Willams» Publ., 2008, 1103 p. (rus)

8. **Kohonen T.** Self-Organized Formation of Topologically Correct Feature Maps, *Biological Cybernetics*, 1982, No. 43(1), Pp. 59–69.

9. **Pham D.T., Ghanbarzadeh A., Koc E., Otri S., Rahim S., Zaidi M.** *The Bees Algorithm – A Novel Tool for Complex Optimisation Problems*, Cardiff: Cardiff University, 2006, Pp. 454–459.

МУХИН Вадим Евгеньевич – *доцент кафедры вычислительной техники Национального технического университета Украины «Киевский политехнический институт», кандидат технических наук.*
02098, Украина, Киев, пр. Победы, д. 37.
E-mail: v_mukhin@mail.ru

MUKHIN, Vadym Ye. *National Technical University of Ukraine «Kiev Polytechnic Institute».*
02098, Pobedy Ave. 37, Kiev, Ukraine.
E-mail: v_mukhin@mail.ru

КОРНАГА Ярослав Игоревич – *старший преподаватель кафедры вычислительной техники Национального технического университета Украины «Киевский политехнический институт».*
02098, Украина, Киев, пр. Победы, д. 37.
E-mail: slovyan_k@ukr.net

KORNAGA, Yaroslav I. *National Technical University of Ukraine «Kiev Polytechnic Institute».*
02098, Pobedy Ave. 37, Kiev, Ukraine.
E-mail: slovyan_k@ukr.net

СТЕШИН Виктор Васильевич – *аспирант кафедры вычислительной техники Национального технического университета Украины «Киевский политехнический институт».*
02098, Украина, Киев, пр. Победы, д. 37.
E-mail: mail@webmarker.com.ua

STESHYN, Viktor V. *National Technical University of Ukraine «Kiev Polytechnic Institute».*
02098, Pobedy Ave. 37, Kiev, Ukraine.
E-mail: mail@webmarker.com.ua

УДК 621.39(075.8)

В.А. Варгаузин

ОЦЕНКА ВЕРОЯТНОСТИ ОШИБКИ В КАНАЛЕ СВЯЗИ НА ОСНОВЕ РЕКУРРЕНТНЫХ СВОЙСТВ СВЕРТОЧНЫХ КОДОВ

V.A. Vargauzin

ESTIMATION OF ERROR PROBABILITY IN A CHANNEL BASED ON THE RECURRENT PROPERTIES OF CONVOLUTIONAL CODES

Рассмотрен метод оценки вероятности ошибки в канале связи, учитывающий рекуррентные свойства сверточных помехоустойчивых кодов. Метод может применяться при вычислении отношения правдоподобия кодовых символов на выходе демодулятора в системах связи, использующих как непосредственно сверточный код, так и сверточные турбокоды и сигнально-кодовые конструкции на основе решетчато-кодовой модуляции.

СВЕРТОЧНЫЕ КОДЫ; ВЕРОЯТНОСТЬ ОШИБКИ ПРИ ПЕРЕДАЧЕ ПО КАНАЛУ; ПРОВЕРОЧНАЯ МАТРИЦА; ОТНОШЕНИЕ СИГНАЛ/ШУМ; ТУРБОКОД.

The method of the estimation of the probability of errors in the channel, based on recurrence property of convolution codes. The method can be used in computation of the likelihood ratio (LLR) of the code symbols at the demodulator output in communication systems that use both directly convolution code and turbo codes and trellis codes modulations.

CONVOLUTION CODES; PROBABILITY OF ERRORS IN THE CHANNEL; PARITY CHECK MATRIX; SIGNAL TO NOISE RATIO; TURBO CODE.

Цель статьи – демонстрация простого метода оценки вероятности ошибки на выходе канала передачи системы связи для случая, когда в ней используется помехоустойчивый сверточный код. Сверточные коды (СК) находят многочисленное применение в современной цифровой радиосвязи [1]. При этом в большинстве приложений применяются декодеры, использующие «мягкие» решения демодулятора. Такие решения пропорциональны отношению функций правдоподобия значений (0 и 1) кодовых двоичных символов, и их вычисление предполагает знание величины отношения сигнал/шум на выходе канало-образующей аппаратуры, т. е. демодулятора. Эта величина также требуется и во многих сопутствующих декодированию алгоритмах в приемном устройстве обработки: синхронизации, адаптивного подавления помех, выравнивания частотной характеристики канала и т. п.

Оценка отношения сигнал/шум на выходе демодулятора, в свою очередь, может основываться на оценке $\hat{p}$ вероятности ошибки p приема кодовых символов (Symbol Error Rate – SER), т. е. вероятности ошибки при «жестких» (двухуровневых) решениях демодулятора. Для такой оценки на практике используются различные методы.

К универсальным методам можно отнести метод передачи известной кодовой последовательности (преамбулы) и метод

регенерации кодовой последовательности приемным устройством по декодированной информационной последовательности при условии известной информационной последовательности.

Используются и менее универсальные методы, учитывающие специфику СК. Например, такой учет возможен при использовании т. н. «почти систематических» кодов или, как их еще называют, *быстропросматриваемых СК* [2], у которых векторы коэффициентов порождающих полиномов отличаются лишь в одном символе.

В этой связи следует обратить внимание на то, что независимо от специфики СК (*несистематический* или *систематический* код) и/или от структуры его формирования (*нерекурсивной* или *рекурсивной*) [1], существует возможность оценки SER по частоте нарушения *рекуррентного* уравнения (или уравнений) проверок на четность, связывающего кодовые символы СК. Это, в свою очередь, эквивалентно оценке частоты наблюдения единиц на выходе логической схемы формирователя синдрома (ФС).

В качестве простого примера рассмотрим несистематический СК с длиной кодового ограничения $K' = 3$ и скоростью $R = 1/2$, формируемый нерекурсивным кодером с порождающими полиномами $g_1(z) = 1 + z^{-1} + z^{-2}$, $g_2(z) = 1 + z^{-2}$, или, что эквивалентно, *порождающей матрицей* вида $\mathbf{G}(z) = [g_1(z) \quad g_2(z)]$, элементами которой являются полиномы. Такой нерекурсивный несистематический кодер (ННК) в ответ на каждый поступивший в момент времени t_i информационный символ $u_1^{(i)} = u^{(i)}$ формирует группу из двух символов $y_1^{(i)}$ и $y_2^{(i)}$ по правилу $y_1^{(i)} = u^{(i)} + u^{(i-1)} + u^{(i-2)}$, $y_2^{(i)} = u^{(i)} + u^{(i-2)}$.

Нетрудно видеть, что при этом кодовые символы удовлетворяют следующему *рекуррентному* уравнению:

$$(y_1^{(i-2)} + y_1^{(i)}) + (y_2^{(i-2)} + y_2^{(i-1)} + y_2^{(i)}) = 0, \quad (1)$$

включающему символы в пределах длины кодового ограничения. Вид такого уравнения можно описать матрицей

$$\mathbf{H}(z) = [h_1(z) \quad h_2(z)] = [g_2(z) \quad g_1(z)],$$

которую называют *проверочной матрицей СК*. Как и для линейных блоковых кодов, порождающая и проверочная матрицы СК связаны соотношением *ортогональности*. Для рассматриваемого кода оно имеет вид:

$$\mathbf{G}(z)\mathbf{H}^{\mathrm{T}}(z) = [g_1(z) \quad g_2(z)] \cdot \begin{bmatrix} g_2(z) \\ g_1(z) \end{bmatrix} = \\ = g_1(z)g_2(z) + g_1(z)g_2(z) = 0.$$

Здесь все операции над коэффициентами полиномов выполняются по правилу вычислений в арифметике по модулю два, то есть $z^{-l} + z^{-l} = 0$.

Рекуррентному уравнению удовлетворяет и систематический СК, формируемый рекурсивным кодером, для которого $y_1^{(i)} = u^{(i)}$, а символ $y_2^{(i)}$ удовлетворяет рекуррентному уравнению второго порядка

$$(u^{(i-2)} + u^{(i)}) + (y_2^{(i-2)} + y_2^{(i-1)} + y_2^{(i)}) = 0$$

или, что эквивалентно, порождающей матрицей вида

$$\mathbf{G}(z) = \begin{bmatrix} 1 & \dfrac{g_2(z)}{g_1(z)} \end{bmatrix}.$$

Это непосредственно следует из уравнения ортогональности:

$$\mathbf{G}(z)\mathbf{H}^{\mathrm{T}}(z) = \begin{bmatrix} 1 & \dfrac{g_2(z)}{g_1(z)} \end{bmatrix} \cdot \begin{bmatrix} g_2(z) \\ g_1(z) \end{bmatrix} = \\ = g_2(z) + g_2(z) = 0.$$

В результате для СК, формируемого как ННК, так и РСК, логическая схема ФС может быть построена на основании одного проверочного рекуррентного уравнения (1), в котором переданные кодовые символы $y_1^{(i)}$ и $y_2^{(i)}$ следует заменить принятыми символами $y_1^{(i)\prime}$ и $y_2^{(i)\prime}$, т. е. «жесткими» решениями демодулятора. При этом оценка вероятности ошибки в канале сводится к оценке частоты нарушения такой проверки.

Безусловно, такая оценка не лишена недостатка. Действительно, при наличии нескольких ошибочных «жестких» решений в пределах длины кодового ограничения СК, нарушение проверки может быть не обнаружено.

В то же время в реальных условиях этот факт может оказать незначительное влияние на достоверность оценки $\widehat{p}$ величины p. Действительно, рассматриваемый СК име-

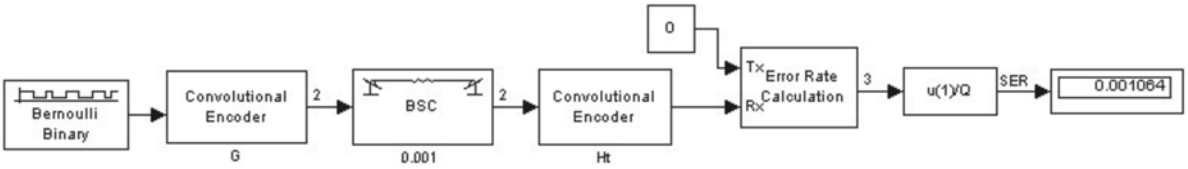


Рис. 1. Модель оценки вероятности ошибки для СК со скоростью $R = ½$

ет достаточно малую величину длины кодового ограничения $K' = 3$. Учтем, что в реальных каналах с независимыми ошибками, как правило, значение $p < 10^{-2}$. В таких условиях величина вероятности необнаруженного нарушения проверки при наличии двух и более ошибок в «жестких» решениях демодулятора в пределах длины кодового ограничения, очевидно, оказывается значительно меньше величины p. Поэтому при учете в оценке $\widehat{p}$ лишь одиночных ошибок в символах $y_1^{(i)\prime}$ и $y_2^{(i)\prime}$, тем не менее следует ожидать выполнения $\widehat{p} \approx p$.

В указанных условиях, как нетрудно видеть, оценка $\widehat{p}$ может быть получена по формуле

$$\widehat{p} = \frac{L}{NQ}, \qquad (2)$$

где N – число испытаний в статистическом эксперименте при получении $\widehat{p}$; L – число зафиксированных нарушений в указанной выше проверке в процессе статистического эксперимента; $Q = q_1 + q_2$; $q_1 = 2$ – число ненулевых коэффициентов полинома $h_1(z)$; $q_2 = 3$ – число ненулевых коэффициентов полинома $h_2(z)$. Последние две величины учитывают тот факт, что каждая одиночная ошибка в символах $y_1^{(i)\prime}$ и $y_2^{(i)\prime}$ проявляет себя q_1 и q_2 раз соответственно на выходе схемы ФС.

Далее нетрудно оценить и значение величины отношения сигнал/шум, для чего следует учесть ее связь с величиной p. Например, для канала с аддитивным белым гауссовым шумом и модуляции ФМ-2 имеем $p = Q(\sqrt{2h})$, где $Q(\cdot)$ – так называемая «ку-функция» [1]; $2h^2$ – величина отношения сигнал/шум на выходе демодулятора. Поэтому оценкой отношения сигнал/шум, очевидно, является величина $[Q^{-1}(\widehat{p})]^2$.

Рассмотренный выше подход, в принципе, относится ко всем широко используемым СК со скоростью $R = 1/2$ с небольшими длинами кодового ограничения $K' \le 9$, а также к турбокодам, для которых, как правило, $K' \le 5$ [3, 4].

На рис. 1 представлена модель, созданная в Simulink, иллюстрирующая метод оценки вероятности ошибки для широко распространенного на практике СК с длиной $K' = 7$.

Модель инициализируются следующими строками на языке MATLAB:

```
G=poly2trellis(7, [171 133]);
Ht=poly2trellis([7 7], [133; 171]);
Q=5+5;
```

Видно, что ФС представляет собой кодер с порождающей матрицей Ht. На рис. 2 представлена зависимость оценки $\widehat{p}$ вероятности ошибки в канале от вероятности ошибки p. Как и следовало ожидать, $\widehat{p} \approx p$ в области $p < 10^{-2}$.

Метод можно обобщить и на случай СК с произвольной скоростью. Для примера рассмотрим СК со скоростью $R = 2/3$ и порождающей матрицей

$$\mathbf{G}(z) = \begin{bmatrix} 1 & z^{-1} & 0 \\ z^{-2} & 1 & z^{-1} \end{bmatrix}.$$

Это порождающая матрица несистематического СК, решетчатая диаграмма которого имеет $2^{\nu} = 8$ состояний, где ν – число ячеек памяти при канонической реализации кодера. Кодовые символы этого СК удовлетворяют проверочному рекуррентному уравнению порядка $\nu = 3$

$$y_1^{(i-2)} + y_2^{(i-1)} + (y_3^{(i-3)} + y_3^{(i)}) = 0,$$

что эквивалентно следующей проверочной матрице вида

$$\mathbf{H}(z) = [h_1(z) \quad h_2(z) \quad h_3(z)] = \\ = [z^{-2} \quad z^{-1} \quad z^{-3} + 1].$$

Это следует и из непосредственной проверки свойства ортогональности порожда-

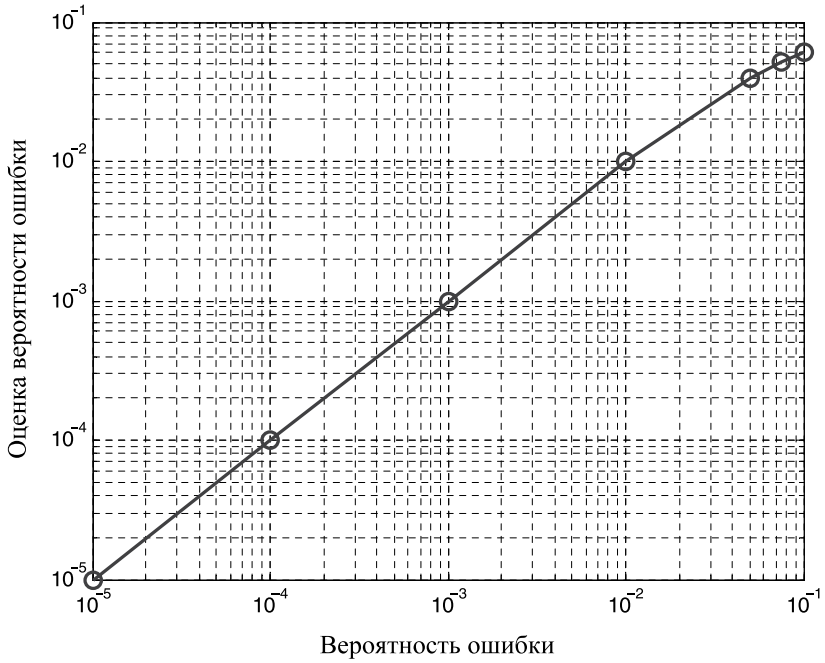


Рис. 2. Зависимость оценки вероятности ошибки в канале от вероятности ошибки

ющей и проверочных матриц:

$$\mathbf{G}(z)\mathbf{H}^{\mathrm{T}}(z) = \begin{bmatrix} 1 & z^{-1} & 0 \\ z^{-2} & 1 & z^{-1} \end{bmatrix} \cdot \begin{bmatrix} z^{-2} \\ z^{-1} \\ z^{-3}+1 \end{bmatrix} =$$

$$= (z^{-2} + z^{-2} + 0) + (z^{-4} + z^{-1} + z^{-4} + z^{-1}) = 0.$$

При кодировании таким СК в систематическом виде, когда $y_1^{(i)} = u_1^{(i)}$ и $y_2^{(i)} = u_2^{(i)}$, получаем, что символ $y_3^{(i)}$ должен удовлетворять рекуррентному уравнению $u_1^{(i-2)} + u_2^{(i-1)} + (y_3^{(i-3)} + y_3^{(i)}) = 0$, вид которого можно описать матрицей

$$\mathbf{P}(z) = \begin{bmatrix} \dfrac{z^{-2}}{z^{-3}+1} \\ \dfrac{z^{-1}}{z^{-3}+1} \end{bmatrix},$$

где коэффициенты полинома $(z^{-3}+1)$ определяют обратную связь в схеме РСК. При этом порождающая матрица при таком систематическом кодировании может быть представлена в виде

$$\mathbf{G}(z) = [\mathbf{I}_2 \quad \mathbf{P}(z)] = \left[\begin{array}{cc|c} 1 & 0 & \dfrac{z^{-2}}{z^{-3}+1} \\ 0 & 1 & \dfrac{z^{-1}}{z^{-3}+1} \end{array}\right],$$

где пунктирной вертикальной линией для наглядности единичная подматрица $\mathbf{I}_2$ отделена от подматрицы $\mathbf{P}(z)$. Непосредственной проверкой снова убеждаемся в справедливости свойства ортогональности порождающей и проверочных матриц:

$$\mathbf{G}(z)\mathbf{H}^{\mathrm{T}}(z) = \left[\begin{array}{cc|c} 1 & 0 & \dfrac{z^{-2}}{z^{-3}+1} \\ 0 & 1 & \dfrac{z^{-1}}{z^{-3}+1} \end{array}\right] \cdot \begin{bmatrix} z^{-2} \\ z^{-1} \\ z^{-3}+1 \end{bmatrix} =$$

$$= (z^{-2} + 0 + z^{-2}) + (0 + z^{-1} + z^{-1}) = 0.$$

Рассмотренные последние две порождающие матрицы СК для несистематического и систематического кодирования используются при решетчато-кодовой модуляции (РКМ) сигналами с квадратурной амплитудной модуляцией (КАМ) [5]. На рис. 3 представлена модель, в которой используется такой несистематический СК.

Модель инициализируется следующими строками на языке MATLAB:

```
G=poly2trellis([2 3], [2 1 0; 1 4 2]);
Ht=poly2trellis([1 1 4], [1; 1; 11]);
Q=1+1+2;
```

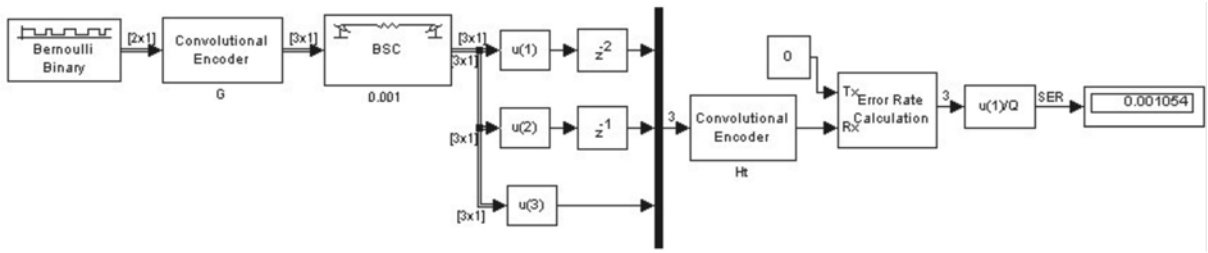


Рис. 3. Модель оценки вероятности ошибки для СК со скоростью R =2/3

В общем случае СК со скоростью $R = k/n$ можно определить как *рекуррентную последовательность*, удовлетворяющую системе проверок, которая описывается проверочной матрицей $\mathbf{H}(z)$ размером $(n-k)\times n$. При этом матрица $\mathbf{G}(z)$, порождающая эту последовательность, удовлетворяет соотношению ортогональности вида

$$\mathbf{G}(z)\mathbf{H}^{\mathrm{T}}(z) = \begin{bmatrix} 0 & 0 & \dots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \underbrace{0 & 0 & \cdots & 0}_{n-k} \end{bmatrix},$$

где справа – «нулевая» квадратная матрица размера $(n - k)$. При систематическом кодировании таким кодом правила вычисления $(n - k)$ избыточных символов, очевидно, описываются матрицей $\mathbf{P}(z)$ размера $k\times(n-k)$.

Проиллюстрируем это для случая $k > 1$. Определим СК проверочной матрицей вида

$$\mathbf{H}(z) = [h_1(z) \quad h_2(z) \quad h_3(z)] =$$

$$= \begin{bmatrix} \underbrace{\underbrace{0z^{-4}}_{1} + \underbrace{1z^{-3}}_{6} + \underbrace{1z^{-2} + 1z^{-1} + 0}_{16}} & \underbrace{\underbrace{0z^{-4}}_{0} + \underbrace{0z^{-3}}_{4} + \underbrace{1z^{-2} + 1z^{-1} + 0}_{4}} & \underbrace{\underbrace{1z^{-4}}_{2} + \underbrace{0z^{-3}}_{1} + \underbrace{0z^{-2} + 1z^{-1} + 1}_{23}} \end{bmatrix}.$$

Эта матрица определяет рекуррентное уравнение СК с $2^{\nu} = 16$ состояниями, используемого в РКМ сигналами с КАМ [5]. Здесь также представлена восьмеричная запись полиномов проверочной матрицы, часто применяемая в литературе по сверточному кодированию. Обратим внимание, что в такой записи полиномов, в отличие от записи полиномов порождающей матрицы, старший разряд соответствует минимальной степени полинома, равной –ν. При этом

$$\mathbf{P}(z) = \begin{bmatrix} \dfrac{z^{-3} + z^{-2} + z^{-1}}{z^{-4} + 1} \\ \dfrac{z^{-2} + z^{-1}}{z^{-4} + 1} \\ \dfrac{z^{-4} + z^{-1} + 1}{z^{-4} + 1} \end{bmatrix}$$

и порождающая матрица при систематическом кодировании есть

$$\mathbf{G}(z) = [\mathbf{I}_3 \quad \mathbf{P}(z)] =$$

$$= \left[\begin{array}{ccc|c} 1 & 0 & 0 & \dfrac{z^{-3} + z^{-2} + z^{-1}}{z^{-4} + 1} \\ 0 & 1 & 0 & \dfrac{z^{-2} + z^{-1}}{z^{-4} + 1} \\ 0 & 0 & 1 & \dfrac{z^{-4} + z^{-1} + 1}{z^{-4} + 1} \end{array}\right].$$

Итак, в статье продемонстрирован метод оценки вероятности ошибки двоичного символа на выходе канала. Для реализации метода в приемном устройстве требуется использовать простой в реализации формирователь синдрома сверточного кода. Метод может использоваться при вычислении отношения правдоподобия значений кодовых символов для декодеров с «мягкими» решениями демодулятора.

СПИСОК ЛИТЕРАТУРЫ

1. **Варгаузин В.А., Цикин И.А.** Методы повышения энергетической и спектральной эффективности в цифровой радиосвязи. СПб.: БХВ-Петербург, 2013. 352 с.

2. **Johannesson R., Zigangirov K.** Fundamentals of Convolutional Coding. IEEE Press, 1999. 426 p.

3. **Berrou C., Glavieux A., Thitimajshima P.** Near Shannon limit error-correcting coding: Turbo codes // In Proc. IEEE Internat. Conf. Commun. Geneva, Switzerland, 1993. Pp. 1064–1070.

4. **Viterbi A.J**. An Intuitive Justification and a Simplified Implementation of the MAP Decoder for Convolutional Codes // IEEE Journal on Selected Areas in Communications. 1998. Vol. 16. No. 2. Pp. 260–264.

5. **Ungerboeck G**. Channel coding with multilevel/phase signals // IEEE Transactions on Information Theory. 1982. Vol. IT-28. No. 1. Pp. 55–67.

REFERENCES

1. **Vargauzin V.A., Tsikin I.A.** *Metody povyisheniya energeticheskoy i spektralnoy effektivnosti v tsifrovoy radiosvyazi*, St. Petersburg: BHV-Petersburg Publ., 2013, 352 p. (rus)

2. **Johannesson R., Zigangirov K.** *Fundamentals of Convolutional Coding*, IEEE Press, 1999, 426 p.

3. **Berrou C., Glavieux A., Thitimajshima P.** Near Shannon limit error-correcting coding: Turbo codes. *In Proc. IEEE Internat. Conf. Commun.*, Geneva, Switzerland, 1993, Pp. 1064–1070.

4. **Viterbi A.J**. An Intuitive Justification and a Simplified Implementation of the MAP Decoder for Convolutional Codes. *IEEE Journal on Selected Areas in Communications*. 1998, Vol. 16, No. 2, Pp. 260–264.

5. **Ungerboeck G**. Channel coding with multilevel/phase signals. *IEEE Transactions on Information Theory*, 1982, Vol. IT-28, No. 1, Pp. 55–67.

ВАРГАУЗИН Виктор Анатольевич – *доцент кафедры радиотехники и телекоммуникаций Института физики, нанотехнологий и телекоммуникаций Санкт-Петербургского государственного политехнического университета.*

195251, Россия, Санкт-Петербург, ул. Политехническая, д. 29.

E-mail: var@mail.spbstu.ru

VARGAUZIN, Victor A. *St. Petersburg State Polytechnical University.*

195251, Politekhnicheskaya Str. 29, St. Petersburg, Russia.

E-mail: var@mail.spbstu.ru

УДК 621.391.8

С.Б. Макаров, С.В. Завьялов

ПОВЫШЕНИЕ ПОМЕХОУСТОЙЧИВОСТИ КОГЕРЕНТНОГО ПРИЕМА НЕОРТОГОНАЛЬНЫХ МНОГОЧАСТОТНЫХ СИГНАЛОВ

S.B. Makarov, S.V. Zavjalov

IMPROVING BER PERFORMANCE FOR COHERENT DETECTION OF NONORTHOGONAL MULTIFREQUENCY SIGNALS

Проведено исследование помехоустойчивости когерентного поэлементного приема неортогональных многочастотных сигналов с ФМ-4. Предложен алгоритм поэлементного когерентного приема с компенсацией межканальной интерференции (и его многошаговая модификация) и по диаграмме состояний.

НЕОРТОГОНАЛЬНЫЕ МНОГОЧАСТОТНЫЕ СИСТЕМЫ; КОГЕРЕНТНЫЙ АЛГОРИТМ ОБРАБОТКИ; КОМПЕНСАЦИЯ МЕЖКАНАЛЬНОЙ ИНТЕРФЕРЕНЦИИ; ДИАГРАММА СОСТОЯНИЙ; ПОМЕХОУСТОЙЧИВОСТЬ ПРИЕМА.

Research of the BER performance of multifrequency PSK-4 signals with nonorthogonal location of subcarriers has been conducted. Elementwise coherent detection algorithm with compensation for inter-channel interference (and multistep modification) and algorithm according to the state diagram are proposed.

NONORTHOGONAL MULTIFREQUENCY SYSTEMS; COHERENT DETECTION ALGORITHM; COMPENSATION FOR INTER-CHANNEL INTERFERENCE; STATE DIAGRAM; BER PERFORMANCE.

Применение многочастотных сигналов с неортогональным частотным разносом (часто используемые обозначения N-OFDM или Fast-OFDM [1–3]) позволяет повысить спектральную эффективность многочастотных систем передачи сообщений за счет уменьшения занимаемой полосы частот по сравнению с системой передачи с ортогональным частотным разносом. Повышение спектральной эффективности приводит к дополнительным энергетическим потерям в канале с постоянными параметрами, коэффициентом передачи μ и прямоугольной формой амплитудно-частотной характеристики [4]. Снижение энергетических потерь возможно путем применения алгоритмов когерентного поэлементного приема, учитывающих межканальную интерференцию.

Цель настоящей работы – исследование помехоустойчивости поэлементного когерентного приема неортогональных многочастотных сигналов длительностью T при использовании алгоритма обработки с компенсацией межканальной интерференции и алгоритма с последовательной процедурой вычислений, на примере 4-позиционных сигналов с фазовой манипуляцией (ФМ–4).

Алгоритмы поэлементного когерентного приема

В общем виде j-я реализация из N сдвинутых по частоте сигналов ФМ-4 с амплитудой A_0, произвольным частотным разносом между поднесущими Δf, средней несущей частотой ω_0 при условии независимого формирования квадратурных составляющих может быть записана на интервале $[-T/2;\ T/2]$ следующим образом:

$$s_j(t) = s(t; d_{i(-N/2)}, d_{q(-N/2)};\ \ldots;\ d_{i(N/2)}, d_{q(N/2)}) = \sum_{n=-(N-1)/2}^{(N-1)/2} s(t, d_{in}, d_{qn}), \quad (1)$$

где $j = 1, 2, 3, \ldots, 4^N$, а значения символов сообщения d_{in} зависят от индекса $i = 1, 2$

и индекса $n = -(N-1)/2, \ldots, (N-1)/2$; $N = 1, 3, 5, \ldots$. В частности, $d_{1n} = 1$; $d_{2n} = -1$. В (1) имеем:

$$s(t, d_{in}, d_{qn}) = s(t, d_{in}) + s(t, d_{qn}) = \\ = A_0[d_{in}\cos((\omega_0 + n\Delta\omega)t) + \\ + d_{qn}\sin((\omega_0 + n\Delta\omega)t)].$$

При неортогональном разносе поднесущих частот каждому значению канального символа d_{rn} на рассматриваемом интервале времени будет соответствовать одна из 4^N возможных форм сигнала $s_j(t)$ (1).

Рассмотрим поэлементный прием сигнала ФМ-4, расположенного на частоте ω_0. На форму этого сигнала оказывают влияние сигналы на поднесущих частотах $\omega_0 + n\Delta\omega$ ($n = \pm 1, \ldots, \pm(N-1)/2$). При наличии на входе приемного устройства смеси полезного сигнала и аддитивного гауссовского шума $n(t)$ со спектральной плотностью мощности $N_0/2$, имеем:

$$x(t) = A_0 \sum_{n=-(N-1)/2}^{(N-1)/2} [d_{in}\cos((\omega_0 + n\Delta\omega)t) + \\ + d_{qn}\sin((\omega_0 + n\Delta\omega)t)]n(t). \qquad (2)$$

Тогда функционал отношения правдоподобия может быть записан следующим образом:,

$$\Delta_{pr} = \frac{\exp\left\{\frac{2}{N_0}\int\limits_{-\frac{T}{2}}^{\frac{T}{2}} x(t)[y_l^{(-)}(t) + s_p(t) + y_l^{(+)}(t)]dt - \rightarrow \right.}{\exp\left\{\frac{2}{N_0}\int\limits_{-\frac{T}{2}}^{\frac{T}{2}} x(t)[y_l^{(-)}(t) + s_r(t) + y_l^{(+)}(t)]dt - \rightarrow \right.} \\ \frac{\left. \rightarrow -\frac{1}{N_0}\int\limits_{-\frac{T}{2}}^{\frac{T}{2}} [y_l^{(-)}(t) + s_p(t) + y_l^{(+)}(t)]^2 dt\right\}}{\left. \rightarrow -\frac{1}{N_0}\int\limits_{-\frac{T}{2}}^{\frac{T}{2}} \left[y_l^{(-)}(t) + s_r(t) + y_l^{(+)}(t)\right]^2 dt\right\}}, \qquad (3)$$

где $s_p(t) = A_0[d_{i0}\cos(\omega_0 t) + d_{q0}\sin(\omega_0 t)]$; $p = 1, 2, 3, 4$;

$$y_l^{(-)}(t) = A_0 \sum_{n=-(N-1)/2}^{-1} [d_{in}z_{in}(t)\cos(\omega_0 t) + \\ + d_{qn}z_{qn}(t)\sin(\omega_0 t)]; l = 1, 2, \ldots, 4^{N/2};$$

$$y_l^{(+)}(t) = A_0 \sum_{n=+1}^{(N-1)/2} [d_{in}z_{in}(t)\cos(\omega_0 t) + \\ + d_{qn}z_{qn}(t)\sin(\omega_0 t)]; l = 1, 2, \ldots, 4^{N/2};$$

$z_{in}(t)$ и $z_{qn}(t)$ — вещественные огибающие сигналов ФМ-4 $s(t, d_{in}, d_{qn})$ ($n \neq 0$) на поднесущей частоте ω_0.

Для конкретной l-й реализации сигналов, расположенных слева $y_l^{(-)}(t)$ и справа $y_l^{(+)}(t)$ по оси частот от анализируемого $s_p(t)$, усредняя числитель и знаменатель по всем комбинациям $y_l^{(-)}(t)$ и $y_l^{(+)}(t)$ в (3) получим:

$$\sum_{\substack{\{d_{in}, d_{qn}\} \\ n \neq 0}} \exp\left\{B_{pl}(t) - \frac{1}{N_0}E_{pl}\right\} > \\ > \sum_{\substack{\{d_{in}, d_{qn}\} \\ n \neq 0}} \exp\left\{B_{rl}(t) - \frac{1}{N_0}E_{rl}\right\}, \qquad (4)$$

где $y_l(t) = y_l^{(-)}(t) + y_l^{(+)}(t)$

$$y_l(t) = A_0 \sum_{\substack{n=-(N-1)/2 \\ n \neq 0}}^{(N-1)/2} [d_{in}z_{in}(t)\cos(\omega_0 t) + \\ + d_{qn}z_{qn}(t)\sin(\omega_0 t)],$$

$$B_{pl}(t) = \frac{2}{N_0}\int\limits_{-\frac{T}{2}}^{\frac{T}{2}} x(t)[s_p(t) + y_l(t)]dt;$$

$$E_{pl} = \int\limits_{-\frac{T}{2}}^{\frac{T}{2}} [s_p(t) + y_l(t)]^2 dt.$$

При условии что сигналы, расположенные на соседних поднесущих частотах, являются помехой, алгоритм поэлементного когерентного приема сигналов на поднесущей частоте ω_0 имеет следующий вид:

$$A_0 \int\limits_{-T/2}^{T/2} x(t)\ a(t)\cos(\omega_0 t)dt \underset{<}{\overset{d_{10}}{\gtrless}} 0; \\ A_0 \int\limits_{-T/2}^{T/2} x(t)\ a(t)\sin(\omega_0 t)dt \underset{<}{\overset{d_{10}}{\gtrless}} 0, \qquad (5)$$

где $x(t)$ определяется (2), а $d_j = -d_r$.

Рассмотрим алгоритм когерентного поэлементного приема с компенсацией межканальной интерференции. Положим, что

при приеме квадратурной составляющей $s(t, d_{i0})$ (или $s(t, d_{q0})$) сигнала $s(t, d_{i0}, d_{q0})$ в (1), все символы последовательности сигналов $y_l(t)$ приняты правильно. Тогда, используя (4), для $s(t, d_{i0})$ получим из (5):

$$A_0 \int_{-\frac{T}{2}}^{\frac{T}{2}} x(t)a(t)[d_{10}\cos(\omega_0 t)]dt - \frac{E^*_{1l}}{2} \underset{d_{20}}{\overset{d_{10}}{\gtrless}}$$

$$\underset{d_{20}}{\overset{d_{10}}{\gtrless}} A_0 \int_{-\frac{T}{2}}^{\frac{T}{2}} x(t)a(t)[d_{20}\cos(\omega_0 t)]dt - \frac{E^*_{2l}}{2},$$

или

$$\int_{-\frac{T}{2}}^{\frac{T}{2}} x(t)s_{i0}(t)dt - E^*_{ip} \underset{d_{20}}{\overset{d_{10}}{\gtrless}} 0, \qquad (6)$$

где

$$E^*_{il} = \int_{-\frac{T}{2}}^{\frac{T}{2}} (s(t, d_{i0}) + y^*_{il}(t))^2 dt, \; i = 1, 2;$$

$$y^*_{il}(t) = A_0 \sum_{\substack{n=-(N-1)/2 \\ n\neq 0}}^{(N-1)/2} d^*_{in} z_{in}(t)\cos(\omega_0 t);$$

$s_{i0}(t) = A_0 a(t)\cos(\omega_0 t)$; $\quad E^*_{ip} = (E^*_{1l} - E^*_{2l})/4$;
d^*_{in} – оценки принятых информационных символов при использовании алгоритма приема (5).

Аналогично можно получить алгоритм приема для другой квадратурной составляющей $s(t, d_{q0})$ сигнала $s(t, d_{i0}, d_{q0})$.

Рассмотрим многошаговую модификацию алгоритма когерентного поэлементного приема с компенсацией межканальной интерференции. На первом шаге используется алгоритм (5). На втором шаге применяется алгоритм (6). В качестве оценок принятых символов используются решения, полученные на первом шаге. На каждом последующем шаге будут использоваться оценки принятых символов, полученные на предыдущем шаге. На последнем шаге формируются решения о принятых информационных символах. Для удобства записи алгоритма обозначим оценку принятого символа на r-м шаге $d_{in}{}^*(r)$. Тогда имеем:

на первом шаге

$$\int_{-\frac{T}{2}}^{\frac{T}{2}} x(t)s_{i0}(t)dt \underset{d^*_{20}(1)}{\overset{d^*_{10}(1)}{\gtrless}} 0;$$

на r-м шаге

$$\int_{-\frac{T}{2}}^{\frac{T}{2}} x(t)s_{i0}(t)dt - E^*_{ip}(r-1) \underset{d^*_{20}(r)}{\overset{d^*_{10}(r)}{\gtrless}} 0, \qquad (7)$$

$$r = 2...L-1;$$

на последнем L-м шаге

$$\int_{-\frac{T}{2}}^{\frac{T}{2}} x(t)s_{i0}(t)dt - E^*_{ip}(L-1) \underset{d_{20}}{\overset{d_{10}}{\gtrless}} 0,$$

где

$$E^*_{il}(r-1) = \int_{-\frac{T}{2}}^{\frac{T}{2}} \left(s(t, d_{i0}) + \right.$$

$$\left. + A_0 \sum_{\substack{n=-(N-1)/2 \\ n\neq 0}}^{(N-1)/2} d^*_{in}(r-1) z_{in}(t)\cos(\omega_0 t) \right)^2 dt \qquad (8)$$

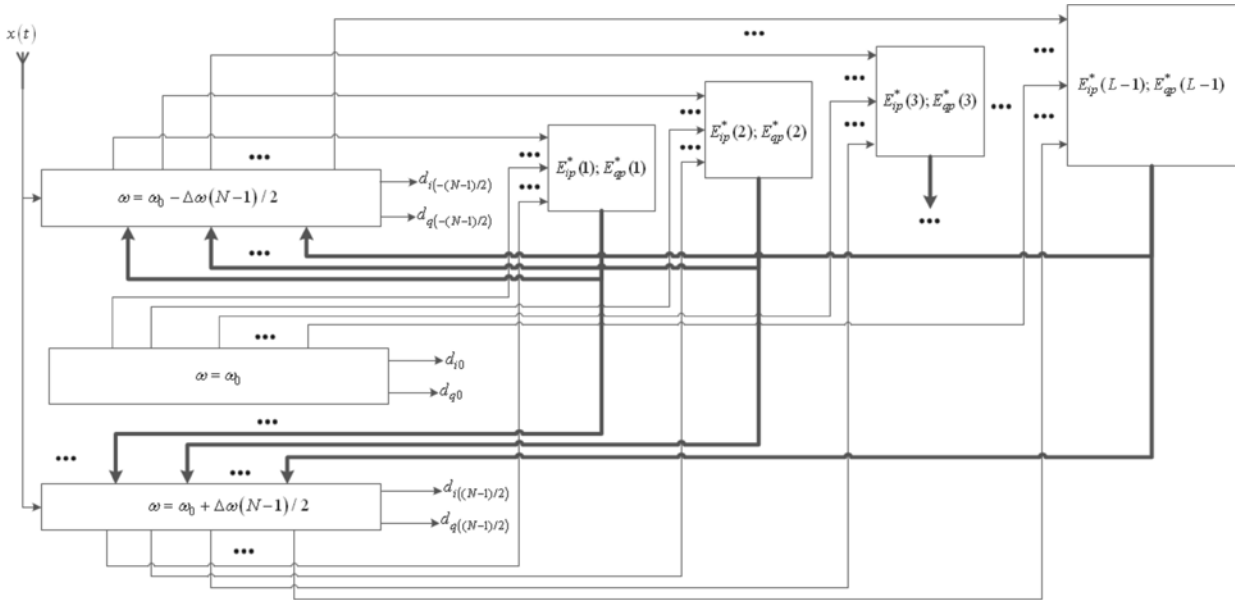
$$i = 1, 2; \; s_{i0}(t) = A_0 a(t)\cos(\omega_0 t);$$

$$E^*_{ip}(r-1) = (E^*_{1l}(r-1) - E^*_{2l}(r-1))/4.$$

При реализации алгоритма приема (7) используется N каналов ($\omega_0 \pm n\Delta\omega$, $n = -(N-1)/2, ..., 0, ..., (N-1)/2$) приема многочастотных сигналов ФМ-4 (рис. 1 *а*). В каждом канале применяются корреляторы (рис. 1 *б*), в которых в качестве опорного напряжения перемножителя используются сигналы $\cos((\omega_0 + n\Delta\omega)t)$ и $\sin((\omega_0 + n\Delta\omega)t)$. С выходов корреляторов значения результатов корреляционной обработки через линии задержки $r\Delta T_0$ ($r = 0, 1, ..., L-1$) и вычитающие устройства поступают на входы решающих устройств РУ. В РУ происходит сравнение результата корреляционной обработки с нулевым порогом. На выходах РУ формируются оценки $d_{in}{}^*(r)$ принятых информационных символов.

В блоках памяти $E^*_{ip}(k)E^*_{qp}(k)$ записаны значения разности энергий (8), с помощью

а)



б)

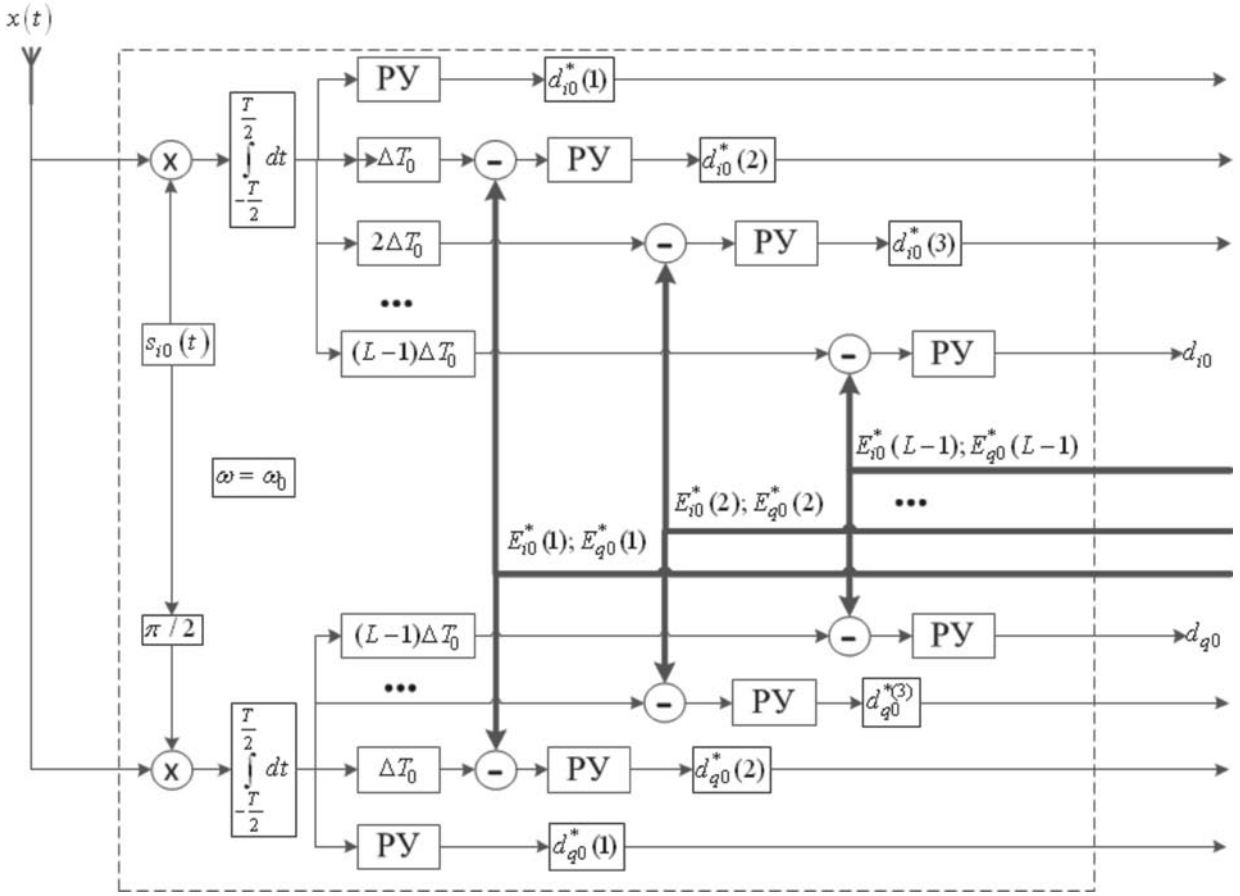


Рис. 1. Блок-схема алгоритма когерентного приема с компенсацией межканальной интерференции (7)

которых происходит компенсация межканальной интерференции. Такая компенсация осуществляется по цепи обратной связи. Линии задержки (рис. 1 *б*) необходимы для компенсации временной задержки, вызванной считыванием величин $E^*_{ip}(k-1); E^*_{qp}(k-1)$ из блоков памяти.

При правильных значениях оценок принятых символов происходит коррекция результатов корреляционной обработки путем вычитания значений разности энергий (8) в соответствии с алгоритмом (7). При ошибочных значениях оценок, разумеется, условия приема сигналов существенно ухудшаются.

Рассмотрим алгоритм когерентного поэлементного приема многочастотных неортогональных сигналов по диаграмме состояний [6, 7]. Необходимо найти последовательность символов $d_{i(-(N-1)/2)}$, $d_{q(-(N-1)/2)}$, ..., $d_{i((N-1)/2)}$, $d_{q((N-1)/2)}$, которая минимизирует значение целевой функции [9]:

$$Q = \int_{-T/2}^{T/2} \left| x(t) - s(t, d_{i(-(N-1)/2)}, d_{q(-(N-1)/2)}; \ldots; d_{i((N-1)/2)}, d_{q((N-1)/2)}) \right| dt, \quad (9)$$

где $x(t)$ определяется из (2).

Диаграмма состояний приведена на рис. 2. В узлах данной диаграммы расположены все возможные комбинации символов d_{in} и d_{qn} (+1 +1; −1 +1; +1 −1; −1 −1), которые передаются на каждой поднесущей частоте $\omega_{0+} n\Delta\omega$ ($n = -(N-1)/2, \ldots, (N-1)/2$). Общее количество узлов равно $4N$. Из каждого выбранного узла возможно четыре перехода к узлам, соответствующим комбинациям символов, передаваемых на соседних поднесущих частотах. Каждый путь по диаграмме состояний образуется от узла $d_{i(-(N-1)/2)}$, $d_{q(-(N-1)/2)}$ до узла $d_{i((N-1)/2)}$, $d_{q((N-1)/2)}$ и соответствует последовательности символов, по которой производится определение целевой функции (9). Количество возможных переходов между узлами в диаграмме состояний растет экспоненциально с ростом N.

Рассмотрим последовательное вычисление целевой функции (9). Для этого представим (9) в форме последовательной процедуры вычислений (по аналогии с алгоритмом Витерби).

На каждом шаге обработки производится расчет промежуточных значений целевой функции (обработка производится от поднесущей с номером $-(N-1)/2$ до поднесущей с номером $(N-1)/2$):

$$\begin{gathered} Q(r) = \int_{-T/2}^{T/2} \left| x(t) - s(t, d_{i(-(N-1)/2)}, \right. \\ \left. d_{q(-(N-1)/2)}; \ldots; d_{i(r)}, d_{q(r)}) \right| dt \\ s(t, d_{i(-N/2)}, d_{q(-N/2)}; \ldots; d_{i(r)}, d_{q(r)}) = \\ = \sum_{n=-(N-1)/2}^{r} s(t, d_{in}, d_{qn}), \\ r = -(N-1)/2, \ldots, (N-1)/2. \end{gathered} \quad (10)$$

Рекуррентная запись выражения (10) имеет вид:

$$Q(r) = Q(r-1) + V(r),$$

$$V(r) = \int_{-T/2}^{T/2} \left| x(t) - s(t, d_{ir} d_{qr}) \right| dt,$$

$$r = -(N-1)/2, \ldots, (N-1)/2.$$

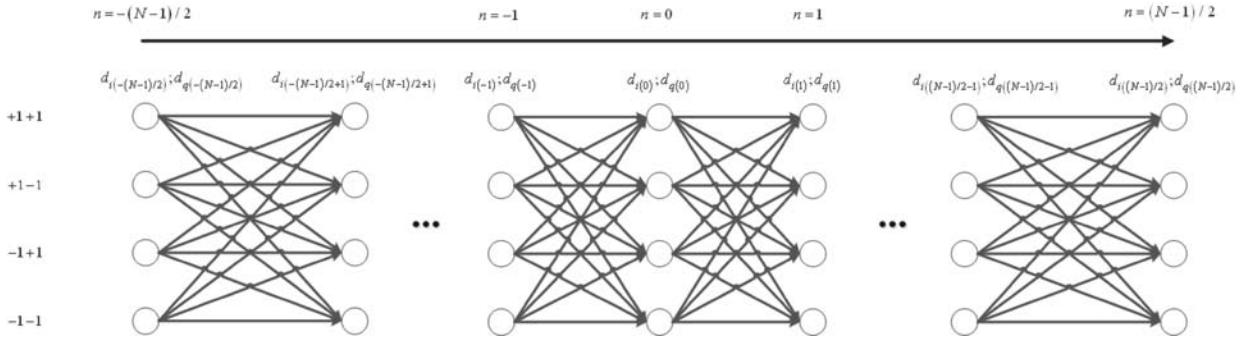


Рис. 2. Диаграмма состояний

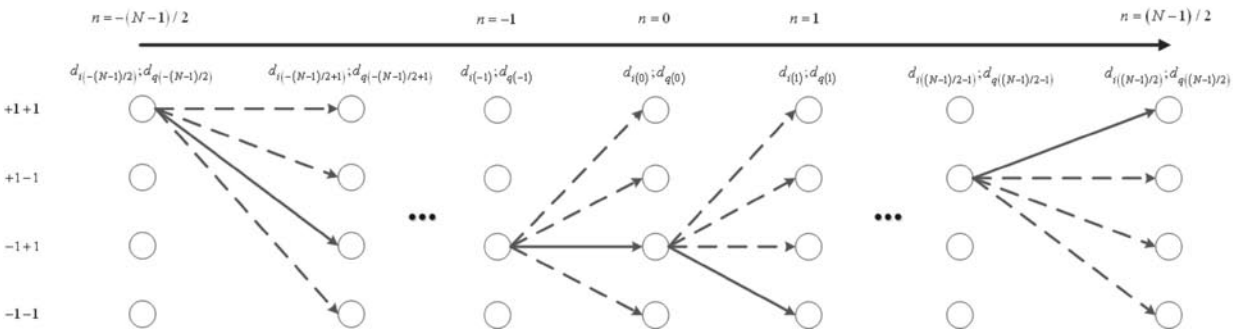


Рис. 3. Диаграмма состояний и переходов для случая последовательной процедуры вычисления целевой функции (10)

$$Q(-(N-1)/2) = V(-(N-1)/2) = \int_{-T/2}^{T/2} \left| x(t) - s(t, d_{i(-(N-1)/2)} d_{q(-(N-1)/2)}) \right| dt.$$

Таким образом, на каждом переходе между соседними поднесущими необходимо использовать при вычислении целевой функции сигналы, соответствующие уже принятым символам на предыдущих поднесущих частотах и всем возможным символам, передаваемым на данной поднесущей частоте.

Диаграмма состояний для случая выполнения последовательной процедуры вычисления целевой функции приведена на рис. 3. На каждом шаге вычисляется промежуточное значение целевой функции (10) для каждого из четырех возможных переходов. Далее выбирается переход, реализующий минимальное значение целевой функции. При любых принятых впоследствии символах результирующее значение целевой функции этих путей не станет лучше значения целевой функции «выжившего» пути (на рисунке обозначено сплошной линией). Таким образом, в памяти устройства обработки всегда содержится четыре пути, один из которых наиболее правдоподобный.

При реализации последовательной процедуры можно перейти от вычисления промежуточных значений целевой функции к вычислению метрики для каждого пути на основании следующего выражения:

$$D_n^k = \sqrt{\int_{-T/2}^{T/2} (x(t) - s(t, d_{i(-N/2)}, d_{q(-N/2)}; \ldots; d_{i(k-1)}, d_{q(k-1)}, d_{ik}, d_{qk}))^2 dt}. \quad (11)$$

Помехоустойчивость приема

Рассмотрим алгоритм (5) поэлементного когерентного приема сигналов на поднесущей частоте ω_0, когда сигналы, передаваемые на соседних поднесущих, являются помехой. Определим вероятность ошибок при приеме сигналов на одной квадратурной составляющей в (5). Подставляя (2) в (5), получим:

$$A_0^2 \int_{-T/2}^{T/2} a^2(t) \sum_{n=-(N-1)/2}^{(N-1)/2} [d_{in} \cos((\omega_0 + n\Delta\omega)t) + d_{qn} \sin((\omega_0 + n\Delta\omega)t)] \cos((\omega_0)t) dt + A_0 \int_{-T/2}^{T/2} n(t)\, a(t) \cos((\omega_0)t) dt \underset{d_{20}}{\overset{d_{10}}{\gtrless}} 0. \quad (12)$$

Пусть $a(t)$ имеет прямоугольный вид. Тогда (12) можно записать следующим образом:

$$\xi(t) \underset{<}{\overset{>}{\gtrless}} - E_0 d_{i0} - E_0 \sum_{\substack{n=-(N-1)/2 \\ n \neq 0}}^{(N-1)/2} \frac{d_{in}}{n\Delta\omega \frac{T}{2}} \sin\left(n\Delta\omega \frac{T}{2}\right), \quad (13)$$

где $E_0 = \frac{1}{2} A_0^2 T$; $\xi(t) = A_0 \int_{-T/2}^{T/2} n(t) \cos((\omega_0)t) dt \in$

$\in N\left\{0, \frac{N_0}{2} E_0\right\}$ – случайная величина, имеющая нормальное распределение.

Используя следующее обозначение:

$$\alpha(d_{in}\big|_{n\neq 0}) = \sum_{\substack{n=-(N-1)/2 \\ n\neq 0}}^{(N-1)/2} \frac{d_{in}}{n\Delta\omega\frac{T}{2}} \sin\left(n\Delta\omega\frac{T}{2}\right), \quad (14)$$

из (12) получим:

$$\xi(t) \underset{<}{\overset{>}{}} - E_0[d_{ik} + \alpha(d_{in}\big|_{n\neq 0})]. \quad (15)$$

Вероятность ошибочного приема сигнала, расположенного на поднесущей частоте ω_0, вычисляется по следующей формуле:

$$p_0(d_{in}\big|_{n\neq 0}) = \frac{1}{2} - \frac{1}{4}\Phi\{-\sqrt{2}h_0[-1 + \alpha(d_{in}\big|_{n\neq 0})]\} + \frac{1}{4}\Phi\{-\sqrt{2}h_0[1 + \alpha(d_{in}\big|_{n\neq 0})]\}, \quad (16)$$

где $h_0 = \sqrt{E_0 / N_0}$.

Полная вероятность ошибочного приема на поднесущей частоте ω_0 может быть получена путем усреднения (16) по всем возможным реализациям символов, передаваемых на соседних поднесущих частотах. В частном случае ортогонального разноса между поднесущими ($\Delta f = 1/T$) формула (12) легко приводится к [5, 8]:

$$\alpha(d_{in}\big|_{n\neq 0})\big|_{\Delta\omega=\frac{2\pi}{T}} = 0,$$

$$p_0\big|_{\Delta\omega=\frac{2\pi}{T}} = \frac{1}{2} - \frac{1}{2}\Phi\{\sqrt{2}h_0\}.$$

Зависимости полной вероятности ошибок p от отношения сигнал/шум для $N = 5$ при разных значениях ΔfT для алгоритма (5) приведены на рис. 4 *а*. Как следует из анализа представленных кривых, уменьшение частотного разноса между поднесущими (значения ΔfT) приводит к появлению дополнительных энергетических потерь по отношению сигнал/шум при фиксированной вероятности ошибок. Так, например, при переходе от значения $\Delta fT = 0{,}9$ к значению $\Delta fT = 0{,}825$ в области вероятностей ошибок $p = 10^{-3} - 10^{-4}$ дополнительные энергетические потери составляют более 5 дБ.

Определим помехоустойчивость приема при использовании алгоритма (8) с компенсацией межканальной интерференции путем имитационного моделирования для $N = 5$ (рис. 4 *б*). Видно, что при использовании (8) для $\Delta f T = 0{,}875$ дополнительные энергетические потери в области $p = 10^{-3} - 10^{-4}$ составляют не более 1 дБ. При разносе частот между поднесущими $\Delta f = 0{,}75/T$ для $p = 10^{-3}$ дополнительные энергетические потери составляют уже около 13 дБ. Данные результаты согласуются с [1, 2], согласно которым предельным

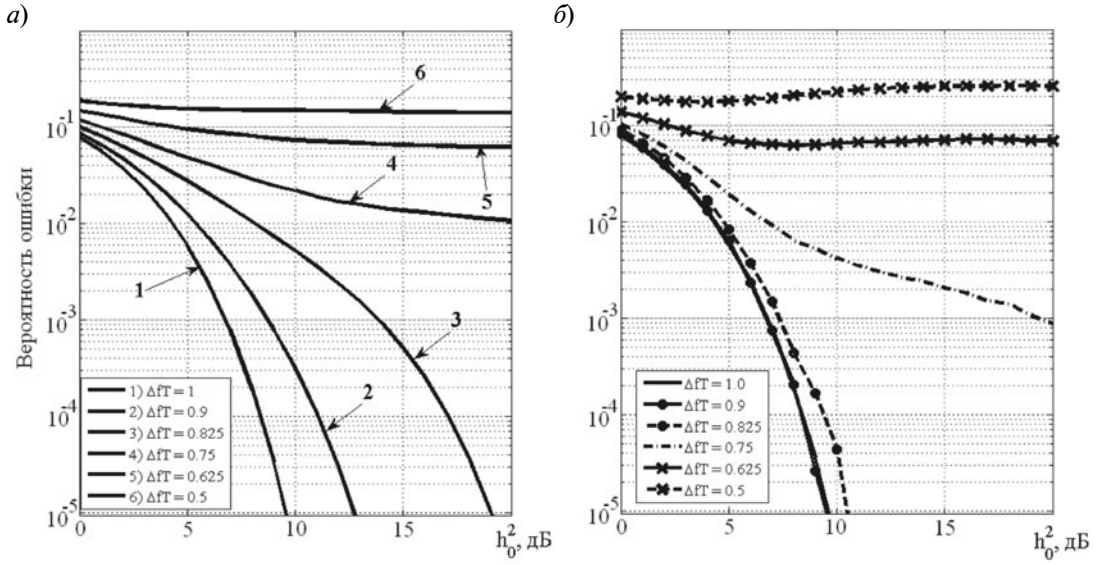


Рис. 4. Помехоустойчивость приема при использовании алгоритма (5) для $N = 5$ (*а*); результаты имитационного моделирования приема сигналов в соответствии с алгоритмом (8) с компенсацией межканальной интерференции для $N = 5$ (*б*)

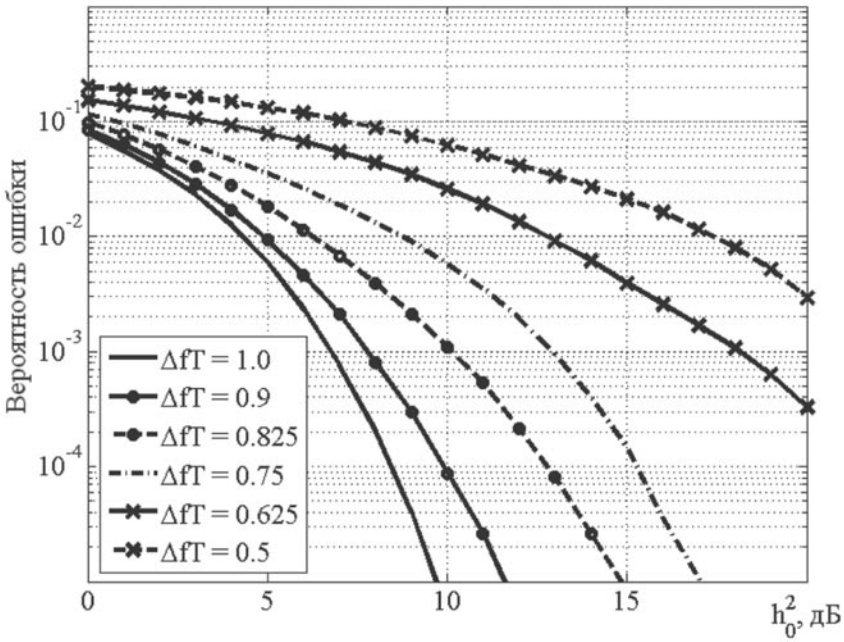


Рис. 5. Помехоустойчивость приема для алгоритма (10) при $N = 5$

значением Δf, при котором помехоустойчивость приема сопоставима с помехоустойчивостью приема ортогональных сигналов, является $\Delta f = 0{,}802/T$ (при дальнейшем уменьшении частотного разноса наблюдается значительное увеличение энергетического проигрыша относительно ортогонального разноса частот).

При еще большем уменьшении разноса частот между поднесущими достоверность приема существенно снижается и при значении $\Delta f = 0{,}5/T$ применение алгоритма (8) с компенсацией межканальной интерференции становится неэффективным. Заметим, что при использовании алгоритма с компенсацией межканальной интерференции из-за наличия цепи обратной связи (см. рис. 1) должен проявляться пороговый эффект, связанный с явлением группирования ошибочных решений символов d_{in}*. Это приводит к тому, что эффективность применения алгоритма (8) должна становиться меньше, чем алгоритма (6). Как показывают результаты имитационного моделирования, данный эффект проявляется в областях малых (меньше единицы) отношений сигнал/шум.

Перейдем к оценке помехоустойчивости приема при использовании последовательной процедуры вычислений (10). Результаты имитационного моделирования для $N = 5$ приведены на рис. 5. Как видно, в отличие от алгоритма с компенсацией межканальной интерференции при уменьшении значения ΔfT до значения 0,5 данный алгоритм остается эффективным.

При значении $\Delta fT = 0{,}5$ дополнительные энергетические потери оказываются достаточно высокими и составляют в области $p = 10^{-3}$ не менее 15 дБ. Кроме того, при значениях $\Delta fT = 0{,}9$ алгоритм (10) оказывается менее эффективным, чем алгоритм с компенсацией межканальной интерференции (см. рис. 4 и 5). Это связано с тем, что при использовании последовательной процедуры вычислений (10) не удается в значительной степени компенсировать межканальную интерференцию в отличие от алгоритма (8).

Для повышения помехоустойчивости когерентного приема неортогональных многочастотных сигналов предложен алгоритм с компенсацией межканальной интерференции. Структура такого алгоритма предполагает использование обратной связи по решению.

Получена последовательная процедура

вычислений целевой функции, на основе которой найдены диаграммы состояний. Последовательная процедура вычислений позволяет существенно повысить достоверность когерентного приема неортогональных многочастотных сигналов.

Показано, что при использовании алгоритма с компенсацией межканальной интерференции для $\Delta fT = 0{,}875$ дополнительные энергетические потери в области $p = 10^{-3} - 10^{-4}$ составляют не более 1 дБ.

Для последовательной процедуры вычисления целевой функции при значении $\Delta fT = 0{,}5$ дополнительные энергетические потери составляют в области вероятностей ошибок $p = 10^{-3}$ не более 15 дБ. Кроме того, при значениях $\Delta fT = 0{,}9$ эта процедура оказывается менее эффективной, чем алгоритм с компенсацией межканальной интерференции.

СПИСОК ЛИТЕРАТУРЫ

1. **Kanaras Y., Chorti A., Rodrigues M., Darwazeh I.** An overview of optimal and sub-optimal detection techniques for a non orthogonal spectrally efficient FDM // LCS/NEMS'09. London, 2009.

2. **Clegg R.G., Isam S., Kanaris I., Darwazeh I.** A practical system for improved efficiency in frequency division multiplexed wireless networks // IET Communications. 2012. Vol. 6. Iss. 4. Pp. 449–457.

3. **Слюсар В.И.** Неортогональное частотное мультиплексирование (N-OFDM) сигналов. Ч. 1. //Технологии и средства связи. 2013. № 5. С. 1–7.

4. **Макаров С.Б, Цикин И.А.** Передача дискретных сообщений по радиоканалам с ограниченной полосой пропускания. М.: Радио и связь, 1988.

5. **Макаров С.Б., Рашич А.В.** Снижение пик-фактора сигналов с ортогональным частотным уплотнением // Научно-технические ведомости СПбГПУ. Информатика. Телекоммуникации. Управление. СПб.: Изд-во СПбГПУ, 2008. № 2(55). С. 79–84.

6. **Макаров С.Б., Рашич А.В.** Метод формирования спектрально-эффективных OFDM-сигналов на основе неортогональных базисных функций // Научно-технические ведомости СПбГПУ. Информатика. Телекоммуникации. Управление. СПб.: Изд-во СПбГПУ, 2009. № 2(76). С. 94–98.

7. **Макаров С.Б., Рашич А.В.** Формирование и прием спектрально-эффективных сигналов с OFDM // Научно-технические ведомости СПбГПУ. Информатика. Телекоммуникации. Управление. СПб.: Изд-во СПбГПУ, 2011. № 6-2(138). С. 19–26.

8. **Феер К.** Беспроводная цифровая связь. Методы модуляции и расширения спектра. Пер. с англ. М.: Радио и связь, 2000. 520 с.

9. **Макаров С.Б., Марков А.М.** Прием в «целом» случайных последовательностей частотно-манипулированных сигналов с межсимвольной интерференцией // Радиотехника. 2011. № 3. С. 46–51.

REFERENCES

1. **Kanaras Y., Chorti A., Rodrigues M., Darwazeh I.** An overview of optimal and sub-optimal detection techniques for a non orthogonal spectrally efficient FDM. *LCS/NEMS'09,* London, UK, 2009.

2. **Clegg R.G., Isam S., Kanaris I., Darwazeh I.** A practical system for improved efficiency in frequency division multiplexed wireless networks, *IET Communications,* 2012, Vol. 6, Iss. 4, Pp. 449–457.

3. **Clyusar V.I.** Neortogonalnoye chastotnoye multipleksirovaniye (N-OFDM) signalov. Chast 1, *Tekhnologii i sredstva svyazi*, 2013, No. 5, Pp. 1–7. (rus)

4. **Makarov S.B, Tsikin I.A.** Peredacha diskretnykh soobshcheniy po radiokanalam s ogranichennoy polosoy propuskaniya. Moscow: Radio i svyaz Publ., 1988. (rus)

5. **Makarov S.B., Rashich A.V.** Snizheniye pik-faktora signalov s ortogonalnym chastotnym uplotneniyem, *Nauchno-tekhnicheskiye vedomosti SPbGPU. Informatika. Telekommunikatsii. Upravlenie*, St. Petersburg: SPbGPU Publ., 2008, No. 2(55), Pp. 79–84. (rus)

6. **Makarov S.B., Rashich A.V.** Metod formirovaniya spektralno-effektivnykh OFDM-signalov na osnove neortogonalnykh bazisnykh funktsiy. *Nauchno-tekhnicheskiye vedomosti SPbGPU. Informatika. Telekommunikatsii. Upravlenie*, St. Petersburg: SPbGPU Publ., 2009, No. 2(76), Pp. 94–98. (rus)

7. **Makarov S.B., Rashich A.V.** Formirovaniye i priyem spektralno-effektivnykh signalov s OFDM. *Nauchno-tekhnicheskiye vedomosti SPbGPU. Informatika. Telekommunikatsii. Upravlenie*, St. Petersburg: SPbGPU Publ., 2011, No. 6-2(138), Pp. 19–26. (rus)

8. **Feyer K.** Besprovodnaya tsifrovaya svyaz. Metody modulyatsii i rasshireniya spektra. Per. s angl.

Moscow: Radio i svyaz Publ., 2000, 520 p. (rus)

9. **Makarov S.B. Markov A.M.** Priyem v «tselom» sluchaynykh posledovatelnostey chastotno-manipulirovannykh signalov s mezhsimvolnoy interferentsiyey, *Radiotekhnika*, 2011, No. 3, Pp. 46—51. (rus)

МАКАРОВ Сергей Борисович — *директор Института физики, нанотехнологий и телекоммуникаций Санкт-Петербургского государственного политехнического университета, доктор технических наук, профессор.*
195251, Санкт-Петербург, ул. Политехническая, д. 29.
E-mail: makarov@cee.spbstu.ru

MAKAROV, Sergey B. *St. Petersburg State Polytechnical University.*
195251, Politekhnicheskaya Str. 29, St. Petersburg, Russia.
E-mail: makarov@cee.spbstu.ru

ЗАВЬЯЛОВ Сергей Викторович — *ассистент кафедры радиоэлектронных средств защиты информации, Института физики, нанотехнологий и телекоммуникаций Санкт-Петербургского государственного политехнического университета.*
195251, Санкт-Петербург, ул. Политехническая, д. 29.
E-mail: sz5@mail.ru

ZAVYALOV, Sergey V. *St. Petersburg State Polytechnical University.*
195251, Politekhnicheskaya Str. 29, St. Petersburg, Russia.
E-mail: sz5@mail.ru

УДК 621.37

В.М. Малышев, Ю.А. Матвеев, А.Б. Никитин, А.В. Худяков

МОДЕЛЬ ВАРИКАПА ДЛЯ РАЗРАБОТКИ СВЕРХШИРОКОПОЛОСНЫХ ПЕРЕСТРАИВАЕМЫХ ГЕНЕРАТОРОВ СВЧ

V.M. Malyshev, Yu.A. Matveev, A.B. Nikitin, A.V. Khudyakov

VARACTOR DIODE MODEL USED TO DESIGN WIDEBAND MICROWAVE VOLTAGE-CONTROLLED OSCILLATORS

Рассмотрены модели варикапов, предназначенных для сверхширокополосной перестройки частоты твердотельных генераторов сантиметрового диапазона длин волн. На основе экспериментальных данных уточнены параметры модели варикапа, адекватно отражающей его свойства и позволяющей осуществлять разработку перестраиваемых генераторов с помощью программ автоматизированного проектирования.

ВАРИКАП; СВЧ-ГЕНЕРАТОР; ПЕРЕСТРОЙКА ЧАСТОТЫ.

This article describes the nonlinear model of the varactor diode for use in broadband microwave voltage-controlled oscillator. The proposed model has been verified based on measurement data and adequately reflects the characteristics of the varactor diode. This varactor model can be used in various types of CAD for accurately design broadband microwave VCO and frequency-tuning circuits.

VARACTOR DIODE; MICROWAVE OSCILLATOR; FREQUENCY TUNING.

Генератор, управляемый напряжением (ГУН), – один из неотъемлемых компонентов современных синтезаторов частоты с ФАПЧ. Использование в качестве такого ГУН сверхширокополосных твердотельных СВЧ-генераторов с варикапами дает возможность создавать быстродействующие широкодиапазонные источники сигналов [1, 2].

Одним из ключевых элементов сверхширокополосных ГУН, с полосой перестройки, достигающей октавы и более, является варикап. Параметры варикапа в существенной степени определяют такие характеристики ГУН, как диапазон перестройки, нелинейность зависимости частоты генерации от управляющего напряжения, флуктуационные характеристики генератора и ряд других [2, 3]. Поэтому при построении сверхширокополосных ГУН требуется наличие нелинейной модели варикапа, адекватно отражающей его свойства и позволяющей осуществлять процесс разработки генератора с помощью современных программ автоматизированного проектирования СВЧ-устройств. В данной статье рассматриваются модели варикапов, предназначенных для сверхширокополосной перестройки частоты твердотельных генераторов сантиметрового диапазона волн, выполненных по гибридной технологии.

Генераторы с октавной перестройкой частоты предъявляют повышенные требования к управляющим элементам – варикапам. Так, например, коэффициент перекрытия емкости диода $k = C_{max}/C_{min}$, необходимый для получения такой перестройки, должен быть не менее четырех, даже в случае идеа-

лизированного представления колебательной системы генератора одиночным колебательным контуром на сосредоточенных параметрах, вся емкость которого является управляемой. В реальности наличие паразитных параметров варикапа и компонентов схемы генератора, а также учет влияния реактивной составляющей сопротивления транзистора приводят к существенному повышению требуемых значений коэффициента перекрытия емкости k до величин порядка 6-8. Впрочем, конкретные требования к СВЧ-диодам, обеспечивающим октавную перестройку частоты, могут быть сформулированы на основе результатов анализа ГУН при наличии модели активного элемента генератора.

Выбор варикапов с такими неординарными характеристиками еще более сужается из-за необходимости обеспечения столь существенного изменения емкости в области значений менее 1 пФ, а в коротковолновой части сантиметрового диапазона и до минимальных величин менее 0,1 пФ.

К числу диодов с требуемыми параметрами можно отнести, например, варикапы BB857 (фирма Infineon [4]) с коэффициентом перекрытия емкости $k = 12$ и минимальным значением емкости $C_{min} = 0{,}5$ пФ, варикапы серии MA46H07X (фирма MACOM [5]) с $k = 5{,}5, \ldots, 7{,}5$ и минимальным значением емкости менее 0,4 пФ, варикапы серии MGV125 (фирма Aeroflex [6]) с коэффициентом перекрытия емкости $k = 10$, отличающиеся от других аналогичных приборов рекордно малыми значениями емкостей. Например, для диода MGV125-08 величина C_{min} по данным производителя менее 0,06 пФ [6]. Кроме того, эти приборы MGV125 поставляются не только в корпусном исполнении, но и в форме чипа (модификация MGV125-XX-C01A), представляющем собой параллелепипед размером 0,3 × 0,3 × 0,12 мм. Столь малые размеры снижают влияние паразитных параметров и делают удобным применение таких диодов в гибридно-интегральных микросхемах сантиметрового диапазона.

На рис. 1 представлен один из вариантов общеупотребительной эквивалентной схемы варикапа, которая может использоваться для моделирования ГУН [3].

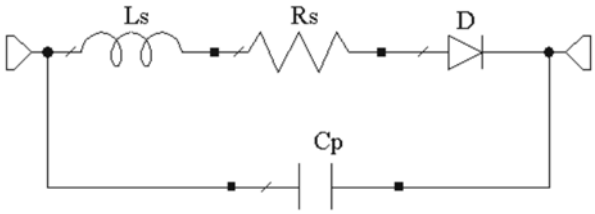


Рис. 1. Эквивалентная схема варикапа

Данная модель включает в себя в качестве подсхемы, обозначенной на рисунке символом диода D, SPICE-модель выпрямляющего перехода. Основные параметры SPICE-модели, определяющие характеристики варикапа, – барьерная емкость при нулевом смещении C_0, контактная разность потенциалов U_J и степень нелинейности выпрямляющего перехода M, входящие в аппроксимацию вольт-фарадной характеристики диода $C(U)$ [3]:

$$C(U) = C_0 \cdot (1 - U/U_J)^{-M}.$$

Некоторые производители предоставляют данные, позволяющие построить модель, необходимую для проектирования ГУН. Например, приводят вольт-фарадные характеристики варикапов. Такой подход, базирующийся на имеющихся в технической документации на полупроводниковые приборы данных, был использован при разработке сверхширокополосных ГУН сантиметрового диапазона длин волн для построения моделей варикапов BB857, MA46H070, MA46H071 в полосе частот 2...12 ГГц и MGV125 в полосе частот 4...16 ГГц. На основе предоставляемых производителями вольт-фарадных характеристик диодов методами параметрической оптимизации были найдены значения параметров их моделей, обеспечивающие хорошее совпадение вольт-фарадных характеристик с данными производителя [4–6]. Однако результаты экспериментальных исследований макетов генераторов, разработанных на основе таких моделей, показали существенное отличие полученных характеристик генератора от рассчитанных. Дополнительные измерения S-параметров, используемых в перестраиваемых генераторах варикапов, показали, что причиной полученных расхождений

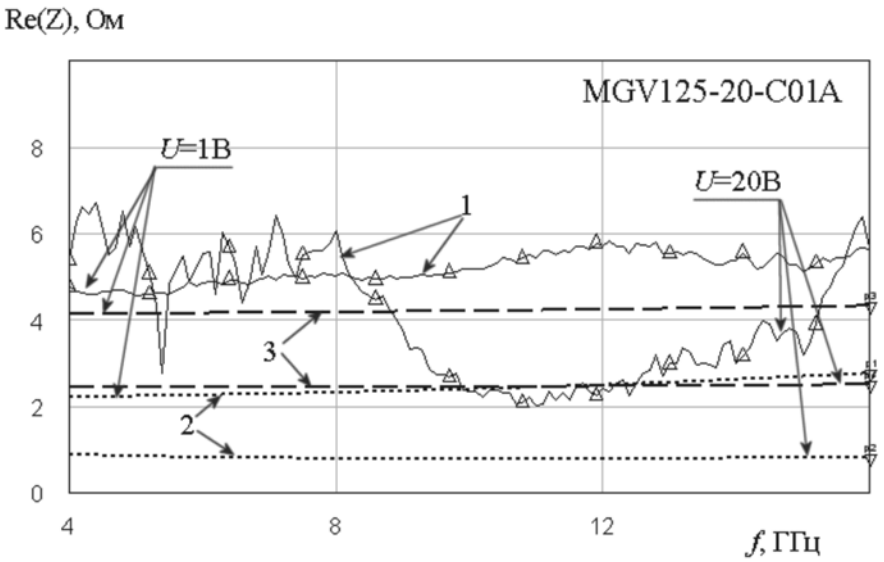


Рис. 2. Частотные зависимости вещественной составляющей сопротивления варикапа:
1– эксперимент (Δ——Δ); 2 – модель на основе паспортных данных (·····); 3 – уточненная модель (– – –)

являются неточности моделей диодов, созданных на основе их паспортных данных. В качестве примера на рис. 2 приведены частотные зависимости вещественной, а на рис. 3 – мнимой составляющей сопротивления варикапов MGV125-20-C01A для двух крайних значений напряжения смещения на диоде (U = 1 В и 20 В).

На графиках экспериментальные зависимости отмечены цифрой 1, а результаты моделирования – 2. Как видно из графиков, отличие экспериментальных данных от характеристик моделей проявляются не только в изменении среднего значения емкости варикапа (рис. 3), но и в увеличении потерь (рис. 2), а также в уменьшении эквивалентного коэффициента перекрытия емкости k. Так, например, если вещественная составляющая сопротивления Re(Z) модели диода MGV125-20 изменяется в преде-

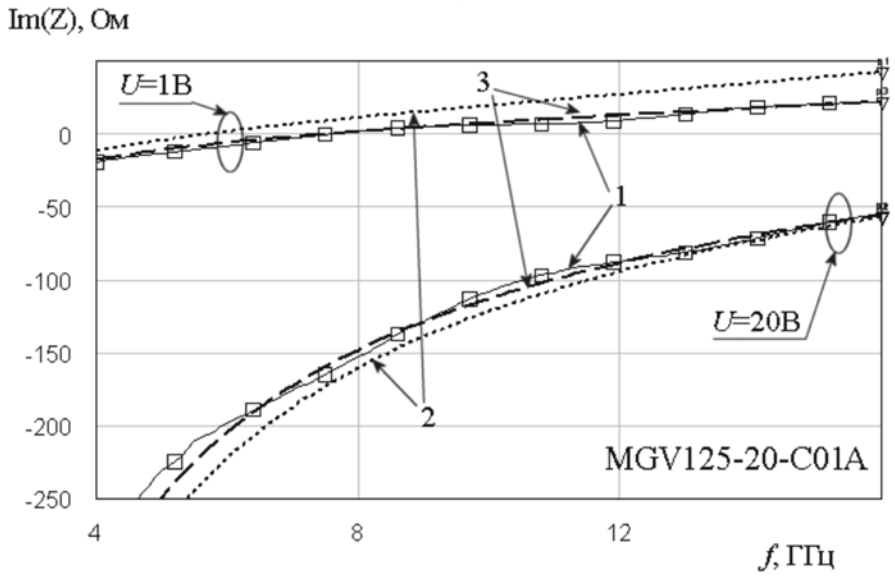


Рис. 3. Частотные зависимости мнимой составляющей сопротивления варикапа:
1– эксперимент (□——□); 2 – модель на основе паспортных данных (·····); 3 – уточненная модель (– – –)

лах 1...2,5 Ом, то по данным измерений 2 Ом < Re(Z) < 6 Ом (см. рис. 2). Еще более существенно увеличение сопротивления для диодов других типов: MGV125-08 и MGV125-09. Так, для диода MGV125-09 наблюдается увеличение потерь в 3–5 и более раз.

Указанные обстоятельства обусловлены, очевидно, отличием условий измерений его характеристик, приведенных в паспортных данных, от требуемых, соответствующих реальным режимам работы варикапа при перестройке контура генератора. К таким наиболее важным отличиям следует отнести частотный диапазон и способ монтажа диода на плате генератора.

Первое обстоятельство может сказываться на неверном определении последовательного сопротивления R_S (см. рис. 1, 2). Как, например, отмечается в [3], нахождение R_S из приведенных в паспортных данных величин добротности диода может приводить к существенно заниженным значениям этого сопротивления. Учет в дополнение к частотному диапазону особенностей монтажа диода на плате генератора (например, влияния контактных площадок микрополосковой схемы) может существенно влиять на общий вид эквивалентной схемы варикапа.

Данные измерений показали необходимость создания скорректированных моделей варикапов, требуемых для разработки перестраиваемых генераторов. Коррекция моделей варикапов потребовала не только подбора новых значений параметров, но и изменения структуры эквивалентной схемы. Упрощенный вариант модели диодов серии MGV125, обеспечивающий удовлетворительное совпадение с экспериментальными данными, представлен на рис. 4.

На рис. 4 *а* приведена эквивалентная схема чипа варикапа. На данной схеме символом диода D обозначена его нелинейная SPICE-модель. Для обеспечения надлежащего соответствия характеристик модели результатам измерений последовательное сопротивление R_S пришлось разделить на две составляющие: R_S и R. Для учета влияния на импедансные характеристики варикапа неоднородностей, возникающих при включении диода в микрополосковую линию, использовалась схема, приведенная

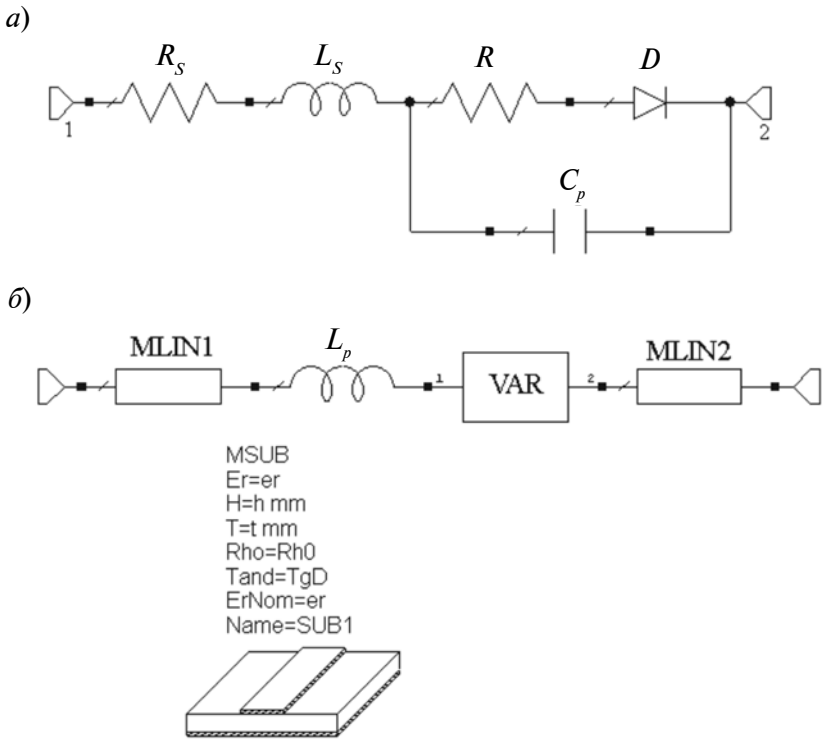


Рис. 4. Модель варикапа

Параметры диодов

Варикап	M	C_0, пФ	U_J, В	L_S, нГн	R_S, Ом	R, Ом	C_p, пФ	l_1, мм	l_2, мм	L_p, нГн
MGV125_08	1,315	1,81	0,926	0,072	2,0-6,5	3,5	0,033	0,092	0,181	0,15
MGV125_09	1,355	2,2	1,016	0,07	1,8-6,0	3,0	0,035	0,084	0,155	0,18
MGV125_20	1,372	3,872	1,075	0,03	1,8-3,2	2,0	0,05	0,08	0,106	0,175
MGV125_21	1,315	6,0	0,877	0,075	1,0-1,8	1,5	0,08	0,084	0,155	0,173

на рис. 4 *б*. На этой схеме элементы MLIN1 и MLIN2 – это отрезки микрополосковых линий шириной $w = 0{,}5$ мм и длиной l_1 и l_2, соответственно, которые моделируют контактные площадки, L_p – индуктивность проволочного контакта линии с анодом диода. Блоком VAR на рис. 4 *б* обозначена модель диодного чипа, представленная на рис. 4 *а*.

В таблице приведены параметры скорректированной нелинейной SPICE-модели диодов (C_0, U_J и M) и значения элементов эквивалентных схем варикапов (рис. 4), вычисленные на основе экспериментальных данных.

Импедансные характеристики, полученные для скорректированных моделей варикапов, обозначены на рис. 2, 3 цифрой 3. Как видно из графиков, скорректированный вариант модели более адекватно описывает характеристики варикапов в исследуемом частотном диапазоне. Так, например, диапазон изменения вещественной составляющей сопротивления Re(Z) для модели MGV125-20-C01A составляет 2,4...4,2 Ом, а по результатам измерений 2...6 Ом. Для мнимой составляющей Im(Z) максимальное отличие от экспериментальных данных снижается в несколько раз (рис. 3). Особенно это заметно для малых значений управляющего напряжения (U = 1...3 В), где несовпадение импедансных характеристик уменьшается с сотен до единиц процентов.

Таким образом, в результате проделанной работы уточнены модели варикапов, включенных в разрыв микрополосковой линии, которые могут использоваться при проектировании сверхширокополосных перестраиваемых генераторов СВЧ диапазона, построенных по гибридной технологии. Полученные модели прошли апробацию в процессе разработки микрополосковых ГУН с октавной полосой перестройки в сантиметровом диапазоне волн, выполненных на SiGe-транзисторе.

СПИСОК ЛИТЕРАТУРЫ

1. **Chenakin A.** Frequency synthesizers: Concept to Product. Artech House, Inc., 2011. 254 p.

2. **Горевой А.** Выбор генераторов для построения малошумящих СВЧ-синтезаторов // Компоненты и технологии. 2012. № 6. С. 87–92.

3. **Stauffer G.H.** Finding the Lumped Element Varactor Diode Model // High Frequency Electronics. 2003. Vol. 2. No. 6. Pp. 22–28.

4. Infineon BB837/BB857 Silicon Tuning Diode. Datasheet [электронный ресурс] /Infineon Technologies AG // URL: http://www.infineon.com

5. MA46 Series. Surface Mount GaAs Tuning Varactors. Datasheet Rev. V6 [электронный ресурс] / M/A-COM Technology Solutions Inc. // URL: http://www.macomtech.com/varactor-tuning-diodes

6. GaAs Hyperabrupt Varactor Diodes MGV Series. Datasheet [электронный ресурс] / Aeroflex Microelectronic Solutions // URL: http://www.aeroflex.com

REFERENCES

1. **Chenakin A.** *Frequency synthesizers: Concept to Product*, Artech House, Inc., 2011, 254 p.

2. **Gorevoy A.** Vybor generatorov dlya postroyeniya maloshumyashchikh SVCH-sintezatorov [Choosing generators for constructing low-noise microwave synthesizers], *Komponenty i tekhnologii* [Components & Technologies], 2012, No. 6, Pp. 87–92. (rus)

3. **Stauffer G.H.** Finding the Lumped Element Varactor Diode Model, *High Frequency Electronics*, 2003, Vol. 2, No. 6, Pp. 22–28.

4. Infineon BB837/BB857 Silicon Tuning Diode. Datasheet. *Infineon Technologies AG*. Available: http://www.infineon.com

5. MA46 Series. Surface Mount GaAs Tuning Varactors. Datasheet Rev. V6. *M/A-COM Technology Solutions Inc*. Available: https://www.macomtech.com/varactor-tuning-diodes

6. GaAs Hyperabrupt Varactor Diodes MGV Series. Datasheet. *Aeroflex Microelectronic Solutions*. Available: http://www.aeroflex.com

---

МАЛЫШЕВ Виктор Михайлович – *доцент кафедры радиотехники и телекоммуникаций Санкт-Петербургского государственного политехнического университета, кандидат физико-математических наук*.
195251, Россия, Санкт-Петербург, ул. Политехническая, д. 29.
E-mail: uhmal@mail.ru

MALYSHEV, Victor M. *St. Petersburg State Polytechnical University*.
195251, Politekhnicheskaya Str. 29, St. Petersburg, Russia.
E-mail: uhmal@mail.ru

МАТВЕЕВ Юрий Александрович – *руководитель отделения разработки РПУ, ООО «Специальный Технологический Центр»*.
195256, Россия, Санкт-Петербург, ул. Софьи Ковалевской, д. 22.
E-mail: matveev.rf@gmail.com

MATVEEV, Yuriy A. *Special Technological Center JSC*.
195256, Sofia Kovalevskaya Str. 22, St. Petersburg, Russia.
E-mail: matveev.rf@gmail.com

НИКИТИН Александр Борисович – *доцент кафедры радиотехники и телекоммуникаций Санкт-Петербургского государственного политехнического университета, кандидат технических наук*.
195251, Россия, Санкт-Петербург, ул. Политехническая, д. 29.
E-mail: nikitin@mail.spbstu.ru

NIKITIN, Aleksandr B. *St. Petersburg State Polytechnical University*.
195251, Politekhnicheskaya Str. 29, St. Petersburg, Russia.
E-mail: nikitin@mail.spbstu.ru

ХУДЯКОВ Артем Владимирович – *студент кафедры радиотехники и телекоммуникаций Санкт-Петербургского государственного политехнического университета*.
195251, Россия, Санкт-Петербург, ул. Политехническая, д. 29.

KHUDYAKOV, Artem V. *St. Petersburg State Polytechnical University*.
195251, Politekhnicheskaya Str. 29, St. Petersburg, Russia.

УДК 621. 315.1, 621.372.2

Р.Г. Минуллин, В.А. Касимов, Т.К. Филимонова, М.Р. Яруллин

ЛОКАЦИОННОЕ ОБНАРУЖЕНИЕ ГОЛОЛЕДА НА ВОЗДУШНЫХ ЛИНИЯХ ЭЛЕКТРОПЕРЕДАЧИ. ЧАСТЬ 1. СПОСОБЫ ОБНАРУЖЕНИЯ ГОЛОЛЕДА

R.G. Minullin, V.A. Kasimov, T.K. Filimonova, M.R. Yarullin

LOCATION DETECTION OF GLAZE ICE ON OVERHEAD ELECTRIC POWER LINES. PART 1. METHODS OF GLAZE ICE DETECTION

Описаны реализованные на практике способы обнаружения гололеда на проводах воздушных линий электропередачи. Подробно рассмотрен метод локационного обнаружения гололеда. Приведены примеры локационного обнаружения гололеда в результате многолетних исследований на действующих линиях электропередачи. Дана оценка преимущества локационного способа обнаружения гололеда перед другими способами.

ЛИНИИ ЭЛЕКТРОПЕРЕДАЧИ; ГОЛОЛЕД НА ПРОВОДАХ; СПОСОБЫ ОБНАРУЖЕНИЯ; ЛОКАЦИОННОЕ ЗОНДИРОВАНИЕ; МЕТОДИКА ОБНАРУЖЕНИЯ ГОЛОЛЕДА.

The paper presents practically implemented methods of detection of glaze ice on wires, such as overhead electric power lines. The paper describes in detail the method of location detection of glaze ice. Authors give several examples of glaze ice detection resulting from multiannual investigations of the active electric power lines. Authors discuss the advantages of the location detection over other methods of glaze ice detection.

ELECTRIC POWER LINES; GLAZE ICE ON WIRES; DETECTION METHODS; LOCATION PROBING; GLAZE ICE DETECTION TECHNOLOGY.

Гололедные отложения на проводах воздушных линий электропередачи (ЛЭП) обычно образуются на территории нескольких энергосистем. При этом возникают массовые провисания и обрывы проводов, разрушения арматуры, поломки опор воздушных линий электропередачи. Гололедные аварии на ЛЭП имеют массовый характер и приносят огромный материальный ущерб из-за недоотпуска электроэнергии потребителям и необходимости проведения ремонтно-восстановительных работ.

Эти аварии составляют для территории России около 25 % от общего количества повреждений на воздушных линиях, а их продолжительность – около 40 % от общей продолжительности всех аварийных отключений [1]. При гололедных нагрузках ниже нормы в результате аэродинамического воздействия могут возникать колебания (пляска) проводов при одностороннем отложении гололеда или вибрации при цилиндрической форме гололеда [2].

Многочисленные аварии показали, что оптимально спроектировать линию (сведя этот процесс только к расчету и определению геометрических параметров линии) без использования различных способов и устройств, ограничивающих и предупреждающих атмосферные воздействия на нее, невозможно.

Процесс образования гололеда на проводах воздушных линий зависит от климатического района и подчиняется определенным метеорологическим закономерностям: зависит от влажности и температуры окружающего воздуха, ветрового режима.

На образование гололеда влияют также размеры диаметра проводов, высота их подвеса, жесткость их крепления, исключающая закручивание, величина протекающего нагрузочного тока.

Поэтому в настоящее время существуют два направления обнаружения гололеда:

1) прогнозирование вероятности возможного гололедообразования на основе метеорологических данных воздушной среды, окружающей провода, с учетом технических параметров ЛЭП;

2) непосредственный контроль процесса гололедообразования на проводах с помощью датчиков и устройств обнаружения гололеда [3, 4], что позволяет достаточно точно определять момент начала его необходимой плавки.

Прогнозирование гололедообразования на основе метеорологических данных воздушной среды применяют во многих странах, где обледенение линий электропередачи является актуальной проблемой, чтобы смягчить или избежать его влияния на работоспособность этих линий.

Работы по прогнозированию гололедообразования на проводах воздушных линий ведутся в исследовательских центрах Чехии [5, 6], Исландии [7, 8], Канады [9, 10], Франции [11], Венгрии [12], Великобритании [13–15], Италии [16] и др. Основой прогноза являются модельные закономерности таких метеорологических явлений, как влажность и температура окружающего воздуха, ветровые давления, их изменения с высотой от поверхности земли. При этом учитываются рельеф местности, где проходит трасса воздушной ЛЭП, высота трассы над уровнем моря, а также климатические и погодные условия. Прогноз осуществляется применительно к техническим параметрам воздушной линии.

В применяемых моделях гололедообразования делается ряд допущений, т. к. неизвестны точный диапазон температур для условий выпадения влажного снега, продолжительность осадков, содержание жидкой воды в обледенении. Все это снижает прогностические возможности модели. Кроме того, важным фактором, который должен быть принят во внимание в модели, является нагревание провода за счет эффекта Джоуля, вызванное протекающим электрическим током.

В настоящее время нет определенной модели возникновения гололедных отложений, которая может достоверно учитывать все физические и механические процессы, участвующие при обледенении, поэтому количество ложных тревог высоко.

К сожалению, данные прогноза являются предупреждением о возможной угрозе возникновения гололедообразования и не могут служить конкретным указанием о начале плавки гололеда, образовавшегося на проводах воздушных линий электропередачи.

На сегодняшний день имеется огромное количество патентов, предлагающих методы и датчики для обнаружения гололеда [4].

Датчики при появлении гололеда на ЛЭП реагируют на изменения:

- физических параметров среды, окружающей провода;
- электрических характеристик проводов;
- веса или натяжения проводов;
- условий распространения высокочастотных и импульсных сигналов по проводам воздушных линий.

Практическое применение для обнаружения гололеда нашли метод взвешивания проводов [17–20] и метод локационного зондирования линий электропередачи [4, 21, 22].

Известно, что нарастание гололеда до аварийных пределов может произойти за считанные часы. Подготовка к действию устройства для плавки гололеда требует времени от одного часа и более (зависит от схемы плавки, потребителя, протяженности ЛЭП и т. д.). Поэтому для эффективной борьбы с гололедом важны два фактора: раннее обнаружение начала гололедообразования и достоверная и надежная информация о динамике гололедообразования.

Несвоевременная и недостоверная информация о гололедообразовании ведет к авариям и громадным убыткам.

Наиболее объективным методом измерения величины гололедной нагрузки на проводах воздушной линии является метод измерения веса одного или нескольких про-

летов провода воздушной линии. Величина натяжения провода при этом определяется нагрузками от гололеда и ветра, а также температурой окружающей среды. Оценка степени напряженного состояния провода и сравнение ее с предельно допустимым значением осуществляется с помощью весового (тензометрического) датчика. Показания датчика передаются на диспетчерский пункт с использованием канала связи [17–20].

В настоящее время в ОАО «МЭС Юга», а также в ряде энергосистем России – «Ростовэнерго», «Ставропольэнерго», «Кубаньэнерго», «Волгоградэнерго», «Башкирэнерго», «Сахалинэнерго» находится в эксплуатации автоматизированная информационная система контроля гололедообразования на воздушных линиях (АИСКГ), в которой используются весовые датчики и которая разработана и внедрена творческим коллективом сотрудников ЮРГТУ (НПИ) и СКБПиСА (г. Невинномысск) [19].

За рубежом используется подобная система обнаружения гололеда «CAT-1», разработанная в 1991 г. В настоящее время «CAT-1» производится концерном NEXANS [20].

В упомянутой системе АИСКГ для обнаружения гололеда используются весовые точечные датчики, определяющие вес гололедного отложения только около одной опоры ЛЭП. Для расширения зоны контроля применяются устройства видеонаблюдения. Общая гололедная ситуация на ЛЭП определяется путем прогнозирования опасных гололедных отложений на основе текущих метеорологических данных (температура и влажность окружающей среды, направление и скорость ветра), а также данных о температуре токонесущего провода.

АИСКГ состоит из пунктов контроля, расположенных на линиях электропередачи в местах наиболее вероятного гололедообразования, и приемных пунктов, расположенных в диспетчерских центрах.

Пункты контроля включают:

микропроцессорный линейный преобразователь;

датчики гололедной нагрузки на проводах и грозозащитных тросах,

устройства видеонаблюдения;

автоматические метеопосты с датчиками температуры и влажности воздуха, скорости и направления ветра;

датчики температуры провода;

устройства передачи и приема данных через каналы связи (радиоканал в УКВ диапазоне, канал сотовой связи, спутниковый канал, волоконно-оптический канал связи или каналы телемеханики).

В состав АИСКГ входят пункты контроля с выдачей до 30 параметров с каждого пункта. Приемные пункты обеспечивают циклический опрос пунктов контроля с заданной дискретностью (1–30 мин) и передачу информации в подсистему сбора данных АИСКГ.

Данные с пунктов контроля могут отображаться различными способами в зависимости от решаемой задачи: на карте местности, где нанесена схема сети (позволяет оценить картину распространения гололеда); в виде таблицы с текущими данными по всем пунктам контроля; в виде графиков по каждому пункту контроля за любой период времени (позволяет увидеть эволюцию гололедообразования).

Стратегия борьбы с гололедом на проводах аппаратурными методами состоит из нескольких этапов.

1. Сбор и отображение данных о гололедообразовании и о метеопараметрах.

2. Раннее обнаружение гололедообразования, определение направления его развития, сигнализация на диспетчерский пункт.

3. Прогноз изменения гололедной нагрузки после его образования.

4. Расчеты механических параметров линии с учетом их запаса прочности.

5. Определение необходимости плавки гололеда на ЛЭП при текущих и прогнозируемых климатических условиях. Определение длительности и рекомендуемой очередности плавки гололеда на ЛЭП энергорайона с учетом скорости нарастания гололеда и ответственности линии.

Реализация стратегии осуществляется техническим персоналом и автоматическими системами на ЛЭП.

К сожалению, вес провода с гололедными отложениями измеряется на отдельных

пролетах воздушной линии, в то время как гололед может образоваться и на других неконтролируемых пролетах линии, где он не будет обнаружен. Поэтому для повышения достоверности измерений необходимо увеличивать количество датчиков и устройств, передающих их показания на пункт управления, что является сложной технической задачей.

Другой недостаток весовых датчиков заключается в том, что они не являются универсальными. Они не могут использоваться без настройки для любого типа ЛЭП, имеющих различные параметры в зависимости от мощности самой линии (длина пролета между опорами, диаметр и количество проводов в фазе, количество и габариты изоляторов).

Но, несмотря на недостатки, этот метод из-за отсутствия других промышленно разработанных методов используется в ряде энергорайонов, как было показано в начале статьи.

Исследования по локационным методам обнаружения на линиях электропередачи гололедных отложений [4, 21, 22] и повреждений [3, 22–24] ведутся в Казанском государственном энергетическом университете (КГЭУ) более 15 лет с 1998 г. Как показывает анализ научно-технической литературы, ни в России, ни за рубежом подобные локационные устройства обнаружения гололеда на воздушных проводах ЛЭП не применяются.

Метод локационного зондирования заключается в подаче импульсного сигнала в линию и определении суммарного времени, затраченного на его распространение вдоль провода в прямом и обратном направлениях после отражения от конца линии либо от высокочастотного (ВЧ) заградителя.

При локационном способе обнаружения гололеда информацию о появлении гололеда несут импульсы, отраженные от любой неоднородности волнового сопротивления линии (реперной точки), имеющейся на ней. Неоднородностями являются концы линий или ответвлений от них, ВЧ заградители, места присоединения ответвлений к линии электропередачи, места соединения воздушных линий с кабельными вставками и т. д.

При образовании гололедных отложений увеличивается время прохождения $\Delta\tau$ отраженным импульсом заданного участка линии с одновременным уменьшением его амплитуды U.

Локационное устройство, осуществляющее зондирование ЛЭП, называется *рефлектометром*, функциональная схема которого представлена на рис. 1.

С генератора импульсов рефлектометра импульсы подаются в линию. Отраженные импульсы поступают с линии в приемник. С выхода приемника преобразованные сигналы поступают на графический индикатор. Все блоки рефлектометра функционируют по сигналам блока управления. На графическом индикаторе воспроизводится рефлектограмма линии – изображения от-

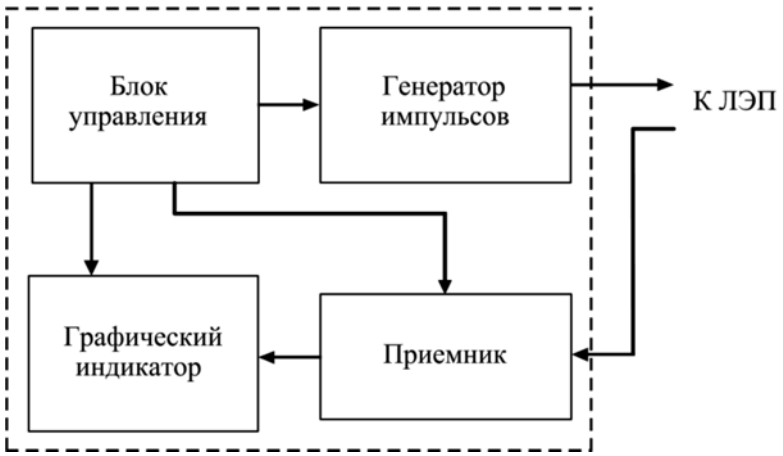


Рис. 1. Функциональная схема локационного устройства

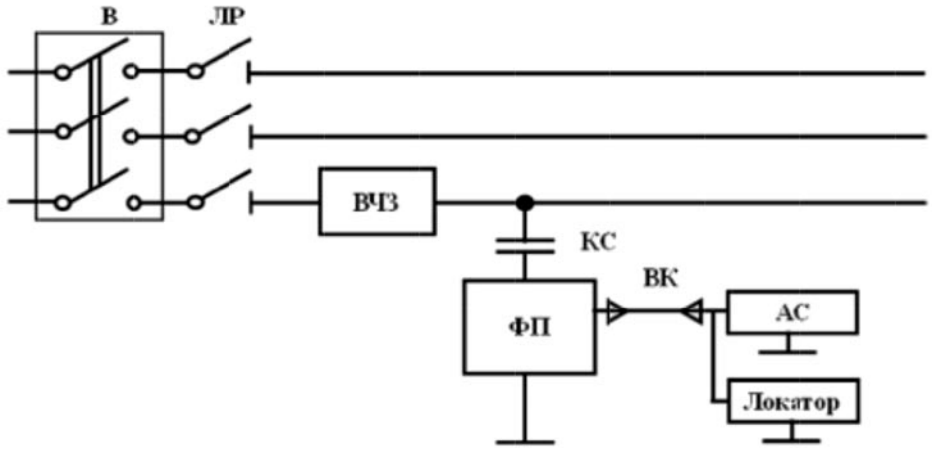


Рис. 2. Схема подключения локационного устройства к линии, имеющей элементы высокочастотной обработки

раженных импульсов на временной оси как реакция неоднородностей линии на зондирующий импульс.

Схема подключения локационного устройства (локатора) к фазному проводу ЛЭП с использованием оборудования высокочастотного тракта показана на рис. 2 [25]. На рисунке следующие обозначения: В – выключатель; ЛР – линейный разъединитель; ВЧЗ – высокочастотный заградитель; КС – конденсатор связи; ФП – фильтр присоединения; ВК – высокочастотный кабель; АС – аппаратура связи; локатор – локационное устройство.

Гололедные образования на проводах представляют собой неоднородный диэлектрик, уменьшающий скорость распространения сигнала вдоль линии и вызывающий его дополнительное затухание, обусловленное диэлектрическими потерями энергии электромагнитной волны, которая расходуется на нагрев слоя гололедного покрытия. Локационный метод позволяет определить появление гололедных образований на проводах ЛЭП путем сравнения времени распространения отраженных сигналов или их амплитуд при наличии и при отсутствии гололедных образований.

При зондировании линии импульсным локатором, упрощенная схема подключения которого к линии показана на рис. 3 *а*, совокупность отраженных импульсов образует рефлектограмму, изменяющуюся при появлении гололедных отложений на линии. Если из штатной (эталонной) рефлектограммы (рис. 3 *б* – сплошная линия) вычесть текущую рефлектограмму (рис. 3 *б* – пунктирная линия), то разностные изменения надежно обнаруживаются по появлению сигнала, соответствующего концу линии в точке Б (рис. 3 *в*). Чем больше волновое сопротивление линии будет изменяться под действием толщины гололедных отложений из-за изменения диэлектрической проницаемости между проводами линии, тем больше будет разность между рефлектограммами, тем больше будут увеличение задержки импульса $\Delta\tau$ и уменьшение амплитуды импульса ΔU (рис. 3 *б*).

При появлении гололедных отложений величины U и $\Delta\tau$ изменяются синхронно, как это видно на рис. 4 (отмечено штрихпунктирными овалами). Использование двух критериев повышает надежность обнаружения гололеда на проводах ЛЭП.

На отсчеты амплитуды U и запаздывания $\Delta\tau$ отраженного импульса кроме гололедных отложений могут влиять погодные условия, изменения температуры окружающей среды (пунктирная линия на рисунке, шкала температур с правой стороны), ветровые воздействия и т. д.

Для изучения особенностей обнаружения гололеда на ЛЭП сотрудниками КГЭУ разработан исследовательский автономный локационный комплекс для автоматического обнаружения гололедных отложений на ЛЭП по двум критериям: задержка импуль-

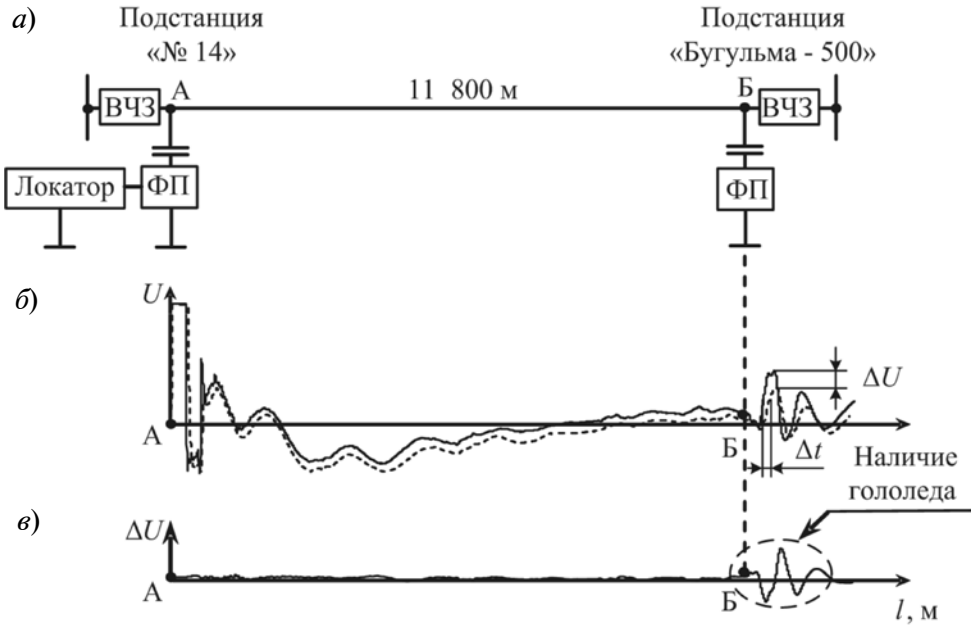


Рис. 3. Режим обнаружения гололеда на воздушной линии 110 кВ длиной 11 800 м между подстанциями «№ 14» и «Бугульма-500»:
а – схема линии; *б* – рефлектограммы линии без гололеда (——) и при наличии гололеда (·····); *в* – разность рефлектограмм линии без гололеда и при наличии гололеда с колебаниями сигнала в точке Б, обусловленными наличием гололедных отложений

са $\Delta\tau$ и уменьшение амплитуды импульса ΔU, как это показано на рис. 4.

Комплекс системы мониторинга гололеда осуществляет следующие операции [4, 21, 22]:

1) генерирование и ввод в линию зондирующих импульсов длительностью 1–10 мкс с напряжением 20–50 В;

2) прием импульсов, отраженных от конца линии;

3) выделение отраженных импульсов на фоне случайных помех и помех, вызванных работой систем связи, телемеханики, релейной защиты и автоматики;

4) определение толщины стенки гололедного отложения по задержке и уменьшению амплитуды отраженного импульса;

5) передачу информации на сервер оператора.

Измерения на линии производятся с периодичностью в 30 мин. Полученные данные передаются на центральный сервер, разме-

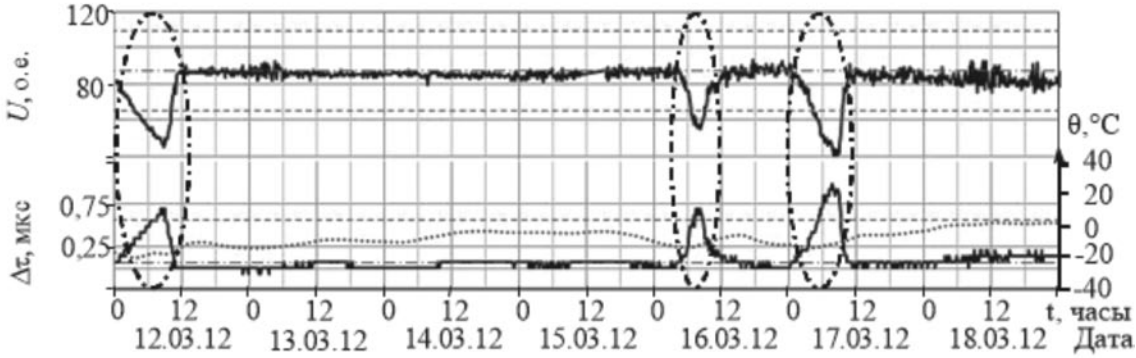


Рис. 4. Суточные изменения амплитуды *U* (верхний график) и запаздывания $\Delta\tau$ (нижний график) отраженных импульсов на ЛЭП 110 кВ «Кутлу Букаш–Рыбная Слобода»;
овалами обозначены регистрации гололедных образований

щенный на территории КГЭУ. Установленная на компьютере сервера программа снимает данные с центрального сервера КГЭУ и отображает графики зависимости измеренных значений $\Delta\tau$ и ΔU от времени. При образовании опасной массы гололеда на экране дисплея появляется предупреждающее сообщение, которое сопровождается звуковой сигнализацией. Программа, установленная на пункте управления (подстанция), также сигнализирует о появлении гололеда с указанием номера линии, если в составе комплекса имеется коммутатор линий.

Исполнительная часть комплекса осуществляет регистрацию и архивацию полученных сведений в виде протокола измерений. Протокол содержит в себе информацию о времени и дате измерений, о состоянии окружающей среды, изменении значений запаздывания $\Delta\tau$ и затухания ΔU импульса. Полная информация может быть представлена за разные периоды времени (сутки, неделя, месяц, квартал, год) в виде графиков зависимости исследуемого параметра от времени.

Для передачи и хранения данных с локационного комплекса на сервер КГЭУ (FTP сервер) используются модемы сотовой связи (GSM).

В настоящее время в автоматическом режиме круглосуточно функционируют четыре локационных комплекса для обнаружения гололеда, контролирующие:

- с 29 октября 2009 г. ЛЭП 110 кВ длиной 10 800 м «Бугульма №14—Бугульма-500», входящую в состав филиала ОАО «Сетевая компания» Бугульминские электрические сети;
- с 8 декабря 2009 г. ЛЭП 110 кВ длиной 40 300 м «Кутлу Букаш—Рыбная Слобода», входящую в состав филиала ОАО «Сетевая компания» Приволжские электрические сети;
- с 1 февраля 2013 г. ЛЭП 330 кВ длиной 63 800 м «Баксан—Прохладная 2», входящую в состав ОАО «МЭС Юга»;
- с 1 марта 2013 г. ЛЭП 110 кВ длиной 14 270 м «Шкапово—Чегодаево», входящую в состав ОАО «Башкирэнерго».

Все четыре системы обнаружения гололеда, упомянутые выше, передают информацию о состоянии ЛЭП на Центр управления в КГЭУ и образуют один общий комплекс обнаружения гололедных отложений, осуществляющий сбор, хранение и дополнительную обработку рефлектограмм.

Пример обнаружения изморозевых образований на проводах линии 110 кВ «Кутлу Букаш—Рыбная Слобода» по запаздыванию $\Delta\tau$ отраженных импульсов показан на рис. 5.

Результаты более длительных наблюдений в течение января 2013 г. за изменениями толщины стенки гололедных образований на проводах ЛЭП 110 кВ «Кутлу Букаш—Рыбная Слобода» представлены на

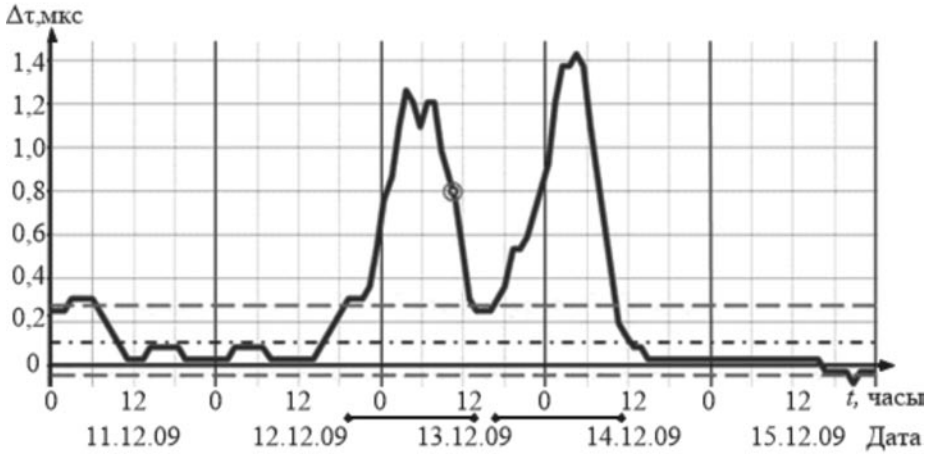


Рис. 5. Обнаружение изморозевых отложений на проводе линии электропередачи 110 кВ «Кутлу Букаш—Рыбная Слобода» по увеличению запаздывания отраженного импульса ($\Delta\tau > 0{,}3$ мкс) 11—15 декабря 2009 г.

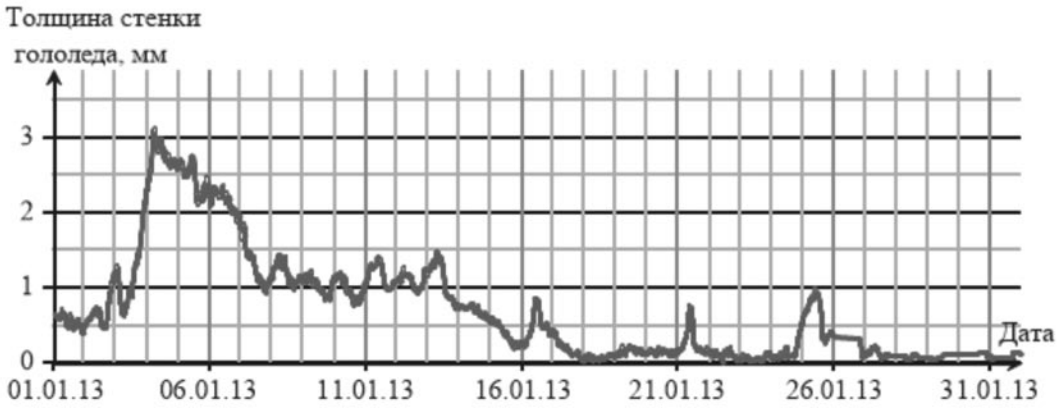


Рис. 6. Контроль локационным методом в течение месяца толщины стенки гололедных образований на проводах ЛЭП 110 кВ «Кутлу Букаш—Рыбная Слобода»

рис. 6. Максимум гололедного образования толщиной в 3 мм наблюдался 4 января 2013 г. Данные гололедные отложения не представляли угрозы целостности проводам воздушных линий, поэтому их плавка не производилась.

Итак, приведенные примеры убедительно характеризуют возможности локационного метода обнаружения гололедных отложений на проводах ЛЭП и подтверждают его высокую чувствительность, обеспечивающую раннее обнаружение гололеда, начиная с толщины стенки 0,5 мм и ниже, как это видно на рис. 6.

Недостатком классического локационного метода является невозможность отличить наличие небольшого по толщине гололедного образования на большой длине воздушной линии от опасной концентрации льда в отдельных ее пролетах. В этом случае применяется метод разбиения воздушной линии на отдельные локационные участки, для них определяется удельная толщина стенки гололедного отложения, по наибольшей величине которого принимается решение о необходимости плавки гололеда [26].

С 1 февраля 2013 г. проводились совместные сравнительные эксперименты по обнаружению гололеда локационным методом и методом взвешивания на ЛЭП 330 кВ «Баксан—Прохладная 2» (ОАО «МЭС Юга», Северный Кавказ) [27].

При локационном зондировании все возникшие на линии гололедные отложения были четко зафиксированы. Три тензометрических датчика, установленных на этой линии, обнаружили гололедные образования только вблизи от места их установки [27].

Можно считать, что локационный метод обнаружения гололеда является объективнее и информативнее метода взвешивания проводов. Определение места гололедного отложения, ведущего к аварии ЛЭП, осуществляется с использованием специальных мер, как было указано выше [26].

Аппаратура локационного зондирования обычно устанавливается около стойки высокочастотной связи, т. к. подключается к его выходной клемме параллельно с высокочастотным кабелем.

Локационный метод позволяет надежно следить в реальном времени за динамикой обледенения проводов и четко определять начало необходимой плавки гололедных отложений для предотвращения обрыва проводов электролиний и обусловленный этим недоотпуск электроэнергии потребителям. Метод позволяет следить за эффективностью плавки гололеда и дает возможность определять момент его своевременного прекращения при исчезновении опасности разрушения линии и обрыва проводов. Оптимизация времени плавки гололеда способствует энергосбережению и позволяет сэкономить значительные финансовые средства, т. к. плавка гололеда требует дорогостоящих энергетических затрат.

В то же время в некоторых ситуаци-

ях можно избежать плавки гололеда, если вес отложений будет меньше нормативной величины и начнется естественный сброс гололеда с проводов. При этом будет исключен недоотпуск электроэнергии потребителям из-за отключения линии на время плавки гололеда и сэкономлена электроэнергия, которая была бы израсходована для его плавки токами повышенного значения.

Локационный метод обнаружения гололеда имеет следующие преимущества перед методом взвешивания проводов:

1) вся аппаратура располагается около начала или конца линии электропередачи в производственных помещениях подстанции и не требует вмешательства в конструкцию ЛЭП, т. к. зондирующий импульсный сигнал одновременно выполняет функции датчика и носителя информации о гололедном отложении на проводе;

2) обеспечивается контроль всей линии, а не только одного пролета;

3) используется меньший, более простой и дешевый состав аппаратуры;

4) отсутствует угроза вандализма, т. к. локационное устройство располагается в помещении подстанции;

5) ввод в действие аппаратуры локационного зондирования занимает несколько минут, если ЛЭП имеет высокочастотную обработку;

6) имеется возможность периодического контроля с помощью коммутатора одним локационным устройством всех линий, отходящих с подстанции.

Локационное зондирование может осуществляться на ЛЭП, находящихся под напряжением и на отключенных линиях, а также на грозотросах, т. е. на любых металлических проводниках. Локационное устройство может функционировать при аварийном отключении питающего напряжения на ЛЭП за счет наличия собственного генератора зондирующих импульсов, на что не способны системы релейной защиты и автоматики.

Сигналы локационного зондирования не влияют на работу аппаратуры релейной защиты, противоаварийной автоматики, телемеханики и связи. В то же время при определенной цифровой обработке эти сигналы перестают быть помехами сигналам локационного зондирования.

Данные о повреждениях и гололедных отложениях могут передаваться через GSM канал или Интернет на рабочее место диспетчера без ограничения расстояния, обеспечивая в удобном интерфейсе наблюдение за динамикой гололедообразования на проводах ЛЭП и за динамикой освобождения проводов от гололедных покрытий при их плавке. Комплекс позволяет четко определять по времени начало необходимой плавки гололедных отложений. Своевременное обнаружение появления гололедно-изморозевых отложений является весьма актуальной проблемой для электроэнергетики нашей страны при решении задач энергосбережения.

Таким образом, разработан и введен в опытную эксплуатацию локационный комплекс по обнаружению гололедных образований на воздушных линиях электропередачи 110–330 кВ.

После рассмотрения трех используемых в настоящее время способов обнаружения гололедных отложений на проводах линий электропередачи можно утверждать, что способ прогнозирования является самым ненадежным и может применяться только из-за отсутствия диагностической аппаратуры с целью примерной оценки возможного появления гололедных образований.

Метод взвешивания проводов позволяет контролировать гололедные образования в пролетах той опоры, у которой установлен датчик. Этот участок линии должен быть характерным для всей линии электропередачи. Если гололед образуется на других участках линии, где нет датчиков, то он не будет обнаружен. Чтобы контролировать линию по всей длине необходимо установить датчики гололеда у каждой опоры. Такая задача, к сожалению, технически невыполнима.

Локационный метод дает возможность контролировать всю линию электропередачи и в реальном времени наблюдать процесс нарастания гололедных отложений и их сброс при плавке гололедной муфты. Метод является самым надежным и информативным из рассмотренных в данной статье.

В настоящее время аппаратура системы мониторинга линий электропередачи имеет несколько вариантов исполнения: настенный, мобильный, настольный и стоечный. Сотрудниками КГЭУ и ОАО «НПО «Радиоэлектроника» имени В.И. Шимко» по заказу ОАО «ФСК ЕЭС» разработан, изготовлен и испытан опытный образец системы мониторинга гололеда на 16 каналов. Намечены предприятия, готовые его тиражировать.

На сегодняшний день метод локационного обнаружения гололеда согласно информационному поиску с глубиной в 40 лет и прошедшей в июне 2013 г. в Канаде XV конференции IWAIS [28] нигде в мире не реализован, наши исследования и их результаты не имеют аналогов в мире и являются уникальными.

В данной статье рассмотрены методы и принципы обнаружения гололеда на проводах воздушных ЛЭП. Особое внимание уделено методу локационного зондирования, как наиболее перспективному из них. Но остались не исследованными вопросы предельной чувствительности локационного метода и временной стабильности параметров высокочастотного канала ЛЭП, определяющие процедуру выбора уставок (порога срабатывания) по амплитуде и запаздыванию отраженных сигналов при раннем обнаружении гололедных отложений на проводах ЛЭП. Результаты этих исследований описаны во второй части данной статьи.

Исследования и разработка аппаратуры обнаружения гололеда на линиях электропередачи выполнены при финансовой поддержке ОАО «Сетевая компания» (Татарстан), АН Республики Татарстан и ОАО «ФСК ЕЭС».

СПИСОК ЛИТЕРАТУРЫ

1. **Шалыт Г.М.** Определение мест повреждений линий электропередачи импульсным методом. М.: Энергия, 1968. 216 с.

2. **Яковлев Л.В.** Комплексные методы и устройства для защиты проводов и грозозащитных тросов воздушных линий от вибрации, «пляски» и гололедообразования // Энергетик. 2004. № 3. С. 15–17.

3. **Минуллин Р.Г., Фардиев И.Ш.** Локационная диагностика воздушных линий электропередачи. Казань: Изд-во КГЭУ, 2008. 202 с.

4. **Минуллин Р.Г. и др.** Обнаружение гололедных образований на линиях электропередачи локационным зондированием. Казань: Изд-во КГЭУ, 2010. 207 с.

5. System for prediction and monitoring of ice shedding, antiicing and de-icing for overhead lines. CIGRE` Working Group B2.29, 2009.

6. **Olafsson H., Eliasson A.J., Thorsteins E.** Orographic influence on wet-snow icing. Part 2: Downstream of mountains // Proc. 10th Internat. Workshop on Atmospheric Icing of Structures. Brno, Check Republic. 2002. Paper 2-3.

7. **Farzaneh M.** Atmospheric Icing of Power Networks. Springer Science, 2008.

8. **Olafsson H., Eliasson A.J., Thorsteins E.** Orographic influence on wet-snow icing, Part 1: Upstream of mountains //Proc. 10th Internat. Workshop on Atmospheric Icing of Structures. Brno, Check Republic. 2002. Paper 2-2.

9. **Fikke S.** Cost Action 727. Measuring and forecasting atmospheric icing on structures // Proc. 11th Internat. Workshop on Atmospheric Icing of Structures. Montreal, Canada. 2005. Paper IW64.

10. **Fikke S.** Modern meteorology and atmospheric icing // Proc. 11th Internat. Workshop on Atmospheric Icing of Structures. Montreal, Canada. 2005. Paper IW73.

11. **Gland H., Admirat P.** Meteorological conditions for wet snow occurrence in France. Calculated and measured results in a recent case study on March 5th, 1985 // Proc. 3rd Internat. Workshop on Atmospheric Icing of Structures. Vancouver, Canada. 1986.

12. **Toth K., Lakatos M., Kollath K., Fulop R., Simon A.** Climatology and forecasting of severe wet snow icing in Hungary // Proc. 13th Internat. Workshop on Atmospheric Icing of Structures. Andermatt, Switzerland. 2009.

13. **Wareing B.J., Nygaard, B.E.** WRF Simulation of wet snow and rime icing incidents in the UK // Proc. 13th Internat. Workshop on Atmospheric Icing of Structures. Andermatt, Switzerland. 2009.

14. **Makkonen L.** Models for the growth of rime, glaze, icicles andwet-snow on structures // Phil. Trans. R. Soc. London, UK. 2000. No. A358. Pp. 2913–2939.

15. **Sakamoto Y.** Snow accretion on overhead wires //Phil. Trans. R. Soc. London, UK. 2000. No. 358(1776). Pp. 2941–2970.

16. **Bonelli P., Lacavalla M.** Experimental activity and investigation of wet-snow accretion on

overhead power lines in Italy // Proc. 13th Internat. Workshop on Atmospheric Iicing of Structures. Andermatt, Switzerland. 2009.

17. **Дьяков А.Ф., Левченко И.И., Засыпкин А.С. и др.** Информационная система контроля гололедообразования на воздушных линиях электропередачи // Энергетик. 2005. № 11. С. 20–25.

18. **Левченко И.И, Засыпкин А.С., Аллилуев А.А., Сацук Е.И.** Диагностика, реконструкция и эксплуатация воздушных линий электропередачи в гололедных районах. Учеб. пособие. М.: ИД МЭИ, 2007. 445 с.

19. **Левченко И.И., Сацук Е.И.** Система прогнозирования и контроля гололедообразования // Электроэнергия. Передача и распределение. 2011. № 1. С. 14–18.

20. **Костиков И.** Система мониторинга «САТ-1» – эффективная защита ВЛЭП от гололеда // Электроэнергия. Передача и распределение. 2011. № 1. С. 32–35.

21. **Минуллин Р.Г., Губаев Д.Ф.** Критерии и индикаторы обнаружения гололеда на линиях электропередачи при локационном зондировании // Научно-технические ведомости Санкт-Петербургского государственного политехнического университета. Информатика. Телекоммуникации. Управление. СПб.: Изд-во СПбГПУ, 2009. № 4 (82). С. 191–197.

22. **Минуллин Р.Г.** Обнаружение гололеда и повреждений на воздушных линиях электропередачи локационным методом //Энергетика Татарстана. 2011. № 2 (22). С. 15–17.

23. **Минуллин Р.Г., Закамский Е.В.** Определение мест повреждения в электрических сетях напряжением 6-35 кВ импульсным методом // Доклады Российского национального симп. по энергетике, III Междунар. симп. по энергетике, окружающей среде и экономике. Казань: Изд-во КГЭУ, 2001. Т. 2. С. 62–64.

24. **Минуллин Р.Г., Петрушенко Ю.Я., Фардиев И.Ш.** Зондирование воздушных линий электропередачи локационным методом // Электротехника. 2008. № 7. С. 42–50.

25. **Минуллин Р.Г., Фардиев И.Ш., Лукин Э.И. и др.** Особенности подключения рефлектометра к линиям электропередачи при локационном зондировании // Электротехника. 2008. № 2. С. 34–43.

26. **Касимов В.А., Минуллин Р.Г., Яруллин М.Р.** Методы расчета толщины стенки гололеда на проводах линий электропередачи при локационном зондировании // Научные труды IV Междунар. науч.-технич. конф. Энергетика глазами молодежи. Новочеркасск: Изд-во ЮРГПУ (НПИ), 2013. Т. 1. С. 488–492.

27. **Минуллин Р.Г., Касимов В.А., Яруллин М.Р., Аскаров Р.Р., Губаренко В.П.** Сравнение систем обнаружения гололеда на линиях электропередачи, использующих методы взвешивания проводов и локационного зондирования // Научные труды IV Междунар. науч.-технич. конф. Энергетика глазами молодежи. Новочеркасск: Изд-во ЮРГПУ (НПИ), 2013. Т. 1. С. 514–518.

28. Proc. 15th International Workshop on Atmospheric Icing of Structures. St. John's, Canada. 2013.

REFERENCES

1. **Shalyt G.M.** *Opredeleniye mest povrezhdeniy liniy elektroperedachi impulsnym metodom*. Moscow: Energiya Publ., 1968, 216 p. (rus)

2. **Yakovlev L.V.** Kompleksnyye metody i ustroystva dlya zashchity provodov i gozozashchitnykh trosov vozdushnykh liniy ot vibratsii, «plyaski» i gololedoobrazovaniya, *Energetik*, 2004, No. 3, Pp. 15–17. (rus)

3. **Minullin R.G., Fardiyev I.Sh.** *Lokatsionnaya diagnostika vozdushnykh liniy elektroperedachi*. Kazan: KGEU Publ., 2008, 202 p. (rus)

4. **Minullin R.G. i dr.** *Obnaruzheniye gololednykh obrazovaniy na liniyakh elektroperedachi lokatsionnym zondirovaniyem*. Kazan: KGEU Publ., 2010, 207 p. (rus)

5. *System for prediction and monitoring of ice shedding, antiicing and de-icing for overhead lines.* CIGRE` Working Group B2.29, 2009.

6. **Olafsson H., Eliasson A.J., Thorsteins E.** Orographic influence on wet-snow icing. Part 2: Downstream of mountains, *Proc. 10th International Workshop on Atmospheric Icing of Structures*, Brno, Check Republic, 2002, Paper 2-3.

7. **Farzaneh M.** *Atmospheric Icing of Power Networks*, Springer Science, 2008.

8. **Olafsson H., Eliasson A.J., Thorsteins E.** Orographic influence on wet-snow icing, Part 1: Upstream of mountains, *Proc. 10th International Workshop on Atmospheric Icing of Structures*, Brno, Check Republic, 2002, Paper 2-2.

9. **Fikke S.** Cost Action 727. Measuring and forecasting atmospheric icing on structures, *Proc. 11th International Workshop on Atmospheric Icing of Structures*, Montreal, Canada, 2005, Paper IW64.

10. **Fikke S.** Modern meteorology and atmospheric icing, *Proc. 11th International Workshop on Atmospheric Icing of Structures.* Montreal, Canada, 2005. Paper IW73.

11. **Gland H., Admirat P.** Meteorological

conditions for wet snow occurrence in France. Calculated and measured results in a recent case study on March 5th, 1985, *Proc. 3rd International Workshop on Atmospheric Icing of Structures*, Vancouver, Canada, 1986.

12. **Toth K., Lakatos M., Kollath K., Fulop R., Simon A.** Climatology and forecasting of severe wet snow icing in Hungary, *Proc. 13th International Workshop on Atmospheric Icing of Structures*, Andermatt, Switzerland, 2009.

13. **Wareing B.J., Nygaard, B.E.** WRF Simulation of wet snow and rime icing incidents in the UK, *Proc. 13th International Workshop on Atmospheric Icing of Structures*, Andermatt, Switzerland, 2009.

14. **Makkonen L.** Models for the growth of rime, glaze, icicles andwet-snow on structures, *Phil. Trans. R. Soc*, London, UK, 2000, No. A358, Pp. 2913–2939.

15. **Sakamoto Y.** Snow accretion on overhead wires, *Phil. Trans. R. Soc*, London, UK, 2000, No. 358(1776), Pp. 2941–2970.

16. **Bonelli P., Lacavalla M.** Experimental activity and investigation of wet-snow accretion on overhead power lines in Italy, *Proc. 13th International Workshop on Atmospheric Iicing of Structures*, Andermatt, Switzerland, 2009.

17. **Dyakov A.F., Levchenko I.I., Zasypkin A.S. i dr.** Informatsionnaya sistema kontrolya gololedoobrazovaniya na vozdushnykh liniyakh elektroperedachi, *Energetik*, 2005, No. 11, Pp. 20–25. (rus)

18. **Levchenko I.I, Zasypkin A.S., Alliluyev A.A., Satsuk Ye.I.** *Diagnostika, rekonstruktsiya i ekspluatatsiya vozdushnykh liniy elektroperedachi v gololednykh rayonakh*, Ucheb. posobiye, Moscow: ID MEI Publ., 2007, 445 p. (rus)

19. **Levchenko I.I., Satsuk Ye.I.** Sistema prognozirovaniya i kontrolya gololedoobrazovaniya, *Elektroenergiya. Peredacha i raspredeleniye,* 2011, No. 1, Pp. 14–18. (rus)

20. **Kostikov I.** Sistema monitoringa «SAT-1» – effektivnaya zashchita VLEP ot gololeda, *Elektroenergiya. Peredacha i raspredeleniye*, 2011, No. 1, Pp. 32–35. (rus)

21. **Minullin R.G., Gubaev D.F.** Kriterii i indikatory obnaruzheniya gololeda na liniyakh elektroperedachi pri lokatsionnom zondirovanii, *Nauchno-tekhnicheskiye vedomosti SPbGPU. Informatika Telekommunikatsii. Upravleniye*, St. Petersburg: SPbGPU Publ., 2009, No. 4 (82), Pp. 191–197. (rus)

22. **Minullin R.G.** Obnaruzheniye gololeda i povrezhdeniy na vozdushnykh liniyakh elektroperedachi lokatsionnym metodom, *Energetika Tatarstana*, 2011, No. 2 (22), Pp. 15–17. (rus)

23. **Minullin R.G., Zakamskiy Ye.V.** Opredeleniye mest povrezhdeniya v elektricheskikh setyakh napryazheniyem 6-35 kV impulsnym metodom, *Doklady Rossiyskogo natsionalnogo simpoziuma po energetike, III Mezhdunarodnogo simpoziuma po energetike, okruzhayushchey srede i ekonomike*, Kazan: KGEU Publ., 2001, Vol. 2, Pp. 62–64. (rus)

24. **Minullin R.G., Petrushenko Yu.Ya., Fardiyev I.Sh.** Zondirovaniye vozdushnykh liniy elektroperedachi lokatsionnym metodom, *Elektrotekhnika,* 2008, No. 7, Pp. 42–50. (rus)

25. **Minullin R.G., Fardiyev I.Sh., Lukin E.I. i dr.** Osobennosti podklyucheniya reflektometra k liniyam elektroperedachi pri lokatsionnom zondirovanii, *Elektrotekhnika,* 2008, No. 2, Pp. 34–43. (rus)

26. **Kasimov V.A., Minullin R.G., Yarullin M.R.** Metody rascheta tolshchiny stenki gololeda na provodakh liniy elektroperedachi pri lokatsionnom zondirovanii, *Nauchnyye trudy IV Mezhdunarodnoy nauchno-tekhnicheskoy konferentsii Energetika glazami molodezhi*, Novocherkassk: YuRGPU (NPI), 2013, Vol. 1, Pp. 488–492. (rus)

27. **Minullin R.G., Kasimov V.A., Yarullin M.R., Askarov R.R., Gubarenko V.P.** Sravneniye sistem obnaruzheniya gololeda na liniyakh elektroperedachi, ispolzuyushchikh metody vzveshivaniya provodov i lokatsionnogo zondirovaniya, *Nauchnyye trudy IV Mezhdunarodnoy nauchno-tekhnicheskoy konferentsii Energetika glazami molodezhi*, Novocherkassk: YuRGPU (NPI) Publ., 2013, Vol. 1, Pp. 514–518.

28. *Proc. 15th International Workshop on Atmospheric Icing of Structures*, St. John's, Canada, 2013.

МИНУЛЛИН Ренат Гизатуллович – *заведующий научно-исследовательской лабораторией локационной диагностики состояния линий электропередачи Казанского государственного энергетического университета.*

420066, Россия, г. Казань, ул. Красносельская, д. 51.

E-mail: Minullin@mail.ru

MINULLIN, Renat G. *Kazan State Power Engineering University.*

420066, Krasnoselskaya Str. 51, Kazan, Russia.

E-mail: Minullin@mail.ru

КАСИМОВ Василь Амирович – *аспирант научно-исследовательской лаборатории локационной диагностики состояния линий электропередачи Казанского государственного энергетического университета.*
420066, Россия, г. Казань, ул. Красносельская, д. 51.
E-mail: VasilKasimov@yandex.ru

KASIMOV, Vasil A. *Kazan State Power Engineering University.*
420066, Krasnoselskaya Str. 51, Kazan, Russia.
E-mail: VasilKasimov@yandex.ru

ФИЛИМОНОВА Тамара Константиновна – *доцент кафедры инженерной кибернетики Института экономики и информационных технологий Казанского государственного энергетического университета.*
420066, Россия, г. Казань, ул. Красносельская, д. 51.
E-mail: Filimonova.Tamara@bk.ru

FILIMONOVA, Tamara K. *Kazan State Power Engineering University.*
420066, Krasnoselskaya Str. 51, Kazan, Russia.
E-mail: Filimonova.Tamara@bk.ru

ЯРУЛЛИН Марсель Рашитович – *аспирант научно-исследовательской лаборатории локационной диагностики состояния линий электропередачи Казанского государственного энергетического университета.*
420066, Россия, г. Казань, ул. Красносельская, д. 51.
E-mail: Marsel.Jarullin@gmail.com

YARULLIN, Marsel R. *Kazan State Power Engineering University.*
420066, Krasnoselskaya Str. 51, Kazan, Russia.
E-mail: Marsel.Jarullin@gmail.com

УДК 621.315.1, 621.372.2

Р.Г. Минуллин, В.А. Касимов, Т.К. Филимонова, М.Р. Яруллин

ЛОКАЦИОННОЕ ОБНАРУЖЕНИЕ ГОЛОЛЕДА НА ВОЗДУШНЫХ ЛИНИЯХ ЭЛЕКТРОПЕРЕДАЧИ. ЧАСТЬ 2. ПРЕДЕЛЬНАЯ ЧУВСТВИТЕЛЬНОСТЬ И ВЫБОР УСТАВОК

R.G. Minullin, V.A. Kasimov, T.K. Filimonova, M.R. Yarullin

LOCATION DETECTION OF GLAZE ICE ON OVERHEAD ELECTRIC POWER LINES. PART 2. NOISE-LIMITED SENSITIVITY AND OPERATION THRESHOLD SETTING IN DETECTING ICE ACCRETION BY LOCATION PROBING

Исследованы стабильность амплитуды и запаздывания отраженных сигналов в штатных условиях при отсутствии гололеда и предельная чувствительность локационного метода обнаружения гололеда. В результате многолетних измерений на действующих линиях электропередачи (ЛЭП) обнаружены суточно-годовые вариации амплитуды и запаздывания отраженных сигналов. Показано, что эти вариации в совокупности со случайными флуктуациями в штатном режиме намного меньше изменений амплитуды и запаздывания сигналов при раннем появлении гололедных отложений. Даны рекомендации о выборе значений уставок (порогов срабатывания) по амплитуде и запаздыванию при обнаружении гололеда и повреждений на ЛЭП локационным методом.

ЛИНИИ ЭЛЕКТРОПЕРЕДАЧИ; ЛОКАЦИОННОЕ ЗОНДИРОВАНИЕ; АМПЛИТУДА И ЗАПАЗДЫВАНИЕ ОТРАЖЕННЫХ СИГНАЛОВ; ОБНАРУЖЕНИЕ ПОВРЕЖДЕНИЙ И ГОЛОЛЕДА НА ПРОВОДАХ; ЧУВСТВИТЕЛЬНОСТЬ ЛОКАЦИОННОГО МЕТОДА; ВЫБОР ПОРОГА СРАБАТЫВАНИЯ (УСТАВКИ) ПО АМПЛИТУДЕ И ЗАПАЗДЫВАНИЮ.

The authors study the threshold sensitivity of detecting ice accretion by location probing and stability of amplitude and transmission delay of reflected signals in the normal operation mode in the absence of ice. Due to long-term investigations of the active power lines annual and daily variations were discovered in amplitude and transmission delay of reflected signals. It was shown that these variations as well as sporadic fluctuations in the normal operation mode are far less than the changes in signal amplitude and transmission delay at the early stage of ice accumulation. Guidelines to set operation threshold for detecting early ice accretion on electric power lines and wire breakage by amplitude and transmission delay are given.

ELECTRIC POWER LINES; LOCATION MONITORING; AMPLITUDE AND TRANSMISSION DELAY OF REFLECTED SIGNALS; DETECTING ICE ACCRETION AND WIRE BREAKAGE; LOCATION METHOD SENSITIVITY; SETTING OPERATION THRESHOLD FOR AMPLITUDE AND TRANSMISSION DELAY.

В первой части статьи описаны используемые в настоящее время способы обнаружения гололеда на проводах воздушных линий электропередачи, отмечены их достоинства и недостатки [1]. Среди них подробно рассмотрен локационный метод, как наиболее прогрессивный по своим возможностям [2]. В данной части статьи приводятся результаты многолетних экспериментальных исследований на действующих линиях электропередачи локационным методом флуктуаций амплитуды и запаз-

дывания отраженных сигналов в штатных условиях при отсутствии гололедных образований на проводах. Эти флуктуации определяют предельную чувствительность локационного метода при зондировании линий электропередачи с целью раннего обнаружения появления гололедных отложений на проводах. С учетом предела этих флюктуаций для каждой линии электропередачи определяются уставки (пороги срабатывания аппаратуры) для обнаружения и последующего отслеживания динамики образования появившихся гололедных отложений.

Показателем текущего состояния проводов ЛЭП является рефлектограмма – реакция линии во времени на зондирующий локационный импульс [3, 4]. При анализе рефлектограммы исследуются по отношению к эталонному сигналу изменения амплитуды U и запаздывания $\Delta\tau$ отраженного импульсного сигнала (рис. 1), по которым определяются величина гололедных отложений на проводах линий электропередачи, а также их повреждения (обрыв, короткое замыкание) [3–6].

В статье описываются результаты исследований предельной чувствительности метода локационного зондирования и стабильности каналов высокочастотной (ВЧ) связи линий электропередачи путем анализа изменений *амплитуды* U и *запаздывания* $\Delta\tau$ отраженных локационных импульсов во времени в штатных условиях при отсутствии гололеда на проводах и при отсутствии сигналов аппаратуры релейной защиты, противоаварийной автоматики, телемеханики и связи в ВЧ тракте. Необходимо установить следующее: не окажутся ли из-за нестабильности рефлектограмм величины случайных вариаций U и $\Delta\tau$ больше изменений этих параметров при появлении гололедных отложений на проводах ЛЭП и не будут ли они мешать обнаружению начала образования гололедной муфты и слежению за последующей динамикой ее увеличения.

Кроме того, флуктуации значений амплитуды ΔU и запаздывания $\Delta\tau$ импульсных сигналов, происходящие в штатных условиях, необходимо учитывать при выборе уставок (порога) для своевременного обнаружения гололеда и повреждений на проводах линий электропередачи.

Результаты исследований лежат в основе технологии локационного обнаружения раннего гололеда и повреждений на проводах ЛЭП.

Исследования особенностей применения локационного зондирования для обнаружения гололедных образований и повреждений на проводах воздушных ЛЭП ведутся сотрудниками Казанского государственного энергетического университета (КГЭУ) более 15 лет и не имеют аналогов в мировой практике. С помощью изготовленного сотрудниками КГЭУ исследовательского локационного комплекса в течение четырех лет с 2009 по 2014 гг. осуществляется непрерывный мониторинг линий 110 кВ «Кутлу Букаш–Рыбная Слобода–Камская» и «Бугульма №14–Бугульма 500», а также линий 110 кВ «Кутлу Букаш–Кулущи», «Кутлу Букаш–Ныpты», «Кутлу Букаш–

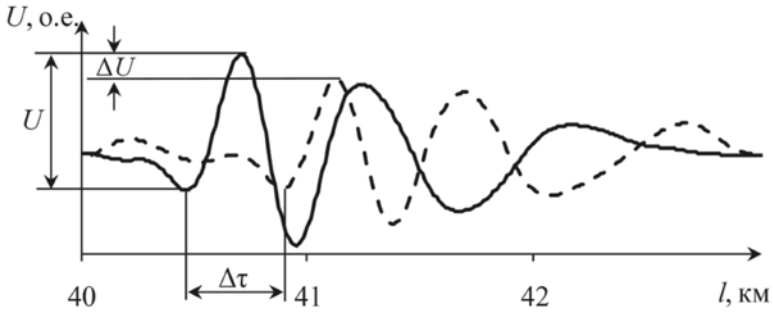


Рис. 1. Отсчеты уменьшения ΔU амплитуды и увеличения запаздывания $\Delta\tau$ отраженного локационного сигнала (- - - -) по отношению к эталонному сигналу (——)

Богатые Сабы» (ОАО «Сетевая компания») и др. [3–8].

Рефлектограммы регистрируются в круглосуточном режиме через 30 мин, при этом отсчитываются величины изменений амплитуды ΔU и дополнительного запаздывания $\Delta\tau$ отраженных сигналов локационного зондирования. За время наблюдений было снято свыше 50 000 рефлектограмм при различных временных и погодных условиях.

Гололедные отложения при локационном зондировании ЛЭП обнаруживаются по дополнительному уменьшению амплитуды ΔU и дополнительному увеличению времени прихода $\Delta\tau$ (запаздыванию) отраженного импульсного сигнала по отношению к эталонному, как это показано на рис. 1.

ΔU и $\Delta\tau$ – это критерии обнаружения образования гололедных отложений. Уменьшение амплитуды импульса ΔU обусловлено потерями энергии импульса в несовершенном диэлектрике, каковым является образовавшееся гололедное покрытие. Запаздывание $\Delta\tau$ импульса возникает из-за увеличения времени прохождения сигнала в контролируемой ЛЭП за счет уменьшения скорости его распространения при появлении гололедного образования. Изменения амплитуды ΔU и времени распространения $\Delta\tau$ импульсного сигнала определяются путем сравнения текущей рефлектограммы линии с ее эталонным аналогом, который хранится в памяти компьютера (см. рис. 1) [5–8].

В штатной ситуации при отсутствии гололеда на проводах ЛЭП нестабильности U и $\Delta\tau$ зависят от механического удлинения проводов ЛЭП под влиянием увеличения температуры окружающей среды, солнечного нагрева, существующей нагрузки ЛЭП, порывов ветра. На нестабильность параметров U и $\Delta\tau$ влияют погодные условия в виде тумана, мороси, дождя и снега. Кроме того, флуктуации отсчетов значений U и $\Delta\tau$ возникают под влиянием шумов и помех, постоянно присутствующих в ВЧ канале ЛЭП [9].

В настоящее время нет расчетных методик, учитывающих временные изменения значений U и $\Delta\tau$ для отраженных локационных импульсов, распространяющихся в ВЧ тракте ЛЭП. Поэтому нестабильность этих параметров определим экспериментально.

На рис. 2 показаны суточные вариации параметров отраженных импульсных сигналов, измеренные в штатном режиме через 30 мин в течение трех суток с 21 по 23 июля 2011 г. на линии «Кутлу Букаш–Богатые Сабы»: *а* – амплитуды U в относительных единицах (о. е.); *б* – запаздывания $\Delta\tau$ в мкс.

На приведенных графиках хорошо видны периодические суточные изменения (тренд) текущих значений $U(t)$ и $\Delta\tau(t)$: в полдень значения U минимальны, а $\Delta\tau$ – максимальны, в полночь – наоборот. Эти полуденные уменьшения U обусловлены, видимо, увеличением сопротивления проводов, а увеличения $\Delta\tau$ – удлинением проводов при увеличении температуры окружающей среды, что имеет место в полдень. На линии «Кутлу Букаш–Богатые

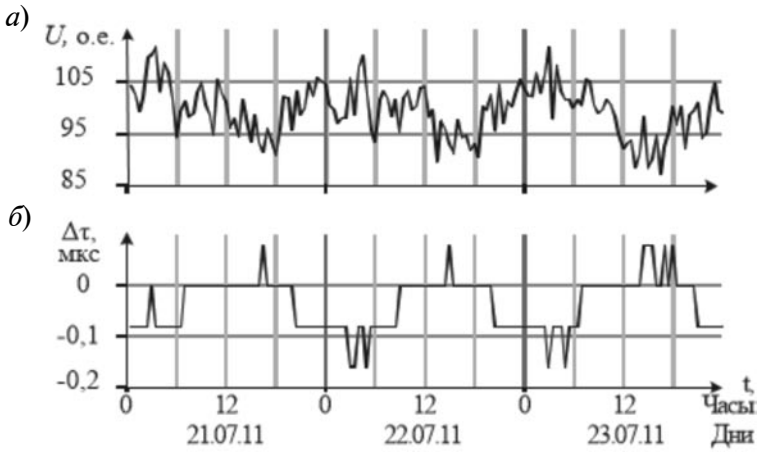


Рис. 2. Суточные изменения амплитуды $U(t)$ (*а*) и запаздывания $\Delta\tau(t)$ (*б*) отраженных сигналов

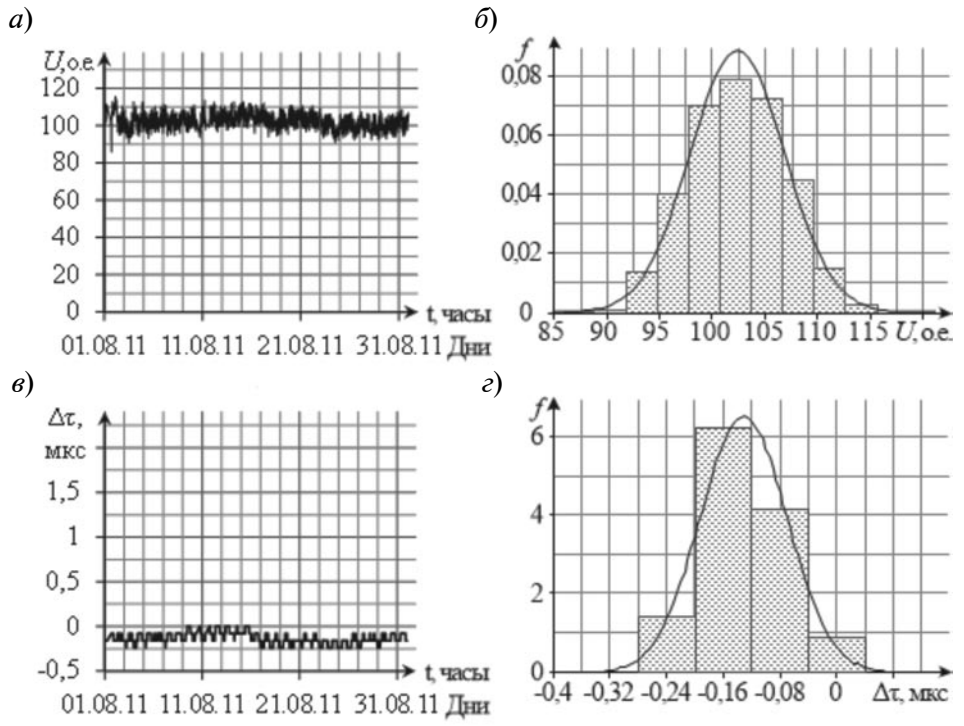


Рис. 3. Текущие значения амплитуд $U(t)$ (*а*) и запаздываний $\Delta\tau(t)$ (*в*) отраженных сигналов и соответствующие им гистограммы распределений плотностей вероятностей $f_Э(U)$ (*б*) и $f_Э(\Delta\tau)$ (*г*)

Сабы» максимальный разброс суточных изменений U от штатного состояния составил примерно $\pm$ 10 %, а $\Delta\tau$ изменялось в пределах $\pm$ 0,08 мкс (шаг квантования).

На суточный тренд $U(t)$ и $\Delta\tau(t)$ накладываются хаотические флуктуации этих значений, которые, видимо, обусловлены шумами и помехами, постоянно присутствующими в ВЧ тракте ЛЭП. Поэтому случайные флуктуации $U(t)$ и $\Delta\tau(t)$ отраженных сигналов, вероятно, подчиняются, подобно белому шуму, закону Гаусса или близки к этому.

Для проверки этого положения исследовались месячные массивы текущих значений U и $\Delta\tau$, измеренных через полчаса за три года наблюдений с 2010 по 2012 гг. на четырех упомянутых выше линиях подстанции «Кутлу Букаш». В качестве примера графики таких сигналов $U(t)$ и $\Delta\tau(t)$ за август 2011 г. для линии 110 кВ «Кутлу Букаш–Рыбная Слобода» показаны на рис. 3 *а* и *в* соответственно.

Месячные массивы текущих значений U и $\Delta\tau$ представлены в виде дифференциальных распределений $P(U)$ и $P(\Delta\tau)$. Примеры экспериментальных распределений плотностей вероятностей $f_Э(U)$ и $f_Э(\Delta\tau)$ в виде гистограмм за август 2011 г. для линии «Кутлу Букаш–Рыбная Слобода» приведены на рис. 3 *б* и *г* соответственно, где сплошной линией нанесены Гауссовские распределения плотностей вероятностей $f_P(U)$ и $f_P(\Delta\tau)$, рассчитанные с использованием математического ожидания $U_{ср}$ и $\Delta\tau_{ср}$, а также среднеквадратических значений σ_U и $\sigma_{\Delta\tau}$ соответствующих гистограмм.

Эмпирические распределения $P(U)$ и $P(\Delta\tau)$ были подвергнуты детальной статистической обработке с использованием пакета STATISTICA [10] для установления вида закона распределения. Для месячных распределений $P(U)$ и $P(\Delta\tau)$ были подсчитаны средние значения $U_{ср}$ и $\Delta\tau_{ср}$, среднеквадратические значения σ_U и $\sigma_{\Delta\tau}$, коэффициенты асимметрии A_U и $A_{\Delta\tau}$, коэффициенты эксцесса E_U и $E_{\Delta\tau}$, а также σ_A, σ_E и отношения $\frac{|A|}{\sigma_A}$. Если A < 0,5 и $\frac{|A|}{\sigma_A} < 3$, то можно считать,

что распределение соответствует Гауссовскому закону.

Выполненная оценка показателей A и E для эмпирических распределений $P(U)$ и $P(\Delta\tau)$ с использованием пакета STATISTICA [10] и статистическая проверка $P(U)$ и $P(\Delta\tau)$ с помощью критериев согласия Пирсона (χ^2) и Колмогорова–Смирнова [11] всего массива данных, полученных на четырех линиях подстанции «Кутлу Букаш», позволяют сделать вывод, что 80 % распределений $P(U)$ и $P(\Delta\tau)$ соответствуют закону Гаусса, а остальная часть близка к нему. Поэтому с большой степенью достоверности можно считать, что 99,7 % флуктуаций значений U и $\Delta\tau$ в штатных условиях при отсутствии гололедных отложений на проводах ЛЭП находятся в пределах доверительных интервалов, соответствующих уровню $\pm 3\sigma$ относительно среднего значения.

Конечные результаты исследования распределений $P(U)$ и $P(\Delta\tau)$ сведены в таблицу, где представлены средние и среднеквадратические значения для амплитуд и запаздываний отраженных сигналов, усредненные за интервал наблюдений на четырех линиях подстанции «Кутлу Букаш».

Согласно анализу экспериментальных данных значения U и $\Delta\tau$ в штатных условиях при отсутствии гололеда имеют достаточно высокую стабильность: флуктуации амплитуды U находятся в пределах примерно $\pm$ 10 % от среднего значения при вероятности 0,997, флуктуации $\Delta\tau$ в среднем не превышают примерно $\pm$ 0,2 мкс ($\pm$ 30 м) также при вероятности 0,997.

Кроме того, локационный метод измерений $\Delta\tau$ обладает высокой чувствительностью, т. к. измерения осуществляются с погрешностью, не превышающей $\pm$ 0,08 мкс, (интервал квантования АЦП равен 0,08 мкс), что соответствует при локационном зондировании изменению длины линии на $\pm$ 12 м ($\pm$ 0,03 %), это в несколько раз меньше длины пролета ЛЭП.

Значения $U_{ср}$ и $\Delta\tau_{ср}$, а также σ_U и $\sigma_{\Delta\tau}$ для разных воздушных линий близки друг к другу, как это видно из таблицы, и мало зависят от года измерений. Эти значения можно брать как стабильные исходные параметры отраженных сигналов локационного зондирования ВЧ трактов ЛЭП при определении уставок по амплитуде и запаздыванию для обнаружения гололедных отложений на проводах ЛЭП.

При появлении гололедных отложений текущие значения U уменьшаются на десятки процентов, а $\Delta\tau$ увеличиваются во много раз и надежно обнаруживаются на фоне случайных флуктуаций данных параметров.

Изменения значений U и $\Delta\tau$ отраженных сигналов в течение суток имеют периодический характер, как это видно на рис. 2. Если эти изменения U и $\Delta\tau$ обусловлены удлинением проводов при повышении температуры окружающей среды в полдень, то следует ожидать, что они будут иметь и периодические годовые вариации в сумме с вариациями, которые будут обусловлены периодическими ветровыми нагрузками.

Параметры амплитуд и запаздываний отраженных сигналов с усреднением за указанные в таблице периоды измерений для воздушных линий разной длины

Воздушная линия	Длина линии, м	Период измерений	$U_{ср}$, о.е.	$\pm 3\sigma_U$, о.е.	$\Delta\tau_{ср}$, мкс	$\pm 3\sigma_{\Delta\tau}$, мкс
Кутлу Букаш–Кулущи	16 000	07–10. 2011	96,5	12,6	–0,04	0,15
Кутлу Букаш–Нырты	37 800	07–10. 2011	102,8	8,1	–0,09	0,12
Кутлу Букаш–Рыбная Слобода	40 000	02–12. 2010	100,5	10,8	0,19	0,21
		01–12. 2011	104,4	13,8	0.04	0,18
		01–11. 2012	101,5	12,3	0,02	0,24
Кутлу Букаш–Богатые Сабы	45 700	07–10. 2011	109,5	14,1	–0,09	0,18
Усредненное значение			102,5	11.9	–0,01	018

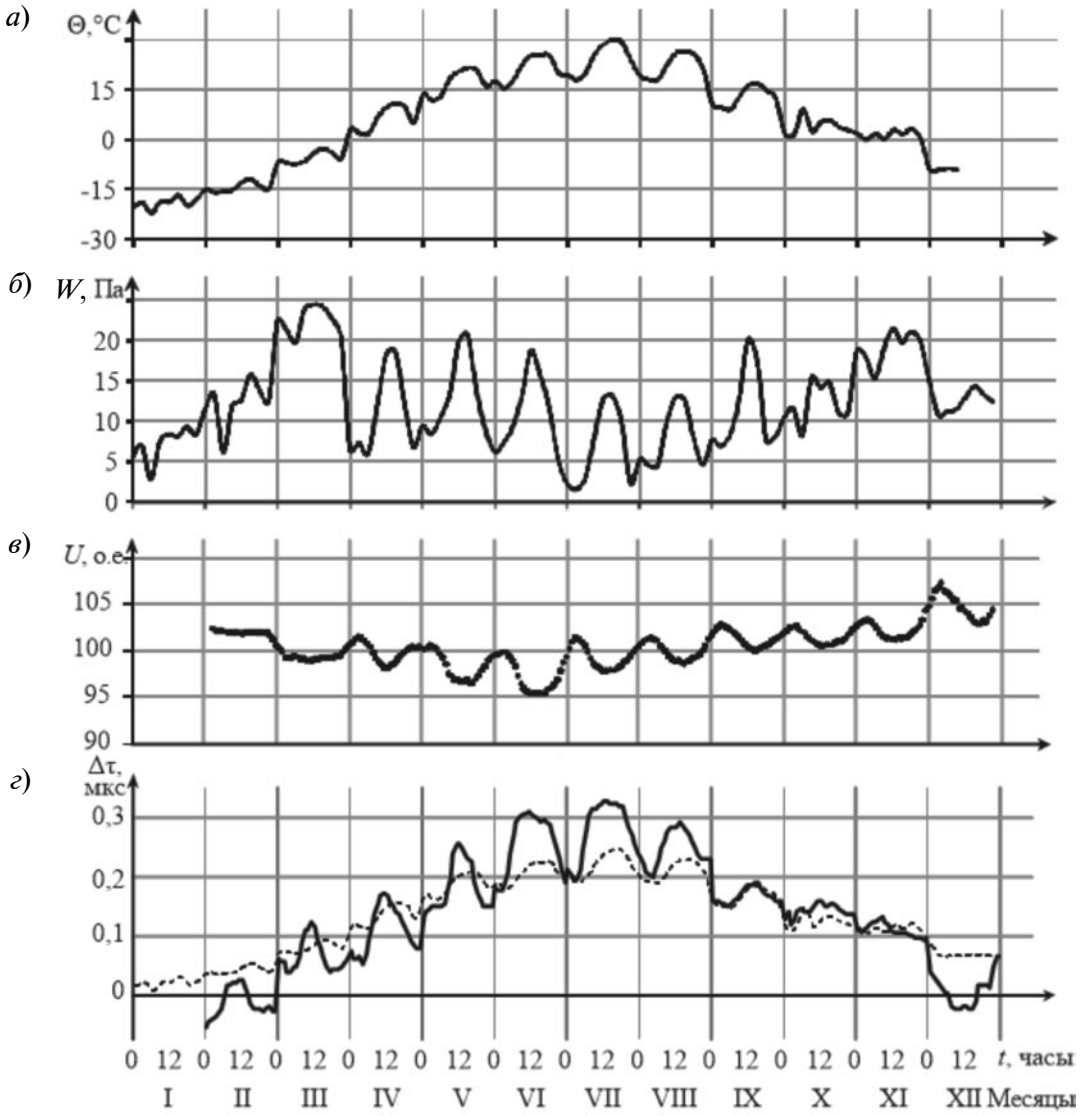


Рис. 4. Суточно-годовые вариации усредненных за месяц часовых значений температуры θ окружающей среды (*а*), ветрового давления W (*б*), измеренных амплитуд U (*в*), измеренных (———) и расчетных (• • • • • •) запаздываний Δτ (*г*) отраженных сигналов

На рис. 4 в качестве примера показаны измеренные в 2010 г. суточно-годовые вариации усредненных за месяц часовых значений температуры $\theta(t)$ окружающей среды (*а*) и ветрового давления $W(t)$ (б) в районе подстанции «Кутлу Букаш», а также суточно-годовые вариации усредненных за месяц часовых значений амплитуд $U(t)$ (*в*) и запаздываний $\Delta\tau(t)$ (*г* – сплошная линия) сигналов, измеренных на линии 110 кВ «Кутлу Букаш–Рыбная Слобода». Ко всем графикам рис. 4 применено скользящее усреднение по шести часовым значениям.

Как видно на рис. 4 *в*, суточные вариации $\theta(t)$ и $W(t)$, а также $U(t)$ и $\Delta\tau(t)$ наиболее ярко выражены в летнее время.

На рис. 4 *в* явно проявляется годовой тренд значений $U(t)$: в полдень и летом амплитуды из-за увеличения затухания сигналов минимальны, а в полночь – максимальны, разность между ними составляет 2–5 % от штатного значения амплитуды. Изменения в течение суток усредненных за месяц часовых значений амплитуд хотя и незначительны, но явно обнаруживаются на всех месячных графиках $U(t)$: летом они

достигают 7 о.е.; зимой − 4 о.е.; годовые изменения составляют 8 о.е., при этом $U_{ср}$ примерно равно 100 о.е.

Согласно рис 4 *г* измеренные значения запаздывания $\Delta\tau$ (сплошная линия) в полдень и летом, видимо, из-за удлинения проводов увеличиваются, а в полночь и зимой уменьшаются, при этом хорошо просматривается годовой тренд этих значений. Изменения в течение суток усредненных за месяц часовых значений запаздываний $\Delta\tau$ на линии «Кутлу Букаш−Рыбная Слобода» летом достигают 0,12 мкс, зимой − 0,05 мкс; годовые изменения составляют 0,35 мкс.

Для линии «Кутлу Букаш−Рыбная Слобода» были рассчитаны суточно-годовые вариации часовых значений запаздывания $\Delta\tau(t)$ по методу допустимых напряжений (объемные нагрузки расчетные, а не нормативные) [12]. Результаты расчета $\Delta\tau(t)$ с учетом изменения температуры θ и ветрового давления W по данным рис. 4 *а* и *б* представлены на рис. 4 *г* (пунктирная линия). Коэффициент взаимной корреляции между расчетными и измеренными значениями $\Delta\tau(t)$ равен 0,8; некоторое расхождение между ними можно объяснить тем, что при расчете не учтены удлинения проводов под влиянием нагрузочного тока.

Максимальное удлинение проводов в течение года на линии «Кутлу Букаш−Рыбная Слобода» длиной 40 000 м в спокойных условиях равно 53 м (0,35 мкс), при учете максимально возможного ветрового давления в 300 Па удлинение увеличивается до 56 м (0,37 мкс), т. е. на 0,14 %. Такие изменения длины проводов под влиянием ветрового давления при выборе уставки по $\Delta\tau$ несущественны и в реальной ситуации их можно не учитывать.

Анализ пределов флуктуаций амплитуд $U(t)$ и запаздываний $\Delta\tau(t)$ отраженных сигналов в штатных условиях при отсутствии гололеда выполнен на линии «Кутлу Букаш−Рыбная Слобода» с использованием массива измерений за 2010−2012 гг., а также на линиях «Кутлу Букаш−Кулущи», «Кутлу Букаш−Ныpты» и «Кутлу Букаш−Богатые Сабы» за 2011 г.

В качестве примера на рис. 5 показаны для линии «Кутлу Букаш−Рыбная Слобода» за 2010 г. графики годовых изменений среднемесячных значений U (*а*) и $\Delta\tau$ (*б*) с доверительными интервалами $\pm$ 3σ, а также годовых изменений сглаженных часовых с усреднением за месяц значений температуры окружающей среды. Как и ранее, на рис. 5 *а* обнаруживается обратная связь между годовыми изменениями значений U и θ ($r = -0{,}7$) и прямая связь между годовыми изменениями значений $\Delta\tau$ и θ ($r = 0{,}8$).

Такие же взаимосвязи между U и θ, а также $\Delta\tau$ и θ были обнаружены на линии «Кутлу Букаш−Рыбная Слобода» в 2011−2012 гг., а также на линиях «Кутлу Букаш−Кулущи», «Кутлу Букаш−Ныpты» и «Кутлу Букаш−Богатые Сабы» в 2011 г.

В результате анализа всего массива измерений U за трехгодичный цикл установлено, что существуют устойчивые годовые уменьшения среднемесячных значений

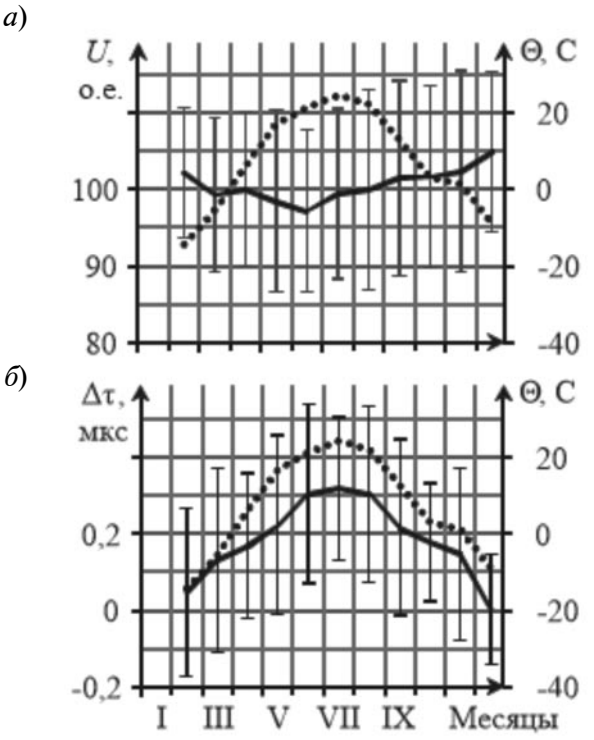


Рис. 5. Графики изменений в 2010 г. среднемесячных значений амплитуд U (*а*) и запаздываний $\Delta\tau$ (*б*) отраженных сигналов (———) с доверительными интервалами $\pm$ 3σ (вертикальные линии) для линии «Кутлу Букаш−Рыбная Слобода» и среднечасовых с усреднением за месяц значений температуры θ (--------) окружающей среды

амплитуд U сигнала, достигающих 10 % от величины амплитуды штатного сигнала в зимнее время. Кроме того, имеют место случайные флуктуации текущих значений амплитуд U в доверительных границах ± 15 % относительно среднемсесячного значения с вероятностью 0,997. При этом погрешность измерения амплитуды отраженных сигналов не превышала ± 1 %.

Также установлено, что при измерениях запаздывания $\Delta\tau$ для линий длиной около 40 000 м годовой тренд среднемесячных значений не превышает 0,3 мкс, суточные случайные вариации $\Delta\tau$ находятся в пределах ± 0,25 мкс с вероятностью 0,997. При этом погрешность измерения запаздывания $\Delta\tau$ отраженных импульсов определяется дискретностью отсчетов АЦП, которая равна 0,08 мкс. Можно утверждать, чтл среди причин, определяющих запаздывание $\Delta\tau$ отраженных импульсных сигналов, температурные влияния являются превалирующими.

При выборе уставок по амплитуде U и запаздыванию $\Delta\tau$ для раннего обнаружения гололеда с целью реализации потенциально высокой чувствительности и большой стабильности локационного метода можно учитывать суточно-годовые изменения этих параметров в виде устойчивых вариаций и в виде случайных отклонений.

Проведены исследования зависимости величины запаздываний $\Delta\tau$ от длины воздушных линий. С использованием [12, 13] рассчитаны в зависимости от температуры окружающей среды среднесуточные часовые значения $\Delta\tau$ за каждый месяц 2010 г. для четырех контролируемых линий «Кутлу Букаш–Кулущи», «Кутлу Букаш–Нырты», «Кутлу Букаш–Рыбная Слобода» и «Кутлу Букаш–Богатые Сабы» с последующим скользящим усреднением по 9 значениям (рис. 6). Таким образом, удалось выделить годовой тренд значений $\Delta\tau$, исключив их суточные вариации. Наименьшие годовые отклонения $\Delta\tau$ в штатных условиях без гололеда (0,08 мкс) имеют место на короткой линии «Кутлу Букаш–Кулущи» (16 000 м) и наибольшие (0,22 мкс) – на длинной линии «Кутлу Букаш–Богатые Сабы» (45 700 м). Эти особенности необходимо учитывать при определении уставок локационного зондирования.

Для выявления статистической зависимости изменений амплитуды U от изменения температуры θ окружающей среды (за счет увеличения длины проводов и повышения их сопротивления из-за нагрева) найдена корреляционная зависимость $U = f(\theta)$ для линии «Кутлу Букаш–Рыбная Слобода» за 2010–2012 гг. с использованием 42 000 измерений в виде $U = 99{,}8 - 0{,}09\ \theta$.

Наблюдается слабая связь между изменениями текущей амплитуды отраженного импульса и температурой окружающей среды (коэффициент взаимной корреляции меньше 0,5), т. к. эта связь маскируется влиянием других причин, рассмотренных выше. Полученную зависимость можно использовать для уточнения величины устав-

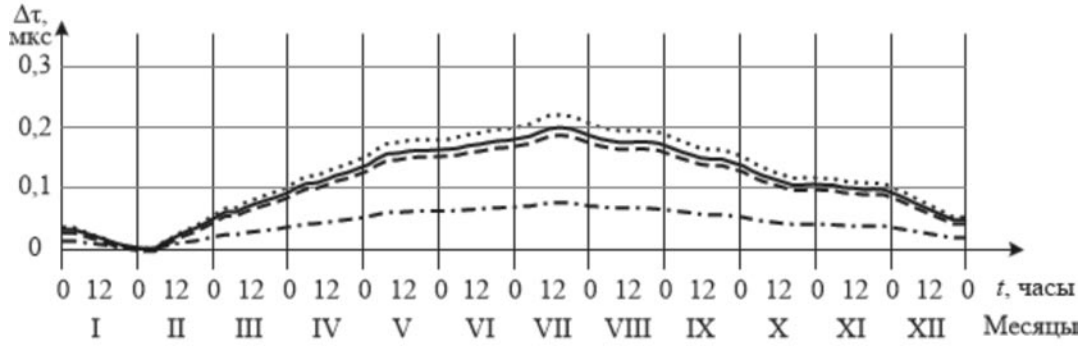


Рис. 6. Расчетные суточно-годовые изменения сглаженных (в интервале 9 ч) усредненных за месяц часовых значений запаздывания $\Delta\tau$ отраженных сигналов на линиях «Кутлу Букаш–Кулущи» (—·—·–), «Кутлу Букаш–Нырты» (--------), «Кутлу Букаш–Рыбная Слобода» (———), «Кутлу Букаш–Богатые Сабы» (·····) в течение 2010 г.

ки по амплитуде с целью обнаружения гололедных отложений.

Для подтверждения температурной зависимости запаздываний $\Delta\tau$ исследованы корреляционные взаимосвязи текущих получасовых значений $\Delta\tau$ и соответствующих значений температуры θ окружающего воздуха на линиях «Кутлу Букаш–Рыбная Слобода», «Кутлу Букаш–Ныpты» и «Кутлу Букаш–Богатые Сабы». Для данных линий получены коэффициенты взаимной корреляции r в пределах 0,69–0,91.

На рис. 7 в качестве примера представлена корреляционная зависимость измеренных среднесуточных значений $\Delta\tau$ от изменения среднесуточной температуры θ окружающей среды на линии «Кутлу Букаш–Рыбная Слобода» за период с февраля по декабрь 2010 г. При этом использовано 11 000 измерений текущих значений $\Delta\tau$. Уравнение регрессии (аппроксимирующая зависимость) имеет вид $\Delta\tau = 0{,}103 + 0{,}005\ \theta$, при коэффициенте корреляции $r = 0{,}85$.

Найденные значения коэффициентов взаимной корреляции между $\Delta\tau$ и θ близки к единице, поэтому они подтверждают высокую статистическую взаимосвязь этих параметров. Проверка коэффициентов r по t-критерию Стьюдента показала, что с доверительной вероятностью $P = 0{,}99$ они значимы для анализируемых воздушных линий, поэтому зависимость между изменениями запаздывания $\Delta\tau$ и изменениями суточной температуры θ является существенной. Проверка

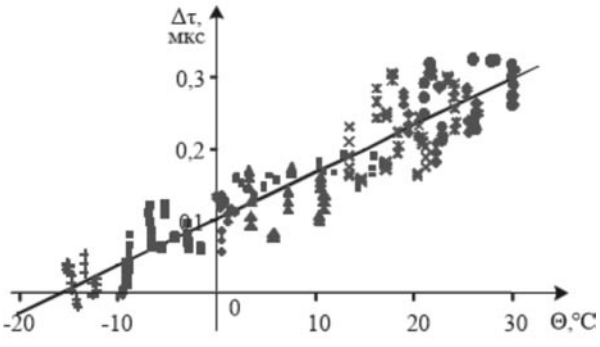


Рис. 7. Корреляционная зависимость измеренных среднесуточных значений $\Delta\tau$ от изменения среднесуточной температуры θ окружающей среды
(——) – аппроксимирующая зависимость

адекватности полученных уравнений тренда $\Delta\tau = f(\theta)$ по F-критерию Фишера показала, что все уравнения статистически значимы с вероятностью $P = 0{,}95$ и пригодны для практического применения.

В результате выполненного анализа можно утверждать, что локационный метод зондирования имеет высокую чувствительность, обеспечивающую обнаружение запаздывания $\Delta\tau$ отраженных сигналов из-за температурных изменений длины проводов ЛЭП в течение суток и года.

Также был определен коэффициент взаимной корреляции случайных флуктуаций амплитуды U и запаздывания $\Delta\tau$ за одинаковые моменты времени в штатных условиях при отсутствии гололеда: он равен –0,37. Поэтому можно утверждать, что взаимосвязь параметров U и $\Delta\tau$ является не очень устойчивой в штатных условиях работы линии, т. к. изменения текущих параметров U и $\Delta\tau$ обусловлены разными физическими причинами, как было показано выше.

Но при обнаружении гололедообразования для повышения его достоверности необходимо использовать оба параметра U и $\Delta\tau$, которые изменяются при этом синхронно и являются надежными и эффективными критериями появления гололедных отложений на проводах воздушных линий.

Предельная чувствительность обнаружения в штатном режиме (при отсутствии гололедных образований) уменьшения амплитуды отраженных локационных сигналов составляет $U = 5$ о.е. или 5 % от штатного значения амплитуды, а увеличения запаздывания состаляет 0,08 мкс.

Эксперименты показывают, что даже при появлении слабой изморози на проводах воздушных линий уменьшения амплитуды отраженного сигнала составляют 50–60 % от штатного режима, а запаздывания достигают 1 мкс и выше, что намного больше температурных изменений этих параметров. Поэтому небольшие (менее 15 %) случайные флуктуации амплитуды и незначительные (менее 0,2 мкс) изменения запаздывания сигнгала в штатных условиях не будут маскировать обнаружение гололеда на проводах ЛЭП.

Итак, приведенные данные характеризуют высокую чувствительность локационного метода и достаточную стабильность высокочастотного канала линии электропередачи при регистрации амплитуды U и запаздываний $\Delta\tau$ отраженного сигнала. При выборе значений уставок по этим параметрам можно учесть их суточные и годовые тренды, что еще больше повысит чувствительность системы локационного мониторинга воздушных линий электропередачи. Флуктуационные изменения параметров U и $\Delta\tau$, а также их суточные и годовые вариации в штатных условиях из-за незначительной величины не будут мешать обнаружению повреждений и гололедных образований на проводах ЛЭП. В некоторых случаях при определении уставок по U и $\Delta\tau$ для обнаружения гололеда можно пренебречь их суточно-годовыми вариациями и флуктуационными изменениями в виду их малости по величине

Авторы выражают благодарность Э.И. Лукину, Р.Г. Мустафину, Ю.В. Писковацкому, Э.Ф. Хакимзянову, С.Г. Ведерникову, И.С. Лаврентьеву, В.Л. Соболевой и А.Я. Латыповой за помощь в проведении измерений.

Исследования и разработки аппаратуры обнаружения гололеда на линиях электропередачи выполнены при финансовой поддержке ОАО «Сетевая компания» (Татарстан), АН Республики Татарстан и ОАО «ФСК ЕЭС».

СПИСОК ЛИТЕРАТУРЫ

1. **Минуллин Р.Г., Касимов В.А., Яруллин М.Р., Филимонова Т.К.** Локационное обнаружение гололеда на воздушных линиях электропередачи. Часть 1. Способы обнаружения гололеда // Научно-технические ведомости СПбГПУ. Информатика. Телекоммуникации. Управление. СПб.: Изд-во СПбГПУ, 2014. № 2 (193). С. 61–73.

2. **Шалыт Г.М.** Определение мест повреждений линий электропередачи импульсным методом. М.: Энергия, 1968. 216 с.

3. **Минуллин Р.Г., Закамский Е.В., Андреев В.В.** Исследования условий отражения импульсных сигналов в распределительных электрических сетях с древовидной топологией // Электротехника. 2003. № 10. С. 39–44.

4. **Минуллин Р.Г., Фардиев И.Ш.** Локационная диагностика воздушных линий электропередачи. Казань: Изд-во КГЭУ, 2008. 202 с.

5. **Минуллин Р.Г., Фардиев И.Ш., Петрушенко Ю.Я., Мезиков А.К., Коровин А.В. и др.** Локационный способ обнаружения появления гололеда на проводах линии электропередачи // Электротехника. 2007. № 12. С. 17–23.

6. **Минуллин Р.Г. и др.** Обнаружение гололедных образований на линиях электропередачи локационным зондированием. Казань: Изд-во КГЭУ, 2010. 207 с.

7. **Минуллин Р.Г.** Обнаружение гололеда и повреждений на воздушных линиях электропередачи локационным методом // Энергетика Татарстана. 2011. № 2 (22). С. 15–17.

8. **Минуллин Р.Г., Лукин Э.И., Сухомяткин М.О. и др.** Особенности обнаружения гололедных отложений на проводах линий электропередачи локационным зондированием // Электротехника. 2011. № 5. С. 6–12.

9. **Минуллин Р.Г., Касимов В.А., Яруллин М.Р., Филимонова Т.К.** Исследование параметров высокочастотного тракта линии электропередачи локационным методом в штатных условиях при отсутствии гололеда // Энергетика Татарстана. 2012. № 4 (28). С. 44–50.

10. **Боровиков В.** Statistica. Искусство анализа данных на компьютере. СПб.: Питер, 2001. 656 с.

11. **Гмурман В.Е.** Теория вероятностей и математическая статистика. М.: Высш. школа, 2007. 479 с.

12. Правила устройства электроустановок. 7-е издание. СПб., 2005.

13. **Кессельман Л.М.** Основы механики воздушных линий электропередачи. М.: Энергоатомиздат, 1992. 352 с.

REFERENCES

1. **Minullin R.G., Kasimov V.A., Yarullin M.R., Filimonova T.K.** Lokatsionnoe obnaruzhenie gololeda na vozdushnykh liniyakh elektroperedachi. Chast 1. Sposoby obnaruzheniya gololeda [Location detection of glaze ice on overhead electric power lines. Part 1. Methods of glaze ice detection], *Nauchno-tekhnicheskiye vedomosti SPbGPU. Informatika. Telekommunikatsii. Upravleniye* [*St. Petersburg State Polytechnical University Journal. Computer Science. Telecommunications and Control System*], St. Petersburg: SPbGPU Publ., 2014, No. 2(193), Pp. 61–73. (rus)

2. **Shalyt G.M.** *Opredeleniye mest povrezhdeniy liniy elektroperedachi impulsnym metodom*. Moscow: Energiya Publ., 1968, 216 p. (rus)

3. **Minullin R.G., Zakamskiy Ye.V., Andreyev V.V.** Issledovaniya usloviy otrazheniya impulsnykh signalov v raspredelitelnykh elektricheskikh setyakh

s drevovidnoy topologiyey, *Elektrotekhnika*, 2003, No. 10, Pp. 39–44. (rus)

4. **Minullin R.G., Fardiyev I.Sh.** *Lokatsionnaya diagnostika vozdushnykh liniy elektroperedachi.* Kazan: KGEU Publ., 2008, 202 p. (rus)

5. **Minullin R.G., Fardiyev I.Sh., Petrushenko Yu.Ya., Mezikov A.K., Korovin A.V. i dr.** Lokatsionnyy sposob obnaruzheniya poyavleniya gololeda na provodakh linii elektroperedachi, *Elektrotekhnika,* 2007, No. 12, Pp. 17–23. (rus)

6. **Minullin R.G. i dr.** *Obnaruzheniye gololednykh obrazovaniy na liniyakh elektroperedachi lokatsionnym zondirovaniyem*, Kazan: KGEU Publ., 2010. 207 p. (rus)

7. **Minullin R.G.** Obnaruzheniye gololeda i povrezhdeniy na vozdushnykh liniyakh elektroperedachi lokatsionnym metodom, *Energetika Tatarstana*, 2011, No. 2 (22), Pp. 15–17. (rus)

8. **Minullin R.G., Lukin E.I., Sukhomyatkin M.O. i dr.** Osobennosti obnaruzheniya gololednykh otlozheniy na provodakh liniy elektroperedachi lokatsionnym zondirovaniyem, *Elektrotekhnika*, 2011, No. 5, Pp. 6–12. (rus)

9. **Minullin R.G., Kasimov V.A., Yarullin M.R., Filimonova T.K.** Issledovaniye parametrov vysokochastotnogo trakta linii elektroperedachi lokatsionnym metodom v shtatnykh usloviyakh pri otsutstvii gololeda, *Energetika Tatarstana*, 2012, No. 4 (28), Pp. 44–50. (rus)

10. **Borovikov V.** *Statistica; Iskusstvo analiza dannykh na kompyutere*, St. Petersburg: Piter Publ., 2001, 656 p. (rus)

11. **Gmurman V.Ye.** *Teoriya veroyatnostey i matematicheskaya statistika*, Moscow: Vysshaya shkola Publ., 2007, 479 p. (rus)

12. *Pravila ustroystva elektroustanovok*. St. Petersburg, 2005. (rus)

13. **Kesselman L.M.** *Osnovy mekhaniki vozdushnykh liniy elektroperedachi.* Moscow: Energoatomizdat Publ., 1992, 352 p. (rus)

МИНУЛЛИН Ренат Гизатуллович – *заведующий научно-исследовательской лабораторией локационной диагностики состояния линий электропередачи Казанского государственного энергетического университета.*
420066, Россия, г. Казань, ул. Красносельская, д. 51.
E-mail: Minullin@mail.ru

MINULLIN, Renat G. *Kazan State Power Engineering University.*
420066, Krasnoselskaya Str. 51, Kazan, Russia.
E-mail: Minullin@mail.ru

КАСИМОВ Василь Амирович – *аспирант научно-исследовательской лаборатории локационной диагностики состояния линий электропередачи Казанского государственного энергетического университета.*
420066, Россия, г. Казань, ул. Красносельская, д. 51.
E-mail: VasilKasimov@yandex.ru

KASIMOV, Vasil A. *Kazan State Power Engineering University.*
420066, Krasnoselskaya Str. 51, Kazan, Russia.
E-mail: VasilKasimov@yandex.ru

ФИЛИМОНОВА Тамара Константиновна – *доцент кафедры инженерной кибернетики Института экономики и информационных технологий Казанского государственного энергетического университета.*
420066, Россия, г. Казань, ул. Красносельская, д. 51.
E-mail: Filimonova.Tamara@bk.ru

FILIMONOVA, Tamara K. *Kazan State Power Engineering University.*
420066, Krasnoselskaya Str. 51, Kazan, Russia.
E-mail: Filimonova.Tamara@bk.ru

ЯРУЛЛИН Марсель Рашитович – *аспирант научно-исследовательской лаборатории локационной диагностики состояния линий электропередачи Казанского государственного энергетического университета.*
420066, Россия, г. Казань, ул. Красносельская, д. 51.
E-mail: Marsel.Jarullin@gmail.com

YARULLIN, Marsel R. *Kazan State Power Engineering University.*
420066, Krasnoselskaya Str. 51, Kazan, Russia.
E-mail: Marsel.Jarullin@gmail.com

УДК 621.374

Ю.К. Рыбин

СИНТЕЗ СИГНАЛОВ С ЗАДАННЫМ КОЭФФИЦИЕНТОМ ГАРМОНИК

Yu.K. Rybin

SYNTHESIS OF SIGNALS WITH A GIVEN HARMONIC COEFFICIENT

Проведен обзор проблемы формирования измерительных сигналов с заданным коэффициентом гармоник. Показано, что перспективным способом формирования подобных сигналов является синтез формы с помощью генераторов синусоидальных сигналов с возможностью управления длительностью полупериодов. Синтезируемые сигналы имеют постоянные амплитудное и среднеквадратическое значения выходного сигнала напряжения при разном установленном коэффициенте гармоник.

ИЗМЕРИТЕЛЬНЫЙ СИГНАЛ; КОЭФФИЦИЕНТ НЕЛИНЕЙНЫХ ИСКАЖЕНИЙ; КОЭФФИЦИЕНТ ГАРМОНИК.

A problem of the measured signals with a predetermined ratio of harmonics formation is described. It is shown that a promising way of forming such shaped signals is a synthesis using sinusoidal signal generator to control the duration of half-periods. Synthesized signals have constant amplitude and RMS voltage at a different set THD.

MEASURING SIGNAL; NONLINEAR DISTORTION FACTOR; TOTAL HARMONIC DISTORTION.

Измерительные сигналы с нормированным коэффициентом гармоник K_g используются в средствах поверки измерителей нелинейных искажений [1] при определении погрешности цифровых вольтметров и преобразователей переменного напряжения и тока [2, 3], фазометров и частотомеров, обусловленной искажениями сигнала. По ГОСТ 14014–82 в технических условиях на цифровые преобразователи среднеквадратических значений, реагирующие на средневыпрямленное значение, должны содержаться значения коэффициента гармоник, при которых сохраняются установленные метрологические характеристики, следовательно, для оценки погрешности преобразователей нужны сигналы с заданным коэффициентом гармоник.

Определение коэффициента гармоник

Коэффициент гармоник определяется как отношение корня квадратного из суммы квадратов амплитуд высших гармоник, начиная со второй, к амплитуде первой гармоники. Обычно он выражается в процентах или в децибелах. Для расчета коэффициента гармоник с помощью анализатора спектра определяют амплитуды гармоник, а затем вычисляют его значение:

$$K_{g\%} = \frac{\sqrt{\sum_{n=2}^{\infty} V_n^2}}{V_1} 100\ \%,$$
$$K_{gdB} = 20 \lg \left(\frac{\sqrt{\sum_{n=2}^{\infty} V_n^2}}{V_1} \right). \quad (1)$$

Как известно, при измерениях число гармоник ограничивают, например, десятью гармониками, поэтому коэффициент гармоник определяется с погрешностью. Данный метод реализован в измерителе 2016 THD Multimeter [4]. В нем аналоговый входной сигнал преобразуется в цифровой, по которому далее вычисляется спектр и рассчитывается суммарное значение коэффициента гармоник при учете от двух до 62-х гармоник. Этот же метод положен в основу измерителя СК6-20 [5, 6] и калибра-

тора СК6-21 [7]. В этом случае прибор измеряет коэффициент гармоник в соответствии с формулой (1) не при бесконечном, а при конечном числе гармоник.

Другой метод измерения коэффициента гармоник реализован в аналоговых измерителях нелинейных искажений типа С6-5, С6-7. Суть его заключается в измерении среднеквадратического напряжения сигнала, подавлении первой гармоники режекторным фильтром, измерении среднеквадратического напряжения высших гармоник и вычислении отношения измеренных значений по формуле:

$$K_{\text{THD}} = \frac{\sqrt{\sum_{n=2}^{\infty} V_n^2}}{\sqrt{\sum_{n=1}^{\infty} V_n^2}} 100\ \%,$$

$$K_{\text{THD+N}} = \frac{\sqrt{\sum_{n=2}^{\infty} V_n^2 + V_{noise}^2}}{\sqrt{\sum_{n=1}^{\infty} V_n^2 + V_{noise}^2}} 100\ \%. \qquad (2)$$

Однако одновременно с измерением высших гармоник измеряются и шумы, сопровождающие сигнал, и собственные шумы прибора. Понятно, что и здесь возникает погрешность измерения за счет влияния шумов. Это влияние может оказаться полезным, т. к. собственные шумы сигнала также характеризуют его искажения и должны учитываться. Поэтому вторая формула (2) используется также широко, как и первая формула в (2). Именно она более точно отражает результат измерения искажений. Обратная величина, выраженная в децибелах, часто называется Signal-to-noise and distortion ratio – SINAD [8].

Проблемы формирования сигналов с заданным коэффициентом гармоник

Для поверки анализаторов спектра и измерителей нелинейных искажений формируют сигналы с заданным (точно известным коэффициентом гармоник). Но при их формировании возникают определенные трудности, т. к. коэффициент гармоник, являясь интегральным параметром сигнала, инвариантен к его форме при условии равенства амплитуд первых гармоник и среднеквадратических значений высших гармоник. Поэтому сигналов с заданным коэффициентом гармоник может быть бесконечно много. Это, с одной стороны, дает возможность использовать в качестве измерительных сигналов с нормированным коэффициентом гармоник сигналы прямоугольной, треугольной или синусоидальной формы типа «усеченный синус», «разновеликий синус» [9] и т. д. Но, с другой стороны, это приводит к неопределенности выбора измерительного сигнала, порождает непрекращающиеся дискуссии по поводу его наилучшей формы. Очевидно, что однозначного ответа на вопрос, какой должна быть форма сигнала, нет, т. к. форма не ограничивается самим определением коэффициента гармоник по формулам (1, 2).

Мы убеждены, что поиск оптимальной в том или ином смысле формы должен проводиться, исходя из цели использования измерительного сигнала, например, с учетом требования согласования спектра сигнала, с диапазоном рабочих частот поверяемого средства измерения. Несоблюдение этого требования влечет за собой появление дополнительных погрешностей измерения, которые невозможно или достаточно сложно оценить. Поэтому оптимальным будем считать такой измерительный сигнал, спектр которого полностью находится в диапазоне рабочих частот поверяемого средства измерений. Ясно также, что это не один сигнал, а целый класс оптимальных измерительных сигналов, т. к. число гармоник и их соотношение могут быть различными. Так, иногда рекомендуется использовать в измерительном сигнале только две гармоники, однако известен калибратор нелинейных искажений с тремя гармониками [10], кроме того, в новом эталоне предлагается работать и с большим их числом (до 50).

Наряду с оптимальными, в измерительной практике находят применение и другие сигналы, формирование которых может осуществляться более простыми средствами. Так, в государственном эталоне коэффициента нелинейных искажений

[9] использовались измерительные сигналы типа «усеченный синус» и «разновеликий синус», предложенные для этого и исследованные Н.Б. Петровым [9]. Общий недостаток известных измерительных сигналов – то, что коэффициент гармоник в них определяется отношением, например, среднеквадратических значений напряжения второй и первой гармоник, либо через отношение уровня ограничения синусоидального сигнала к его амплитудному значению [9]. Известно, что точное задание отношений переменных напряжений зависит от погрешности деления используемых для этой цели делителей. Так, это может быть индуктивный делитель напряжения [10] либо точный резистивный делитель. Нет нужды объяснять, что данные устройства весьма трудоемки в изготовлении и громоздки, неудобны для автоматизации их изготовления. Поэтому нами предложен способ задания коэффициента гармоник, реализованный в [11], не через отношение напряжений, а через отношение длительностей интервалов времени. Известно, что погрешность измерения и воспроизведения интервалов времени уже сегодня достигает долей пикосекунд, и нет принципиальных ограничений по ее дальнейшему снижению.

Сущность предложенного способа поясним на модели композитного измерительного сигнала, составленного из сигналов синусоидальной или косинусоидальной формы равной амплитуды, но разной длительности:

$$x(t) = \sum_{i=-\infty}^{\infty} a_i \cos\left(\frac{t-\tau_i}{T_i}\right) \times \times [H(t-\tau_i) - H(t-\tau_{i+1})], \quad (3)$$

где $T_i = \tau_{i+1} - \tau_i$, $a_i = a(-1)^i$.

На рис. 1 *а* показаны сигналы синусоидальной и косинусоидальной форм разной длительности полупериодов или периодов. Эти сигналы являются частным случаем, описанным выражением (3), и представляют собой периодическую чередующуюся последовательность импульсов, длительность которых на периоде T можно изменять. При $T_{-i} = ... = T_{-1} = T_0 = T_1 = T_2 = ... = T_i = T/2$ выражение (3) служит моделью строго синусоидального периодического сигнала, для которого $K_\Gamma = 0$.

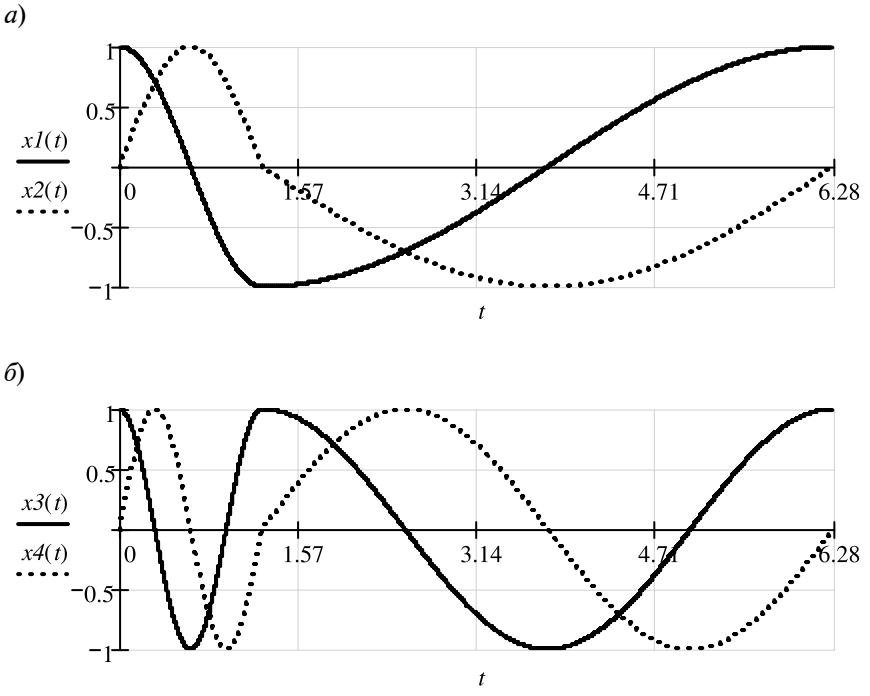


Рис. 1. Измерительные сигналы с заданным коэффициентом гармоник

На рис. 1 *б* показаны сигналы, составленные из двух полупериодов косинусоидальной и синусоидальной форм, которые описываются выражениями:

$$x_1(t) = \begin{cases} \cos\left(\frac{\pi}{\alpha T} t\right) & if \quad 0 \le t < \alpha T; \\ -\cos\left[\frac{\pi}{(1-\alpha)}\left(\frac{t}{T} - \alpha\right)\right] & if \quad \alpha T \le t < T; \end{cases} \tag{4}$$

$$x_2(t) = \begin{cases} \sin\left(\frac{\pi}{\alpha T} t\right) & if \quad 0 \le t < \alpha T; \\ -\sin\left[\frac{\pi}{(1-\alpha)}\left(\frac{t}{T} - \alpha\right)\right] & if \quad \alpha T \le t < T; \end{cases} \tag{5}$$

$$x_3(t) = \begin{cases} \cos\left(\frac{2\pi}{\alpha T} t\right) & if \quad 0 \le t < \alpha T; \\ \cos\left[\frac{2\pi}{(1-\alpha)}\left(\frac{t}{T} - \alpha\right)\right] & if \quad \alpha T \le t < T; \end{cases} \tag{6}$$

$$x_4(t) = \begin{cases} \sin\left(\frac{2\pi}{\alpha T} t\right) & if \quad 0 \le t < \alpha T; \\ \sin\left[\frac{2\pi}{(1-\alpha)}\left(\frac{t}{T} - \alpha\right)\right] & if \quad \alpha T \le t < T, \end{cases} \tag{7}$$

где α – коэффициент, характеризующий неравенство длительностей полупериодов или периодов.

Как видно из формул (5–7), в сигналах $x_1(t)$ и $x_2(t)$ на периоде частота изменяется с $\omega_1 = \pi/\alpha T$ к $\omega_2 = \pi/(1-\alpha)T$. При этом частота повторения составного сигнала равна $\omega_0 = 2\pi/T$, а соотношение частот равно $\omega_1 = \omega_0/2\alpha$ и $\omega_2 = \omega_0/2(1-\alpha)$. Например, при $\alpha = 0,2$ на рис. 1 *а* (сплошная линия) $\omega_0 = 1$, $\omega_1 = 2,5$ и $\omega_2 = 0,625$, а отношение частот равно $\omega_1/\omega_2 = (1-\alpha)/\alpha = 4$. При этом из графика на рис. 2 *а* K_Γ сигнала $x_1(t)$ равен 35,193 %. Интересно, что при $\alpha = 0,5$ все частоты равны $\omega_0 = \omega_1 = \omega_1 = 1$, а $K_\Gamma = 0$ %. Максимальное значение $K_\Gamma = 61,832$ % достигается при $\alpha = 0$.

Несмотря на незначительную нелинейность графиков, для конкретных значений K_Γ значения соотношения длительностей могут быть рассчитаны точно.

Меняя отношение длительности полуволны к длительности периода α, можно получать сигналы с различными значениями K_Γ. Замечательная особенность этих сигналов – однозначная связь коэффициента гармоник и отношения длительностей, постоянство его амплитудного, средневыпрямленного и среднеквадратического значений при изменении длительностей полуволн. При использовании таких сигналов в средствах измерений для поверки измерителей нелинейных искажений значительно повышается производительность за счет уменьшения времени калибровки по уровню среднеквадратического значения, автоматизации процесса, что реализовано в генераторе, защищенном авторским свидетельством [11].

Все сигналы могут быть получены с помощью аналоговых и цифровых генераторов синусоидальных сигналов с возможностью управления длительностью обоих полупериодов. Особенность синусоидальных сигналов с $K_\Gamma = 0$ % состоит в том, что они получаются как частный случай при равных длительностях полупериодов. Наименьшие значения K_Γ реальных синусоидальных сигналов достигаются в аналоговых генераторах. Так, в генераторе ГС-50 это значение менее 0,0001 %.

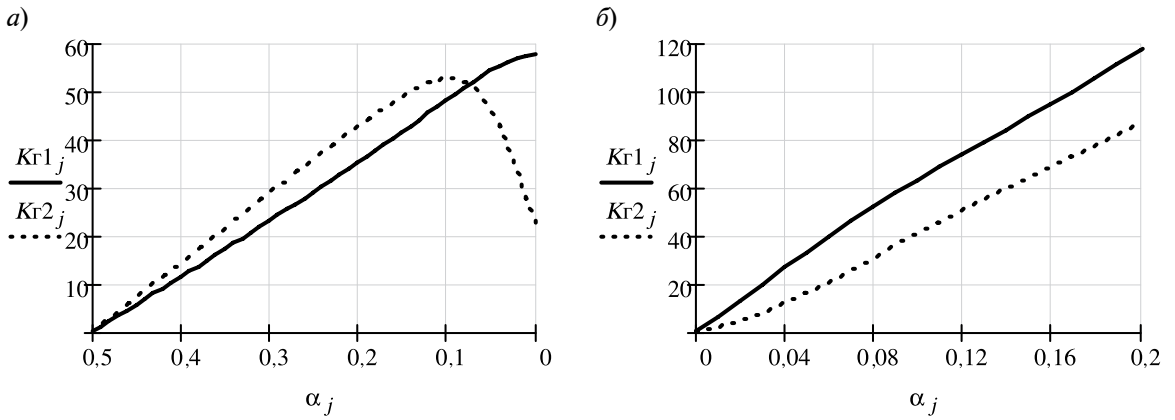


Рис. 2. Графики зависимости коэффициента гармоник сигналов $x_1(t)$ и $x_2(t)$ (*а*) и $x_3(t)$ и $x_4(t)$ (*б*) от соотношения длительностей полуволны и периода сигнала

В статье проведен анализ проблемы формирования измерительных сигналов с заданным коэффициентом гармоник. Задача синтеза измерительного сигнала рассмотрена для аналогового и цифрового способов формирования сигнала с последующим его цифро-аналоговым преобразованием. При этом способе задания коэффициента гармоник конкретные формы сигнала реализуются через отношение длительностей интервалов времени. Сущность предложенного способа проиллюстрирована на модели композитного измерительного сигнала, составленного из сигналов синусоидальной или косинусоидальной формы равной амплитуды, но разной длительности.

СПИСОК ЛИТЕРАТУРЫ

1. **ГОСТ 8.331–99.** Межгосударственный стандарт. Государственная система обеспечения единства измерений. Измерители нелинейных искажений. Методы и средства поверки. М.: Изд-во стандартов, 2001. 18 с.

2. **ГОСТ 8.118–85.** Государственная система обеспечения единства измерений. Вольтметры электронные аналоговые переменного тока. Методика поверки. М.: Изд-во стандартов, 1986. 16 с.

3. **ГОСТ 23854–79.** Измерители уровня электрических сигналов. Общие технические требования и методы испытаний. М.: Изд-во стандартов. 2002. 15 с.

4. Model 2016 THD Multimeter User's Manual 2016-900-01 Rev. C [электронный ресурс] / URL: http://www.keithley.com/products/dcac/audioanalyzer/?path=2016/Documents#5: (дата обращения 08.11.2013).

5. **ГОСТ Р 8.762-2011 ГСИ.** Государственная поверочная схема для средств измерений коэффициента гармоник. Высшее звено поверочной схемы – Государственный первичный эталон единицы коэффициента гармоник. М.: Изд-во стандартов, 2012. 16 с.

6. **Казанцев Ю.И., Музалевский В.Е., Пруглo А.В.** Метрологическое обеспечение измерений коэффициента гармоник низкочастотных радиотехнических сигналов [электронный ресурс] / URL: http://www.npcentre.ru/view_docs.php?doc=26 (дата обращения 08.11.2013).

7. **Kester W.** Understand SINAD, ENOB, SNR, THD, THD + N [электронный ресурс] / URL: http://www.analog.com/static/imported-files/tutorials/MT-003.pdf (дата обращения 11.11.2013).

8. **Петров Н.Б., Ковалев Г.Г., Жеваго А.А., Яковлева В.Я., Бойченко В.Д., Заруба Л.И., Амброзевич О.Н.** Государственный первичный эталон единицы коэффициента нелинейных искажений // Измерительная техника. 1974. № 5. С. 9–11.

9. **Рыбин Ю.К., Ройтман М.С., Литвак Э.С.** А. с. 584259 СССР, МКИ[3] G 01 R 23/20. Устройство для получения сигнала, калиброванного по коэффициенту нелинейных искажений. Патент № 2393943/18–21; заявл. 01.08.76; опубл. 07.12.77.

10. Калибратор коэффициента гармоник (СК6-21) [электронный ресурс] / URL: http://www.rpis.ru/index.php?option=content&task=view&id=40: (дата обращения 08.11.2013).

11. **Рыбин Ю.К.** А. с. 1114970 СССР, МКИ[3] G 01 R 23/20. Устройство для получения сигнала, калиброванного по коэффициенту нелинейных искажений. Патент № 3558386/18–21; заявл. 28.02.83; опубл. 23.09.84.

REFERENCES

1. **GOST 8.331–99.** *Mezhgosudarstvennyy standart. Gosudarstvennaya sistema obespecheniya yedinstva izmereniy. Izmeriteli nelineynykh iskazheniy. Metody i sredstva poverki*, Moscow: Izd-vo standartov Publ., 2001, 18 p. (rus)

2. **GOST 8.118–85.** *Gosudarstvennaya sistema obespecheniya yedinstva izmereniy. Voltmetry elektronnyye analogovyye peremennogo toka. Metodika poverki*, Moscow: Izd-vo standartov Publ., 1986, 16 p. (rus)

3. **GOST 23854–79.** *Izmeriteli urovnya elektricheskikh signalov. Obshchiye tekhnicheskiye trebovaniya i metody ispytaniy*, Moscow: Izd-vo standartov Publ., 2002, 15 p. (rus)

4. *Model 2016 THD Multimeter User's Manual 2016-900-01 Rev. C*. Available: http://www.keithley.com/products/dcac/audioanalyzer/?path=2016/Documents#5: (Accessed 08.11.2013).

5. **GOST R 8.762-2011 GSI.** *Gosudarstvennaya poverochnaya shema dlya sredstv izmereniy koeffitsiyenta garmonik. Vyssheye zveno poverochnoy skhemy – Gosudarstvennyy pervichnyy etalon yedinitsy koeffitsiyenta garmonik*, Moscow: Izd-vo standartov Publ., 2012, 16 p. (rus)

6. **Kazantsev Yu.I., Muzalevskiy V.Ye., Pruglo A.V.** *Metrologicheskoye obespecheniye izmereniy koeffitsiyenta garmonik nizkochastotnykh radiotekhnicheskikh signalov*. Available: http://www.npcentre.ru/view_docs.php?doc=26 (Accessed 08.11.2013). (rus)

7. **Kester W.** Understand SINAD, ENOB, SNR, THD, THD + N. Available: http://www.analog.com/static/imported-files/tutorials/MT-003.pdf (Accessed 11.11.2013).

8. **Petrov, N.B., Kovalev G.G., Zhevago A.A., Yakovleva V.Ya., Boychenko V.D., Zaruba L.I., Ambrozevich O.N.** Gosudarstvennyy pervichnyy etalon yedinitsy koeffitsiyenta nelineynykh iskazheniy, *Izmeritelnaya tekhnika*, 1974, No. 5, Pp. 9–11. (rus)

9. **Rybin Yu.K., Roytman M.S., Litvak E.S.** A. s. 584259 USSR, MKI3 G 01 R 23/20. *Ustroystvo dlya polucheniya signala, kalibrovannogo po koeffitsiyentu nelineynykh iskazheniy*. Patent No. 2393943/18–21; zayavl. 01.08.76 opubl. 07.12.77. (rus)

10. *Kalibrator koeffitsiyenta garmonik (SK6-21)*. Available: http://www.rpis.ru/index.php?option=content&task=view&id=40: (Accessed 08.11.2013). (rus)

11. **Rybin Yu.K.** A. s. 1114970 USSR, MKI3 G 01 R 23/20. *Ustroystvo dlya polucheniya signala, kalibrovannogo po koeffitsiyentu nelineynykh iskazheniy*. Patent No. 3558386/18–21; zayavl. 28.02.83 opubl. 23.09.84. (rus)

РЫБИН Юрий Константинович – *доцент кафедры компьютерных измерительных систем и метрологии Национального исследовательского Томского политехнического университета, кандидат технических наук.*

634050, Россия, г. Томск, пр. Ленина, д. 30.
E-mail: rybin@tpu.ru

RYBIN, Yurii K. *National Research Tomsk Polytechnic University.*
634050, Lenina Ave. 30, Tomsk, Russia.
E-mail: rybin@tpu.ru

УДК 681.3 (075.8)

Н.В. Ростов

МНОГОКРИТЕРИАЛЬНАЯ ПАРАМЕТРИЧЕСКАЯ ОПТИМИЗАЦИЯ ЦИФРОВЫХ РЕГУЛЯТОРОВ С УЧЕТОМ НЕЛИНЕЙНОСТЕЙ И ДЕЙСТВИЯ ВНЕШНИХ ВОЗМУЩЕНИЙ

N.V. Rostov

MULTIOBJECTIVE PARAMETER OPTIMIZATION OF DIGITAL CONTROLLERS WITH REGARD TO THE INFLUENCE OF NONLINEARITIES AND EXTERNAL DISTURBANCES

Предложена методика многокритериальной компьютерной настройки параметров цифровых регуляторов в системах стабилизации и следящих системах с электромеханическими объектами управления в различных динамических режимах с учетом нелинейностей и при действии внешних возмущений. Приведены примеры настройки параметров типовых цифровых регуляторов в контурах позиционной следящей системы.

СИСТЕМЫ АВТОМАТИЧЕСКОГО УПРАВЛЕНИЯ; ЦИФРОВЫЕ РЕГУЛЯТОРЫ; МНОГОКРИТЕРИАЛЬНАЯ ПАРАМЕТРИЧЕСКАЯ ОПТИМИЗАЦИЯ; ПРЯМОЕ ЗОНДИРОВАНИЕ; ПАРЕТО-РЕШЕНИЯ; НЕЛИНЕЙНОСТИ; ВНЕШНИЕ ВОЗМУЩЕНИЯ.

Some formulations and computer-aided technologies for multiobjective parameter optimization of digital stabilization and tracking control systems with electromechanical plants are proposed, taking into account nonlinearities and disturbances in different dynamical modes. The practical examples are given to multiobjective parameter tuning of typical digital controllers.

AUTOMATIC CONTROL SYSTEMS; DIGITAL CONTROLLERS; MULTIOBJECTIVE PARAMETER OPTIMIZATION; DIRECT SOUNDING; PARETO-SOLUTIONS; NONLINEARITIES; DISTURBANCES.

В многоконтурных микропроцессорных системах автоматического управления (САУ), таких как системы стабилизации скорости и позиционные следящие системы, широко применяются цифровые регуляторы низкого порядка ПИ, ПД и ПИД-типа. Их линейный параметрический синтез обычно проводится аналитически, начиная с внутреннего контура, по выражениям, обеспечивающим настройку переходных процессов в контурах момента (или тока), скорости и положения по модульному, симметричному или другим критериям оптимальности [1]. Однако при компьютерном проектировании таких САУ с использованием нелинейных моделей контуров расчетные значения параметров регуляторов приходится корректировать. При этом целесообразно осуществлять многокритериальную (МК) компромиссную настройку регуляторов из-за противоречивости частных критериев – интегральных и прямых показателей динамических процессов, оцениваемых в основных режимах работы кон-

туров при входных и возмущающих воздействиях разных видов.

В статьях [3, 4] предложены методики МК-оптимизации последовательных и модальных цифровых регуляторов высокого порядка на основе косвенного зондирования, совмещенного с алгебраическим синтезом и анализом устойчивости на сетке полюсов линейной дискретной модели системы и оцениванием частных критериев по результатам нелинейного моделирования САУ. В статье [5] изложена методика последовательной (поэтапной) МК-оптимизации цифровых САУ с предварительной итерационной скалярной оптимизацией САУ на начальных этапах для локализации Парето-области и проведением в ограниченной ее окрестности прямого зондирования пространства параметров регулятора.

В предлагаемой ниже практической методике МК-настройка параметров цифровых регуляторов проводится на основе *многорежимного зондирования динамики* с оцениванием показателей переходных и установившихся процессов по результатам дискретно-непрерывного моделирования цифровых САУ при разных видах входных воздействий с учетом присущих реальным системам нелинейностей, а также при действии внешних возмущений. Применение методики иллюстрируется на примерах МК-настройки параметров цифровых регуляторов в контурах следящей системы со статическими и астатическими объектами управления (ОУ).

Дискретно-непрерывные модели цифровых САУ. На обобщенной структурной схеме цифровой САУ (рис. 1) обозначены: НЧ – непрерывная часть системы (объект управления); ДЧ – дискретная часть; ЦР, ЦК – цифровой регулятор и компенсатор; АЦП, ЦАП – аналого-цифровой и цифро-аналоговый преобразователи; ЭЗ – элемент временного запаздывания; $Э_0$ – экстраполятор нулевого порядка; T_0 – период квантования сигналов по времени; $НЭ_1$, $НЭ_2$ – нелинейные элементы, учитывающие квантование по уровню входного воздействия $g[n]$ и сигнала обратной связи $y[n]$; $НЭ_3$ – нелинейный элемент, учитывающий квантование по уровню и ограничение цифрового сигнала управления $u[n]$; $f(t)$ – внешнее возмущение.

В моделях электромеханических ОУ необходимо учитывать падение напряжения на выходе силового преобразователя и на щетках двигателя, трение и момент нагрузки на валу двигателя, люфт в механической передаче.

Постановки задач МК-оптимизации цифровых САУ. В зависимости от целевого назначения системы векторные критерии могут составляться из различных пар частных критериев $F_i(\theta) = [f_1,\ f_2]^T$, зависящих от значений θ – вектора настраиваемых параметров регулятора.

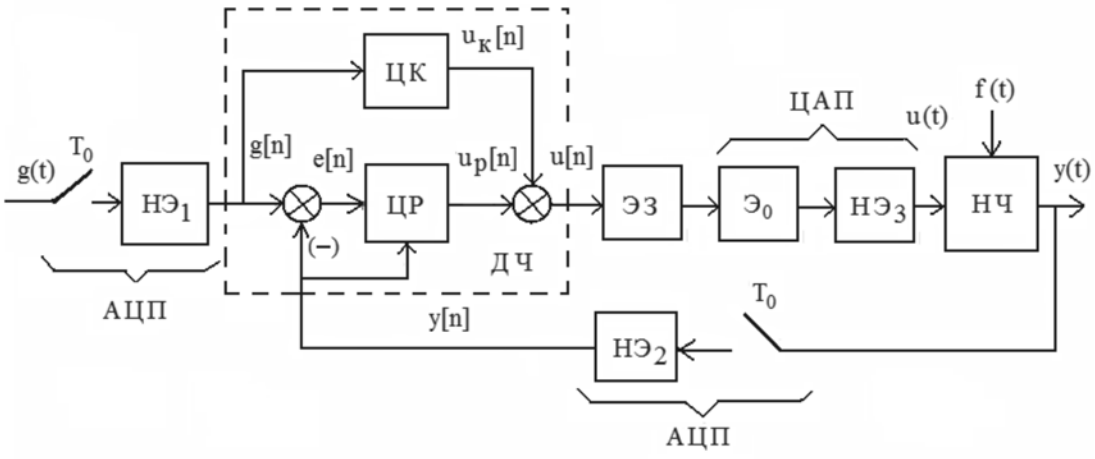


Рис. 1. Структурная схема цифровой системы

Для систем любого класса должны оцениваться прямые показатели переходного процесса по входному воздействию:

$$F_1(\theta) = [T_g, \ \sigma]^{\mathrm{T}} \to \min, \qquad (1)$$

где T_g – время переходного процесса; σ – перерегулирование.

Но в случаях сильно колебательной или расходящейся переходной характеристики САУ они не могут быть определены по результатам моделирования.

Векторный критерий может включать в себя интегральные квадратичные оценки переходного процесса произвольного вида, вычисляемые в дискретном времени:

$$F_2(\theta) = \left[\frac{1}{N} \sum_{n=0}^{N} e^2[n], \frac{1}{N} \sum_{n=0}^{N} u^2[n] \right] \to \min, \qquad (2)$$

где $e[n] = (g[n] - y[n])$ – ошибка системы; $u[n]$ – управляющее воздействие; N – число периодов дискретности протекания переходного процесса. Первый частный критерий косвенно оценивает быстродействие САУ, а второй – энергозатраты на управление.

Векторный критерий может содержать также интегральную оценку производной выходной переменной системы, характеризующую ее колебательность:

$$F_3(\theta) = \left[\frac{1}{N} \sum_{n=0}^{N} e^2[n], \frac{1}{T} \int_0^T \left(\frac{dy}{dt}\right)^2 dt \right] \to \min, \qquad (3)$$

где T – время протекания динамического процесса.

Для *систем стабилизации момента или скорости* векторный критерий должен содержать оценки показателей инвариантности, определяемых при действии ступенчатого внешнего возмущения:

$$F_4(\theta) = [T_f, \ |e_f|_{\max}]^{\mathrm{T}} \to \min, \qquad (4)$$

где T_f – время переходного процесса по возмущению; $|e_f|_{\max}$ – соответствующая максимальная динамическая ошибка.

Для *позиционных следящих систем* векторный критерий должен включать в себя оценки точностных показателей, определяемых при гармоническом входном воздействии:

$$F_5(\theta) = \left[\frac{1}{K_v}, E_{\mathrm{sq}} \right]^{\mathrm{T}} \to \min, \qquad (5)$$

где $E_{\mathrm{sq}} = \sqrt{\left(1 / N \sum_{1}^{N} e^2[n]\right)}$ – среднеквадратичная динамическая ошибка; K_v – добротность по скорости.

Для оценивания указанных выше критериев необходимо проводить зондирование динамики цифровой САУ с моделированием в следующих трех режимах:

- при ступенчатом входном воздействии $g[n]$ и нулевом внешнем возмущении $f(t)$;
- при действии ступенчатого внешнего возмущения $f(t)$;
- при гармоническом воздействии $g[n] = g_m \sin(\omega \cdot nT_0)$, при этом

$$K_v = (g_m \cdot \omega) / |e[n]|_{\max}.$$

В конкретных практических задачах МК-оптимизации полный векторный критерий может формироваться из разных комбинаций пар частных критериев (1)–(5) в соответствии с классом САУ.

МК-оптимизация регуляторов трехконтурной следящей системы. *Настройка цифрового ПИ-регулятора тока.* На рис. 2 представлена схема Simulink-модели циф-

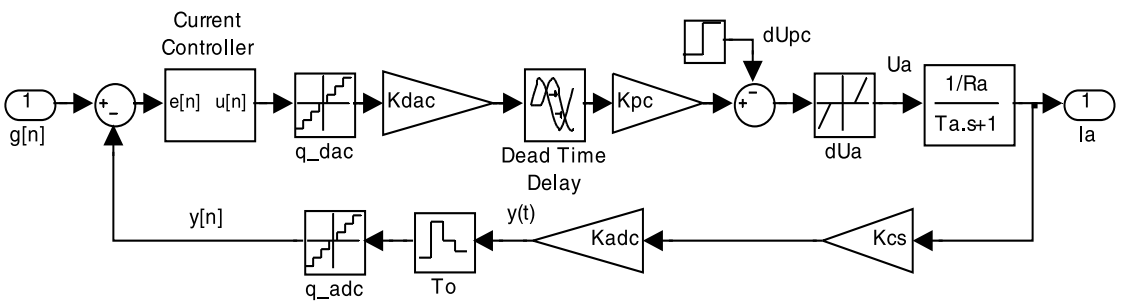


Рис. 2. Simulink-модель контура тока

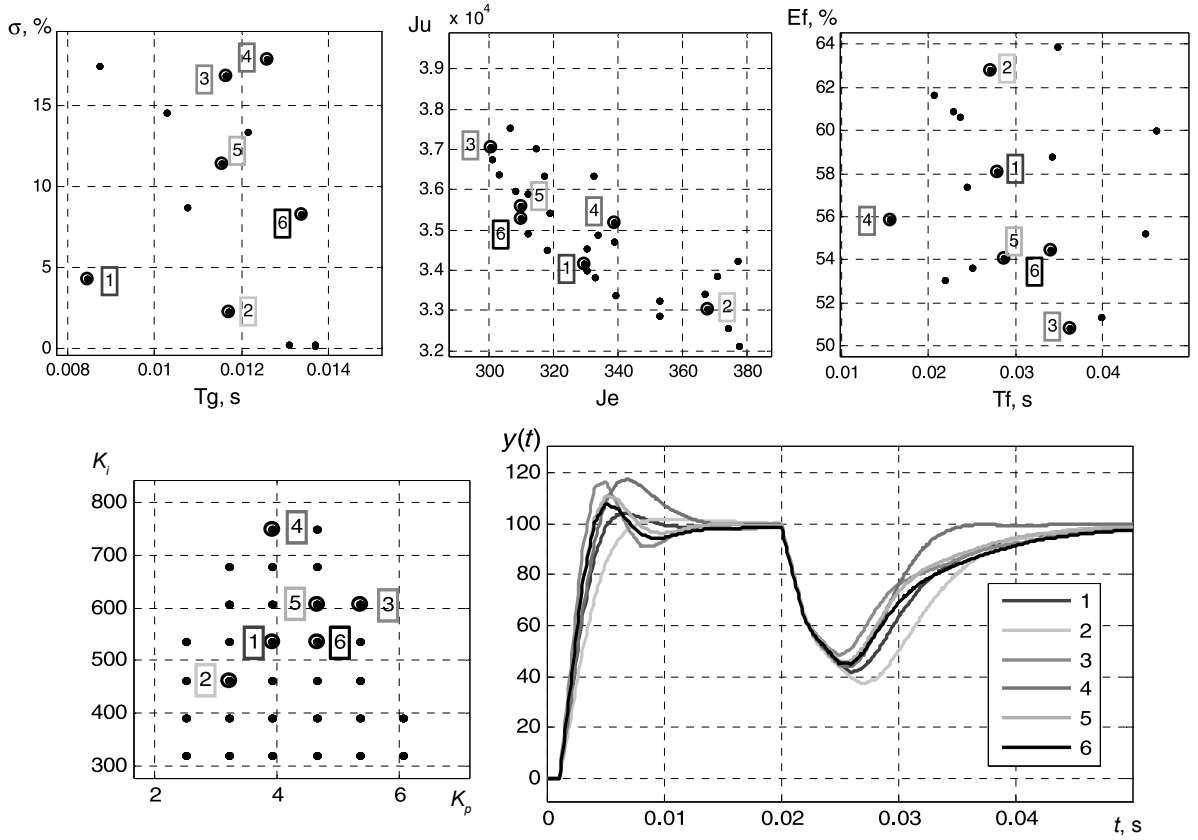


Рис. 3. Результаты зондирования и варианты настройки ПИ-регулятора тока

рового контура тока, в нелинейной модели статического ОУ которого учитываются падение напряжения якоря на щетках двигателя постоянного тока dUa (dead zone) и временное запаздывание (dead time delay) силового преобразователя (power converter). Внешним возмущением является падение напряжения на выходе силового преобразователя dU_{pc}.

Для оптимизации электромагнитных переходных процессов в контуре тока проведено прямое зондирование в пространстве двух параметров (K_p, K_i) с оцениванием критериев (F_1, F_2, F_4). Результаты зондирования представлены на рис. 3, где выделены узлы для шести вариантов настроек параметров ПИ-регулятора тока и приведены кривые соответствующих пере-

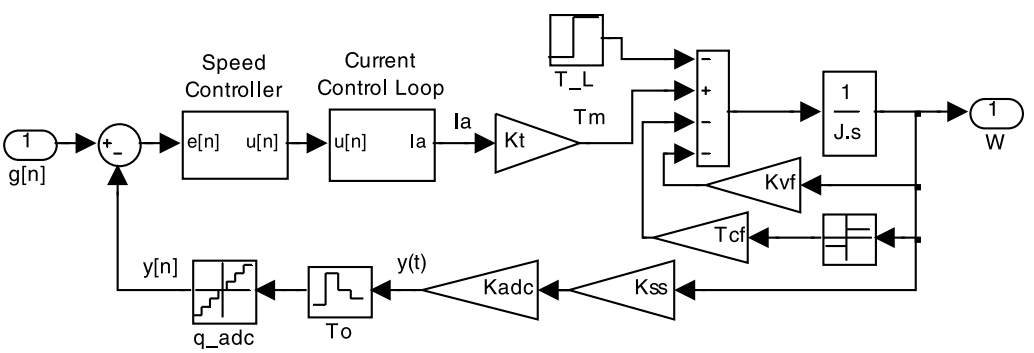


Рис. 4. Simulink-модель контура скорости

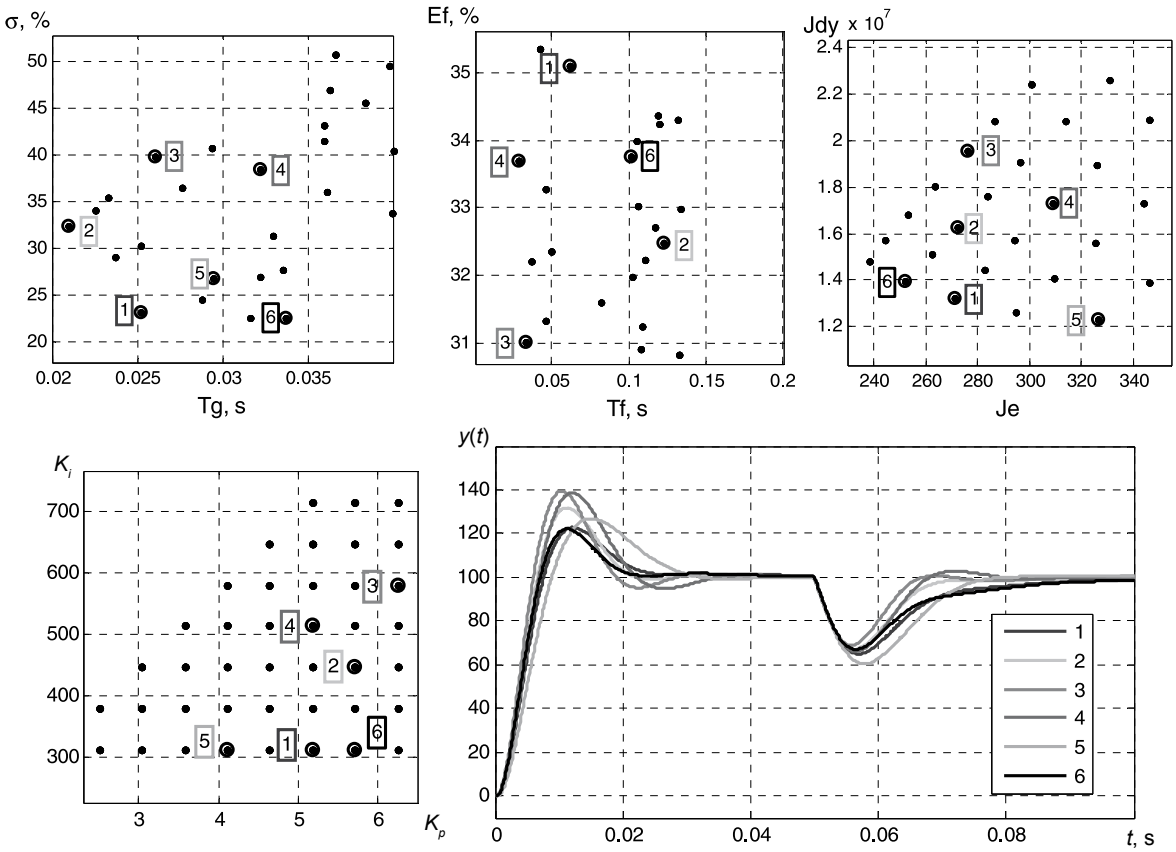


Рис. 5. Результаты зондирования и варианты настройки ПИ-регулятора скорости

ходных процессов.

Полученные варианты настроек противоречивы. Они являются Парето-оптимальными решениями, но по разным критериям: 1 и 2 – по критерию F_1; 3 и 4 – по критерию F_4; 5 и 6 – по критерию F_2.

Настройка цифрового ПИ-регулятора скорости. На рис. 4 представлена схема Simulink-модели цифрового контура скорости, в нелинейной модели ОУ которого учитывается вязкое и сухое трение на валу двигателя (K_{vf}, T_{cf}). Внешним возмущением является момент нагрузки T_L.

Для оптимизации электромеханических переходных процессов в контуре скорости необходимо прямое зондирование в пространстве двух параметров (K_p, K_i) с оцениванием критериев (F_1, F_3, F_4). Результаты зондирования представлены на рис. 5, где выделены узлы для шести вариантов настроек параметров ПИ-регулятора скорости и приведены кривые соответствующих переходных процессов.

Выбранные варианты настроек являются Парето-оптимальными по разным парам критериев: 1 и 2 – по критерию F_1; 3 и 4 – по критерию F_4; 5 и 6 – по критерию F_3.

Настройка цифрового ПИД-регулятора положения. На рис. 6 представлена схема Simulink-модели цифрового контура положения с астатическим нелинейным ОУ, в модели которого учитывается люфт в редукторе (backlash). Внешнее возмущение имитируется резким снижением скорости двигателя dW при ступенчатых моментах нагрузки.

Для оптимизации механических переходных процессов в контуре положения необходимо прямое зондирование в пространстве трех параметров (K_p, K_i, K_d) с оцениванием критериев (F_1, F_5, F_3) в соответствующих режимах. Для упрощения

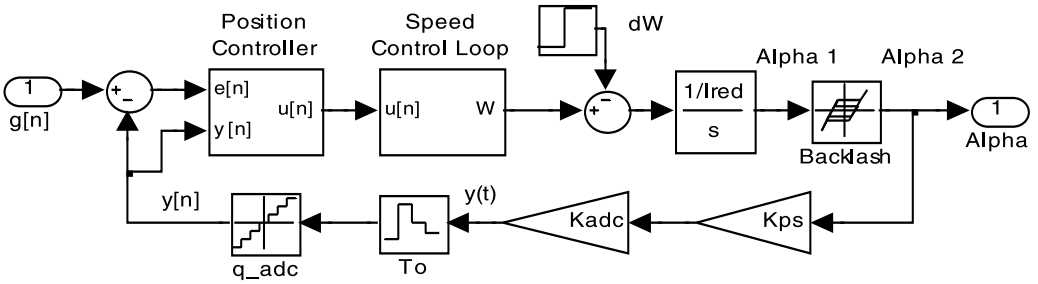


Рис. 6. Simulink-модель контура положения

на рис. 7 представлены результаты зондирования в пространстве только двух параметров (K_p, K_i) а третий параметр K_d задавался постоянным. Для выделенных трех вариантов настроек параметров ПИД-регулятора положения приведены кривые соответствующих переходных и установившихся процессов.

Выбранные варианты настроек являются Парето-оптимальными: 1 – по критерию F_1; 2 – по критерию F_5; 3 – по критерию F_3.

По результатам оптимизации контуров тока, скорости и положения можно сформулировать следующие важные утверждения.

Утверждение 1. Настройка ПИ-регулятора тока по показателям переходного процесса при ступенчатом входном воздействии, близким к модульному критерию (σ = 4,3 %), энергетически также выгодна,

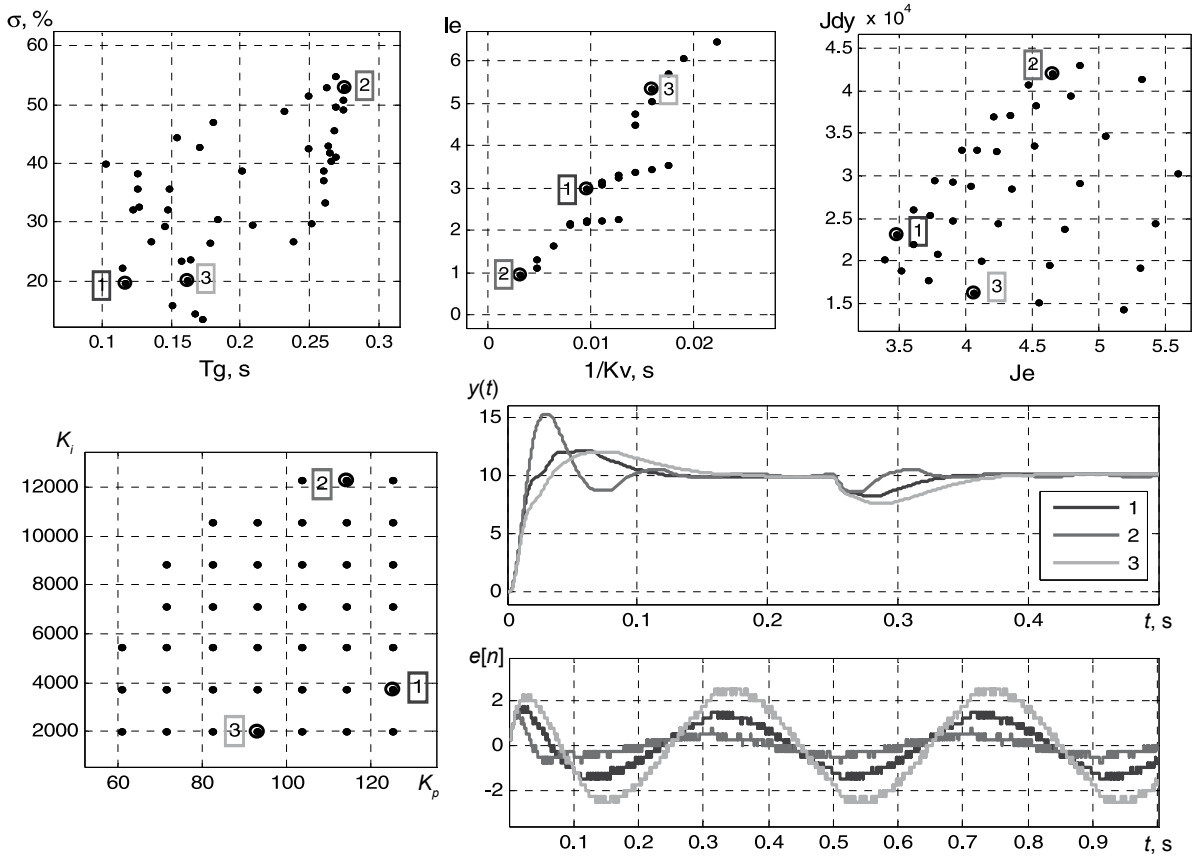


Рис. 7. Результаты зондирования и варианты настройки ПИД-регулятора положения

но не является Парето-оптимальной по показателям переходного процесса при ступенчатом возмущающем воздействии.

Утверждение 2. Настройка ПИ-регулятора скорости по показателям переходного процесса при ступенчатом входном воздействии, близким к симметричному критерию ($\sigma = 43\ \%$), является Парето-оптимальной по показателям при ступенчатом возмущающем воздействии.

Утверждение 3. Настройка ПИД-регулятора положения по критериям максимального быстродействия и наименьшего перерегулирования переходного процесса при ступенчатом входном воздействии не является Парето-оптимальной по показателям переходного процесса при ступенчатом возмущающем воздействии. Высокая добротность K_v контура положения при гармоническом входном воздействии обеспечивается настройкой переходного процесса при ступенчатом входном воздействии на большое перерегулирование ($\sigma = 50\ \%$).

Эти утверждения сделаны на основе эмпирического анализа результатов зондирования нелинейных цифровых контуров и приведены здесь без строгих доказательств, но они не противоречат известным положениям теории электропривода и линейной теории управления [1, 2]. В случаях существенного влияния нелинейностей (при малых уровнях сигналов) их справедливость на практике целесообразно проверять путем компьютерного моделирования или физическими экспериментами.

По изложенной выше методике можно сделать следующее заключение.

Парето-оптимальные параметры регуляторов потенциально обеспечивают наилучшие показатели САУ, поэтому их поиск по результатам многорежимного зондирования является практически важной задачей. При этом векторные критерии должны формироваться в соответствии с классами САУ и учетом приоритетов их показателей. В отличие от скалярной оптимизации по интегральному критерию с субъективно задаваемыми весовыми коэффициентами МК-оптимизация позволяет более объективно принимать компромиссные, технически рациональные решения по настройке параметров регуляторов.

Предложенная методика имеет общий характер, т. к. она может применяться при МК-настройке не только типовых цифровых регуляторов, но и цифровых САУ с более сложными структурами.

СПИСОК ЛИТЕРАТУРЫ

1. **Ковчин С.А., Сабинин Ю.А.** Теория электропривода: учеб. для вузов. СПб.: Энергоатомиздат, СПбО, 1994.

2. Синтез регуляторов и теория оптимизации систем автоматического управления // Методы классической и современной теории автоматического управления: учеб. в 3 т. Т. 2. Под ред. Н.Д. Егупова. М.: Изд-во МГТУ им. Н.Э. Баумана, 2000.

3. **Ростов Н.В.** Параметрическая оптимизация цифровых модальных регуляторов // Научно-технические ведомости СПбГПУ. Информатика. Телекоммуникации. Управление. СПб.: Изд-во СПбГПУ, 2010. № 3(101). С. 39–45.

4. **Ростов Н.В.** Синтез и компьютерная оптимизация цифровых последовательных регуляторов высокого порядка // Научно-технические ведомости СПбГПУ. Информатика. Телекоммуникации. Управление. СПб.: Изд-во СПбГПУ, 2010. № 4(103). С. 53–58.

5. **Ростов, Н.В.** Последовательная многокритериальная оптимизация регуляторов нелинейных систем автоматического управления // Научно-технические ведомости СПбГПУ. Информатика. Телекоммуникации. Управление. СПб.: Изд-во СПбГПУ, 2010. № 6(113). С. 44–50.

REFERENCES

1. **Kovchin S.A., Sabinin Iu.A.** *Teoriia elektroprivoda: uchebnik dlia vuzov*. St. Petersburg: Energoatomizdat, SPbO Publ., 1994. (rus)

2. Sintez reguliatorov i teoriia optimizatsii sistem avtomaticheskogo upravleniia, *Metody klassicheskoi i sovremennoi teorii avtomaticheskogo upravleniia*: uchebnik v 3 t. T. 2. Pod red. N.D. Egupova. Moscow: Izd-vo MGTU im. N.E. Baumana Publ., 2000. (rus)

3. **Rostov N.V.** Parametricheskaia optimizatsiia tsifrovykh modal'nykh reguliatorov, *Nauchno-tekhnicheskie vedomosti SPbGPU. Informatika. Telekom-*

munikatsii. Upravlenie, St. Petersburg: SPbGPU Publ., 2010, No. 3(101), Pp. 39–45. (rus)

4. **Rostov N.V.** Sintez i komp'iuternaia optimizatsiia tsifrovykh posledovatel'nykh reguliatorov vysokogo poriadka. *Nauchno-tekhnicheskie vedomosti SPbGPU. Informatika. Telekommunikatsii. Upravlenie*, St. Petersburg: SPbGPU Publ., 2010, No. 4.(103) Pp. 53–58. (rus)

5. **Rostov N.V.** Posledovatel'naia mnogokriterial'naia optimizatsiia reguliatorov nelineinykh sistem avtomaticheskogo upravleniia, *Nauchno-tekhnicheskie vedomosti SPbGPU. Informatika. Telekommunikatsii. Upravlenie*, St. Petersburg: SPbGPU Publ., 2010. No. 6(113). Pp. 44–50. (rus)

РОСТОВ Николай Васильевич – *доцент кафедры систем и технологий управления Института информационных технологий и управления Санкт-Петербургского государственного политехнического университета.*

195251, Россия, Санкт-Петербург, ул. Политехническая, д. 29.
E-mail: rostovnv@mail.ru

ROSTOV, Nikolay V. *St. Petersburg State Polytechnical University.*
195251, Politekhnicheskaya Str. 29, St. Petersburg, Russia.
E-mail: rostovnv@mail.ru

Вычислительные машины и программное обеспечение

УДК 004.052.3

А.В. Игумнов, С.Э. Сараджишвили

ВОССТАНОВЛЕНИЕ РАБОТОСПОСОБНОСТИ РЕЗЕРВИРОВАННЫХ МНОГОАГЕНТНЫХ СИСТЕМ

A.V. Igumnov, S.E. Saradgishvili

FAULT RECOVERY IN REDUNDANT MULTIAGENT SYSTEMS

Разработана модель резервированной многоагентной системы (МАС) как распределенного программно-аппаратного комплекса, основанная на избыточности задач, а не агентов. Предложена методика восстановления работоспособности резервированной МАС, включающая в себя протокол взаимодействия между компонентами МАС, обеспечивающий выполнение задачи независимо от ее расположения в компонентах МАС, технику поддержания неактивных резервных задач в актуальном состоянии и техники обнаружения отказов и восстановления работоспособности МАС. Достоверность разработанных моделей и методики восстановления работоспособности подтверждена сформулированной и доказанной теоремой о свойстве сохранения работоспособности МАС.

МНОГОАГЕНТНАЯ СИСТЕМА; ИЗБЫТОЧНОСТЬ; РЕЗЕРВИРОВАНИЕ; ОТКАЗ; ВОССТАНОВЛЕНИЕ РАБОТОСПОСОБНОСТИ.

The model of redundant multiagent system (MAS) as distributed hardware-software complex, which is based on replication of tasks instead of replication of agents, is developed. The developed fault recovery methodology presented in the article includes communication protocol for components of MAS, which enables task execution independently of its deployment or belonging to a particular component of MAS, technique for keeping inactive task replicas in actual state and techniques for fault detection and fault recovery. Developed models and methodology are validated by stated and proved theorem on fault tolerance property of MAS.

MULTIAGENT SYSTEM; REDUNDANCY; REPLICATION; FAULT; FAULT RECOVERY.

Применение агентно-ориентированных технологий [1] в промышленных системах определяет необходимость исследования надежности многоагентных систем (МАС) и разработки средств и методов обеспечения отказоустойчивости МАС. Введение избыточности является основным средством обеспечения отказоустойчивости технических систем [2]. Известные методики обеспечения отказоустойчивости МАС, такие как DARX [3], Meta-Agent [4] и Brokered MAS [5], используют избыточность агентов, однако в работе [6] предложено использование избыточности не только агентов, но и задач. Существующие методики обеспечения отказоустойчивости МАС обладают следующими недостатками:

рассматривают МАС только как систему, состоящую из взаимодействующих агентов [7], в то время как современные прикладные и промышленные системы, как правило, являются программно-аппаратными комплексами;

используют резервирование как метод повышения уровня надежности, но не определяют множество отказов и типы отказов, к которым МАС должна быть гарантированно устойчива.

Цель исследования – обеспечить восстановление работоспособности МАС,

реализованной в виде распределенного программно-аппаратного комплекса. В рамках работы решены следующие задачи:

- разработана формальная модель МАС как программно-аппаратного комплекса, учитывающая распределение задач по агентам и агентов по аппаратным компонентам, а также наличие доступа к ресурсам;
- разработана формальная модель резервированной МАС, основанная на введении избыточности задач, а не агентов МАС;
- определена классификация отказов компонентов МАС, устойчивость к которым необходимо обеспечить;
- разработана методика восстановления работоспособности МАС, основанная на модели резервированной МАС, включающая в себя:

протокол взаимодействия компонентов программно-аппаратной МАС, обеспечивающий выполнение задачи определенного типа независимо от ее расположения в компонентах МАС и использование исполнительного ресурса определенного типа;

техники обнаружения функциональных и логических отказов задач, отказов аппаратных компонентов МАС;

техники восстановления работоспособности для каждого из компонентов программно-аппаратной МАС;

технику поддержания неактивных резервных компонентов МАС в актуальном состоянии при изменениях условий выполнения своих действий интеллектуальными агентами;

- сформулирована и доказана теорема о свойстве сохранения работоспособности резервированной МАС, подтвердившая достоверность разработанных моделей и методики восстановления работоспособности.

Формальная модель МАС

Существующие методики обеспечения отказоустойчивости МАС основаны на моделях, представляющих МАС как систему взаимодействующих агентов. Поэтому для разработки методики восстановления работоспособности распределенной программно-аппаратной МАС необходимо разработать ее формальную модель, учитывающую не только взаимодействие отдельных агентов, но и необходимость использования аппаратных компонентов как для исполнения программных агентов, так и для выполнения агентами отдельных задач. Кроме того, т. к. агенты МАС способны модифицировать свои реакции на состояние окружающей среды, т. е. изменять условия запуска своих задач, то формальная модель МАС должна включать в себя компоненты, ответственные за определение данных условий.

Разработанная формальная модель МАС основана на следующих элементах модели, предложенной авторами работы [6]: $A = \{a_i\}$ – множество агентов МАС; $E = \{s_j\}$ – воздействий на систему; $T = \{t_q\}$ – задач МАС; $R = \{r_k\}$ – ресурсов, доступных МАС; $resources(t) \subseteq R$ – множество ресурсов, необходимых для выполнения задачи t; $prect(t) = \{c = s_1 \wedge ... \wedge s_u \mid \forall i \in 1...u : s_i \in E\}$ и $post(t)$ – множества предусловий и постусловий задачи t.

Введем понятия агентной платформы (АП) и исполнительного ресурса. Агентная платформа – это программно-аппаратный компонент системы, предназначенный для исполнения агентов. Введем множество $HWP = \{hwp_i\}$ АП МАС. Пусть предикат $cA\_HWP : A \times HWP \to \{true, false\}$ определяет принадлежность агента АП, тогда функция $AofHWP : HWP \to \varphi(A), \varphi(A) \subseteq A$ определяет множество агентов $AofHWP(hwp) = \{a_i \in A \mid \forall i : cA\_HWP(a_i, hwp) = true\}$, исполняющихся на АП hwp. Так как согласно [6] агент МАС a исполняет ряд задач, то введем предикат $cT\_A : T \times A \to \{true, false\}$, определяющий принадлежность задачи агенту. Тогда функция $TofA : A \to \varphi(T)$ определяет множество задач $TofA(a) = \{t_i \in T \mid \forall i : cT\_A(t_i, a) = true\}$, принадлежащих агенту a, а функция $AofT : T \to \varphi(A)$ определяет агента a, которому принадлежит задача t: $a = AofT(t) \Leftrightarrow cT\_A(t, a) = true$. Введем функцию $TofHWP : HWP \to \varphi(T)$, определяющую множество задач $TofHWP(hwp) = \{t_i \in T \mid \forall i : \exists a \in AofHWP(hwp) : cT\_A(t_i, a) = true\}$, исполняемых АП hwp. Исполнительный ресурс (ИР) – это аппаратный компонент системы, необходимый агенту для выполнения

его задач. Введем множество $HWR = \{hwr_i\}$ ИР МАС. Пусть подмножество ресурсов $\{R \backslash HWR\}$, не являющихся ИР, включает только программные ресурсы, которые могут быть предоставлены любой АП, тогда можно считать, что множество ресурсов МАС содержит только ИР. Доступность отдельного ИР определяется конфигурацией МАС, поэтому пусть предикат $cHWR\_HWP : HWR \times HWP \to \{true, false\}$ определяет доступность ИР hwr для АП hwp. Тогда функция $HWRofHWP : HWP \to \varphi(HWR)$ определяет множество ИР $HWRofHWP(hwp) = \{hwr_i \in HWR \mid \forall i : cHWR\_HWP(hwr_i, hwp) = true\}$, доступных АП hwp. Пусть предикат $cT\_HWR : T \times HWR \to \{true, false\}$ задает необходимость использования ИР задачей, тогда функция $rHWRofT : T \to \varphi(HWR)$ определяет множество необходимых для выполнения задачи t ИР $rHWRofT(t) = \{hwr_i \in HWR \mid \forall i : cT\_HWR(t, hwr_i) = true\}$.

В случае закрытой МАС [8] необходимость выполнения задачи может определяться наблюдаемым состоянием МАС [6]. Так как интеллектуальные агенты способны модифицировать свои реакции на состояние МАС, то необходимость выполнения задачи должна определяться блоком принятия решений (БПР) агента. Будем считать, что каждой задаче $t \in TofA(a)$ агента a можно поставить в соответствие БПР, определяющий необходимость ее выполнения. Введем множество $DM = \{dm_i\}$ БПР, тогда предикат $cT\_DM : T \times DM \to \{true, false\}$ определяет соответствие БПР и задачи.

Исходная МАС задана множеством $MAS = <T, A, HWP, HWR, E, DM>$, где A – множество агентов, DM – БПР, T – задач, E – воздействий на систему, HWR – множество ИР (ресурсов), HWP – АП.

Модель резервированной МАС

Разработанная модель МАС позволяет определить ряд отказов отдельных компонентов, приводящих к отказу МАС:

отказ АП hwp, приводящий к отказу задач $TofHWP(hwp)$ данной АП;

программный отказ агента a, приводящий к отказу задач $TOfA(a)$;

программный отказ задачи;

логический отказ задачи t, т. е. невыполнение хотя бы одного из предусловий из множества $prec(t)$;

функциональный отказ задачи t, т. е. невыполнение хотя бы одного из постусловий из множества $post(t)$;

отказ ИР hwr, приводящий к отказу множества задач $\{t_i \in T \mid \forall i : cT\_HWR(t_i, hwr) = true\}$, использующих данный ИР.

В соответствии с [2] основным средством обеспечения отказоустойчивости системы является введение избыточности ее компонентов. Для обеспечения отказоустойчивости программно-аппаратной МАС будем использовать резервирование задач системы. Кроме того, будем считать, что для обеспечения отказоустойчивости может быть использовано резервирование ИР, необходимых для выполнения задач, а также могут быть введены дополнительные агенты и АП. Для разработки техник обнаружения отказов, восстановления работоспособности, поддержания актуального состояния неактивных резервных компонентов необходимо разработать формальную модель резервированной программно-аппаратной МАС, основанную на резервировании задач и ИР и введении дополнительных агентов и АП.

В работе [6] высказано предположение, что логический отказ задачи может быть преодолен, если воздействия на МАС, определяющие предусловия задачи, подконтрольны МАС, однако механизм контроля не был предложен. Определим эффектор как задачу, способную изменять значение одного из воздействий на МАС. Введем множество $EF = \{ef_i\}$ эффекторов и предикат $cEF\_ST : EF \times E \to \{true, false\}$, задающий соответствие эффектора и воздействия на МАС. Определим внутренний контекст МАС как множество $IC = \{s_i \in E \mid \forall i : \exists ef \in EF : cEF\_ST(ef, s_i) = true\}$ тех воздействий, для которых существуют эффекторы.

Эффекторы и задачи исходной МАС образуют расширенное множество задач МАС $ET = T \cup EF$. Для обеспечения отказоустойчивости предлагается вводить

избыточность задач и соответствующих им БПР. Введем множество типов задач $TT = \{tt_i\}$ так, чтобы каждой задаче $et_i \in ET$ соответствовал один тип задачи. Для эффекторов соответствие типа задачи элементу внутреннего контекста задается функцией $TTofST : IC \to \varphi(TT)$. Пусть $RT = \{t_i\}$ – множество задач резервированной МАС, тогда функция $TYofT : RT \to \varphi(TT)$ определяет тип задачи. Из группы задач-реплик одного типа только одна реплика является активной, а агент МАС исполняет только БПР, соответствующие активным репликам. Пусть предикат $cATofTT : RT \times TT \to \{true, false\}$ определяет, является ли задача активной репликой. В резервированной МАС вместо запроса на исполнение задачи вводится запрос на исполнение задачи определенного типа, обработка которого приводит к выполнению активной реплики.

После введения избыточности задач необходимо определить структуру резервированной МАС, при этом должны быть решены следующие задачи: введение избыточности ИР, введение дополнительных агентов и АП, распределение задач по агентам и агентов по АП. Решение поставленных задач должно быть обеспечено некоторой методикой синтеза структуры резервированной МАС, не описанной в данной статье. Так как допускается использование избыточности ИР, то введем множество типов ИР $HWRT = \{hwrt_i\}$ и множество ИР резервированной МАС $RHWR = \{hwr_i\}$, тогда функция $TYofHWR : RHWR \to \varphi(HWRT)$ определяет тип ИР. Пусть предикат $cTT\_HWRT : TT \times HWRT \to \{true, false\}$ задает необходимость использования ИР определенного типа задачей определенного типа, тогда функция $rHWRTofTT : TT \to \varphi(HWRT)$ определяет множество типов ИР $rHWRTofTT(tt) = \{hwrt_i \in HWRT \mid \forall i : cTT\_HWRT(tt, hwrt_i) = true\}$, необходимых для выполнения задачи типа *tt*.

Резервированная МАС задана множеством $RMAS = <RT, RA, RHWP, RHWR, E, DM, IC, TT, HWRT>$, где RA – множество агентов; DM – множество БПР; RT – задач; E – воздействий на систему; IC – внутренний контекст; $RHWR$ – множество ИР; $RHWP$ – АП; TT – множество типов задач; $HWRT$ – типов ИР. Конфигурация резервированной МАС задана предикатами и функциями: $cT\_A : RT \times RA \to \{true, false\}$ определяет распределение задач по агентам; $cA\_HWP : RA \times RHWP \to \{true, false\}$ – распределение агентов по АП; $cHWR\_HWP : RHWR \times RHWP \to \{true, false\}$ – доступность отдельного ИР для АП; cTT_HWRT – необходимость использования ИР определенного типа задачей определенного типа; $cT\_DM : RT \times DM \to \{true, false\}$ определяет соответствие БПР и задачи; $TTofST$ определяет тип эффектора, контролирующего элемент внутреннего контекста; $TYofT$ и $TYofHWR$ определяют тип задачи и ИР; $precTT : TT \to \{c = s_1 \wedge ... \wedge s_u \mid \forall i \in 1...u : s_i \in E\}$ – предусловия для задачи определенного типа; $cATofTT$ – активную реплику для каждого типа задачи.

Протокол взаимодействия компонентов МАС

Для обеспечения взаимодействия компонентов резервированной МАС необходимо разработать соответствующий протокол взаимодействия. Протокол взаимодействия должен обеспечивать прозрачность резервирования, т. е. возможность запуска задачи определенного типа без необходимости явного выбора конкретной задачи из группы реплик, а также независимо от размещения активной реплики данного типа в агентах и АП МАС. Аналогично протокол взаимодействия должен обеспечивать использование ИР определенного типа без явного указания конкретного ИР. Также протокол взаимодействия компонентов МАС необходим для поддержания актуального состояния неактивных резервных компонентов, обнаружения отказов и восстановления работоспособности системы.

Введем протокол обмена сообщениями. Пусть функция *send*(*msg*(*args*), *dest*) означает посылку сообщения *msg* с параметрами *args* получателю *dest*. Передача сообщений возможна между АП, задачей и агентом, агентом и АП. Введем особые значения параметра *dest*: *up* – получателем является вышестоящий компонент МАС, т. е.

агент для задачи и АП для агента; *bcst* – сообщение отправлено всем АП. Функция *dest:recv*(*msg*(*args*), *src*) означает обработку компонентом *dest* сообщения, полученного от отправителя *src*.

Агент МАС должен обладать базой данных (БД), содержащей информацию о задачах, необходимых типах ИР и наличии активных реплик. БД может быть реализована в виде таблиц, содержащих записи вида: (*t*, *tt*), связывающие идентификатор задачи и тип задачи; (*tt,* {*hwrt*}), связывающие тип задачи и необходимые типы ИР; (*tt*, *t*), определяющие наличие активных реплик. БД агента обеспечивает реализацию функций *TofA*, *TYofT*, *rHWRTofTT* и позволяет ввести функцию $ATofTTinA : RA \times TT \to \varphi(RT)$, определяющую задачу, являющуюся активной репликой определенного типа. Агент должен реализовывать процедуры для модификации БД: *a_setATofTT*(*t*, *tt*) обеспечивает обновление таблицы активных реплик так, чтобы типу *tt* соответствовала реплика *t*; *a_rm_t*(*t*) – удаление всех записей о задаче *t*.

АП должна обладать БД, содержащей информацию об агентах, доступных ИР, а также о расположении активных реплик всех типов как в агентах данной АП, так и в агентах удаленных АП. БД АП может быть реализована в виде таблиц, содержащих записи вида: (*a*, *t*, *tt*), определяющие распределение задач (и их типы) по агентам; (*hwr*, *hwrt*), определяющие доступные ИР и их типы; (*tt*, {*hwrt*}), определяющие необходимые типы ИР в соответствии с типом задачи; {*tt*, *hwp*, *a*}, определяющие расположение активных реплик. Пусть функции $AHWPofTT : TT \to \varphi(RHWP)$ и $AAofTT : TT \to \varphi(RA)$ определяют (согласно БД АП) АП и агента, которым принадлежит активная реплика определенного типа. БД АП *hwp* должна содержать информацию о размещении активных реплик каждого типа для обеспечения взаимодействия между любыми задачами и агентами, т. е. должно быть выполнено условие $\forall tt \in TT : \exists hwp^* = AHWPofTT(tt) \wedge ((hwp^* = hwp) \Rightarrow \exists a = AAofTT(tt))$. Информация об агенте и размещенных в нем активных репликах вносится в БД при получении сообщения *a_cfg_anc*, которое посылается агентом при размещении на АП и изменениях БД агента. Информация о размещении активных реплик в удаленных АП – при получении сообщения *arepl_anc*(*tt*, *hwp*), которое посылается каждой АП при активизации реплики. БД АП обеспечивает реализацию функций *AofHWP*, *TofHWP*, *TYofT*, *TofA*, *AofT*, *rHWRTofTT*, *HWRofHWP*, *TYofHWR*. АП должна реализовывать процедуры для модификации БД: *hwp_rm_t*(*t*) обеспечивает удаление записей о задаче *t*; *hwp_rm_hwr*(*hwr*) – удаление записи о ИР *hwr* из таблицы доступных ИР; *hwp_setAAofTT*(*tt*, *a*) – обновление таблицы активных реплик так, чтобы типу *tt* соответствовал агент *a*; *hwp_rmATofTT*(*tt*) – удаление из таблицы активных реплик записи о типе *tt*. АП может инициировать изменение конфигурации любого из ее агентов, т. е. запросить выполнение агентом *a* процедур *a_setATofTT* и *a_rm_t* через отправку сообщений *a_cmd_AT*(*t*) и *a_cmd_rm*.

Задачи МАС могут требовать исполнения других задач, что в рамках модели резервированной МАС означает требование исполнения задачи определенного типа. Агент МАС *a* может запустить выполнение задачи $t \in TofA(a)$ при помощи процедуры *a_pfm_t*(*t*). Запрос на исполнение задачи определенного типа реализуется путем посылки сообщения *pfm*. Сообщение *pfm,* полученное от задачи, обрабатывается агентом: *a:recv*(*pfm*(*tt*), *t*) {if $(\exists t^* = ATofTTinA(a, tt))$ then *a_pfm_t*(*t**) else *send*(*pfm*(*tt*), *up*)}. Обработка сообщения *pfm,* полученного от агента, осуществляется АП: *hwp:recv*(*pfm*(*tt*), *a*) {$hwp^* = AHWPofTT(tt)$; if ($hwp^* == hwp$) then *send*(*pfm*(*tt*), *AAofTT*(*tt*)) else *send*(*pfm*(*tt*), *hwp**)}. Обработка сообщения *pfm*, полученного от удаленной АП, осуществляется АП: *hwp:recv*(*pfm*(*tt*), *hwp**) {*send*(*pfm*(*tt*), *AAofTT*(*tt*)}. Сообщение *pfm*, полученное от АП, обрабатывается агентом: *a:recv*(*pfm*(*tt*), *hwp*) {*a_pfm_t*(*ATofTTinA*(*a*, *tt*))}.

Запрос задачи на использование ИР определенного типа реализуется путем посылки сообщения *send*(*use_hwr*(*hwrt*), *up*), которое пересылается агентом вышестоящей АП: *a:recv*(*use_hwr*(*hwrt*), *t*) {*send*(*use_hwr*(*hwrt*, *up*)}. Так как АП имеет непосредственный

доступ к ИР, то АП *hwp* может использовать ИР $hwr \in HWRofHWP(hwp)$ посредством процедуры *hwp_use_hwr*(*hwr*), возвращаемое значение которой указывает на успешное использование ИР или его отказ. При обработке сообщения *use_hwr* АП должна обеспечить использование ИР или зафиксировать его отказ: *hwp:recv*(*use_hwr*(*hwrt*), *a*) {if not (($\exists hwr \in HWRofHWP(hwp)$: $TYofHWR(hwr) = hwrt$) AND (*hwp_use_hwr*(*hwr*))) then {*hwp_rm_hwr*(*hwr*); *hwp_hwr_fail*(*hwrt*)}.

Поскольку необходимость выполнения задачи определяется БПР, который может быть модифицирован интеллектуальным агентом, и только БПР, соответствующий активной реплике, исполняется агентом, необходимо обеспечить возможность модификации БПР неактивных реплик. Если агент *a* фиксирует изменение БПР *dm*, соответствующего задаче *t* (т. е. $cT\_DM(t, dm) = true$), то он, определив тип задачи $tt = TYofT(t)$, выполняет процедуру *update_dm*(*tt*, *dm*) и посылает сообщение *send*(*upd_dm*(*tt*, *dm*), *up*). Процедура *update_dm*(*tt*, *dm*) заключается в модификации БПР всех задач агента определенного типа $\{t \in TofA(a) \mid TYofT(t) = tt\}$. АП пересылает сообщение *upd_dm* всем своим агентам и удаленным АП: *hwp:recv*(*upd_dm*(*tt*, *dm*), *a*) {foreach ($a^* \in AofHWP(hwp)$) *send*(*upd_dm*(*tt*, *dm*), *a*); *send*(*upd_dm*(*tt*, *dm*), *bcst*)}. Согласно алгоритму обработки сообщения *upd_dm*, оно будет доставлено всем агентам МАС, которые при его получении выполнят процедуру *update_dm*. Этот механизм позволяет поддерживать актуальное состояние всех неактивных реплик, т. к. такие задачи информируются об изменениях в БПР активной реплики, т. е. об изменениях условий своего запуска.

Обнаружение отказов компонентов МАС

Для восстановления работоспособности системы необходимо зафиксировать состояние отказа. Поэтому определим техники обнаружения отказов, которым подвержена программно-аппаратная МАС, заданная предложенной нами формальной моделью.

Для обнаружения программных отказов задач и агентов предлагается использовать существующие техники сторожевых watchdog-таймеров [9] в агентах и АП. Для обнаружения функционального отказа задачи агент должен контролировать выполнение постусловий после завершения задачи. Функциональный отказ задачи определяется условием $\exists c = s_1 \wedge ... \wedge s_k \in post(t) : (\forall i \in 1...k : s_i \in E) \quad \wedge(\exists u \in 1...k : s_u = false)$. Для обнаружения логического отказа задачи агент должен контролировать выполнение предусловий перед запуском задачи. Логический отказа задачи определяется условием $\exists c \in precTT(TYofT(t)) : c = false$. Отказ ИР фиксируется АП, взаимодействующей с этим ИР, в процессе обработки сообщения *use_hwr*. Условием отказа ИР является отсутствие доступного АП ИР запрошенного типа $\forall hwr \in HWRofHWP(hwp) : TYofHWR(hwr) \neq \neq hwrt$ или отрицательный результат выполнение процедуры *hwp_use_hwr*().

Для обнаружения отказа АП необходимо использовать протокол рукопожатия. Пусть каждая АП *hwp* МАС сообщает о своей жизнеспособности путем посылки сообщения *send*(*alv*(*hwp*), *bcst*). Если сообщения *alv* посылаются каждой АП с периодичностью *t_a* и моменты посылки сообщений не синхронизированы, то любая АП, анализируя принятые сообщения в течение периода времени $t\_a + \Delta t$, может определить множество АП *ALV*, с которыми имеется связь, и множество АП *DEAD*, с которыми связь отсутствует. Пусть в начале цикла обнаружения отказов АП выполняет процедуру *hwp_startd* {$ALV = 0$}, а в течение цикла происходит следующая обработка *alv* сообщений: *hwp:recv*(*alv*(*hwp**), *hwp**) {$ALV+=hwp^*$}, тогда при завершении цикла может быть определено множество $DEAD = (RHWP \backslash ALV) \backslash \{hwp\}$. Если АП *hwp* фиксирует отсутствие связи с АП *hwp**, то возможен либо отказ *hwp**, либо отказ одной из линий связи между *hwp* и *hwp**. Будем считать, что все АП МАС объединены сетью, организованной согласно топологии звезда или шина [10]. Тогда отказ линии связи между АП *hwp* и центральным звеном сети (общей шиной или коммутатором) означает, что *hwp* не получит *alv* сообщений ни от одной другой АП МАС, а ни одна другая АП МАС *hwp** не получит *alv*

сообщения от *hwp*. Если АП *hwp* фиксирует отсутствие связи со всеми удаленными АП, что определяется условием $DEAD = RHWP \setminus \{hwp\}$, то АП не может определить находятся ли все удаленные АП в состоянии отказа или отсутствие связи обусловлено отказом линии связи между самой АП и центральным звеном сети, поэтому АП фиксирует свой собственный отказ и останавливает работу всех размещенных в ней агентов. Если АП фиксирует отказ части удаленных АП, то она должна выполнить процедуру *hwp_rhwp_fail* для восстановления, значит, процедура окончания цикла обнаружения отказов должна быть реализована следующим образом: *hwp_endd*() {$DEAD = (RHWP \setminus ALV) \setminus \{hwp\}$; if ($DEAD \neq RHWP \setminus \{hwp\}$) then foreach ($hwp^* \in DEAD$) *hwp_rhwp_fail*($hwp^*$)}.

Восстановление работоспособности резервированной МАС

Для восстановления работоспособности резервированной МАС, заданной разработанной нами формальной моделью, необходимо разработать процедуры, которые должны быть выполнены различными компонентами МАС в случае обнаружения отказов. При этом будем считать, что процесс восстановления работоспособности должен быть организован иерархически. Так, первая попытка восстановления работоспособности предпринимается компонентом, зафиксировавшим отказ. В случае неудачи задача восстановления работоспособности делегируется компоненту, в котором размещен компонент, зафиксировавший отказ. Если восстановление работоспособности невозможно в рамках одной АП, то решение данной задачи обеспечивается совместными действиями всех АП МАС.

Обнаруженный логический отказ задачи может быть преодолен, если невыполнение предусловий обусловлено воздействиями на МАС $s_i \in IC$, принадлежащими внутреннему контексту. Множество воздействий, обусловливающих невыполнение предусловий задачи *t*, определяется как $SF = \{sf_i \mid \forall i : sf_i = false \wedge (\exists c = s_1 \wedge \ldots \wedge sf_i \wedge \ldots \wedge s_k \in precTT(TYofT(t)))\}$. Определим условие восстановления работоспособности после логического отказа задачи: $\forall sf \in SF : sf \in IC$. При обнаружении логического отказа задачи *t* агент *a* выполняет следующую процедуру для вызова необходимых эффекторов *a_t_lfail*(*t*) {if ($\forall sf \in SF : sf \in IC$) then foreach ($sf \in SF$) {$tt = TTofST(sf)$; if ($\exists t^* = ATofTTinA(a, tt)$) then *a_pfm_t*($t^*$) else *send*(*pfm*(*tt*), *up*)}}.

Так как любой из отказов компонентов МАС приводит к отказу одной или нескольких активных реплик, то восстановление работоспособности сводится к нахождению и активизации задачи-реплики соответствующего типа. Если реплика не найдена в рамках агента, зафиксировавшего отказ задачи, то запрос на поиск реплики передается АП, в которой размещен агент, в виде сообщения *req_repl*. При обнаружении функционального или программного отказа задачи *t* агент *a* выполняет процедуру *a_t_fail*(*t*) {$tt = TYofT(t)$; *a_rm_t*(*t*); if ($\exists t^* \in TofA(a) : TYofT(t^*) = tt$) then *a_setATofTT*($t^*$, *tt*) else *send*(*req_repl*(*tt*), *up*)}. При получении от агента сообщения *req_repl* АП выполняет процедуру поиска и активизации реплики: *hwp:recv*(*req_repl*(*tt*), *a*) {*hwp_t_fail*(*tt*)}. Если реплика не найдена в рамках АП, то запрос на поиск реплики передается всем удаленным АП: *hwp_t_fail*(*tt*) {*hwp_rmATofTT*(*tt*); if ($\exists t \in TofHWP(hwp) : TYofT(t) = tt$) then {$a = AofT(t)$; *hwp_setAAofTT*(*tt*, *a*); *send*(*a_cmd_AT*(*t*), *a*)} else *send*(*req_repl*(*tt*), *bcst*)}. Поиск и активизация реплики в удаленных АП осуществляется при приеме сообщения: *hwp:recv*(*req_repl*(*tt*), *hwp**) {*hwp_rmATofTT*(*tt*); if ($\exists t \in TofHWP(hwp) : TYfT(t) = tt$) then {$a = AofT$ (*t*); *hwp_setAAofTT*(*tt*, *a*); *send*(*a_cmdAT*(*t*), *a*); *send*(*arepl_anc*(*tt*, *hwp*), *bcst*)}}.

Программный отказ агента *a* является причиной отказов активных задач-реплик, размещенных в данном агенте. При обнаружении программного отказа агента *a* АП *hwp* определяет множество типов отказавших активных реплик $TTF = \{tt \in TT \mid AAofTT(tt) = a\}$ и выполняет процедуру, обеспечивающую поиск и активизацию соответствующих задач-реплик: *hwp_a_fail*(*a*) {foreach ($t \in TofA(a)$) *hwp_rm_t*(*t*); foreach ($tt \in TTF$) *hwp_t_fail*(*tt*);}.

Отказ ИР $hwr \in HWRofHWP(hwp)$ при отсутствии других доступных АП *hwp* ИР типа $hwrt = TYofHWR(hwr)$ приводит к от-

казам активных задач-реплик, для выполнения которых необходимо использование ИР типа *hwrt*. Если АП *hwp* фиксирует отказ ИР типа *hwrt*, то АП определяет множество типов отказавших активных задач-реплик $TTFR = \{tt \in TT \mid hwrt \in rHWRTofTT(tt) \wedge \exists a \in AofHWP(hwp) : AAofTT(tt) = a\}$, а также множество задач АП $TFR = \{t \in TofHWP(hwp) \mid$ которые требуют наличия ИР типа *hwrt*. Так как задачи из *TFR* не могут быть выполнены на данной АП в силу недоступности необходимого ИР, то следует удалить информацию о них из БД агентов и АП, чтобы эти задачи не рассматривались в процессе поиска и активизации реплик. Восстановление работоспособности обеспечивается процедурой: *hwp_hwr_fail*(*hwrt*) {foreach ($t \in TFR$) {*hwp_rm_t*(*t*); *send*(*a_cmd_rm_t*(*t*), *AofT*(*t*));}; foreach ($tt \in TTFR$) *hwp_t_fail*(*tt*)}.

Отказ АП *hwp** приводит к отказам всех активных задач-реплик, размещенных на данной АП. Если АП *hwp* фиксирует отказ удаленной АП *hwp**, то необходимо определить множество типов отказавших активных задач-реплик $TTFH = \{tt \in TT \mid AHWPofTT(tt) = hwp^*\}$. Поиск и активизация соответствующих задач-реплик осуществляется процедурой *hwp_rhwp_fail*(*hwp**) {foreach ($tt \in TTFH$) {*hwp_t_fail*(*tt*); if ($\exists a = AAofTT(tt)$) then *send*(*arepl_anc*(*tt*, *hwp*), *bcst*)}}.

Доказательство свойства сохранения работоспособности

Покажем, что использование разработанных процедур, составляющих методику восстановления работоспособности МАС, и разработанного протокола взаимодействия компонентов МАС обеспечивает сохранение работоспособности резервированной программно-аппаратной МАС, заданной предложенной нами формальной моделью.

Сформулируем теорему. При функциональных, логических и программных отказах задач, программных отказах агентов, отказах АП и ИР резервированная МАС сохраняет работоспособность, т. е. обеспечивает выполнение задачи типа *tt*, если после обнаружения отказа в системе существует задача-реплика типа *tt*, имеющая доступ к необходимым для ее выполнения ИР, и логический отказ задачи обусловлен только воздействиями, принадлежащими внутреннему контексту МАС.

Доказательство. Выполнение задачи типа *tt* может быть обусловлено реакцией МАС или запросом другой задачи. В первом случае задача будет выполнена, если существует активная реплика данного типа, т. к. реакция МАС определяется БПР, а агенты МАС исполняют только БПР, соответствующие активным репликам. Во втором случае согласно протоколу взаимодействия компонентов МАС обработка сообщения *pfm*(*tt*) обеспечивает исполнение активной реплики типа *tt* независимо от ее размещения в компонентах МАС. Следовательно, МАС обеспечивает выполнение задачи типа *tt*, если существует активная реплика данного типа t^*: $cATofTT(t^*,tt) = true$.

Рассмотрим случай программного или функционального отказа задачи *t*. В соответствии с условием теоремы существует задача t^*: $TYofT(t^*) = TYofT(t)$, которая должна быть активирована для сохранения работоспособности. Отказ задачи *t* фиксируется агентом *a*: $cT\_A(t,a) = true$. Если задача t^* принадлежит агенту *a*, то ее активизация обеспечивается процедурой *a_t_fail*(*t*). В противном случае агент *a* передает запрос на активизацию АП, в которой он размещен. АП согласно процедуре *hwp_t_fail*(*tt*) обеспечивает активизацию задачи t^*, если она принадлежит данной АП, или передает запрос на активизацию всем удаленным АП. Если задача t^* принадлежит одной из удаленных АП, то она будет активирована при получении запроса *req_repl*. Значит, при обнаружении программного или функционального отказа задачи будет активирована соответствующая реплика, т. е. сохранена работоспособность МАС. Программный отказ агента *a* приводит к отказам всех активных задач-реплик, размещенных в агенте. Отказ агента фиксируется АП, в которой он размещен. Согласно процедуре *hwp_a_fail*(*a*) АП определяет множество отказавших активных реплик и выполняет для каждого из них процедуру *hwp_t_fail*, которая, как было доказано, обеспечивает активизацию задачи t^* типа *tt* независимо от ее размещения в компонен-

тах МАС. Следовательно, если выполнено условие теоремы, при отказе агента *a* для каждой из размещенных в нем активных реплик будет найдена и активирована новая реплика, т. е. будет восстановлена работоспособность МАС. Отказ АП *hwp** приводит к отказам всех активных задач-реплик, размещенных в агентах данной АП. АП *hwp* согласно протоколу рукопожатия фиксирует отказ АП *hwp**, согласно процедуре *hwp_rhwp_fail*(*hwp**) определяет множество типов отказавших активных реплик в соответствии с БД и для каждого типа выполняет процедуру *hwp_t_fail*. Если выполнено условие теоремы, то данная процедура обеспечивает активизацию задачи *t** типа *tt*, следовательно при отказе АП *hwp** будет восстановлена работоспособность МАС. Отказ ИР фиксируется АП и приводит к отказу активных задач-реплик, размещенных в агентах данной АП, которым для их работы необходим данный ИР. Согласно процедуре *hwp_hwr_fail* АП определяет множество типов отказавших активных реплик и для каждого из них выполняет процедуру *hwp_t_fail*, которая при выполнении условия теоремы обеспечивает активизацию задачи соответствующего типа, следовательно, при отказе ИР будет восстановлена работоспособность МАС. Логический отказ задачи фиксируется агентом *a*, в котором задача размещена. Так как согласно условию теоремы логический отказ задачи вызван воздействиями, принадлежащими внутреннему контексту, то согласно процедуре *a_t_lfail* для каждого из воздействий, определяющих невыполнения предусловий задачи, будет вызвана задача-эффектор. Поскольку было доказано, что при отказах задач, агентов и АП, в МАС будут активированы реплики соответствующих типов, то все эффекторы будут выполнены, что согласно определению эффектора обеспечивает выполнение предусловий задачи, находящейся в состоянии логического отказа, следовательно, работоспособность МАС будет восстановлена. Теорема доказана.

Сравнение методик обеспечения отказоустойчивости МАС

Сравнение существующих методик обеспечения отказоустойчивости МАС DARX [3], Meta-Agent [4], Brokered MAS [5] и техники, предложенной в [6], с разработанными методикой восстановления работоспособности МАС и моделью резервированной МАС выявило следующие преимущества:

- в отличие от существующих методик разработанная модель резервированной МАС учитывает распределение задач по агентам и агентов по аппаратным компонентам, а также наличие доступа к исполнительным ресурсам для аппаратных компонентов;
- модель резервированной МАС основана только на избыточности задач и допускает избыточность ресурсов и введение дополнительных агентов и аппаратных компонентов, в то время как DARX, Meta-Agent и Brokered MAS основаны только на избыточности агентов, а техника, предложенная в [6], хоть и предлагает использовать избыточность отдельных задач, но вводит также и избыточность агентов;
- разработанная методика восстановления работоспособности МАС позволяет преодолевать логические отказы задач, в то время как только техника [6] допускает такую возможность;
- в отличие от существующих методик разработанная методика восстановления работоспособности обеспечивает обнаружение функциональных отказов задач и отказов исполнительных ресурсов, а также обеспечивает поддержание неактивных резервных задач в актуальном состоянии в случае изменения условий их выполнения;
- протокол взаимодействия между компонентами МАС обеспечивает возможность вызова задачи независимо от знания о ее расположении в компонентах МАС, что уменьшает сложность реализации агента.

В статье представлены разработанные нами формальная модель МАС как программно-аппаратного комплекса и модель резервированной МАС, основанная на введении избыточности отдельных задач, а не агентов. На основе разработанных моделей предложены протокол взаимодействия между компонентами МАС, техники обнаружения отказов и восстановления работоспособности и техника поддержания резервных задач МАС в актуальном состоя-

нии, образующие методику восстановления работоспособности резервированной МАС. Научная новизна и практическая ценность результатов исследования заключаются в следующем:

модель МАС позволяет рассматривать МАС как распределенный программно-аппаратный комплекс, конфигурация которого задана распределением задач по агентам и агентов по агентным платформам, а также доступностью отдельных ресурсов для агентных платформ МАС;

модель резервированной МАС основана на использовании только избыточности задач вместо избыточности агентов и при этом допускает использование избыточности ресурсов, а также введение дополнительных агентов и агентных платформ;

методика восстановления работоспособности включает в себя техники обнаружения отказов аппаратных компонентов, исполнительных ресурсов и логических и функциональных отказов задач;

методика восстановления работоспособности обеспечивает работоспособность МАС после программных и функциональных отказов задач, программных отказов агентов, отказов аппаратных компонентов и исполнительных ресурсов, а также после логических отказов задач, если в МАС существуют эффекторы, способные удовлетворить предусловия задачи, находящейся в состоянии отказа;

методика восстановления работоспособности обеспечивает поддержание неактивных резервных задач в актуальном состоянии, что позволяет снизить вычислительную нагрузку на систему и обеспечить восстановление работоспособности при изменении условий выполнения отдельных задач, характерном для интеллектуальных агентов;

протокол взаимодействия между компонентами МАС обеспечивает выполнение задачи независимо от ее расположения в компонентах системы, что уменьшает сложность реализации агента, т. к. агент не должен обладать информацией о фактической конфигурации МАС для запроса исполнения определенной задачи;

сформулирована и доказана теорема о свойстве сохранения работоспособности резервированной МАС, подтвердившая достоверность разработанных моделей и методики восстановления работоспособности.

СПИСОК ЛИТЕРАТУРЫ

1. **Jennings N.R.** On agent-based software engineering // Artificial intelligence. 2000. Vol. 117. No. 2. Pp. 277–296.

2. **Pullum L.L.** Software fault tolerance techniques and implementation. Artech House, 2001. 360 p.

3. **Guessoum Z., Briot J.P., Faci N.** Towards fault-tolerant massively multiagent systems // Massively Multiagent Systems I. Springer Berlin Heidelberg, 2005. Pp. 55–69.

4. **Serugendo G.D.M., Romanovsky A.** Designing fault-tolerant mobile systems // Scientific Engineering for Distributed Java Applications, International Workshop. Springer Berlin Heidelberg, 2003. Pp. 185–201.

5. **Kumar S., Cohen P.R.** Towards a fault-tolerant multiagent system architecture // Proc. of the 4th Internat. Conf. on Autonomous Agents, ACM. 2000. Pp. 459–466.

6. **Mellouli S.** A reorganization strategy to build fault-tolerant multiagent systems // Advances in Artificial Intelligence. Springer Berlin Heidelberg, 2007. Pp. 61–72.

7. **Wooldridge M., Jennings N.R.** Intelligent agents: Theory and practice // Knowledge engineering review. 1995. Vol. 10. No. 2. Pp. 115–152.

8. **Faci N., Guessoum Z., Marin O.** DimaX: a fault-tolerant multiagent platform // Proc. of the 2006 Internat. Workshop on Software Engineering for Large-Scale Multiagent Systems, ACM. 2006. Pp. 13–20.

9. **Chen X., Feng J., Hiller M., Lauer V.** Application of software watchdog as a dependability software service for automotive safety relevant systems // Dependable Systems and Networks, 2007. 37th Annual IEEE/IFIP Internat. Conf. on, IEEE. 2007. Pp. 618–624.

10. **Олифер В.Г., Олифер Н.А.** Компьютерные сети. Принципы, технологии, протоколы: Учебн. пособие. 4-е изд. СПб.: Питер, 2013. 944 с.

REFERENCES

1. **Jennings N.R.** On agent-based software engineering, *Artificial intelligence*, 2000, Vol. 117, No. 2, Pp. 277–296.

2. **Pullum L.L.** *Software fault tolerance techniques and implementation*, Artech House, 2001, 360 p.

3. **Guessoum Z., Briot J.P., Faci N.** Towards

fault-tolerant massively multiagent systems, *Massively Multiagent Systems I*, Springer Berlin Heidelberg, 2005, Pp. 55–69.

4. **Serugendo G.D.M., Romanovsky A.** Designing fault-tolerant mobile systems, *Scientific Engineering for Distributed Java Applications, International Workshop*, Springer Berlin Heidelberg, 2003, Pp. 185–201.

5. **Kumar S., Cohen P.R.** Towards a fault-tolerant multiagent system architecture, *Proceedings of the 4th International Conference on Autonomous Agents*, ACM, 2000, Pp. 459–466.

6. **Mellouli S.** A reorganization strategy to build fault-tolerant multiagent systems, *Advances in Artificial Intelligence*, Springer Berlin Heidelberg, 2007, Pp. 61–72.

7. **Wooldridge M., Jennings N.R.** Intelligent agents: Theory and practice, *Knowledge engineering review*, 1995, Vol. 10, No. 2, Pp. 115–152.

8. **Faci N., Guessoum Z., Marin O.** DimaX: a fault-tolerant multiagent platform, *Proceedings of the 2006 International Workshop on Software Engineering for Large-scale Multiagent systems*, ACM, 2006, Pp. 13–20.

9. **Chen X., Feng J., Hiller M., Lauer V.** Application of software watchdog as a dependability software service for automotive safety relevant systems, *Dependable Systems and Networks, 2007. DSN'07. 37th Annual IEEE/IFIP International Conference on*, IEEE, 2007, Pp. 618–624.

10. **Olifer V.G., Olifer N.A.** *Kompyuternye seti. Printsipy, tehnologii, protokoly*, Uchebn. posobie, 4-e izd., St. Petersburg: Piter Publ., 2013, 944 p. (rus)

ИГУМНОВ Алексей Владимирович – *аспирант кафедры информационных и управляющих систем Санкт-Петербургского государственного политехнического университета.*
195251, Россия, Санкт-Петербург, ул. Политехническая, д. 29.
E-mail: Alexei.Igumnov@gmail.com

IGUMNOV, Alexei V. *St. Petersburg State Polytechnical University.*
195251, Politekhnicheskaya Str. 29, St. Petersburg, Russia.
E-mail: Alexei.Igumnov@gmail.com

САРАДЖИШВИЛИ Сергей Эрикович – *доцент кафедры информационных и управляющих систем Санкт-Петербургского государственного политехнического университета, кандидат технических наук.*
195251, Россия, Санкт-Петербург, ул. Политехническая, д. 29.
E-mail: SSaradg@yandex.ru

SARADGISHVILI, Sergey E. *St. Petersburg State Polytechnical University.*
195251, Politekhnicheskaya Str. 29, St. Petersburg, Russia.
E-mail: SSaradg@yandex.ru

УДК 004.41:004.822

А.Н. Кузнецов, Е.В. Пышкин

ОНТОЛОГИЯ СБОРКИ, КОНФИГУРАЦИИ ОКРУЖЕНИЯ И ЗАПУСКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ И ЕЕ ПРИМЕНЕНИЕ ПРИ АВТОМАТИЗАЦИИ РАЗВЕРТЫВАНИЯ КЛИЕНТСКИХ ПРОГРАММ В ВЫЧИСЛИТЕЛЬНЫХ ОБЛАКАХ

A.N. Kuznetsov, E.V. Pyshkin

ONTOLOGY OF SOFTWARE BUILDING, EXECUTION AND ENVIRONMENT CONFIGURATION AND ITS APPLICATION IN SOFTWARE DEPLOYMENT IN COMPUTING CLOUDS

Рассмотрены аспекты сборки, конфигурации, развертывания и запуска приложений в облачной среде. Уделено внимание проблемам, с которыми сталкиваются разработчики, будучи специалистами в определенной предметной области, но не являясь при этом экспертами в области программирования. На примере проблем, возникающих при тестировании алгоритмов в области информационного поиска, проиллюстрированы основные подходы к организации исполнения консольных клиентских программ в облачной платформе. Предложена онтология предметной области, описывающая процессы построения программного кода и его запуска, а также ошибки, возникающие во время построения и запуска, и действия, необходимые для устранения возникших ошибок. Приведены примеры определения онтологий конкретных задач для построения исходного кода средствами maven и запуска java приложений, сформулированы рекомендации по использованию предложенной онтологии для описания правил базы знаний экспертной системы.

СЕРВИСНО-ОРИЕНТИРОВАННАЯ АРХИТЕКТУРА; ОБЛАЧНЫЕ ВЫЧИСЛЕНИЯ; ОНТОЛОГИЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ; ИНФОРМАЦИОННЫЙ ПОИСК; СЕМАНТИЧЕСКАЯ СЕТЬ.

This paper is focused on software applications build, configuration and execution in cloud frameworks. We pay special attention to the case when developers being experts in their subject domain aren't experienced enough in programming. By examples of problems existing in the area of testing information retrieval algorithms we examine major approaches to support CLI client software execution in clouds. We introduce a subject domain ontology describing processes of software code building, its execution, build and execution errors as well as actions which are necessary to fix recognized errors. We cite examples of concrete tasks ontology definition to describe code building activities using maven and software execution activities using java. The recommendations on how to use the proposed ontology to define expert system knowledge base rules are given.

SERVICE-ORIENTED ARCHITECTURE; CLOUD COMPUTING; SOFTWARE ONTOLOGY; INFORMATION RETRIEVAL; SEMANTIC NETWORKS.

Как известно, облачные вычисления реализуют концепцию сервисно-ориентированного подхода к распределенным вычислениям. Она позволяет клиентам быстро получать ресурсы, такие как виртуальные машины или сетевые хранилища, и возвращать их, оплачивая только время или количество использованных ресурсов. До настоящего времени облачные вычисления рассматривались, преимущественно, в контексте проектирования и предоставления web-сервисов. С удешевлением облачных ресурсов различные научные сообщества начинают исследовать возможность при-

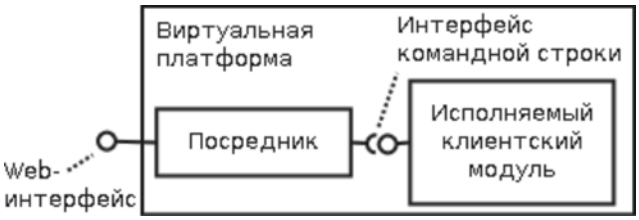


Рис. 1. Применение посредника при развертывании консольного ПО в облачном окружении

менения облачных сервисов в своих областях [1].

Традиционно многие исследователи разрабатывают ПО, реализующее исследуемые методы, с целью проведения сравнений разрабатываемого алгоритма с уже существующими. Поскольку в ряде случаев исполнение программного кода занимает длительное время, целесообразно выполнять такие испытания в облаке, арендовав нужное количество ресурсов на короткое время. Другое преимущество облачного приложения заключается в том, что его можно сделать публичным, предоставив API для доступа к приложению, обеспечивая таким образом доступ к исследуемому алгоритму другим членам научного сообщества.

В целях упрощения использования облачных сервисов в научных исследованиях разрабатываются различные платформы и инструменты для развертывания приложений. Эти инструменты делятся на две группы:

автоматизирующие развертывание приложений, разработанных для исполнения в вычислительных кластерах;

автоматизирующие развертывание приложений, разработанных для исполнения на ПК.

Предметом нашего исследования является вторая группа ПО. Основное допущение, которое делается в рамках работ по автоматизации развертывания приложений этой группы (см., например, [1]), заключается в том, что требуется правильно сконфигурировать облачную инфраструктуру. При этом сами вычислительные среды, предоставленные PaaS или IaaS сервисом (далее эти среды будем называть *виртуальной платформой*, когда различия между PaaS и IaaS несущественны), уже сконфигурированы нужным образом, т. е. все зависимости развертываемого ПО от сторонних компонентов удовлетворены. Часто это предположение оказывается неверным, что значительно усложняет процесс развертывания, равно как и сам механизм использования облачных сервисов для научных исследований.

Особенно ярко указанная проблема проявляется в областях науки, где исследователи, являясь экспертами в своей области, не являются экспертами в разработке компьютерных программ. Примером может служить информационный поиск (ИП). Традиционно, для сравнения алгоритмов, исследователи в области ИП используют метрики, вычисленные для комбинации «алгоритм-корпус». Такой подход предполагает реализацию алгоритма, причем разработанные программные модули чаще всего являются обычными консольными приложениями[1], разработанными для настольных ПК. Для развертывания такого приложения в облаке требуется *посредник*, предоставляющий web-интерфейс к консольному приложению, как показано на рис. 1.

В данной статье мы рассматриваем проблемы автоматического выявления и устранения ошибок запуска приложений, вызванных неправильной конфигурацией *виртуальной платформы*, на которой производится запуск клиентского модуля, а также предлагаем онтологию для описания и решения возникающих проблем с помощью баз знаний.

[1] То есть речь идет о приложениях с интерфейсом командной строки. Такие приложения не предполагают интерактивное взаимодействие с пользователем, а используют аргументы командной строки как основной метод коммуникации с пользователем.

Обзор предметной области

Хотя настоящее исследование посвящено онтологии процесса сборки, запуска ПО и конфигурации среды исполнения (для краткости будем называть эту онтологию *Запуск ПО*), с учетом контекста работы, а именно автоматического развертывания консольного ПО в облачных инфраструктурах и платформах (далее для краткости будем называть их *облачными структурами*, когда различия между PaaS и IaaS несущественны), необходимо обратить внимание на исследования, которые не имеют прямого отношения к онтологиям, но связаны с задачами нашей предметной области.

NEMA (Networked Environment for Music Analysis) [2]. Эта платформа предоставляет распределенную сетевую инфраструктуру для исследований в области информационного поиска музыки (MIR, Music Information Retrieval). Предлагаемая архитектура позволяет авторам публиковать реализации своих алгоритмов в виде сервисов с определенным интерфейсом. Развертывание производится в ручном режиме, поэтому авторы развертываемых программных модулей должны обладать определенными экспертными знаниями и навыками для публикации своего алгоритма в данной инфраструктуре.

MEDEA (Message, Enqueue, Dequeue, Execute, Access) [1]. Данная система позволяет развернуть произвольное консольное приложение в облаке. Основное внимание уделено развертыванию приложений в различных облачных сервисах (Google App Engine, EC2, Eucalyptus, MS Azure), при этом преследуются две основные цели: оценить производительность различных сервисов при решении одной и той же задачи; предоставить возможность развертывать произвольные приложения с интерфейсом командной строки в произвольном облачном сервисе. Для развертывания приложения пользователь системы должен предоставить дополнительную информацию, в т. ч. требуемое окружение, где под «требуемым окружением» понимается строка, описывающая среду исполнения, например «Java» или «Python». При развертывании приложения в виртуальной платформе будет установлено необходимое окружение. Главным недостатком MEDEA является то, что ошибки, возникающие в процессе исполнения клиентского кода (в т. ч. и те, которые связаны с неправильной конфигурацией среды исполнения), не анализируются.

Обнаружение и диагностика ошибок конфигурации приложения, а также зависимостей программного модуля от сторонних библиотек и компонентов обычно выполняется двумя способами: путем статического анализа кода и объектных модулей или путем динамического анализа результатов исполнения программы на тестовом наборе данных. Для первого подхода разработано множество методов и алгоритмов, например ConfAnalyzer [3], ConfDiagnoser [4], ConfDebugger [5] и пр. Известными инструментами, реализующими второй подход, являются AutoBash [6] и ConfAid [7]. Для выявления ошибочных ситуаций используются предикаты (реализованные в виде скриптов), возвращающие значение *ложь*, если обнаружена ошибка конфигурации. AutoBash и ConfAid интенсивно используют модифицированное ядро Linux и программные отладчики бинарного кода, что усложняет использование этих утилит в облачных структурах из-за ограничений политик безопасностей поставщика облачных ресурсов. Необходимость модифицированного ядра и отладчиков можно отнести к недостаткам реализации, в то время как другой, более существенный, недостаток заключается в том, что предметная область диагностирования ошибок конфигурации ПО определена в указанных работах неявно. Ключевыми понятиями являются *ошибка* (определяется с помощью предиката) и *решение* (определяется с помощью скрипта, исправляющего ошибку). Ошибка вызвана *неправильной конфигурацией*, однако само понятие *неправильной конфигурации* и атрибуты этого понятия явно не описаны, при этом они тесно связаны с реализацией утилит и форматом данных, которые эти утилиты собирают и обрабатывают.

Для формального описания связей между ошибкой и решением, а также формального описания ошибки требуется примене-

ние развитых формализмов, используемых в области инженерии знаний. К таким формализмам можно отнести языки описания процессов (BPEL, BPMN, PSL, Cyc и пр.), онтологии, а также базы знаний.

На настоящий момент нет широко известных онтологий[2], описывающих процесс сборки, запуска ПО и конфигурации среды исполнения (для краткости будем называть эту онтологию *Запуск ПО*). Тем не менее существует несколько широко известных онтологий, описывающих область программной инженерии. В работе [9], посвященной классификации онтологий в области программной инженерии, выявлено 19 типов онтологий, покрывающих различные аспекты проектирования, документирования, тестирования и поддержки программных продуктов. Наиболее релевантными данной работе являются следующие типы онтологий: онтология процесса разработки (software process ontology) [10], онтология поведения системы (system behavior ontology) [11], онтология программных артефактов (software artifact ontology) [12], онтология конфигурации системы (system configuration ontology) [13]. Каждая из перечисленных онтологий оперирует определенным подмножеством понятий, необходимых для описания предметной области *Запуск ПО*, но ни одна из представленных онтологий не описывает предметную область целиком.

Помимо онтологий предметной области программной инженерии, широко известны метаонтологии, способные описывать обширный круг задач. Наиболее релевантной нашей области является метаонтология процессов (process ontology), включающая в себя такие понятия, как процесс (process), деятельность (activity) и пр. Примерами подобных онтологий являются языки PSL (process specification language) [14] и Cyc [15]. Кроме того, существуют языки описания процессов, ориентированных на формальное исполнение (например, BPEL [16] и BPMN [17]). Особенность подобного рода метаонтологий заключается в том, что, стремясь описать широкий круг задач, они не содержат конкретики для описания понятий и связей в узкой предметной области, поэтому часто требуется разработка онтологии предметной области и конкретных задач на базе существующей метаонтологии.

Постановка задачи

Рассмотрим трехзвенную архитектуру, состоящую из *виртуальной платформы*, брокера облака и менеджера автоматических развертываний консольных приложений. На *виртуальной платформе* имеется предустановленный компонент – агент менеджера развертываний, – который предоставляет web-интерфейс к консольному приложению, собирает информацию о состоянии консольного приложения и окружения и содержит базу знаний для решения проблем развертывания клиентского ПО. Агент взаимодействует с менеджером посредством высокоуровневых команд, описывающих необходимые действия по реконфигурации платформы (например, «добавить JDK1.7»). Менеджер взаимодействует с брокером посредством низкоуровневых команд, обеспечивающих инсталляцию в платформу необходимых картриджей, или пересоздание виртуальных машин из других образов.

Требуется разработать такую онтологию для использования в базе знаний агента, которая описывает процессы автоматического построения, конфигурирования и запуска клиентского приложения в *виртуальной платформе*, а также ошибки, возникающие во время построения и запуска, и действия, необходимые для устранения выявленных ошибок.

Онтология запуска программ

Цель менеджера – успешное развертывание приложения в *виртуальной платфор-*

[2] «Онтология – формальное явное описание понятий в рассматриваемой предметной области (классов (иногда их называют понятиями)), свойств каждого понятия, описывающих различные свойства и атрибуты понятия (слотов (иногда их называют ролями или свойствами)), и ограничений, наложенных на слоты (фацетов (иногда их называют ограничениями ролей))» [8].

Рис. 2. Семантическая сеть, описывающая онтологию сборки и запуска консольного ПО и конфигурации окружения

ме. Под успешным развертыванием будем понимать успешный запуск клиентского ПО на пробном наборе входных данных. Для успешного запуска менеджеру требуются следующее:

наличие исполняемого модуля клиентского ПО;

тестовый набор данных;

описание формата вызова клиентского модуля из командной строки;

описание ожидаемого результата (набор файлов и их формат) для данного тестового набора данных.

Исполняемый модуль клиентского ПО загружается в *виртуальную платформу* одним из двух возможных способов. Первый способ состоит в том, что исполняемый модуль непосредственно загружается разработчиком клиентского ПО. Во втором случае исполняемый модуль является производным артефактом процесса сборки клиентского ПО из исходных текстов. При этом построение исходного кода можно описать теми же понятиями, что и запуск клиентского ПО, поскольку процесс построения можно рассматривать как запуск консольного приложения (компилятора, или более развитой системы построения кода как make, ant, maven и т. п.), а входными данными являются исходные коды клиентского ПО.

Результатами исполнения консольного приложения (клиентского модуля или системы построения) являются:

сгенерированные файлы;

выходные потоки (stdout, stderr);

код возврата.

Результат запуска может быть признан успешным, если код возврата равен нулю, а сгенерированные файлы и содержимое stdout/stderr соответствует ожиданиям. В противном случае запуск признается неуспешным.

В случае неуспешного запуска менеджер развертываний анализирует причины возникших ошибок (путем разбора выходных потоков, отчетов среды исполнения и т. д.) и пытается их решить самостоятельно путем переконфигурации *виртуальной платформы* и повторного запуска консольного приложения (клиентского ПО или системы построения исходного кода). Если автоматически решить проблему не удается, создается и рассылается уведомление заинтересованным лицам (автору клиентского ПО, администратору системы и т. п.) о невозможности автоматического развертывания.

Для описания подобного рода сценариев и с учетом устоявшейся терминологии в существующих онтологиях, разработана онтология автоматического построения и запуска консольных приложений и устранения ошибок конфигурации среды исполнения. Семантическая сеть, описывающая онтологию предметной области, представлена на рис. 2.

Все множество понятий онтологии можно разбить (по назначению) на три группы: *обычные* факты, факты-действия (action), направленные на внешние, по отношению к машине вывода, компоненты, и факты-запросы (request), описывающие действия, направленные на машину вывода.

Обычные факты описывают текущее состояние системы. Ключевыми фактами являются следующие: **Деятельность** (Activity), **Артефакт** (Artifact) и **Команда** (для интерпретатора командной строки, ExecCommand). Дадим краткое описание ключевых *обычных* фактов:

1. **ИсходныйАртефакт** (OriginalArtifact) – это артефакт, предоставленный разработчиком *клиентского модуля*, представляющий собой файл, каталог, архив или URL. В процессе работы машины вывода он является причиной добавления производного артефакта **Файл** или **Каталог**, представляющего исходный артефакт в виде файла или каталога в локальной файловой системе соответственно после скачивания, копирования или распаковки исходного артефакта.

2. **Артефакт** (Artifact) – Артефакт файловой системы. Это всегда либо один файл (**Файл**), либо один каталог (**Каталог**) в локальной файловой системе.

3. **Деятельность** (Activity) – действие эксперта (или экспертной системы) для достижения цели: **Сборка**, **Запуск** клиентского модуля. **Деятельность** является агрегирующим понятием (при помощи отношения **описывать**), и всегда имеет своей целью

запуск какой-либо **Команды**, результат исполнения которой описывается **СтатусомИсполнения**. В зависимости от кода возврата команды и, возможно, некоторых других факторов, принимается решение об **УспехеДеятельности** или **ОшибкеДеятельности**.

4. **Команда** (ExecCommand) – команда для интерпретатора командной строки. Формат команды состоит из пяти элементов, как показано на рис. 3: из имени программы и четырех групп **Аргументов** командной строки: **Позиционных{1,2}** и именованных **КлючЗначение{1,2}**. Такой формат обусловлен тем, что в качестве среды исполнения часто выступает виртуальная машина (например, Java), и первая пара аргументов – это ключи виртуальной машины, а вторая – ключи исполняемой программы. Команда исполняется в облачном окружении агентом менеджера развертываний в ответ на факт-действие **ДействиеИнтерпретатора**. По завершении исполнения агент добавляет в рабочую память **СтатусИсполнения** с информацией о коде возврата, а также содержимым выходных потоков.

Необходимость выполнить какое-либо действие отражается в онтологии фактом-действием. Такой подход позволяет сохранить правила базы знаний декларативными (а не императивными). Однако добавление фактов в рабочую память машины вывода само по себе не оказывает воздействие ни на окружающую среду, ни на экспертную систему, кроме возможной активации новых правил в базе знаний. Для обработки фактов-действий требуется специальный компонент – *исполнитель* (см. рис. 4). Взаимодействие *исполнителя* с экспертной системой может быть организовано различными способами, а реализация взаимодействия может оказывать влияние на реализацию правил в экспертной базе знаний. В рамках прототипа разрабатываемой системы автоматического развертывания консольных приложений мы предлагаем использовать позднее исполнение и прямой доступ в рабочую память машины вывода. Этот подход отличается простотой реализации и толерантностью к добавлению, удалению и изменению факта-действия вплоть до момента исполнения действия агентом.

Взаимодействие экспертной системы и *исполнителя* изображено на рис. 4.

В тот момент, когда в расписании машины вывода больше нет активных правил (т. е. правил, которые могут сработать), управление передается *исполнителю*, который анализирует рабочую память на наличие в ней фактов-действий. Выполнив необходимое действие, *исполнитель* добавляет в рабочую память машины вывода новые факты, вызванные этим действием. Например, в результате исполнения действия **ДействиеИнтерпретатора**, *исполнитель* добавит в рабочую память **СтатусИсполнения**. В целях упрощения реализации агента, обработанные факты-действия удаляются из рабочей памяти машины вывода.

В рамках онтологии предметной области мы выделили следующие основные факты-действия:

1. **ДействиеИнтерпретатора** (Exec Action) – указание на необходимость исполнить команду. Результат исполнения команды будет добавлен в рабочую память в виде факта **СтатусИсполнения**.

2. **КонфигурацияОкружения** (ConfigAction) – указание на необходимость изменить состояние окружающей среды (добавить или удалить сторонние компоненты). Не предполагается прямое использование этого класса. Вместо этого должны быть определены подклассы данного понятия (например, «ДобавитьJDK7»), которые могут быть интерпретированы менеджером развертываний.

3. **Уведомление** (UsrNfAction) – указание на необходимость отправить пользователю сообщение (например, о невозможности автоматического развертывания приложения). Аналогично предыдущему действию,

```
<имя_прог> [[<поз1>][<ключ1 зн1>]]… [[<поз2>][<ключ2 зн2>]]…
java –verbose:class –jar ./prog.jar -progArg1 val1 -progArg2
```

Рис. 3. Формат командной строки

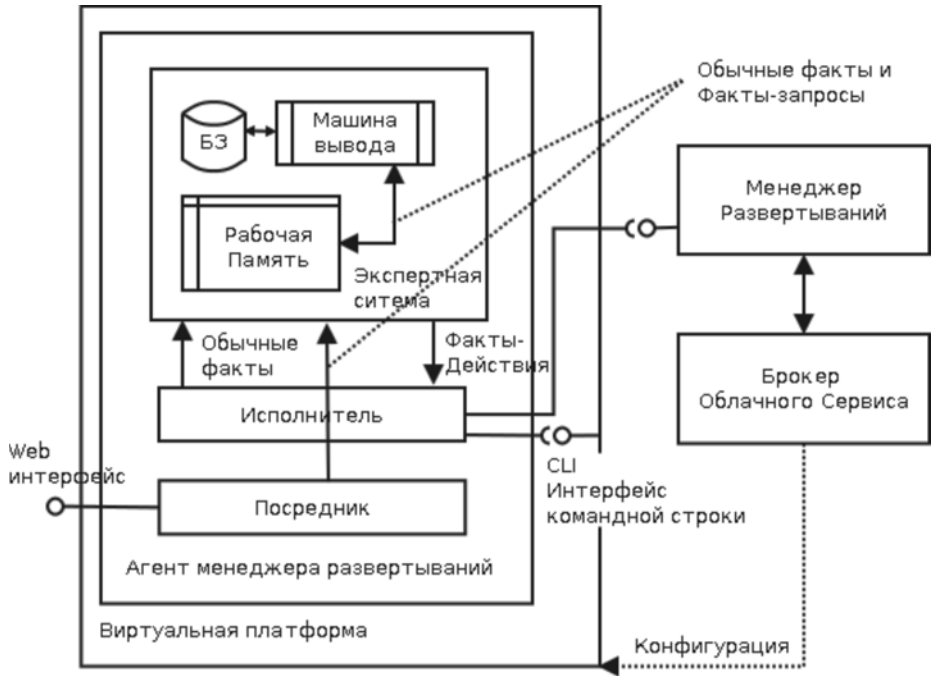


Рис. 4. Взаимодействие экспертной системы (ЭС) и менеджера развертываний

не предполагается прямое использование этого класса.

Факты-запросы по своей сути похожи на факты-действия, однако их принципиальным отличием является то, что запросы направлены на саму экспертную систему и обрабатываются машиной вывода как обычные факты. Запросы могут быть добавлены в систему как извне (например, *посредником*), так и самой системой. Например, если пользователь системы предоставил исходные коды в качестве **ИсходногоАртефакта**, система может сгенерировать сама себе новый **ЗапросСборки**. В предлагаемой онтологии предметной области основным и единственным типом запроса является **ЗапросДеятельности**: **ЗапросЗапуска** или **ЗапросСборки**. Также предполагается, что в онтологиях конкретных задач, базирующихся на онтологии предметной области, могут появиться дополнительные типы запросов.

Онтологии конкретных задач

Онтология предметной области редко используется в экспертных системах без изменений, поскольку она не обладает нужным уровнем детализации для решения конкретных задач. Поэтому на основе онтологии предметной области создается онтология конкретной задачи, обладающая нужным уровнем детализации. В этом разделе мы продемонстрируем расширение предложенной онтологии для описания конкретных задач, на двух примерах:

1) сборка исходных кодов системой maven;

2) запуск Java приложения.

Для описания конкретной задачи мы расширим базовые понятия (классы), введенные в онтологии предметной области, так, как показано на рис. 5.

В частности, расширим понятие **Аргумент** подклассами **JАргумент** и **M2Аргумент**. Некоторые экземпляры этих понятий приведены на рисунке в качестве иллюстрации, однако список приведенных экземпляров неполный. Список экземпляров ограничен набором всех возможных ключей для Java и maven соответственно. Приведенные в качестве иллюстраций понятия имеют следующий смысл:

1. **JOptHeadless** («-Djava.awt.headless=true») – запуск Java машины в AWT Headless режиме;

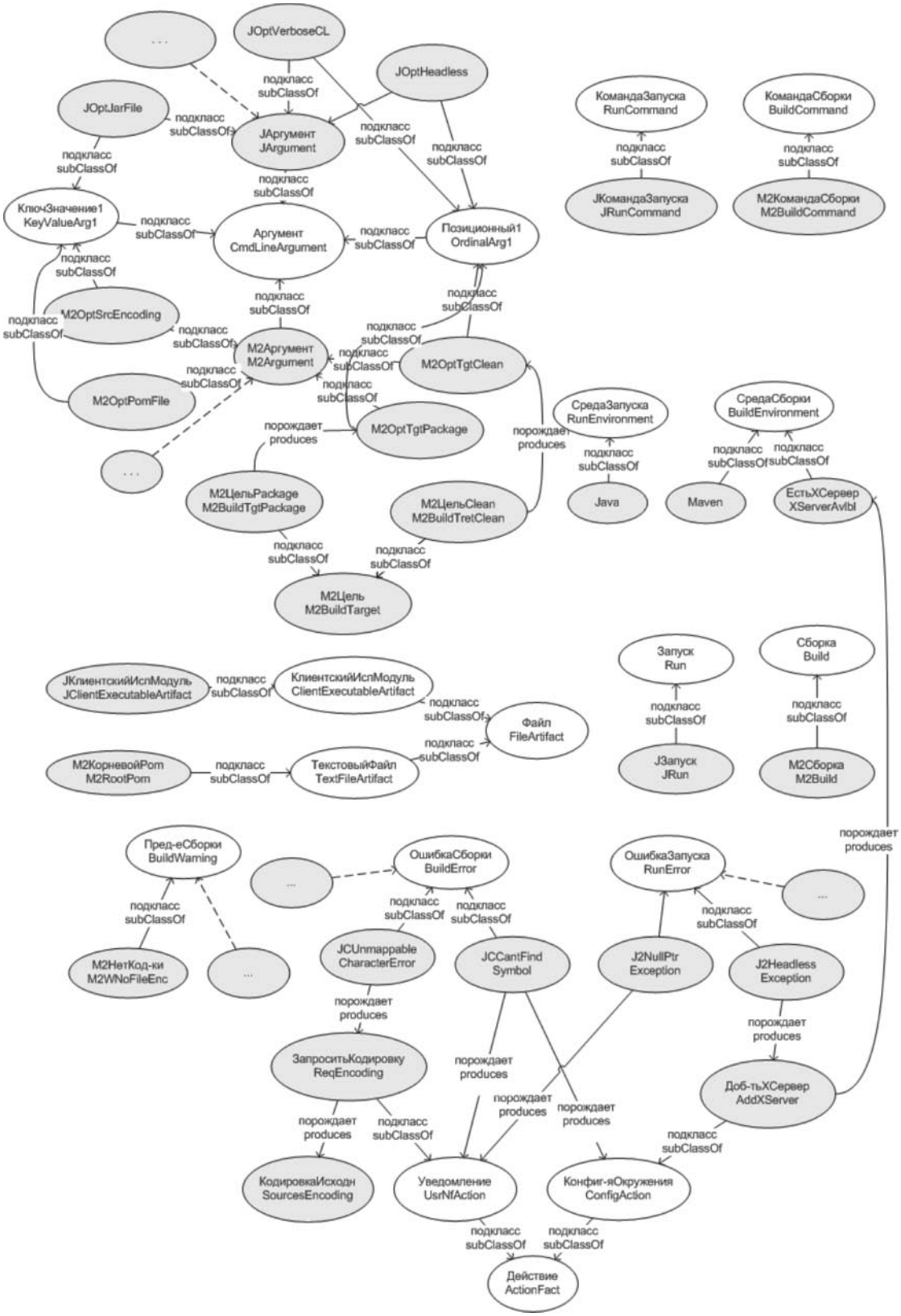


Рис. 5. Фрагмент семантической сети, описывающий онтологию сборки maven и запуска Java приложений

2. **JOptVerboseCL** («-verbose:class») – выводить лог Java компонента Class Loader;

3. **JOptJarFile** («-jar <file>») – местоположение Jar файла;

4. **M2OptTgtClean** («clean»), **M2OptTgtPackage** («package») – maven цели: clean и package соответственно;

5. **M2OptPomFile** («-f <file>») – местоположение pom файла;

6. **M2OptSrcEncoding** («-Dproject.build.sourceEncoding=<encoding>») – кодировка ресурсов maven (в т. ч. исходных файлов).

Другое важное расширение онтологии заключается во введении понятий **JЗапуск** (запуск Java программы) и **M2Сборка** (сборка maven), расширяющие базовые понятия **Запуск** и **Сборка**, а также введение конкретных типов ошибок и предупреждений maven и Java. Аналогично расширению аргументов, перечисленные экземпляры ошибок являются иллюстрацией, поскольку полный набор экземпляров измеряется десятками сотен. В примере на рис. 5 приведены следующие типы maven, Java и Javac (компилятор Java) ошибок:

1. **M2НетКодировки** – предупреждение maven сборки «WarnNoFileEncoding» о том, что для проекта не указана исходная кодировка. Это предупреждение можно игнорировать.

2. **J2HeadlessException** – ошибка запуска Java кода: невозможно исполнение в Headless режиме. В ответ на эту ошибку экспертная система обычно добавляет факт-запрос на реконфигурацию среды: **ДобавитьXСервер**.

3. **J2NullPointerException** – ошибка запуска Java кода: NullPointerException. Обычно эту ошибку невозможно исправить автоматически, т. к. она часто связана с ошибками в клиентском коде. Необходимо отправить уведомление о невозможности продолжения запуска.

4. **JCCantFindSymbol** – ошибка Java компилятора при сборке: невозможно найти символ. Часто связана либо с ошибками компиляции других классов этого же проекта, либо с отсутствующим jar файлом, содержащим необходимые определения. Эту ошибку можно диагностировать по полному имени отсутствующего символа. Для широко известных имен классов (org.apache.*, например) эта ошибка может быть исправлена автоматически путем добавления необходимых jar файлов в classpath во время компиляции. Для нестандартных имен классов эту ошибку автоматически исправить невозможно, поэтому будет создано соответствующее уведомление пользователям системы.

5. **JCUnmappableCharacterError** – ошибка Java компилятора при сборке: невозможно разобрать входной файл. Обычно возникает из-за проблем с кодировками. Например, если в java файле есть комментарии на русском языке и файл сохранен в кодировке cp1251, а javac пытается читать этот же файл как UTF8. Обычно проблема решается путем явного указания кодировки исходного файла.

Приведенный выше список классов, описывающих аргументы и ошибочные ситуации Java и maven, не является полным, а является иллюстрацией того, каким образом можно расширять онтологию предметной области для решения конкретных задач. В следующем разделе мы продемонстрируем как использовать расширенную онтологию при описании правил в продукционных моделях баз знаний.

Применение онтологии в продукционных базах знаний

В этом разделе мы опишем основные шаблоны правил, покрывающие процесс построения, запуска исходного кода и диагностирование ошибок, возникающих в процессе построения и запуска, на примере процесса сборки кода системой maven. Эти шаблоны можно считать рекомендациями по использованию онтологии. Представленный набор правил является упрощенным подмножеством правил, который был опробован на нескольких реальных maven проектах в рамках исследования. Для описания правил мы будем использовать язык Drools [18], задающий правила в форме «когда-тогда» («when-then», «если-то»), который ориентирован на понимание неэкспертами в области программирования.

В общем виде причиной какой-либо деятельности является запрос этой деятельности. Если ЭС определила, что в ка-

честве **ИсходногоАртефакта** были предоставлены исходные коды программы, то ЭС может сама себе создать **ЗапросПостроения**. В результате такого запроса будет создана соответствующая деятельность (Листинг 1).

ReqNewActivity::newActivityInstance является «синтаксическим сахаром». Фактически, если запрос был **ЗапросСборки**, то в рабочую память добавится **Сборка**, а если **ЗапросЗапуска**, то **Запуск**. Не представляет труда составить такое правило, которое, в зависимости **СредыСборки**, будет преобразовывать деятельность **Сборка** в соответствующий подкласс, например, **M2Сборка**.

Выполнение деятельности всегда направлено на выполнение **Команды** интерпретатора командной строки. В общем виде правило для составления команды выглядит как показано в листинге 2.

В рассматриваемом примере команду для построения кода с помощью maven можно составить как в листинге 3.

```
rule "Activity_New"
  when
    $rr : ReqNewActivity(  )
    // если есть запрос новой деятельности
    forall ( $b : Activity( request == $rr )
            // деятельность для данного запроса
            // либо не существует, либо
            // для каждой деятельности справедливо:
            ExecStatus( activity == $b, succeeded == false )
            // предыдущие попытки либо еще не завершились,
            // либо завершились с ошибками
            ActivityErrorFixed( activity == $b ) )
            // и хотя бы одна ошибка исправлена
  then
    insert( $rr.newActivityInstance() );
end
```

Листинг 1: Пример правила базы знаний, добавляющего новую **Деятельность** *в ответ на* **ЗапросДеятельности**

```
rule "SomeCommand" when
  $options : java.util.LinkedList() from collect ( CmdLineArgument() )
then
  insertLogical( new ExecCommand("some_command", $arguments) );
end
```

Листинг 2: Шаблон правила базы знаний, составляющего **Команду** *из* **Аргументов**

```
rule "Maven_BuildCommand" when
    $b : M2Build(  )
    // для каждой M2Сборки
    $options : java.util.LinkedList( size > 0 ) from
    // опции сборки - это коллекция M2Option
            collect ( M2Argument( activity == $b ) )
            // для данной деятельности
  then
    insertLogical( new M2BuildCommand($b, $options) );
end
```

Листинг 3: Пример правила базы знаний, составляющего **M2КомандуСборки** *из* **M2Аргументов** *maven*

На примере этого правила можно продемонстрировать целесообразность определения **Аргументов** как отдельных сущностей, а не атрибутов **Команды**, поскольку зависимость ключей командной строки от внешних факторов (например, кодировки исходных файлов) в этом случае можно декларативно определить так, как это показано в листинге 4.

В случае представления аргументов как атрибутов **Команды** такой декларативности достичь не удается, а в части правила, описывающего действие, пришлось бы изменять состояние атрибутов **Команды**, что ухудшает читаемость правил и производительность ЭС.

Нетрудно записать правило, формирующее **ДействиеИнтерпретатора** для **Команды** (листинг 5).

Агент менеджера развертываний выполнит соответствующее действие и добавит в рабочую память **СтатусИсполнения**, который можно проанализировать, и принять решение об успешности или неуспешности завершившейся **Деятельности** (листинг 6).

Для неуспешной деятельности можно разработать правила, диагностирующие возникшие ошибки. Например, ошибку кодировки исходного файла можно диагностировать следующим правилом (см. листинг 7).

```
rule "Maven_Option_SrcEncoding" when
    $build : M2Build()
    // для каждой M2Сборки
    $srcEnc : SrcEncoding( )
    // Если известна кодировка ресурсов
  then
    insert( new M2OpSrcEncoding($build, $srcEnc.getSrcEncoding()) );
    // Добавить соответствующую опцию M2
end
```

*Листинг 4: Пример правила базы знаний, добавляющего опцию сборки (***M2OpSrcEncoding***) в ответ на выявление нового факта о конфигурации проекта (***SrcEncoding***)*

```
rule "CommandExecutor"
  when
    $cmd : ExecCommand( )
    // Для каждой команды
    not ExecStatus( execCommand == $cmd )
    // для которой нет результата исполнения
  then
    insertLogical( new ExecAction($cmd) );
    // Создать соответствующий факт-действие
end
```

Листинг 5: Пример правила базы знаний, формирующего **ДействиеИнтерпретатора** *для* **Команды**

```
rule "BuildStatus"
  when
    $build : Build( )
    ExecStatus( activity == $build, succeeded == true )
  then
    insertLogical( new BuildSucceeded($build));
end
```

Листинг 6: Пример правила базы знаний, принимающего решение об успешности сборки

```
rule "Maven_Err_UnmappableCharacter"
  when
    $build : Build()
    // для каждой Сборки
    ExecStatus( activity == $build,
                succeeded == false,
                // Если статус указывает на ошибку
                $errLine : getSingleLine(«error: unmappable character for encoding»),
                // и в stdout или stderr есть строка «error: unmappable character...»
                $errLine != null )
  then
    insert( new JCUnmappableCharacterError($build, $errLine) );
    // то имеется проблемы с кодировкой
end
```

Листинг 7: Пример правила базы знаний, диагностирующего ошибку сборки (ошибка кодировки исходного файла)

Предложена онтология предметной области, описывающая процессы построения программного когда, его запуска, ошибки, возникающие во время построения и запуска, а также действия, необходимые для устранения возникших ошибок. Поскольку онтология предметной области оперирует слишком общими понятиями для того, чтобы на ее основе формулировать правила экспертной системы, требуются онтологии конкретных задач. В статье приведены примеры определения онтологий конкретных задач для построения исходного кода средствами maven и запуска Java программ. Также сформулированы рекомендации по использованию предложенной онтологии для описания правил базы знаний экспертной системы.

Рассматриваемые онтологии (предметной области и две онтологии конкретных задач) формально описаны на языке Java и использованы в качестве модели предметной области, применяемой при разработке экспертной системы, являющейся частью системы автоматического развертывания консольных приложений в качестве веб-сервисов. Для описания базы знаний экспертной системы использовался язык Drools и машина вывода Drools Expert 5.0.0.Final [18]. Разработанная система позволяет *автоматически* развертывать консольные приложения в облачных структурах благодаря использованию базы знаний о возможных ошибках развертываний. База знаний, разработанная в рамках исследования, не является полной (т. е. не содержит знаний о решении всех возможных проблем, возникающих в ходе развертывания ПО). Поэтому важным является не то, что развертывание выполняется автоматически с *текущим* набором правил базы, а то, что развертывание возможно выполнить в автоматическом режиме, *дополнив* базу знаний необходимыми правилами. С точки зрения эксплуатации системы, поддержкой базы знаний должен заниматься эксперт в соответствующей области. Эксперт добавляет новые правила в базу, опираясь на конкретные случаи невозможности автоматического развертывания. Добавление новых правил должно обеспечивать возможность автоматического развертывания приложения, тем самым происходит «тиражирование» знаний эксперта.

Прототип системы автоматического развертывания, использующий предлагаемую онтологию, испытывался на нескольких модельных примерах (в которых намеренно были внесены ошибки), а также на двух реальных проектах: [19] и [20]. Проведенные эксперименты не выявили ситуаций невозможности описания ошибок построения или запуска ПО в рамках предлагаемой онтологии и с помощью продукционной модели базы знаний. В настоящее время продолжается испытание прототипа на других реальных проектах, не основанных на maven и Java.

СПИСОК ЛИТЕРАТУРЫ

1. **Bunch C.** Automated Configuration and Deployment of Applications in Heterogeneous Cloud Environments // Ph.D. Dissertation. University of California at Santa Barbara, Santa Barbara, CA, USA. 2012. 245 p.

2. **West K., Kumar A, Shirk A, Guojun Zhu, Downie J.S., Ehmann A., Bay M.** The Networked Environment for Music Analysis (NEMA) // 6th World Congress on Services, IEEE Computer Society. 2010. Pp. 314–317.

3. **Rabkin A., Katz R.** Precomputing possible configuration error diagnoses // In Proc. of the 26th IEEE/ACM Internat. Conf. on Automated Software Engineering. 2011. Pp. 193–202.

4. **Zhang S.** ConfDiagnoser: An Automated Configuration Error Diagnosis Tool for Java Software // In Proc. of the Internat. Conf. on Software Engineering, IEEE Press Piscataway. 2013. Pp. 312–321.

5. **Dong Z., Ghanavati M., Andrzejak A.** Automated Diagnosis of Software Misconfigurations Based on Static Analysis // In Proc. of the 2013 IEEE Internat. Symp. on Software Reliability Engineering Workshops. 2013. Pp. 162–168.

6. **Su Y., Attariyan M., Flinn J.** AutoBash: Improving Configuration Management with Operating System Causality Analysis // In Proc. of the 21st ACM Symp. on Operating Systems Principles. 2007. Pp. 237–250.

7. **Attariyan M., Flinn J.** Automating Configuration Troubleshooting with ConfAid // Usenix ;login: Magazine. 2011. Vol. 36. No. 1.

8. **Noy N.F., Mcguinness D.L.** Ontology Development 101: A Guide to Creating Your First Ontology // Stanford Knowledge Systems Laboratory Technical Report KSL-01-05 and Stanford Medical Informatics Technical Report SMI-2001-0880. 2001.

9. **Zhao Y., Dong J., Peng T.** Ontology Classification for Semantic-Web-Based Software Engineering // IEEE Trans. Serv. Comput. 2009. Vol. 2. No. 4. Pp. 303–317.

10. **Liao L., Qu Y., Leung H.K.N.** A Software Process Ontology and Its Application // In Proc. of the First Int'l Workshop Semantic Web Enabled Software Eng. 2005.

11. **Caralt J.C., Kim J.W.** Ontology Driven Requirement Query // In Proc. of the 40th Ann. Hawaii Int'l Conf. System Sciences. 2007. 197 p.

12. **Ambrosio A.P., de Santos D.C., de Lucena F.N., de Silva J.C.** Software Engineering Documentation: An Ontology-Based Approach // In Proc. of the WebMedia and LA-Web Joint Conf. 10th Brazilian Symp. Multimedia and the Web Second Latin Am. Web Congress. 2004. Pp. 38–40.

13. **Shahri H.H., Hendler J.A., Porter A.A.** Software Configuration Management Using Ontologies // In Proc. of the 3rd Int'l Workshop Semantic Web Enabled Software Eng. 2007.

14. PSL Core [электронный ресурс] / URL: http://www.mel.nist.gov/psl/psl-ontology/psl_core.html (дата обращения 06.04.2014)

15. **Aitken S.** Process Representation and Planning in Cyc: From Scripts and Scenes to Constraints. AIAI, University of Edinburgh, 2001 [электронный ресурс] / URL: http://www.aiai.ed.ac.uk/~stuart/Papers/plan01-ws.pdf (дата обращения 06.04.2014)

16. Web Services Business Process Execution Language Version 2.0 (OASIS Standard 11 April 2007) [электронный ресурс] / URL: http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html (дата обращения 06.04.2014)

17. Business Process Model and Notation Version 2.0 (BPMN 2.0) (OMG Standard 02 January 2011) [электронный ресурс] / URL: http://www.omg.org/spec/BPMN/2.0/PDF/ (дата обращения 06.04.2014)

18. Drools Expert User Guide [электронный ресурс] / URL: http://docs.jboss.org/drools/release/5.5.0.Final/drools-expert-docs/html_single/index.html (дата обращения 14.04.2014)

19. **Glazyrin N.** Audio chord estimation using chroma reduced spectrogram and self-similarity // In Proc. of the Music Information Retrieval Evaluation Exchange. 2012.

20. **Khadkevich M., Omologo M.** Large-Scale Cover Song Identification Using Chord Profiles // In Proc. of the 14th Internat. Society for Music Information Retrieval Conf. 2013. Pp. 233–238.

REFERENCES

1. **Bunch C.** Automated Configuration and Deployment of Applications in Heterogeneous Cloud Environments, *Ph.D. Dissertation*, University of California at Santa Barbara, Santa Barbara, CA, USA, 2012, 245 p.

2. **West K., Kumar A, Shirk A, Guojun Zhu, Downie J.S., Ehmann A., Bay M.** The Networked Environment for Music Analysis, *6th World*

Congress on Services, IEEE Computer Society, 2010, Pp. 314–317.

3. **Rabkin A., Katz R.** Precomputing possible configuration error diagnoses, *In Proceedings of the 26th IEEE/ACM International Conference on Automated Software Engineering*, 2011, Pp. 193–202.

4. **Zhang S.** ConfDiagnoser: An Automated Configuration Error Diagnosis Tool for Java Software, *In Proceedings of the International Conference on Software Engineering*, IEEE Press Piscataway, 2013, Pp. 312–321.

5. **Dong Z., Ghanavati M., Andrzejak A.** Automated Diagnosis of Software Misconfigurations Based on Static Analysis, *In Proceedings of the 2013 IEEE International Symposium on Software Reliability Engineering Workshops*, 2013, Pp. 162–168.

6. **Su Y., Attariyan M., Flinn J.** AutoBash: Improving Configuration Management with Operating System Causality Analysis, *In Proceedings of the 21st ACM Symposium on Operating Systems Principles*, 2007, Pp. 237–250.

7. **Attariyan M., Flinn J.** Automating Configuration Troubleshooting with ConfAid. *Usenix; login: Magazine*, 2011, Vol. 36, No. 1.

8. **Noy N.F., Mcguinness D.L.** Ontology Development 101: A Guide to Creating Your First Ontology, *Stanford Knowledge Systems Laboratory Technical Report KSL-01-05 and Stanford Medical Informatics Technical Report SMI-2001-0880*, 2001.

9. **Zhao Y., Dong J., Peng T.** Ontology Classification for Semantic-Web-Based Software Engineering, *IEEE Trans. Serv. Comput.*, 2009, Vol. 2, No. 4, Pp. 303–317.

10. **Liao L., Qu Y., Leung H.K.N.** A Software Process Ontology and Its Application, *In Proceedings of the First Int'l Workshop Semantic Web Enabled Software Eng.*, 2005.

11. **Caralt J.C., Kim J.W.** Ontology Driven Requirement Query. *In Proceedings of the 40th Ann. Hawaii Int'l Conference System Sciences*, 2007, 197 p.

12. **Ambrosio A.P., de Santos D.C., de Lucena F.N., de Silva J.C.** Software Engineering Documentation: An Ontology-Based Approach, *In Proceedings of the WebMedia and LA-Web Joint Conf. 10th Brazilian Symp. Multimedia and the Web Second Latin Am. Web Congress*, 2004, Pp. 38–40.

13. **Shahri H.H., Hendler J.A., Porter A.A.** Software Configuration Management Using Ontologies, *In Proceedings of the 3rd Int'l Workshop Semantic Web Enabled Software Eng.*, 2007.

14. *PSL Core*. Available: http://www.mel.nist.gov/psl/psl-ontology/psl_core.html (Accessed 06.04.2014)

15. **Aitken S.** *Process Representation and Planning in Cyc: From Scripts and Scenes to Constraints*. AIAI, University of Edinburgh, 2001, 4 p. Available: http://www.aiai.ed.ac.uk/~stuart/Papers/plan01-ws.pdf (Accessed 06.04.2014)

16. *Web Services Business Process Execution Language Version 2.0* (OASIS Standard 11 April 2007). Available: http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html (Accessed 06.04.2014)

17. *Business Process Model and Notation Version 2.0* (BPMN 2.0) (OMG Standard 02 January 2011) Available: http://www.omg.org/spec/BPMN/2.0/PDF/ (Accessed 06.04.2014)

18. *Drools Expert User Guide*. Available: http://docs.jboss.org/drools/release/5.5.0.Final/drools-expert-docs/html_single/index.html (Accessed 14.04.2014)

19. **Glazyrin N.** Audio chord estimation using chroma reduced spectrogram and self-similarity, *In Proceedings of the Music Information Retrieval Evaluation Exchange*, 2012.

20. **Khadkevich M., Omologo M.** Large-Scale Cover Song Identification Using Chord Profiles. *In Proceedings of the 14th International Society for Music Information Retrieval Conference*, 2013, Pp. 233–238.

КУЗНЕЦОВ Андрей Николаевич – *аспирант Санкт-Петербургского государственного политехнического университета.*

195251, Санкт-Петербург, ул. Политехническая, д. 29.

E-mail: andrei.n.kuznetsov@gmail.com

KUZNETSOV, Andrey N. *St. Petersburg State Polytechnical University.*

195251, Politekhnicheskaya Str. 29, St. Petersburg, Russia.

E-mail: andrei.n.kuznetsov@gmail.com

ПЫШКИН Евгений Валерьевич – *доцент Санкт-Петербургского государственного политехнического университета, кандидат технических наук.*
195251, Санкт-Петербург, ул. Политехническая, д. 29.
E-mail: evgeny.pyshkin@gmail.com

PYSHKIN, Evgeniy V. *St. Petersburg State Polytechnical University.*
195251, Politekhnicheskaya Str. 29, St. Petersburg, Russia.
E-mail: evgeny.pyshkin@gmail.com

ISFEE 2014

http://isfee.elth.pub.ro

International Symposium on Fundamentals of Electrical Engineering 2014

University Politehnica of Bucharest, Romania

NOVEMBER 28-29 2014

MAIN TOPICS

- Circuit theory
- Electromagnetic field theory
- Circuit analysis and synthesis
- Electromagnetic field analysis
- Circuit design
- Inverse problems
- Technical magnetism
- Coupled problems
- Electrical machines and drives
- Electromagnetic compatibility
- Sensors and transducers
- Data processing
- Power electronics
- Materials in electrical engineering
- Education and management
- Renewable energy
- Power engineering
- Bioengineering
- Informatics and applied mathematics

AUTHOR SCHEDULE

Abstract or full paper submission date: September 15, 2014
Notification of acceptance date: November 1, 2014
Final paper submission date: November 15, 2014

REGISTRATION AND FEES

Participants in ISFEE 2014 must fill a registration form which can be found on the symposium website. One registration fee includes the IEEE Xplore publication of two full papers, a CD containing all accepted submissions, coffee-breaks, lunch and gala dinner. Papers will be presented orally or as posters.
The symposium fee is 150 euro (125 euro for IEEE members, and 50 euro for students).

SPONSORS

Romania Section CAS/CS Joint Chapter

The Association of Romanian Electrical & Electronics Engineers

CONTACT

ISFEE-2014 Secretariat
University POLITEHNICA of Bucharest
Department of Electrical Engineering
Spl. Independentei 313, room EB 225

BUCHAREST 060042, ROMANIA
Tel/Fax: (0040-21) 4029144
E-mail: isfee@elth.pub.ro
http://isfee.elth.pub.ro

ICECS 2014
1st Call For Papers
7-10 December 2014
Palais du Pharo
Marseille-Provence, France
http://www.ieee-icecs2014.org
Paper submission deadline: **June 13**

Dear Colleagues,
The **IEEE 2014 International Conference on Electronics Circuits and Systems** will be held in the south of France at the Pharo palace, Marseille, December 7-10.
The organization committee is very pleased to invite you to contribute to the 21st edition of ICECS and will be very pleased to welcome you in Provence.
Best regards,
The Organization Committee of IEEE ICECS 2014

TOPICS

- Analog & Mixed Signal Circuits and Signal Processing
- VLSI Systems, Applications and Computer Aided Network Design
- Wireless Communication, RF Circuits and Systems
- Bioengineering Circuits and Systems
- Circuits and Systems for Communications
- Digital Circuits and Signal Processing
- Emerging Technologies
- Low-Power and Harvesting Techniques
- Multimedia Systems and Signal Processing
- Neural Network Circuits and Systems
- Linear and Nonlinear Circuits and Systems
- Photonic and Optoelectronic Circuits
- Sensing and Sensor Networks
- Test and Reliability

AUTHOR'S SCHEDULE
Submission of Tutorials/Special Sessions: May 2
Notification of Tutorials/Special Sessions Proposals: May 23
Submission of Regular/Student Papers: June 13
Notification of Paper Acceptance: September 5
Submission of Camera Ready Papers: September 22

IEEE Visual Communications and Image Processing Conference

(IEEE VCIP 2014)

Valletta, Malta, 7 – 10 December, 2014

http://www.um.edu.mt/events/vcip2014

The IEEE Visual Communications and Image Processing (IEEE VCIP) Conference, sponsored by the IEEE Circuits and Systems Society, IEEE Malta Section and by the University of Malta, will be held in Valletta, the capital city of the Historical Malta, during December 7-10, 2014.

Since 1986, VCIP has served as a premier forum in SPIE for the exchange of fundamental research results and technological advances in the field of visual communications and image processing. The 2014 edition will be the forth time that VCIP will be held under the auspices of the IEEE. VCIP 2014 shall inherit the tradition of previous conferences in providing fertile ground for leading engineers and scientists from around the world to escalate collaboratively the research frontiers within these and related areas. Original and unpublished work relevant to, but are not limited to, the following topics are hereby solicited.

Emerging Techniques for Next Generation Video/Image Coding

Advanced Techniques for 3D Videos

Visual Communications

Systems and Techniques for Human Interaction

Embedded Systems and Architectural Implementations

Cloud Multimedia Systems, Applications and Services

Submission of regular and demo papers

Prospective authors are invited to submit a 4 page, full-length paper, including results, figures, and references. Furthermore, IEEE VCIP 2014 considers the submission of one-page demo paper with the description of the proposed working system.

Tutorial and Special Session proposals

Researchers, developers, and practitioners from related communities are invited to submit proposals for organizing tutorials and special sessions on various emerging topics described above. Tutorials will be held on Sunday, December 7, 2014. Proposals must include title, outline, contact and biography information of the presenter, and a short description of the material to be distributed. Special session proposals must provide title, rationale, contact and biography information of the organizers, and a list of possible papers with tentative title and abstract. Special session papers will be regularly reviewed.

All Papers must be submitted through Microsoft Content Management System following the IEEE format. Papers accepted for regular and special sessions must be registered and presented to be included in the proceedings that will be submitted in the IEEE Xplore® package. Awards will be given to a number of papers with best quality. Please visit http://www.um.edu.mt/events/vcip2014 for additional information regarding paper and proposal formatting instructions and submission details.

Important Dates:

- Submission of Proposals for Tutorial and Special Session – 28th March, 2014
- Acceptance Notification of Tutorial and Special Session Proposals – 11th April, 2014
- Submission of Papers for Regular, Demo, and Special Sessions – 5th May, 2014
- Acceptance Notification – 10th August, 2014
- Submission of Camera Ready Paper and Registration – 29th August, 2014

Конференция «TMPA-2013»

TMPA-2013
EXTENT CONFERENCE
ИНСТРУМЕНТЫ И МЕТОДЫ АНАЛИЗА ПРОГРАММ

В октябре 2013 года в Костроме прошла первая конференция по программной инженерии «Инструменты и методы анализа программ – 2013» (Tools & Methods of Program Analysis, TMPA-2013). Санкт-Петербургский государственный политехнический университет был одним из ее организаторов.

Для Политехнического университета это не просто рядовое событие, а один из важных шагов в цепочке мероприятий, направленных на решение важнейшей долгосрочной задачи по повышению глобальной конкурентоспособности Университета на мировом рынке образования и научных исследований. Став летом 2013 года одним из 15 победителей государственной программы повышения глобальной конкурентоспособности российских университетов к 2020 году (Программа 5-100-2020), Университет поставил перед собой амбициозную цель – кардинально изменить систему образования и уровень проводимых научных исследований. В числе решаемых задач – повышение публикационной и конференционной активности сотрудников Университета. Одним из путей решения этой задачи является качественное изменение уровня конференций, в которых принимают участие сотрудники Университета, а также организация новых конференций по приоритетным для Университета научным направлениям, организованным по международным стандартам. Обязательными требованиями к современным научным конференциям являются:

- интернациональный программный комитет;
- высокий уровень требований к присылаемым статьям;
- профессиональное (peer review) рецензирование;
- поддержка конференции ведущими университетами и научными организациям;
- индексирование конференционных статей ведущими научными базами данных.

Конференция «Инструменты и методы анализа программ – 2013» отвечает всем перечисленным требованиям: в ее организации приняли участие несколько университетов и институтов Академии наук, программный комитет представлял различные научные школы и ведущие высокотехнологичные компании. Конференция прошла под эгидой сообщества IEEE, на 40 докладов было получено более 120 рецензий, для представления на конференции отобрали 26 лучших докладов.

После проведения конференции часть лучших докладов была рекомендована для доработки и последующей публикации в ведущих российских рецензируемых журналах. В рамках этой инициативы в журнале «Научно-технические ведомости СПбГПУ. Информатика. Телекоммуникации. Управление» представлено шесть статей, рекомендованных для публикации программным комитетом конференции.

Вторая конференция «Инструменты и методы анализа программ – 2014» (Tools & Methods of Program Analysis, TMPA-2014) будет проходить 14-15 ноября 2014 года в Костроме. Вся информация о программе конференции, сроках приема докладов и приглашенных докладчиках будет оперативно отражаться на сайте конференции: http://tmpaconf.org

В.М. Ицыксон,
сопредседатель программного комитета конференции TMPA-2013

УДК 004.415.53

В.П. Иванников, А.С. Камкин, М.М. Чупилко

ПРОВЕРКА КОРРЕКТНОСТИ ПОВЕДЕНИЯ HDL-МОДЕЛЕЙ ЦИФРОВОЙ АППАРАТУРЫ НА ОСНОВЕ ДИНАМИЧЕСКОГО СОПОСТАВЛЕНИЯ ТРАСС

V.P. Ivannikov, A.S. Kamkin, M.M. Chupilko

VERIFYING CORRECTNESS OF HDL-MODEL BEHAVIOR ON THE BASIS OF DYNAMICAL TRACE MATCHING

Проверка корректности поведения HDL-моделей является неотъемлемой частью динамической верификации аппаратуры. Как правило, она основана на сравнении поведения HDL-модели с поведением эталонной модели, разработанной на языке программирования. В процессе верификации на обе модели подается одна и та же последовательность стимулов; реакции перехватываются и сравниваются друг с другом. Из-за абстрактности эталонной модели сопоставление трасс не является тривиальной задачей: порядок событий может не совпадать, а некоторые события одной трассы могут отсутствовать в другой. Рассмотрен метод динамического сопоставления трасс для моделей аппаратуры разного уровня абстракции. Метод успешно применен в нескольких промышленных проектах по верификации модулей микропроцессоров.

ДИНАМИЧЕСКАЯ ВЕРИФИКАЦИЯ; МОДУЛЬНАЯ ВЕРИФИКАЦИЯ; HDL; СОПОСТАВЛЕНИЕ ТРАСС; ЧАСТИЧНО-УПОРЯДОЧЕННЫЕ МУЛЬТИМНОЖЕСТВА.

Correctness checking of HDL-model behavior is an integral part of dynamical verification of hardware. As a rule, it is based on comparing of HDL-model behavior and reference model behavior, developed in high-level programming languages. Being verified, both models are applied with the same stimulus sequences; their reactions are caught and checked against each other. Due to the abstractness of the reference model, the checking is not a trivial task as event sequences can be different and some events of one trace may miss in the other one. A methodology of dynamical trace matching for hardware models of different abstraction levels is considered in the paper. The methodology has been successfully used in a number of industrial projects of unit-level microprocessor verification.

DYNAMICAL VERIFICATION; UNIT-LEVEL VERIFICATION; HDL; TRACE MATCHING; PARTIALLY ORDERED MULTISETS.

Несмотря на развитие формальных методов, *динамическая верификация* остается основным методом проверки сложной цифровой аппаратуры [1]. Объектом верификации выступают не сами устройства, а их модели, разработанные на специальных *языках описания аппаратуры* (Hardware Description Languages – HDL), например, на Verilog или VHDL (такие модели называются *HDL-моделями*) [2]. С одной стороны, HDL-модели представляют основу для автоматизированного производства интегральных схем; с другой стороны, это имитационные модели, поведение которых можно анализировать в специальных средах (*симуляторах*). Для автоматизации проверки корректности поведения HDL-моделей разрабатываются *мониторы*, перехватывающие события на входах и выходах (*стимулы* и *реакции*) и проверяющие допустимость получаемой трассы [3].

Существует множество подходов к формальному описанию поведения, позволяющих автоматизировать создание мониторов: *расширенные регулярные выражения* [4], *контрактные спецификации* [5], *системы пра-*

вил [6], *темпоральные утверждения* [7], однако указанные формализмы не покрывают всех потребностей индустрии. Большинство компаний, проектирующих аппаратуру, для оценки проектных решений используют *исполнимые модели*, разработанные на языках программирования (прежде всего, на C и C++) [8]. Экономически целесообразно использовать эти модели и для верификации создаваемой аппаратуры, в частности, для проверки корректности поведения HDL-моделей.

При наличии исполнимой модели применяют следующую схему проверки корректности поведения: эталонная модель (по терминологии тестирования соответствия, *спецификация*) выполняется совместно с проверяемой HDL-моделью (*реализацией*); на спецификацию поступает та же последовательность стимулов, что и на реализацию; реакции обеих моделей перехватываются монитором и сравниваются. Основная проблема указанной схемы связана с обобщенным характером спецификации, из-за чего порядок реакций, выдаваемых спецификацией и реализацией, а также их состав могут различаться. Перед тем как использовать эталонную модель для мониторинга, необходимо понять, насколько она абстрактна и насколько можно доверять производимым ею трассам.

В статье предлагается способ построения мониторов для HDL-моделей аппаратуры, адаптируемый для эталонных моделей разного уровня абстракции. Рассматриваемый подход может быть формализован на основе модели *частично упорядоченных мультимножеств* [9]. Поведение спецификации и реализации описывается *временны́ми трассами* над общим алфавитом. Сведения об абстрактности спецификации позволяют обобщать последовательности выдаваемых реакций в частично упорядоченные мультимножества, в которых для каждой реакции задан допустимый *временной интервал*. Монитор проверяет, что трасса реализации является *линеаризацией* обобщенной трассы спецификации (или ее подмножества) и реакции реализации удовлетворяют ограничениям, заданными временны́ми интервалами.

Основные понятия и обозначения

В дальнейшем будем использовать следующие обозначения: Σ – конечный алфавит *событий*; $\mathbb{T}$ – *временна́я область* (для определенности, $\mathbb{N}$). Последовательности событий называются *словами*. Σ^* обозначает множество всех (конечных) слов над алфавитом Σ.

В контексте динамической верификации HDL-моделей удобно структурировать алфавит событий, разбив его на два множества: множество *входных событий* (*стимулов*) I и множество *выходных событий* (*реакций*) O, а также сопоставив каждому событию его *порт*: $port : \Sigma \to \{1, 2, ..., k\}$. На содержательном уровне стимул представляет собой посылку запроса к устройству, а реакция – выдачу ответа. При этом события на разных портах могут происходить параллельно.

Определение 1 (Временно́е слово [10]). Временны́м словом (timed word) w над алфавитом Σ и временно́й областью $\mathbb{T}$ называется конечная последовательность $(a_1, t_1)...(a_n, t_n)$ временны́х событий $(a_i, t_i) \in \Sigma \times \mathbb{T}$, удовлетворяющая следующим ограничениям:

1) для каждого $1 \le i < n$ выполняется неравенство $t_i \le t_{i+1}$;

2) для всех $1 \le i, j \le n$, таких что $i \ne j$ и $t_i = t_j$, $port(e_i) \ne port(e_j)$. □

В параллельных системах, к которым относится аппаратура, используется концепция *независимости* событий: два события считаются *независимыми*, если между ними нет причинно-следственной связи (для таких событий не накладываются ограничения на их относительный порядок). Эта идея лежит в основе двух формальных моделей параллельных вычислений: (1) *трасс Мазуркевича* [11] и (2) *частично упорядоченных мультимножеств* [9]. В настоящей статье мы будем придерживаться более общей второй модели.

Определение 2 (Частично упорядоченное мультимножество [9]). Σ-размеченным частичным порядком называется кортеж $\langle V, \prec, \lambda \rangle$, где V – конечное множество вершин, $\prec \subseteq V \times V$ – отношение частичного порядка и $\lambda : V \to \Sigma$ – функция

разметки. Два Σ-размеченных частичных порядка называются эквивалентными, если они изоморфны относительно $\prec$ и λ (либо совпадают, либо отличаются только названием вершин). Частично упорядоченным мультимножеством (pomset, partially ordered multiset) над алфавитом Σ называется класс эквивалентности Σ-размеченных частичных порядков. □

Для удобства мы будем использовать конкретного представителя (конкретный размеченный частичный порядок) для обозначения частично упорядоченного мультимножества. *Линеаризацией* частично упорядоченного мультимножества $\langle V, \prec, \lambda\rangle$ называется размеченный полный порядок $\langle V, \leq, \lambda\rangle$, где $\prec \subseteq \leq$.

Определение 3 (Временна́я трасса [12]). Временно́й трассой над алфавитом Σ и временно́й областью $\mathbb{T}$ называется четверка $\langle V, \prec, \lambda, \theta\rangle$, где $\langle V, \prec, \lambda\rangle$ – частично упорядоченное мультимножество, а $\theta : V \to \mathbb{T}$ – функция времени, удовлетворяющая следующему условию: для всех $x, y \in V$, из $x \prec y$ вытекает $\theta(x) \leq \theta(y)$. □

Множество всех трасс над алфавитом Σ и временно́й областью $\mathbb{T}$ обозначается $\mathbb{M}_\theta(\Sigma, \mathbb{T})$ или, для краткости, $\mathbb{M}$. Заметим, что определенные ранее временны́е слова являются частным случаем временны́х трасс (это трассы с тривиальным отношением частичного порядка на множестве вершин – отношением равенства).

Для заданной непустой трассы $\sigma = \langle V, \prec, \lambda, \theta\rangle$ введем обозначения:

$$begin(\sigma) = \min_{x \in V}\{\theta(x)\};$$
$$end(\sigma) = \max_{x \in V}\{\theta(x)\};$$
$$V_{[t,t+\Delta t]} = \{x \in V \mid \theta(x) \in [t, t+\Delta t]\};$$
$$\sigma_{[t,t+\Delta t]} = \langle V_{[t,t+\Delta t]}, \prec_{|V_{[t,t+\Delta t]}}, \lambda_{|V_{[t,t+\Delta t]}}, \theta_{|V_{[t,t+\Delta t]}}\rangle.$$

Пусть $\mathcal{TI}(\mathbb{T})$ – множество временны́х интервалов во временно́й области $\mathbb{T}$, то есть $\mathcal{TI}(\mathbb{T}) = \{[t, t+\Delta t] \mid t, t+\Delta t \in \mathbb{T}\}$.

Определение 4 (Интервальная трасса). Интервальной трассой над алфавитом Σ и временно́й областью $\mathbb{T}$ называется четверка $\sigma = \langle V, \prec, \lambda, \theta\rangle$, где $\langle V, \prec, \lambda\rangle$ – частично упорядоченное мультимножество, а $\delta : V \to \mathcal{TI}(\mathbb{T})$ – функция, сопоставляющая каждой вершине временно́й интервал. □

В дальнейшем мы будем иметь дело с *расширенными интервальными трассами*, описываемыми пятерками $\sigma = \langle V, \prec, \lambda, \theta, \delta\rangle$. В таких трассах с каждой вершиной $x \in V$ связан как момент времени $\theta(x)$, так и временно́й интервал $\delta(x) = [\theta(x) - \Delta t^-(x), \theta(x) + \Delta t^+(x)]$. Будем считать, что функции $\Delta t^\pm : V \to \mathbb{T}$ ограничены: существуют константы $\Delta T^\pm > 0$, такие что $|\Delta t^\pm(x)| \leq \Delta T^\pm$ для всех $x \in V$, и, кроме того, их значения зависят только от событий: если $\lambda(x) = \lambda(x')$, то $\Delta t^\pm(x) = \Delta t^\pm(x')$.

Динамическая верификация на основе исполнимых моделей

Временно́е слово (временна́я трасса с тривиальным частичным порядком) описывает конкретное выполнение HDL-модели (*реализации*), в то время как расширенная интервальная трасса описывает поведение эталонной модели (*спецификации*). Наша цель – проверить *в динамике* (on-the-fly), что трасса реализации w_I (временно́е слово) соответствует трассе спецификации σ_S (расширенной интервальной трассе). Поясним, почему в качестве спецификационной трассы рассматривается расширенная интервальная трасса. В процессе выполнения спецификация порождает конкретную трассу w_S, описываемую временны́м словом. Прямое сравнение двух временны́х слов, w_I и w_S, на равенство возможно только для *потактово точной* спецификации. Как правило, спецификация абстрактнее реализации, особенно в отношении временны́х свойств. Для того чтобы использовать абстрактную спецификацию для проверки поведения реализации, в ее коде с помощью специальных средств задаются ограничения на порядок реакций и время их возникновения (рис. 1). Подробнее об этом говорится в разделе «Инструментальная поддержка и опыт применения».

Отношение соответствия. В дальнейшем мы будем называть временны́е трассы с тривиальным частичным порядком (порядком, заданным отношением равенства) *трассами выполнения*. Если $\Sigma = I \cup O$, то трассы выполнения над алфавитом I будем

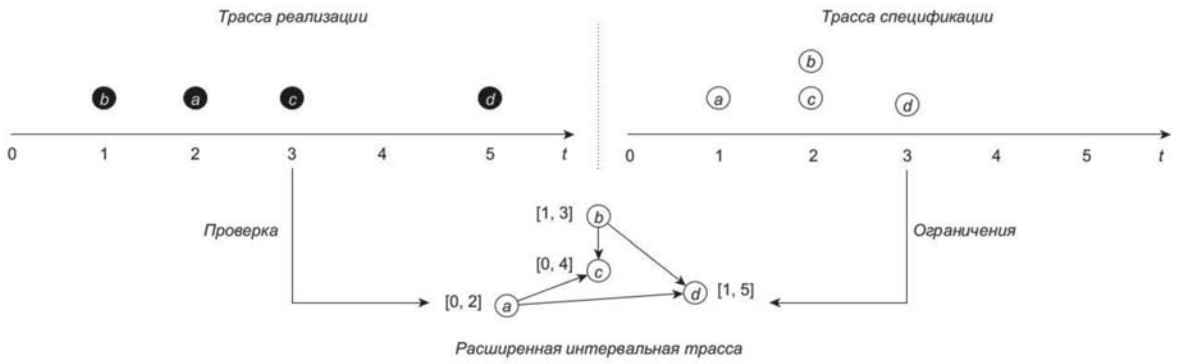


Рис. 1. Схема проверки соответствия между реализацией и спецификацией

называть *входными последовательностями*, а трассы выполнения над алфавитом O – *выходными последовательностями*. Отметим, что тривиальный частичный порядок в трассах выполнения отражает тот факт, что реализация рассматривается как «черный ящик», и причинно-следственные связи между ее событиями если и известны, то не от самой реализации. Множества входных и выходных последовательностей обозначаются $\mathbb{I}_\theta(\Sigma,\mathbb{T})$ и $\mathbb{O}_\theta(\Sigma,\mathbb{T})$ соответственно. Для краткости будем использовать сокращенные обозначения: $\mathbb{I}=\mathbb{I}_\theta(\Sigma,\mathbb{T})$ и $\mathbb{O}=\mathbb{O}_\theta(\Sigma,\mathbb{T})$.

Определение 5 (Поведение). Детерминированным поведением над алфавитом $\Sigma = I \cup O$ и временной областью $\mathbb{T}$ называется (частичная) функция $\mathcal{B}:\mathbb{I}\times\mathbb{T}\to\mathbb{O}$, удовлетворяющая следующим ограничениям:

1) для всех $w\in\mathbb{I}$ и $t\in\mathbb{T}$ выполняется неравенство $end(\mathcal{B}(w,t))\le t$;

2) для всех $w\in\mathbb{I}$ и $t\in\mathbb{T}$ справедливо равенство $\mathcal{B}(w,t)=\mathcal{B}(w_{[0,t]},t)$. □

Поведение описывает, каким образом входная последовательность преобразуется в выходную последовательность, принимая во внимание момент времени, в который производится наблюдение.

Предположим, что у нас имеется исполнимая спецификация. Рассмотрим, как ее можно использовать для динамической проверки поведения реализации. Для этого расширим определение поведения, позволив спецификации возвращать расширенные интервальные трассы. Обозначим множество всех таких трасс символом $\mathbb{O}_{\theta\delta}=\mathbb{O}_{\theta\delta}(\Sigma,\mathbb{T})$. Таким образом, поведение спецификации – это функция $\mathcal{B}:\mathbb{I}\times\mathbb{T}\to\mathbb{O}_{\theta\delta}$, удовлетворяющая указанным в определении ограничениям (используемые обозначения имеют смысл и для расширенных интервальных трасс).

Пусть $\mathcal{I}$ и $\mathcal{S}$ – *поведение реализации* и *поведение спецификации* соответственно. Для заданной входной последовательности $w\in\mathbb{I}$ и момента времени $t\in\mathbb{T}$ рассмотрим выход реализации и спецификации: $\mathcal{I}(w,t)=\langle V_\mathcal{I},=,\lambda_\mathcal{I},\theta_\mathcal{I}\rangle$ и $\mathcal{S}(w,t)=\langle V_\mathcal{S},\prec_\mathcal{S},\lambda_\mathcal{S},\theta_\mathcal{S},\delta_\mathcal{S}\rangle$. Введем обозначения:

$$past_\mathcal{I}^{\Delta t}(w,t)=\{y\in\mathcal{I}(w,t)\mid\theta_\mathcal{I}(y)\le(t-\Delta t^-(y))\};$$

$$past_\mathcal{I}(w,t)=\{y\in\mathcal{I}(w,t)\mid\theta_\mathcal{I}(y)\le t\};$$

$$past_\mathcal{S}^{\Delta t}(w,t)=\{x\in\mathcal{S}(w,t)\mid\theta_\mathcal{S}(x)\le(t-\Delta t^+(x))\};$$

$$past_\mathcal{S}(w,t)=\{x\in\mathcal{S}(w,t)\mid\theta_\mathcal{S}(x)\le t\};$$

$$match(x,y)=(\lambda_\mathcal{I}(y)=\lambda_\mathcal{S}(x))\wedge(\theta_\mathcal{I}(y)\in\delta_\mathcal{S}(x)).$$

Заметим, что в определении $past_\mathcal{I}^{\Delta t}(w,t)$ используется функция Δt^-, заданная для выходной трассы спецификации (для трассы реализации, не являющейся расширенной интервальной трассой, функции $\Delta t^\pm$ не определены).

Определение 6 (Отношение соответствия). Говорят, что поведение реализации $\mathcal{I}$ соответствует поведению спецификации $\mathcal{S}$, если $dom\mathcal{I}=dom\mathcal{S}$ ($\mathcal{I}$ и $\mathcal{S}$ определены на одном множестве входных последовательностей) и для всех $w\in dom\mathcal{S}$ и $t\in\mathbb{T}$ существует бинарное

отношение $\mathcal{M}(w,t) \subseteq \{(x,y) \in past_{\mathcal{S}}(w,t) \times \times past_{\mathcal{I}}(w,t) \mid match(x,y)\}$, называемое сопоставлением, такое что:

1) $\mathcal{M}(w,t)$ взаимно однозначно;

2) для каждой реакции спецификации $x \in past_{\mathcal{S}}^{\Delta t}(w,t)$ существует реакция реализации $y \in past_{\mathcal{I}}(w,t)$, такая что $(x,y) \in \mathcal{M}(w,t)$;

3) для каждой реакции реализации $y \in past_{\mathcal{I}}^{\Delta t}(w,t)$ существует реакция спецификации $x \in past_{\mathcal{S}}(w,t)$, такая что $(x,y) \in \mathcal{M}(w,t)$;

4) для всех $(x,y),(x',y') \in \mathcal{M}(w,t)$ если $x \prec x'$ то $\theta_{\mathcal{I}}(y) \le \theta_{\mathcal{I}}(y')$.

Если для некоторых $w \in dom\mathcal{S}$ и $t \in \mathbb{T}$ сопоставления не существует, то говорят, что $\mathcal{I}$ не соответствует $\mathcal{S}$, при этом w называется контрпримером. □

Рис. 2 иллюстрирует определение отношения соответствия для некоторой входной последовательности (будучи неважной, она на рисунке не показана) и момента времени $t = 4$. Верхняя часть рисунка показывает реакции реализации (черные кружки с белыми надписями: *b*, *a* и *c*), нижняя – реакции спецификации (белые кружки с черными надписями: *a*, *b*, *c* и *d*). Обозначим вершины трассы (собственно, кружки) символами y_b, y_a и y_c (для реализации) и x_a, x_b, x_c и x_d (для спецификации). Между вершинами реализационной трассы нет причинно-следственных связей. Вершины спецификационной трассы частично упорядочены (непосредственное предшествование событий показано стрелками: $x_a \to x_c$, $x_b \to x_c$, $x_a \to x_d$ и $x_b \to x_d$) и помечены временными интервалами ($\delta(x_a) = [0,2]$, $\delta(x_b) = [1,3]$, $\delta(x_c) = [0,4]$ и $\delta(x_d) = [1,5]$). Сопоставление реакций отмечено пунктирными линиями ((x_a, y_a), (x_b, y_b) и (x_c, y_c)). Легко видеть, что изображенное отношение является сопоставлением (в смысле данного выше определения): (1) оно взаимно однозначно; (2 и 3) оно включает все реакции с истекшим «временем жизни»; (4) оно сохраняет спецификационный порядок: (a) $x_a \prec x_c$ и $\theta(y_a) = 2 \le 3 = \theta(y_c)$; (b) $x_b \prec x_c$ и $\theta(y_b) = 1 \le 3 = \theta(y_c)$. Безусловно, каждая пара этого отношения удовлетворяет условию сопоставления реакций: (a) $\lambda(x_a) = \lambda(y_a) = a$ и $\theta(y_a) = 2 \in [0,2] = \delta(x_a)$; (b) $\lambda(x_b) = \lambda(y_b) = b$ и $\theta(y_b) = 1 \in [1,3] = \delta(x_b)$; (c) $\lambda(x_c) = \lambda(y_c) = c$ и $\theta(y_c) = 3 \in [0,4] = \delta(x_c)$.

Динамическое сопоставление трасс. Монитор, осуществляющий сопоставление трасс реализации и спецификации, может быть представлен как *временной автомат* [10] с двумя типами входных портов: (1) порты для получения *спецификационных реакций* и (2) порты для получения *реали-*

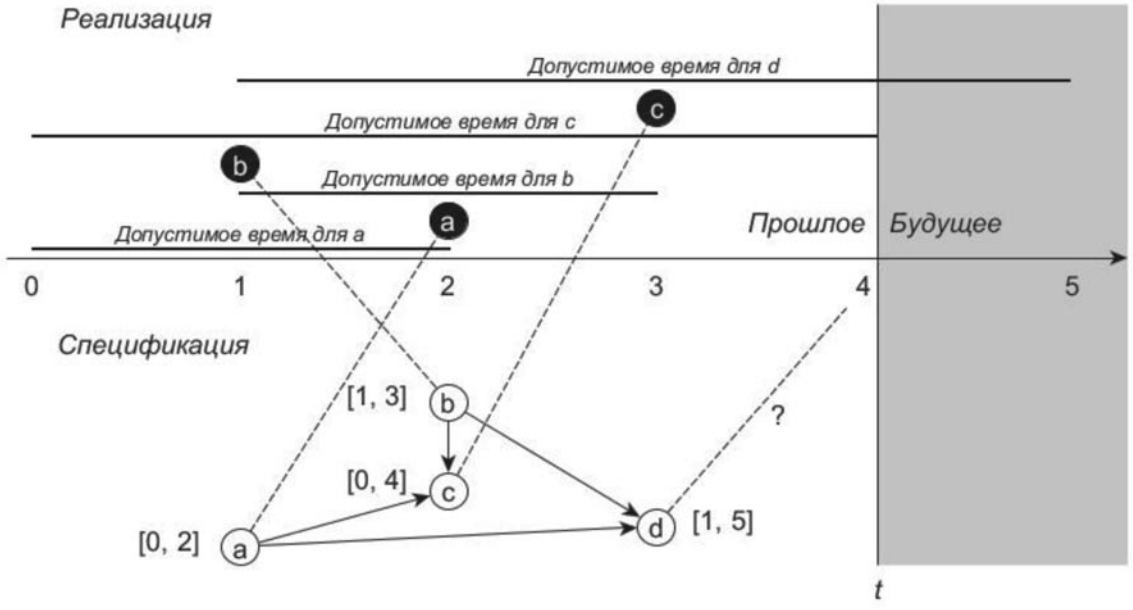


Рис. 2. Соответствие между реализацией и спецификацией

зационных реакций. Когда автомат обнаруживает расхождение в трассах реализации и спецификации, он переходит в определенное состояние, информируя, тем самым, что реализация не соответствует спецификации.

Ниже представлено формальное описание монитора в виде системы *действий с условиями* (guarded actions) [13]. Каждое *действие* атомарно и выполняется, как только соответствующее *условие* становится истинным. Действия и условия зависят от внешней переменной *t*, отражающей *текущее время*, и реакций, выдаваемых реализацией и спецификацией в ответ на одну и ту же последовательность стимулов (I и S соответственно). Значение переменной t монотонно возрастает в процессе верификации. Запись $y \in I[t]$ ($x \in S[t]$) означает, что в момент времени t реализация (спецификация) выдает реакцию $y(x)$. Пометка $[\uparrow \theta_{\mathcal{I}}(y)]$ в действии *onSpecOut* говорит о том, что перебор реакций $y \in past_I$ осуществляется в порядке возрастания $\theta_{\mathcal{I}}(y)$.

Пусть X – подмножество вершин трассы спецификации, $\prec$ – заданное на множестве X отношение частичного порядка, y – вершина трассы реализации. Введем следующие обозначения:

$$\min_{\prec}(X) = \{x \in X \mid \nexists x' \in X.\, x' \neq x \wedge x' \prec x\};$$

$$match(X, y) = \{x \in X \mid match(x, y)\}.$$

Кроме того, определим две функции, на которых основано описание монитора: *первичный арбитр* ($arbiter_S$) и *вторичный арбитр* ($arbiter_I$):

$$arbiter_S(X) = \begin{cases} \min_{\prec}(X) & \text{если } X \neq \varnothing, \\ \phi & \text{иначе } (\phi \notin X); \end{cases}$$

$$arbiter_I(X, y) = \begin{cases} \arg\min_{x \in match(X,y)} \{\theta_S(x)\} \\ \text{если } match(X, y) \neq \varnothing, \\ \phi \text{ иначе.} \end{cases}$$

Первичный арбитр для заданного множества X вершин спецификационной трассы возвращает его подмножество, состоящее из минимальных по отношению частичного порядка $\prec$ элементов. Вторичный арбитр сопоставляет вершину y реализационной трассы с вершинами спецификационной трассы из множества X.

Ниже приведен порядок проверки условий и запуска действий в момент времени t (при несоблюдении этого порядка возможны ложные сообщения об ошибках):

1) инициализация (*onInnitialize*);

2) прием реакции (*onSpecOut*, *onImplOut*);

3) превышение лимита времени (*onSpecTime*, *onImplTime*);

4) завершение (*onFinalize*).

Когда говорят, что некоторое свойство φ выполняется в момент времени t, имеется в виду, что φ выполняется после завершения всех действий, активированных в момент t. Для многопортовых систем (что типично для аппаратуры) монитор можно разбить на слабо связанные компоненты, параллельно обслуживающие отдельные порты.

Утверждение 1 (Корректность). Входная последовательность является контрпримером тогда и только тогда, когда монитор завершается с $verdict = false$.

Схема доказательства. Докажем, что если $verdict = false$, то w, входная последовательность, – контрпример. Согласно описанию монитора, имеются две возможности, когда переменной *verdict* присваивается значение *false*:

1) при выполнении действия *onSpecTime*;

2) при выполнении действия *onImplTime*.

В обоих случаях можно показать, что w – контрпример. Рассмотрим для примера случай 1. Пусть t – время завершения работы монитора (вызова оператора **terminate**). Выполнение действия *onSpecTime* в момент времени t подразумевает, что существует реакция спецификации $x \in past_S$, такая что $(\theta_S(x) + \Delta t^+(x)) \leq t$ (следовательно, $x \in past_S^{\Delta t}(w, t)$). Предположим, что существует сопоставление $\mathcal{M}$ (то есть w – не контрпример). Тогда найдется реакция реализации $y \in past_{\mathcal{I}}(w, t)$, такая что $(x, y) \in \mathcal{M}$ (см. п. 2 определения отношения соответствия). По какой-то причине эта реакция не была сопоставлена монитором с реакцией x. Не ограничивая общности рассуждений, положим, что реакция y пришла раньше реакции x. Возможны две причины, по которым эти реакции не были

Действие 1 *onSpecOut*$[x], x \in S[t]$
Если: *true*
Вход: x
$past_S \Leftarrow past_S \cup \{x\}$
if $x \in arbiter_S(past_S)$ **then**
  for all $y \in past_I$ [$\uparrow \theta_{\mathcal{I}}(y)$] **do**
    if x = arbiter$_I$({x}, y) **then**
      $past_S \Leftarrow past_S \setminus \{x\}$
      $past_I \Leftarrow past_I \setminus \{y\}$
      $match \Leftarrow match \cup \{(x, y)\}$
      break
    end if
  end for
end if

Действие 2 *onImplOut*$[y], y \in I[t]$
Если: *true*
Вход: y
$past_I \Leftarrow past_I \cup \{y\}$
$x \Leftarrow arbiter_I(arbiter_S(past_S), y)$
if $x \neq \phi$ **then**
  $past_S \Leftarrow past_S \setminus \{x\}$
  $past_I \Leftarrow past_I \setminus \{y\}$
  $match \Leftarrow match \cup \{(x, y)\}$
end if

Действие 3 *onSpecTime*$[x], x \in past_S$
Если: $\left(\theta_S(x) + \Delta t^+(x)\right) \leq t$
Вход: x
$past_S \Leftarrow past_S \setminus \{x\}$
verdict $\Leftarrow$ *false*
trace («Missing output»)
terminate

Действие 4 *onImplTime*$[y], y \in past_I$
Если: $\left(\theta_{\mathcal{I}}(y) + \Delta t^-(y)\right) \leq t$
Вход: y
$past_I \Leftarrow past_I \setminus \{y\}$
verdict $\Leftarrow$ *false*
trace («Unexpected output»)
terminate

Действие 5 *onInitialize*
Если $t = 0$
Вход: $\varnothing$
$past_S \Leftarrow \varnothing$
$past_I \Leftarrow \varnothing$
$match \Leftarrow \varnothing$

Действие 6 *onFinalize*
Если
$\mathbf{max}\left(\mathbf{end}(S) + \Delta T^+, \mathbf{end}(I) + \Delta T^-\right) \leq t$
Вход: $\varnothing$
verdict $\Leftarrow$ *true*
terminate

сопоставлены при выполнении действия *onSpecOut* для реакции x:

a. при получении реакции x было выполнено условие $x \notin arbiter_S(past_S)$ (см. описание действия *onSpecOut* и определение функции $arbiter_I$);

b. к этому моменту реакция y уже была сопоставлена с другой реакцией спецификации (обозначим эту реакцию x^*).

Рассмотрим случай *a*. Условие $x \notin arbiter_S(past_S)$ выполняется, только если существует реакция спецификации $x' \neq x,$, такая что $x' \prec x$ и $x' \in arbiter_S(past_S)$ (см. определение функции $arbiter_S$). Если таких реакций несколько, пусть x' – та из них, на которой достигается минимум функции θ_S. Пусть y' – реакция реализации, такая что $(x', y') \in \mathcal{M}$. Отметим, что реакция y' пришла не позже реакции y (это следует из п. 4 определения отношения соответствия), и она не была сопоставлена с реакцией x' (в противном случае в момент прихода реакции x было бы выполнено условие $x' \notin past_S$ и, как следствие, $x' \notin arbiter_S(past_S)$). Заметим, что решение о сопоставлении реакции x' с одной из

реакций реализации должно быть принято не позднее момента времени t, даже если $(\theta_S(x') + \Delta t^+(x')) > t$: поскольку $x' \prec x$, то

$$\theta_I(y'^*) \le \theta_I(y^*) \le (\theta_S(x) + \Delta t^+(x)) \le t,$$

где y^* и y'^* — произвольные реакции реализации, которые могут быть сопоставлены с x и x' соответственно. Причиной, по которой реакции x' и y' не были сопоставлены, может быть только b. Таким образом, случай a для пары реакций (x, y) сводится к случаю b для пары реакций (x, y).

Рассмотрим теперь случай b. Пусть N_I — общее число реакций реализации, произошедших до момента времени t включительно. Очевидно, что реакция x^* (с которой была сопоставлена реакция y) произошла не позже реакции x. Пусть y^* — реакция реализации, такая что $(x^*, y^*) \in \mathcal{M}$. Отметим, что реакция y^* произошла не раньше реакции y (в противном случае, поскольку в действии *onSpecOut* реакции реализации рассматриваются в порядке возрастания времени их прихода, реакция x^* сопоставилась бы с реакцией y^* или иной реакцией, отличной от y, которая произошла раньше). Покажем, что выполнено условие $match(x, y^*)$:

$$\lambda_I(y^*) = \lambda_S(x^*) = \lambda_I(y) = \lambda_S(x);$$

$$(\lambda_S(x^*) = \lambda_S(x)) \Rightarrow (\Delta t^+(x^*) = \Delta t^+(x)) \Rightarrow$$
$$(\theta_S(x) - \Delta t^-(x)) \le \theta_I(y) \le \theta_I(y^*) \le$$
$$\le (\theta_S(x^*) + \Delta t^+(x^*)) \le (\theta_S(x) + \Delta t^+(x)).$$

Тем не менее реакция y^*, как и реакция y, не была сопоставлена монитором с реакцией x. Применяя аналогичные рассуждения в отношении (x, y^*) и последующих пар реакций, получим, что общее число реакций реализации не ограничено числом N_I, что противоречит определению этого числа. Следовательно, сопоставления не существует, а значит, w — контрпример, что и требовалось доказать.

Теперь докажем, что если w — контрпример, то монитор завершается с $verdict = false$. Предположим противное, т. е. то, что монитор не завершается с $verdict = false$ (это равносильно невыполнению действий *onSpecTime* и *onImplTime*). Рассмотрим произвольный момент времени t, не превосходящий время завершения работы монитора (если он завершается). Покажем, что отношение *match*, построенное к моменту t, является сопоставлением (выполнены свойства 1—4 из определения отношения соответствия). Во-первых, *match* — взаимно однозначное отношение (поскольку при каждом обновлении *match* используются только новые, не принадлежащие *match*, реакции). Во-вторых и в-третьих, *match* включает все реакции спецификации $x \in past_S^{\Delta t}(w, t)$ и все реакции реализации $y \in past_I^{\Delta t}(w, t)$ (в противном случае выполнилось бы одно из действий *onSpecTime* или *onImplTime*, что привело бы к завершению монитора с $verdict = false$). В-четвертых, если $(x, y), (x', y') \in match$ и $x \prec x'$, то $\theta_I(y) \le \theta_I(y')$ (если $x \prec x'$ и $x \ne x'$, то сопоставление пары реакций (x, y) осуществляется не позже сопоставления (x', y'): условие $x' \in arbiter_S(past_S)$ может быть выполнено только после того, как реакция x будет удалена из множества $past_S$). Из существования сопоставления вытекает, что w — не контрпример, что противоречит условию утверждения. Следовательно, монитор завершается, причем $verdict = false$, что и требовалось доказать. □

На практике встречается аппаратура, в которой операции в некоторых ситуациях *отменяют* другие операции (конфликтующие с ними и имеющие более низкий приоритет). Например, операция записи может быть отменена другой операцией записи, адресующейся к той же ячейке памяти и запущенной сразу после рассматриваемой операции. Из-за абстрактности спецификации в ее терминах не всегда возможно выразить условия, при которых происходит отмена операций и соответствующие реакции не посылаются наружу. Принимая во внимание сказанное выше, определение спецификационного поведения может быть расширено: каждая спецификационная реакция дополнительно помечается признаком возможной отмены. Предложенный подход к организации мониторов может быть перенесен и на этот случай. Следует, однако, учесть, что если некоторое событие отменяется, то также отменяются все зависимые от него события.

Обзор существующих работ

В работе [14] используется модель *автомата с частично упорядоченными входными/выходными событиями* (Partial Order Input/Output Automaton – POIOA) для представления поведения спецификации и реализации. В статье предложен метод построения тестового набора, гарантирующего обнаружение ошибок определенного типа. Если (1) реализация сообщает о приеме неподдерживаемых стимулов, (2) можно установить порядок выдачи реакций, (3) время ответа реализации ограничено, и (4) каждый единичный переход в спецификации соответствует единичному переходу в реализации, можно определить соответствие между реализацией и спецификацией. Реализация соответствует спецификации, если она принимает допускаемые спецификацией стимулы и выдает описываемые спецификацией реакции в допустимом порядке. Определение отношения соответствия, данное в этой статье, близко к используемому нами. Главными отличиями нашего подхода являются поддержка необязательных реакций и контроль временны́х интервалов.

Статья [15] описывает подход к проверке поведения временны́х систем, основанный на *инвариантах трассы*. Рассматриваются два типа инвариантов: (1) *инварианты ожидания*, выражающие то свойство, что после заданной трассы всегда ожидается определенное событие (в определенном временно́м интервале), и (2) *инварианты наблюдения*, утверждающие, что между двумя заданными событиями всегда наблюдается определенная последовательность событий. Корректность поведения реализации проверяется в два этапа: (1) проверка соответствия инвариантов спецификации, выраженной в форме *временно́го конечного автомата* (Timed Finite State Machine – TFSM); (2) проверка выполнимости инвариантов для трассы реализации. Подход представляет теоретический интерес, однако его промышленное использование может быть затруднено необходимостью постоянного согласования проверяемых инвариантов со спецификацией.

В работе [16] рассматривается метод тестирования параллельных систем с помощью неявно заданных *асинхронных конечных автоматов* (Asynchronous Finite State Machines – AFSM). Поведение реализации проверяется только в *стационарных состояниях*, в которых не ожидается выдача реакций. Используется следующая процедура: (1) собираются все события и определяется их частичный порядок; (2) строятся и проверяются все возможные линеаризации множества событий (проверка базируется на пред- и постусловиях, определенных для каждого события). Считается, что реализация соответствует спецификации, если допускается хотя бы одна из построенных линеаризаций. Применение подхода возможно только для ограниченного класса входных последовательностей: требуется, чтобы регулярно возникали стационарные состояния. В нашем случае такого ограничения нет.

Инструментальная поддержка и опыт применения

Предложенный метод к проверке корректности поведения HDL-моделей реализован в инструменте C++TESK [17]. Библиотека инструмента содержит классы и макросы для создания компонентов систем динамической верификации аппаратуры (эталонных моделей, мониторов, генераторов стимулов и др.). Возможности C++TESK для разработки эталонных моделей аппаратуры (и, соответственно, мониторов) включают средства для посылки и приема пакетов данных (примитивы *send* и *receive*), ветвления и объединения параллельных процессов (*fork* и *join*), моделирования временны́х задержек (*delay*) и задания зависимостей между пакетами (*depends*).

Ниже приведены описания некоторых примитивов C++TESK, наиболее важных для затрагиваемой в статье темы (для наглядности их синтаксис несколько отличается от используемого в инструменте):

- *delay*(*n*) – моделирует временну́ю задержку (выполнение процесса прерывается, а возобновление работы планируется через *n* единиц времени);
- *receive*(*in*) : *pkg* – ждет до тех пор, пока

входной пакет не будет получен на входном порту (*in*), а затем возвращает этот пакет (*pkg*);

• *send*(*out*, *pkg*, *opt*) – посылает выходной пакет (*pkg*) через выходной порт (*out*), указывая, является ли данный пакет обязательным или опциональным (*opt*);

• *depends*(*pkg*1, *pkg*2) – указывает, что выходной пакет (*pkg*1) зависит по данным или связан причинно-следственной связью с другим пакетом (*pkg*2), входным или выходным.

Заметим, что каждый раз, когда пакет данных выдается наружу с помощью примитива *send*, он помечается временны́м интервалом $[t - \Delta t^{-}, t + \Delta t^{+}]$, где t – время начала посылки, а $\Delta t^{\pm}$ – пользовательские параметры выходного порта. Для каждого порта также задается режим работы: *fifo* (режим по умолчанию) или *unordered*. В режиме *fifo* при посылке пакета добавляется зависимость этого пакета от последнего пакета, переданного через тот же выходной порт и находящегося в очереди ожидания (еще не сопоставленного с пакетом реализации), в режиме *unordered* никакие зависимости не добавляются. Дополнительные ограничения на порядок пакетов, если они нужны, задаются с помощью примитива *depends*.

Инструмент позволяет создавать модели аппаратуры на разных уровнях абстракции (относительно точности моделирования времени): (1) *модели без учета времени* (описывающие общие причинно-следственные связи между пакетами данных без моделирования временны́х задержек между ними: $\Delta t^{\pm} = \infty$), (2) *модели с приближенным учетом времени* (частично специфицирующие внутренние схемы арбитража пакетов, но учитывающие время лишь примерно: $\Delta t^{\pm} \le T$, где T имеет значение нескольких десятков тактов) и (3) *потактовые модели* (реализующие точное или почти точное моделирование времени: $\Delta t^{\pm} \le 1$).

Инструмент C++TESK использован для динамической верификации моду-

Опыт применения предложенного метода

Название модуля	Стадия разработки	Точность разработки	
		от	до
Буфер трансляции адресов	Поздняя/ завершающая	Приближенная	Потактовая
Модуль арифметики (FPU)	Поздняя/ завершающая	Без учета времени	–
Кэш-память 2-го уровня (L2)	Промежуточная/ поздняя	Приближенная	–
Коммутатор северного моста	Промежуточная/ завершающая	Приближенная	Потактовая
Модуль доступа к памяти	Ранняя/ промежуточная	Без учета времени	Потактовая
Контроллер прерываний	Ранняя/ промежуточная	Без учета времени	Приближенная
Модуль поиска по таблице страниц	Поздняя	Приближенная	–
Контроллер банка кэш-памяти L2	Поздняя	Потактовая	–
Буфер команд	Поздняя/ завершающая	Потактовая	–
Кэш-память 3-го уровня (L3)	Промежуточная	Приближенная	–

лей промышленных микропроцессоров, разрабатываемых в НИИСИ РАН и ЗАО «МЦСТ». Наш опыт уже был описан в [18], но с тех пор он расширился. Последняя информация о применении инструмента и заложенного в нем метода представлена в таблице. Как видно из нее, подход поддерживает верификацию как с помощью абстрактных эталонных моделей (доступных на ранних стадиях проектирования), так и с помощью потактово точных моделей (доступных на заключительных этапах). Что важно, подход позволяет использовать для верификации C++-модели, изначально создаваемые для других целей (в частности, компоненты системного симулятора микропроцессора). Таким способом, например, был проверен модуль поиска по таблице страниц.

Статья сфокусирована на использовании исполнимых моделей для динамической верификации HDL-моделей. Проблема не так проста, как кажется на первый взгляд. Проверка того, что реализация и спецификация выдают одинаковые реакции в одинаковые моменты времени в большинстве случаев не является адекватной. Спецификация в силу своей абстрактности может не учитывать многих особенностей реализации, таких как упорядочивание событий и их распределение во времени. В работе предложены механизмы, позволяющие настраивать точность проверки поведения, учитывая степень абстрактности спецификации. К ним относятся: (1) описание отношения зависимости событий, (2) расширение временны́х меток событий до временны́х интервалов и (3) пометка некоторых событий как необязательных. Основываясь на предложенных механизмах, разработан метод проверки поведения HDL-моделей. Метод был реализован в инструменте C++TESK и применялся в более чем 10 проектах по верификации модулей микропроцессоров. Наши дальнейшие исследования связаны с диагностикой ошибочного поведения HDL-моделей на основе более глубокого анализа трасс, выполняемого после сеанса верификации. Целью такого анализа является упрощение локализации ошибок путем определения характера расхождений наблюдаемого поведения HDL-модели от эталонного.

СПИСОК ЛИТЕРАТУРЫ

1. **Wile B., Goss J., Roesner W.** Comprehensive Functional Verification: The Complete Industry Cycle. Morgan Kaufmann, 2005.

2. **Bergeron J.** Writing Testbenches: Functional Verification of HDL Models. Kluwer Academic, 2000.

3. **Glasser M.** Open Verification Methodology Cookbook. Springer, 2009.

4. **Sen K., Rosu G.** Generating Optimal Monitors for Extended Regular Expressions // Electronic Notes in Theoretical Computer Science. 2003. No. 89(2). Pp. 162–181.

5. **Ivannikov V., Kamkin A., Kossatchev A., Kuliamin V., Petrenko A.** The Use of Contract Specifications for Representing Requirements and for Functional Testing of Hardware Models // Programming and Computer Software. 2007. No. 33(5). Pp. 272–282.

6. **Barringer H., Rydeheard D., Havelund K.** Rule Systems for Run-Time Monitoring: From Eagle to RuleR // In Proc. of 7th Internat. Workshop on Runtime Verification. Revised Selected Papers. 2007. Pp. 111–125.

7. **Bauer A., Leucker M., Schallhart C.** Runtime Verification for LTL and TLTL // ACM Transactions on Software Engineering and Methodology. 2011. No. 20(4). Pp. 14:1–14:64.

8. **Mintz M., Ekendahl R.** Hardware Verification with C++: A Practitioner's Handbook. Springer Science+Business Media, LLC. 2006.

9. **Pratt V.** The Pomset Model of Parallel Processes: Unifying the Temporal and the Spatial // In Seminar on Concurrency. 1984. Pp. 180–196.

10. **Alur R., Dill D.** A Theory of Timed Automata // Theoretical Computer Science. 1994. No. 126(2). Pp. 183–235.

11. **Mazurkiewicz A.** Trace Theory // In Advances in Petri Nets 1986, Part II on Petri Nets: Applications and Relationships to Other Models of Concurrency. New York: Springer-Verlag, 1987. Pp. 279–324.

12. **Chieu D., Hung D.** Timed Traces and Their Applications in Specification and Verification of Distributed Real-time Systems // In Proc. of the 3rd Symp. on Information and Communication

Technology. 2012. Pp. 31–40.

13. **Dijkstra E.W.** Guarded Commands, Nondeterminacy and Formal Derivation of Programs // Communication of the ACM. 1975. No. 18(8). Pp. 453–457.

14. **von Bochmann G., Haar S., Jard C., Jourdan G.V.** Testing Systems Specified as Partial Order Input/Output Automata // In Proc. of the 20th IFIP TC 6/WG 6.1 Internat. Conf. on Testing of Software and Communicating Systems: 8th Internat. Workshop. 2008. Pp. 169–183.

15. **Andres C., Merayo M., Nunez M.** Formal Passive Testing of Timed Systems: Theory and Tools // Software Testing, Verification & Reliability. 2012. No. 22(6). Pp. 365–405.

16. **Kuliamin V., Petrenko A., Pakoulin N., Kossatchev A., Bourdonov I.** Integration of Functional and Timed Testing of Real-Time and Concurrent Systems // In Perspectives of System Informatics. 2003. Pp. 450–461.

17. ISPRAS: C++TESK Homepage [электронный ресурс] / URL: http://forge.ispras.ru/projects/cpptesk-toolkit/

18. **Chupilko M., Kamkin A.** A TLM-Based Approach to Functional Verification of Hardware Components at Different Abstraction Levels // In Proc. of the 12th Latin-American Test Workshop. 2011. Pp. 1–6.

REFERENCES

1. **Wile B., Goss J., Roesner W.** *Comprehensive Functional Verification: The Complete Industry Cycle*, Morgan Kaufmann, 2005.

2. **Bergeron J.** *Writing Testbenches: Functional Verification of HDL Models*, Kluwer Academic, 2000.

3. **Glasser M.** *Open Verification Methodology Cookbook*, Springer, 2009.

4. **Sen K., Rosu G.** Generating Optimal Monitors for Extended Regular Expressions, *Electronic Notes in Theoretical Computer Science*, 2003, No. 89(2), Pp. 162– 181.

5. **Ivannikov V., Kamkin A., Kossatchev A., Kuliamin V., Petrenko A.** The Use of Contract Specifications for Representing Requirements and for Functional Testing of Hardware Models, *Programming and Computer Software*, 2007, No. 33(5), Pp. 272–282.

6. **Barringer H., Rydeheard D., Havelund K.** Rule Systems for Run-Time Monitoring: From Eagle to RuleR, *In Proceedings of 7th International Workshop on Runtime Verification. Revised Selected Papers*, 2007, Pp. 111–125.

7. **Bauer A., Leucker M., Schallhart C.** Runtime Verification for LTL and TLTL, *ACM Transactions on Software Engineering and Methodology*, 2011, No. 20(4), Pp. 14:1–14:64.

8. **Mintz M., Ekendahl R.** *Hardware Verification with C++: A Practitioner's Handbook*, Springer Science+Business Media, LLC, 2006.

9. **Pratt V.** The Pomset Model of Parallel Processes: Unifying the Temporal and the Spatial, *In Seminar on Concurrency*, 1984, Pp. 180–196.

10. **Alur R., Dill D.** A Theory of Timed Automata, *Theoretical Computer Science*, 1994, No. 126(2), Pp. 183–235.

11. **Mazurkiewicz A.** Trace Theory, *In Advances in Petri Nets 1986, Part II on Petri Nets: Applications and Relationships to Other Models of Concurrency*, New York, USA, Springer-Verlag, 1987, Pp. 279–324.

12. **Chieu D., Hung D.** Timed Traces and Their Applications in Specification and Verification of Distributed Real-time Systems, *In Proceedings of the 3rd Symposium on Information and Communication Technology*, 2012, Pp. 31–40.

13. **Dijkstra E.W.** Guarded Commands, Nondeterminacy and Formal Derivation of Programs, *Communication of the ACM*, 1975, No. 18(8), Pp. 453–457.

14. **von Bochmann G., Haar S., Jard C., Jourdan G.V.** Testing Systems Specified as Partial Order Input/Output Automata, *In Proceedings of the 20th IFIP TC 6/WG 6.1 International Conference on Testing of Software and Communicating Systems: 8th International Workshop*, 2008, Pp. 169–183.

15. **Andres C., Merayo M., Nunez M.** Formal Passive Testing of Timed Systems: Theory and Tools, *Software Testing, Verification & Reliability*, 2012, No. 22(6), Pp. 365–405.

16. **Kuliamin V., Petrenko A., Pakoulin N., Kossatchev A., Bourdonov I.** Integration of Functional and Timed Testing of Real-Time and Concurrent Systems, *In Perspectives of System Informatics*, 2003, Pp. 450–461.

17. ISPRAS: C++TESK Homepage. Available: http://forge.ispras.ru/projects/cpptesk-toolkit/

18. **Chupilko M., Kamkin A.** A TLM-Based Approach to Functional Verification of Hardware Components at Different Abstraction Levels, *In Proceedings of the 12th Latin-American Test Workshop*, 2011, Pp. 1–6.

ИВАННИКОВ Виктор Петрович – *директор Института системного программирования РАН.*
109004, Россия, Москва, ул. Александра Солженицына, д. 25.
E-mail: ivan@ispras.ru

IVANNIKOV, Viktor P. *Institute for System Programming of the Russian Academy of Sciences.*
109004, Alexander Solzhenitsyn Str. 25, Moscow, Russia.
E-mail: ivan@ispras.ru

КАМКИН Александр Сергеевич – *старший научный сотрудник отдела технологий программирования Института системного программирования РАН.*
109004, Россия, Москва, ул. Александра Солженицына, д. 25.
E-mail: kamkin@ispras.ru

KAMKIN, Alexander S. *Institute for System Programming of the Russian Academy of Sciences.*
109004, Alexander Solzhenitsyn Str. 25, Moscow, Russia.
E-mail: kamkin@ispras.ru

ЧУПИЛКО Михаил Михайлович – *научный сотрудник отдела технологий программирования Института системного программирования РАН.*
109004, Россия, Москва, ул. Александра Солженицына, д. 25.
E-mail: chupilko@ispras.ru

CHUPILKO, Mikhail M. *Institute for System Programming of the Russian Academy of Sciences.*
109004, Alexander Solzhenitsyn Str. 25, Moscow, Russia.
E-mail: chupilko@ispras.ru

УДК 004

А.А. Аверина, Н.А. Антонов, И.Л. Иткин

ОСОБЕННОСТИ ИНСТРУМЕНТОВ ДЛЯ ТЕСТИРОВАНИЯ, ПРИМЕНИМЫХ ПРИ ПРОМЫШЛЕННОЙ ЭКСПЛУАТАЦИИ ТРЕЙДИНГОВЫХ СИСТЕМ

A.A. Averina, N.A. Antonov, I.L. Itkin

PARTICULAR QUALITIES OF TESTING TOOLS USED IN INDUSTRIAL OPERATION OF TRADING SYSTEMS

Рассмотрены основные требования к инструментам, разработанным для верификации корректной работы электронных трейдинговых систем с использованием методов большого объема автоматизированного тестирования (HiVAT); проанализирована применимость подобных инструментов при промышленной эксплуатации трейдинговых систем.

АВТОМАТИЗАЦИЯ ТЕСТИРОВАНИЯ; ТРЕЙДИНГОВЫЕ СИСТЕМЫ; HIVAT.

This article discusses the basic requirements for tools developed to verify the correct operation of electronic trading systems using a large amount of automated testing (NiVAT); analyzes the applicability of such tools for industrial operation trading systems.

TEST AUTOMATION; TRADING SYSTEMS; HIVAT.

Ускоренное развитие технологий, используемых в электронных трейдинговых системах, признается одним из наиболее существенных факторов, влияющих на изменения в структуре и стабильности функционирования международных финансовых рынков [1]. Решения, обеспечивающие качество программных и архитектурных компонентов, становятся все более востребованными участниками рынка и организациями, регулирующими финансовый сектор [2, 3].

С технологической точки зрения трейдинговые системы в большинстве случаев являются сложными адаптивными распределенными программно-аппаратными комплексами, осуществляющими параллельное выполнение большого количества транзакций в режиме реального времени [4]. Современные системы обеспечивают время отклика, измеряемое долями миллисекунды, и при своей работе генерируют значительные объемы данных [5].

В связи с возрастающей долей финансовых транзакций, осуществляемых посредством автоматизированных компьютерных систем [6], основные взаимодействия в трейдинговых системах базируются на различных открытых и закрытых сетевых протоколах и программных интерфейсах доступа [7]. Специфика автоматизированных рабочих мест пользователя трейдинговой системы состоит, в частности, в высокой частоте обновления данных о котировках, статусе выполнения финансовых транзакций, текущих позициях и рисках [8].

Рассмотрим несколько ключевых аспектов обеспечения качества трейдинговых систем.

- Как и другие сложные многопоточные системы, трейдинговые платформы предрасположены к появлению в них трудновоспроизводимых сбоев [9], проявляющихся исключительно тогда, когда система находится под нагрузкой. Часто речь идет о функциональных проблемах, не связанных с исчерпанием ресурсов системы, но вызванных ситуацией конфликтующих друг с другом по времени условий (race conditions) при обработке параллельных транзакций или проявлением статистически маловероятных внутренних нарушений целостности

[10]. Нахождение таких проблем требует проведения верификации на границе между функциональным и нагрузочным тестированием [11].

• Обеспечение полноты покрытия тестами функциональности трейдинговой системы требует большого количества сценариев в библиотеке автоматизированных тестов [12]. Количество нуждающихся в проверке комбинаций особенно велико для производных и составных финансовых инструментов [13]. Даже однократный прогон сценариев такой библиотеки тестов через систему приводит к продолжительному исполнению последовательности автоматических тестов. Многократный запуск позволяет выделить скрытые проблемы с внутренними состояниями системы.

• Регулирующие органы, акционеры и участники торгов ожидают высокой устойчивости биржевых и брокерских систем к непредусмотренным внешним воздействиям [3]. В научной литературе детально проработаны методы тестирования устойчивости, основанные на прогоне большого количества случайно созданных данных через систему – фаззинге (fuzzing) [14].

Описанные выше свойства приводят к необходимости отправки большого количества созданных для тестирования сообщений в систему в течение продолжительного периода времени. Для обнаружения дефектов, связанных с обработкой потоков сообщений, необходима доскональная функциональная верификация выходящих сообщений и внутренних состояний системы.

Применимые для этого методы известны под аббревиатурой HiVAT – большие объемы автоматизированного тестирования (High Volume Automated Testing) [15–17]. Эти методы направлены на выявление проблем, которые с большой долей вероятности могут остаться незамеченными при использовании других подходов, требующих ручного создания сценариев для автоматического тестирования и приводящих к статистически недостаточному количеству выходных данных.

Для высоконагруженных трейдинговых систем использование HiVAT-методов служит не столько возможным способом расширения покрытия тестированием, сколько обязательным методом их тестирования в условиях, приближенных к использованию в режиме промышленной эксплуатации.

Требования к инструментам тестирования

По влиянию на тестируемую систему инструменты тестирования можно разделить на два типа: пассивные и активные [18]. Пассивное тестирование – это процесс обнаружения неисправностей в тестируемой системе путем исследования ее поведения без оказания воздействий на нормальный процесс работы. Активные инструменты непосредственно воздействуют на трейдинговую систему и используются для обмена сообщениями, анализа полученных от системы откликов, нагрузочного тестирования.

Пассивные инструменты тестирования. Пассивные инструменты тестирования используются для автоматизированного сбора логов, структурирования данных, мониторинга и анализа поведения системы, сертификации пользователей. Инструменты тестирования позволяют оперативно исследовать большой объем данных, реагировать на отклонения в поведении системы от установленных требований, локализировать неполадки.

Инструмент тестирования позволяет эффективно решать эти задачи, если выполнены следующие требования:

• обеспечена полнота сбора данных обо всех событиях в системе;

• минимизировано влияние на систему;

• обеспечено оповещение в реальном режиме времени о нестандартных ситуациях;

• реализованы гибко настраиваемые критерии распознавания образов;

• обеспечено хранение и предоставление доступа к историческим данным;

• предоставлена возможность отслеживания последовательности событий и внутренних состояний системы на определенный момент времени.

Нахождение первопричины сбоя, произошедшего при использовании HiVAT-методов, нередко является более сложным процессом по сравнению с обычными методиками ручного и автоматизированного тестирования. Эта сложность обусловлена, в основном,

двумя факторами: автоматическим созданием сценариев тестирования и большим объемом разнородных данных, пропущенных через систему. Тестировщику необходимы удобные инструменты, информирующие его о возникновении проблем и позволяющие детально исследовать состояние системы до и после возникновения сбоев.

Недетерминированный характер входного потока сообщений, являющийся отличительной чертой использования HiVAT-методов, подразумевает необходимость в гибкости сценариев распознавания проблем и предоставление тестировщику возможностей по их настройке.

Невозможно обеспечить полноту сбора информации без присутствия эффекта измерения для высоконагруженной системы. Задача инструмента тестирования состоит в том, чтобы минимизировать влияние самого инструмента на функциональность и производительность тестируемой системы.

Инструмент должен предоставлять доступ к детальным и агрегированным данным, полученным в результате выполнения текущей и предыдущих итераций тестирования. Диаграмма на рис. 1 показывает компоненты, необходимые для реализации перечисленных требований к пассивным инструментам тестирования.

Активные инструменты тестирования. Инструмент для активного тестирования должен обладать универсальностью, т. е. способностью подключаться к различным тестируемым окружениям, используя разнообразные протоколы. Применение фаззинга для распределенных трейдинговых систем накладывает ограничения на создаваемые сообщения [25] с целью обеспечить их прохождение к ядру и другим внутренним компонентам без блокирования их на внешних шлюзах протокольных подключений или на начальных ступенях защиты модулей контроля рисков.

Автоматический характер создания сценариев тестирования и их значительный объем требует сохранения информации об отправленных сообщениях и внутренних состояниях инструмента тестирования, его настройках. При использовании сложных техник необходима также сверка данных между инструментом и тестируемой системой.

Использование HiVAT-методов для высоконагруженных трейдинговых систем требует создания масштабной инфраструктуры для тестирования. Эффективность использования такой инфраструктуры может быть обеспечена только в случае одновременной

Рис. 1. Высокоуровневая схема основных компонентов инструмента для пассивного тестирования трейдинговых систем

Рис. 2. Высокоуровневая схема основных компонентов инструмента для активного тестирования трейдинговых систем

доступности ее для выполнения различных задач тестирования. Инструменты для тестирования должны позволять выполнять вручную и ставить в расписание прогонов наборы автоматизированных сценариев даже в случаях, когда параллельно многократно выполняются сценарии библиотеки регрессионного тестирования или тесты, основанные на техниках случайной генерации тестов.

Для создания полноценных функциональных тестов инструмент должен предоставить удобные программируемые возможности для управления процессом генерации автоматизированных сценариев тестирования. Диаграмма на рис. 2 показывает компоненты, необходимые для реализации перечисленных требований к активным инструментам тестирования.

Общие требования к инструментам для тестирования трейдинговых систем. Следующие характеристики являются обязательными как для пассивных, так и для активных инструментов при использовании HiVAT-методов:

- масштабируемость и высокая пропускная способность;
- устойчивость;
- адаптивность;
- удобство и интерактивность.

Эти обязательные характеристики совпадают с требованиями, предъявляемыми к промышленным трейдинговым системам. Более детально требования к системам мониторинга и контроля финансовых рисков, а также к системам выполнения биржевых и алгоритмических заявок описаны в табл. 1 и 2.

Использование инструментов тестирования в промышленной эксплуатации трейдинговых систем

В этой части статьи мы сопоставим требования, предъявляемые к инструментам для тестирования трейдинговых систем с использованием методов HiVAT, с требованиями к трейдинговым системам и системам мониторинга, используемым в промышленной эксплуатации.

Из схемы на рис. 3 видна концептуальная близость системы к инструментам для пассивного тестирования трейдинговых платформ.

Схема на рис. 4 показывает концептуальную близость системы к инструментам для активного тестирования трейдинговых платформ.

Направление дальнейших исследований

Сравнение требований и концептуальных схем инструментов для тестирования с соответствующими промышленными систе-

Рис. 3. Высокоуровневая схема основных компонентов системы мониторинга и контроля финансовых рисков, создаваемой в компании Exactpro Systems, LLC

мами позволяет утверждать, что, начиная с определенного уровня зрелости инструменты для тестирования могут использоваться как подсистема трейдинговой платформы. Но основной задачей тестирования является не нахождение правильно работающих подсистем, а выявление проблем и недостатков в исследуемом приложении. Превращение инструментов для тестирования в промышленные трейдинговые системы может потенциально привести к концентрации взаимодействия на корректно работающих участках, снижению покрытия тестами и избыточному фокусу на позитивных сценариях. Основное направление дальнейшей работы – это нахождение ме-

Рис. 4. Высокоуровневая схема основных компонентов системы выполнения биржевых и алгоритмических заявок, создаваемой в компании Exactpro Systems, LLC

Таблица 1

Требования к системам мониторинга и контроля финансовых рисков

Требование	Использование в промышленной эксплуатации
Полнота сбора данных обо всех событиях в системе	Основная задача системы мониторинга и контроля финансовых рынков – поддерживать аналитическую работу отделов, отвечающих за выявление возможных манипуляций [19]. Система мониторинга должна собирать информацию обо всех входящих заявках, ответах системы, данных, поступающих из внешних источников, а также о релевантных внутренних состояниях трейдинговой платформы
Минимизация влияния на трейдинговую систему	Сбор большого объема информации невозможен без масштабируемой мониторинговой инфраструктуры. Наблюдение за рынком исключительно важный процесс, являющийся обязательным в абсолютном большинстве финансовых юрисдикций. Тем не менее для обеспечения минимальных времен отклика на приходящие в реальном режиме времени сообщения операторы трейдинговых систем стараются избегать негативного влияния эффекта измерения на основную функциональность трейдинговой платформы
Оповещение в реальном режиме времени о нестандартных ситуациях	Операторы системы должны быть немедленно уведомлены при возникновении технических проблем или подозрительных транзакций. Такие уведомления называются *оповещениями системы мониторинга* (surveillance alerts). Эффективно работающая система оповещения – ключевой компонент системы мониторинга и контроля рисков
Гибко настраиваемые критерии распознавания образов	Очень часто недобросовестные участники рынка пытаются скрыть злоупотребления и манипуляции рынком под видом легитимных финансовых транзакций. Системам мониторинга и контроля рынков приходится делать аналитические заключения на основе нечеткой логики, параметры которой приходится непрерывно адаптировать под новые трейдинговые ситуации и схемы поведения участников торгов [20]
Хранение и предоставление доступа к историческим данным	По запросам аудита или регулирующих органов операторы трейдинговых систем обязаны предоставлять исторические данные и результаты их обработки
Возможность отследить последовательность событий и внутренние состояния системы на определенный момент времени	Формальные критерии сами по себе не являются достаточным доказательством наличия манипуляции. Операторы системы нуждаются в возможности восстанавливать последовательность событий, относящихся к конкретным эпизодам, вызвавшим оповещение о проблеме. Данная функция называется *проигрыванием книги заявок* (order book replay). Удобная реализация позволяет операторам исследовать большее количество событий и делать выводы о правильности работы механизмов распознавания образов

тодов преодоления данной тенденции.

Если инструменты для тестирования становятся частью промышленной инфраструктуры, то внесение небольших изменений в их код и настройки является по сути изменением содержащей их трейдинговой платформы. Таким образом, предложенный подход открывает новые возможности по примене-

Таблица 2

Требования к системам выполнения биржевых и алгоритмических заявок

Требование	Использование в промышленной эксплуатации
Универсальность. Подключение ко множеству других систем и поддержка используемых ими протоколов	В условиях фрагментации финансовых рынков брокерские системы должны обеспечивать возможность подключения к разным биржевым системам и сторонним брокерам [21]
Предоставление одновременного доступа	Высоконагруженная трейдинговая система должна обеспечить одновременный доступ большому количеству участников торгов
Возможность выполнения сложных сценариев	Основная функция систем алгоритмической торговли – предоставление пользователям возможности задавать стратегию посылки заявок на выполнение финансовых транзакций в зависимости от состояния рынка, портфеля, параметров риска, информационных сообщений и т. д.
Хранение информации обо всех отправленных сообщениях и внутренних состояниях. Возможность сверки этих данных с данными, поступающими из внешних систем, включая клиринг и депозитарии	Оператор трейдинговой системы, предоставляющий доступ своим клиентам на финансовые рынки, обязан хранить информацию обо всех совершенных финансовых транзакциях [22] и производить сверку этих данных с данными клиентов и контрагентов, а также с информацией из постторговых систем

нию методов мутационного тестирования на системном уровне к исследованию сложных интегрированных трейдинговых систем [23]. Внесенные изменения называются *мутациями* и основываются на мутационных операторах [24], имитирующих типичные ошибки или нежелательные воздействия. Мутации также позволяют оценить эффективность существующего набора тестов и качество инструментов для тестирования.

Операторами мутации трейдинговой системы для проверки нефункциональных свойств могут стать следующие изменения:

- введение случайных задержек во внутренние компоненты, логические алгоритмы и в функционирование внешних подключений;
- замена оптимизированных TCP/IP потоков данных на множество параметризованных библиотек на различных языках;
- заполнение памяти или дискового пространства на компьютере с исследуемой системой большим количеством логов;
- загрузка сети паразитными сообщениями или внедрение ошибок в структуры данных.

Второе направление дальнейших исследований – это изучение методов фаззинга, совместимых со структурой и консистентностью промышленных систем [25].

На основе обобщения опыта по верификации корректной работы высоконагруженных электронных трейдинговых систем с функциональной и нефункциональной точек зрения в статье проанализированы методы обеспечения их качества, основанные на большом объеме автоматизированного тестирования (HiVAT).

Практическое использование этих методов нами позволило определить набор основных требований к инструментам пассивного и активного тестирования, применяемым для верификации подобного рода систем. В статье сделан вывод о том, что инструмент для тестирования, соответствующий определенному набору требований, является по своей сути системой, применимой при промышленной эксплуатации трейдинговых систем.

Полученные нами результаты обосновали оправданность финансирования соз-

дания инструментов для тестирования, основанных на описанных выше методах и принципах. В рамках проектов компании Exactpro Systems, LLC разрабатываются система мониторинга и контроля финансовых рынков и система поддержки алгоритмической торговли. Обе системы имеют двойное назначение и могут использоваться и как инструмент для тестирования, и как самостоятельный элемент промышленной трейдинговой инфраструктуры [26].

Выше были рассмотрены также дополнительные возможности по использованию методов мутационного тестирования на системном уровне для анализа и расширения полноты покрытия функциональными и нефункциональными тестами. Указанные возможности открываются при включении инструментов тестирования в состав трейдинговой платформы.

СПИСОК ЛИТЕРАТУРЫ

1. The Future of Computer Trading in Financial Markets. Final Project Report. Foresight. The Government Office for Science, London [электронный ресурс] / URL: http://www.bis.gov.uk/assets/foresight/docs/computer-trading/12-1086-future-of-computer-trading-in-financial-markets-report.pdf

2. **Bloomfield R., Wetherilt A.** Computer trading and systemic risk: a nuclear perspective. Foresight. The Government Office for Science, London [электронный ресурс] / URL: http://www.bis.gov.uk/assets/foresight/docs/computer-trading/12-1059-dr26-computer-trading-and-systemic-risk-nuclear-perspective.pdf (дата обращения 02.10.2012)

3. Commission Roundtable on Technology and Trading: Promoting Stability in Today's Markets. U.S. Securities and Exchange, 2012 [электронный ресурс] / URL: http://www.sec.gov/news/otherwebcasts/2012/ttr100212-transcript.pdf

4. **Иткин И.Л.** Высоконагруженные трейдинговые системы и их тестирование. Конференция разработчиков высоконагруженных систем HighLoad++ [электронный ресурс] / URL: http://www.highload.ru/2012/abstracts/388.html

5. **Penhaligan P.** Equity Trading: Performance, Latency & Throughput. ExTENT Conference [электронный ресурс] / URL: http://www.slideshare.net/extentconf/extent3-turquoise-equitytrading2012

6. **Avellaneda M.** Algorithmic and High-frequency trading: an overview. New York University & Finance Concepts LLC Quant Congress USA 2011 [электронный ресурс] / URL: http://www.math.nyu.edu/faculty/avellane/QuantCongressUSA2011AlgoTradingLAST.pdf

7. Millennium Exchange Technical [электронный ресурс] / URL: http://www.londonstockexchange.com/products-and-services/technical-library/millennium-exchange-technical-specifications/millennium-exchange-technical-specifications.htm

8. **Zverev A., Bobrov I., Pryadkina N.** Testing of HFT GUI. ExTENT Conference [электронный ресурс] / URL: http://www.slideshare.net/extentconf/extent3-exactpro-testingofhftgui-12944204

9. **Yu T.** An observable and controllable testing framework for modern systems // Proc. of the 2013 Internat. Conf. on Software Engineering. IEEE Press, 2013.

10. **Netzer R.H.B., Miller B.P.** What are race conditions?: Some issues and formalizations // Letters on Programming Languages and Systems. 1992. Vol. 1. Iss. 1.

11. **Zverev A., Bulda A., Bobrov I.** Trading Systems: Testing at the Confluence of FT&NFT. ExTENT Conference [электронный ресурс] / URL: http://www.slideshare.net/extentconf/extent-2013-obninsk-trading-systems-testing-at-the-confluence-of-ft-nft-17082512

12. **Heider W., Rabiser R., Grьnbacher P., Lettner D.** Using regression testing to analyze the impact of changes to variability models on products // Proc. of the 16th Internat. Software Product Line Conf. 2012. Vol. 1.

13. Eurodollars on CME Globex: Implied Price Functionality [электронный ресурс] / URL: http://www.cmegroup.com/trading/interest-rates/files/Butterflies.pdf

14. **Sutton M., Greene A., Amini P.** Fuzzing: Brute Force Vulnerability Discovery. Addison-Wesley Professional, 2007.

15. **McGee P., Kaner C.** Experiments with high volume test automation // SIGSOFT Software Engineering Notes. 2004.

16. **Kaner C.** An Overview of High Volume Automated Testing [электронный ресурс] / URL: http://kaner.com/?p=278

17. Teaching High Volume Automated Testing (HiVAT). 12th Workshop on Teaching Software Testing, 2013, Melbourne, Florida [электронный ресурс] / URL: http://wtst.org/

18. **Cavalli R., Montes de Oca E., Mallouli W., Lallali M.** Two Complementary Tools for the Formal Testing of Distributed Systems with Time Constraints // 12th 2008 IEEE/ACM Internat.

Symp. on Distributed Simulation and Real-Time Applications.

19. **Diaz D., Zaki M., Theodoulidis B., Sampaio P.** A Systematic Framework for the Analysis and Development of Financial Market Monitoring Systems // Annual SRII Global Conf. 2011.

20. **Иткин И., Прядкина Н., Крюков А.** Анализ данных в высоконагруженных трейдинговых системах [электронный ресурс] // Конференция АИСТ-2013 / URL: http://clubqa.ru/blogs/?p=436

21. Fidessa Fragmentation Index [электронный ресурс] / URL: http://fragmentation.fidessa.com/

22. OATS Reporting Technical Specifications [электронный ресурс] / URL: http://www.finra.org/web/groups/industry/@ip/@comp/@regis/documents/appsupportdocs/p197473.pdf

23. **Mateo P.R., Usaola M.P., Alemarn J.L.F.**Validating Second-Order Mutation at System Level // IEEE Transactions on Softvare Engineering. 2013. Vol. 39. No. 4.

24. **Mateo P.R., Usaola M.P., Offutt J.** Mutation at System and Functional Levels // 3rd Internat. Conf. on Software Testing, Verification, and Validation Workshops.

25. **Wang T., Wei T., Gu G., Zou W.** Checksum-Aware Fuzzing Combined with Dynamic Taint Analysis and Symbolic Execution // ACM Transactions on Information and System Security. 2011. Vol. 14. No. 2. Article 15.

26. **Itkin I., Matveeva A., Barch A.** Test Tools Evolution. ExTENT Conference [электронный ресурс] / URL: http://www.slideshare.net/extentconf/extent-2013-obninsk-test-tools-for-trading-systems-evolution-theory-17007184

REFERENCES

1. *The Future of Computer Trading in Financial Markets, Final Project Report*, Foresight, The Government Office for Science, London. Available: http://www.bis.gov.uk/assets/foresight/docs/computer-trading/12-1086-future-of-computer-trading-in-financial-markets-report.pdf

2. **Bloomfield R., Wetherilt A.** *Computer trading and systemic risk: a nuclear perspective, Foresight*, The Government Office for Science, London. Available: http://www.bis.gov.uk/assets/foresight/docs/computer-trading/12-1059-dr26-computer-trading-and-systemic-risk-nuclear-perspective.pdf

3. *Commission Roundtable on Technology and Trading: Promoting Stability in Today's Markets.* U.S. Securities and Exchange, 2012. Available: http://www.sec.gov/news/otherwebcasts/2012/ttr100212-transcript.pdf

4. **Itkin I.L.** Vysokonagruzhennyye treydingovyye sistemy i ikh testirovaniye. *Konferentsiya razrabotchikov vysokonagruzhennykh sistem HighLoad++*. Available: http://www.highload.ru/2012/abstracts/388.html (rus)

5. **Penhaligan P.** Equity Trading: Performance, Latency & Throughput, *ExTENT Conference*. Available: http://www.slideshare.net/extentconf/extent3-turquoise-equitytrading2012

6. **Avellaneda M.** Algorithmic and High-frequency trading: an overview, *New York University & Finance Concepts LLC Quant Congress USA 2011*. Available: http://www.math.nyu.edu/faculty/avellane/QuantCongressUSA2011AlgoTradingLAST.pdf

7. *Millennium Exchange Technical Specifications*. Available: http://www.londonstockexchange.com/products-and-services/technical-library/millennium-exchange-technical-specifications/millennium-exchange-technical-specifications.htm

8. **Zverev A., Bobrov I., Pryadkina N.** Testing of HFT GUI, *ExTENT Conference*. Available: http://www.slideshare.net/extentconf/extent3-exactpro-testingofhftgui-12944204

9. **Yu T.** An observable and controllable testing framework for modern systems, *Proceedings of the 2013 International Conference on Software Engineering Publisher*, IEEE Press, 2013.

10. **Netzer R.H.B., Miller B.P.** What are race conditions? Some issues and formalizations, *Letters on Programming Languages and Systems*, 1992, Vol. 1, Iss. 1.

11. **Zverev A., Bulda A., Bobrov I.** Trading Systems: Testing at the Confluence of FT&NFT, *ExTENT Conference*. Available: http://www.slideshare.net/extentconf/extent-2013-obninsk-trading-systems-testing-at-the-confluence-of-ft-nft-17082512

12. **Heider W., Rabiser R., Grьnbacher P., Lettner D.** Using regression testing to analyze the impact of changes to variability models on products, *Proceedings of the 16th International Software Product Line Conference*, 2012, Vol. 1.

13. *Eurodollars on CME Globex: Implied Price Functionality*. Available: http://www.cmegroup.com/trading/interest-rates/files/Butterflies.pdf

14. **Sutton M., Greene A., Amini P.** *Fuzzing: Brute Force Vulnerability Discovery*. Addison-Wesley Professional, 2007.

15. **McGee P., Kaner C.** Experiments with high volume test automation, *SIGSOFT Software Engineering Notes*, 2004.

16. **Kaner C.** *An Overview of High Volume Automated Testing*. Available: http://kaner.

com/?p=278

17. Teaching High Volume Automated Testing, *12th Workshop on Teaching Software Testing*, 2013, Melbourne, Florida. Available: http://wtst.org/

18. **Cavalli R., Montes de Oca E., Mallouli W., Lallali M.** Two Complementary Tools for the Formal Testing of Distributed Systems with Time Constraints, *12th 2008 IEEE/ACM International Symposium on Distributed Simulation and Real-Time Applications.*

19. **Diaz D., Zaki M., Theodoulidis B., Sampaio P.** A Systematic Framework for the Analysis and Development of Financial Market Monitoring Systems, *Annual SRII Global Conference*, 2011.

20. **Itkin I., Pryadkina N., Kryukov A.** Analiz dannykh v vysokonagruzhennykh treydingovykh sistemakh, *Konferentsiya AIST-2013.* Available: http://clubqa.ru/blogs/?p=436 (rus)

21. *Fidessa Fragmentation Index.* Available: http://fragmentation.fidessa.com/

22. *OATS Reporting Technical Specifications.* Available: http://www.finra.org/web/groups/industry/@ip/@comp/@regis/documents/appsupportdocs/p197473.pdf

23. **Mateo P.R., Usaola M.P., Alemarn J.L.F.**Validating Second-Order Mutation at System Level, *IEEE Transactions on softvare engineering*, 2013, Vol. 39, No. 4.

24. **Mateo P.R., Usaola M.P., Offutt J.** Mutation at System and Functional Levels, *3rd International Conference on Software Testing, Verification, and Validation Workshops.*

25. **Wang T., Wei T., Gu G., Zou W.** Checksum-Aware Fuzzing Combined with Dynamic Taint Analysis and Symbolic Execution, ACM Transactions on Information and System Security, 2011, Vol. 14, No. 2, Article 15.

26. **Itkin I., Matveeva A., Barch A.** Test Tools evolution, *ExTENT Conference.* Available: http://www.slideshare.net/extentconf/extent-2013-obninsk-test-tools-for-trading-systems-evolution-theory-17007184

АВЕРИНА Анастасия Андреевна — *аналитик ООО «ИТС-Эксперт».*
115088, Россия, Москва, 2-й Южнопортовый проезд, д. 20А/4.
E-mail: Anastasia.Matveeva@exactprosystems.com

AVERINA, Anastasiya A. *ITS-Expert, LLC.*
115088, 2nd Yuzhnoportovy proyezd, 20A/4, Moscow, Russia.
E-mail: Anastasia.Matveeva@exactprosystems.com

АНТОНОВ Николай Андреевич — *аналитик Костромского государственного технологического университета.*
156005, Россия, г. Кострома, ул. Дзержинского, д. 17.
E-mail: nikolay.antonov@exactprosystems.com

ANTONOV, Nikolay A. *Kostroma State Technological University.*
156005, Dzerzhinsky Str. 17, Kostroma, Russia.
E-mail: nikolay.antonov@exactprosystems.com

ИТКИН Иосиф Леонидович — *генеральный директор Exactpro Systems, LLC.*
4040 Civic Center Drive, Suite 200 San Rafael, CA 94903.
E-mail: sasha.khodchenko@exactprosystems.com

ITKIN, Iosif I. *Exactpro Systems, LLC.*
4040 Civic Center Drive, Suite 200 San Rafael, CA 94903.
E-mail: sasha.khodchenko@exactprosystems.com

УДК 004.052.2

А.Ю. Булда, О.А. Буянова, А.В. Зверев

ПРИМЕНЕНИЕ СИМУЛЯТОРОВ РЫНКА ЦЕННЫХ БУМАГ ДЛЯ ТЕСТИРОВАНИЯ СИСТЕМ АГРЕГАЦИИ И РАСПРЕДЕЛЕНИЯ ИНФОРМАЦИИ О КОТИРОВКАХ (TICKER PLANT)

A.Yu. Bulda, O.A. Buyanova, A.V. Zverev

USING OF EXCHANGE SIMULATORS AND TEST EXCHANGES AS TOOLS TO TEST TICKER PLANT SYSTEMS

Системы агрегации и распределения информации о котировках широко применяются в современном трейдинге. Они позволяют собирать в реальном времени котировки с нескольких рынков, предоставлять данные о них в едином формате и распространять их в электронном виде в зависимости от запросов и целей внешних клиентов, трейдеров. Рассмотрена возможность применения симуляторов рынка к тестированию таких систем. Проведен анализ основных тестовых сценариев для систем распределения информации о котировках. Выделен набор основных функциональных и нефункциональных сценариев тестирования, необходимых для контроля качества распределения информации о котировках. Произведена сравнительная оценка симуляторов рынка и реальных тестовых площадок.

СИСТЕМА АГРЕГАЦИИ И РАСПРЕДЕЛЕНИЯ ИНФОРМАЦИИ О КОТИРОВКАХ (TICKER PLANT, MARKET DATA); БИРЖА; ТЕСТОВАЯ ПЛАТФОРМА; СИМУЛЯТОР РЫНКА; ФУНКЦИОНАЛЬНОЕ ТЕСТИРОВАНИЕ; НЕФУНКЦИОНАЛЬНОЕ ТЕСТИРОВАНИЕ.

Ticker Plant systems are widely used in modern day trading. They allow collecting in real-time quotes from several markets, present the data in a unified format, and disseminate it electronically depending on requests and goals of external clients, traders. This paper presents a view on the possibility of using market simulators for testing such systems. A set of main functional and non-functional test scenarios required to control the quality of quote dissemination has been identified. A comparison of market simulators and real test markets has been presented.

TICKER PLANT; MARKET DATA; EXCHANGE; TEST MARKET; MARKET SIMULATOR; FUNCTIONAL TESTING; NON-FUNCTIONAL TESTING.

Современный электронный трейдинг невозможно представить себе без актуальных сведений о финансовых инструментах, заявках и сделках на них. Высоконагруженные биржевые и брокерские системы предоставляют рыночные данные о торгуемых на них финансовых инструментах через собственные специальные компоненты, называющиеся *каналами распространения рыночных данных* (Market Data Feeds). Каждый финансовый инструмент – это значительное количество информации, генерирующейся ежесекундно, поэтому способность распространять весь поток рыночных данных и скорость распространения информации о котировках и финансовых сделках для каждого финансового инструмента являются основными характеристиками для такого рода компонентов высоконагруженных систем.

Рыночная информация (Market Data) и ее основные представления. Обычно рыночная информация включает в себя следующий набор параметров, специфичных для определенного финансового инструмента: краткое название инструмента (ticker symbol), цена последней сделки (last trade price), текущие лучшие цены спроса и предложения (Best Bid & Offer), идентификационный номер инструмента (ISIN), биржа, на которой произошла сделка (exchange code), время последней сделки (Trade Time), цена

сделки при закрытии торговой сессии на инструменте (Close Price). В зависимости от сложности высоконагруженных биржевых систем, рыночная информация о котировках может обрабатываться внутренними компонентами электронных бирж и обогащаться дополнительной информацией: например, об общем объеме сделок в течение биржевого дня (Turnover), о средневзвешенной цене по общему объему и количеству сделок (VWAP), а также более детальной информацией об акции или деривативе – справочные данные, такие как параметры трейдеров, рынка, торговых сессий, инструментов (или Reference Data). Справочные данные (Reference Data или Static Data) представляют собой информацию об инструменте, которая не меняется в режиме реального времени: например, международный идентификационный код ценной бумаги (ISIN), цена сделки при закрытии трейдинговой сессии за предыдущий день (Close Price), валюта, в которой данный финансовый инструмент торгуется на бирже (Currency), параметры т. н. «прерывателей торгов», которые чаще указываются в процентных соотношениях к цене последней сделки (Dynamic or Static Circuit Breaker Tolerances, %), и т. д. [1]

Для предоставления перечисленной выше информации существует ряд стандартных протоколов распространения котировок (например, FIX/FAST [2]), т. н. протоколов распространенния котировок с фиксированной длиной сообщений (например, ITCH [3]), или кодированные протоколы передачи данных (например, HTTPS (HyperText Transfer Protocol Secure) [4, 5]).

Многие электронные биржи распространяют информацию о котировках как через стандартные протоколы, так и через специальные. Трейдеры часто не могут позволить себе дорогостоящую разработку программ, которые собирали бы необходимую им для работы информацию о котировках. Таким образом, возникает необходимость создания систем сбора и агрегации рыночных данных с различных бирж и по различным финансовым протоколам передачи данных.

Система агрегации информации (Ticker Plant) и ее основные функции. *Ticker Plant* – это система агрегации информации о котировках с различных электронных торговых площадок (или бирж) и ее распределения. Данная система предоставляет рыночные данные трейдерам в нормализованном или унифицированном виде [6]. Нередко Ticker Plant рассчитывает дополнительные параметры на основе предоставленных рыночных данных, т. е. обогащает статистическую информацию об акциях и деривативах. Кроме того, для таких систем характерна способность объединять однородные величины распространяемых котировок: ценовые уровни (Price Levels), предоставление котировок согласно запросам клиентов (например, широко распространены такие сервисы, как Level 1, Level 2, Index, T&S, News) в режиме реального времени, а также хранение переданной рыночной информации.

Указанные выше типы сервисов по предоставлению рыночной информации распространены в электронной торговле финансовыми инструментами. Рассмотрим каждый из них подробнее:

- Level1 – информация о последних Bid/Ask, OHLC (Open, High, Low, Close), Volume;
- Level2 – информация о Level1 + Order Book (5–10 уровней глубины торговой книжки, Bid/Ask Px&Qty). Level2 может содержать атрибутированные или анонимные заявки) и использовать модели MBO (Market by Order, т. е. индивидуальное отображение всех заявок в пределах отдельно взятого ценового уровня) и MBP (Market by Price, т. е. агрегированное отображение заявок в пределах отдельно взятого ценового уровня) [7];
- Level2 – Total View – более полная информация по сравнению с Level2;
- News – последние новости о компании [8];
- Index – данные об индексах [9].

Схематичное представление системы агрегации и распределения информации о котировках показано на рис. 1.

Для того чтобы оценить сложность решаемых задач, определим параметры режи-

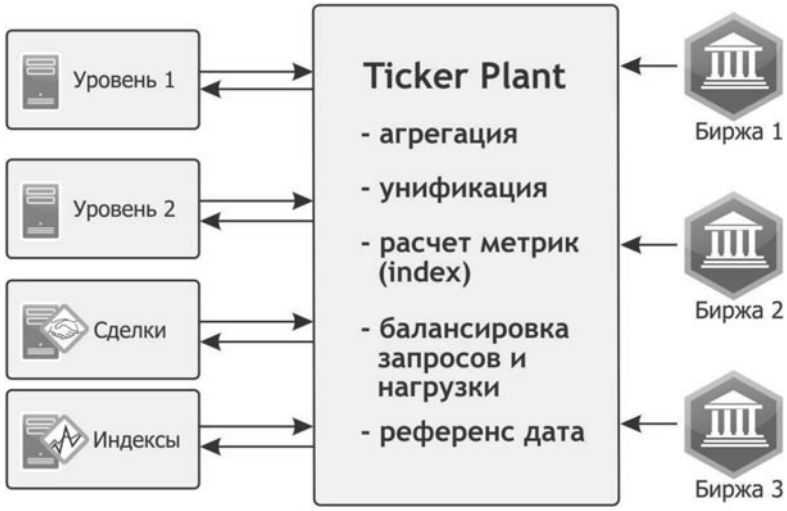


Рис. 1. Схема системы Ticker Plant

ма реального времени. Наиболее значимыми являются:

критический срок обслуживания или предельный срок завершения какой-либо работы;

время отклика (время задержки) системы на внешние события;

разброс значений времени отклика.

События, происходящие в системе реального времени, могут относиться к одной из трех категорий:

1) непредсказуемые события;

2) предсказуемые события, случающиеся с определенной регулярностью;

3) регулярные события (разновидность предсказуемых событий), случающиеся в течение интервала времени.

Основные требования к системе

Исходя из описания системы агрегации и распределения информации о котировках и основных ее характеристик, выделенных выше, мы определяем набор требований, которым такая система должна соответствовать. При тестировании данной системы мы будем обращаться к этому набору характеристик. Для того чтобы было удобно оценивать каждую характеристику, мы распределили их на функциональные и нефункциональные.

Итак, система должна решать следующие задачи.

С функциональной точки зрения:

1) собирать рыночную информацию о котировках из нескольких источников (поставщиков рыночной информации – бирж, банков);

2) обрабатывать справочные данные (reference data), предоставляемые биржами;

3) обрабатывать информацию о котировках, предоставляемую по различным протоколам передачи данных, в режиме реального времени;

4) преобразовывать полученную информацию в один формат;

5) агрегировать данные о котировках согласно различным методам, описанным выше;

6) обрабатывать эти данные с целью расширения функциональности системы: например, предоставлять статистику (VWAP, Turnover, Trade High/Low, 52 week Trade High/Low);

7) предоставлять данные согласно запросам клиентов: Level1, Level2, T&S, News, Index, Option chains и др.;

8) предоставлять записанную историческую информацию о котировках.

С нефункциональной точки зрения:

1) быструю обработку потоков информации о котировках, поступающей в режиме реального времени от электронных бирж;

2) быструю обработку запросов, по-

ступающих от клиентов, и предоставление данных о котировках в зависимости от вида запроса клиента (например, отдельно на торгуемый инструмент или на группу инструментов, или на весь рынок);

3) бесперебойную работоспособность системы;

4) управляемость системой (operability);

5) возможность мониторинга систем (наличие приложений, с помощью которых можно наблюдать за системой или управлять ее компонентами);

6) пропускную способность (throughput);

7) временную задержку (latency);

8) отказоустойчивость (fault tolerance).

Определив набор функциональных и нефункциональных характеристик системы агрегации и распределения информации о котировках, нам необходимо понять, как будет проходить процесс тестирования отдельно взятой характеристики. Очевидными становятся два пути тестирования такого рода систем.

Первый – это тестирование с реальными тестовыми площадками (Customer Development Service – CDS [10]). Обычно биржи предоставляют своим клиентам тестовые окружения (test environments), аналогичные реальной электронной трейдинговой платформе. Чаще всего это делается с целью прохождения сертификационного процесса или предоставления клиентам возможности для отладки своего клиентского программного обеспечения.

И второй – это тестирование с помощью симуляторов трейдинговых платформ, разработанных на основе данных о бирже, находящихся в публичном доступе.

«Тестовая» биржа: описание и основные возможности

«Тестовая» биржа – это своего рода полная копия реальной трейдинговой биржи [11]. Такой аналог трейдинговой биржи должен наиболее полно представлять реальную электронную платформу. Тестовые биржи предоставляются клиентам, которым необходимо отладить разработанное ими трейдинговое и информационное программное обеспечение [12]. Заявки и сделки

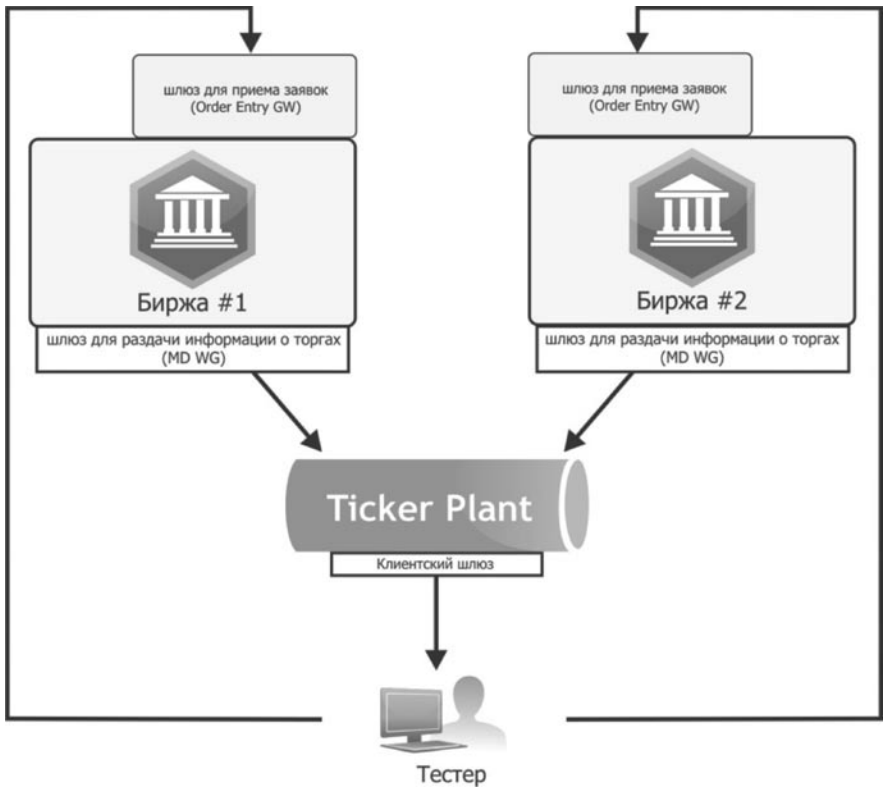


Рис. 2. Тестирование Ticker Plant с помощью тестовой биржи

в таких тестовых биржах генерируются как следствие взаимодействия самих клиентов, тестирующих собственное программное обеспечение, друг с другом.

В идеале тестовая биржа состоит из тех же компонентов, что и реальная биржевая система [13]. Торговые сессии, основные параметры торгуемых инструментов, правила трейдинга, графические приложения для управления настройками справочных данных (reference data) и т. д – все эти характеристики полностью аналогичны компонентам реальной системы. Поэтому такое тестовое окружение (test environment) позволяет проверять программное обеспечение, заведомо ориентируясь на аналогичные настройки в реальной системе.

На рис. 2 приведена схема тестирования Ticker Plant с применением тестовой биржи.

Симулятор рынка ценных бумаг: описание и основные возможности

Симулятор рынка ценных бумаг (или *биржевой симулятор*) – это интерактивная программа, разработанная для имитации возможностей и свойств реальных моделей рынка [14]. Симулятор рынка должен уметь эмулировать реальные действия, происходящие на бирже: сам трейдинг (размещение заявок, генерацию сделок, изменения статусов инструментов и т. д), симуляцию мониторинга за ходом торгов на рынке и внесение изменений (market operations: отмена/изменение заявок и сделок, остановка торгов и т. д.), режим работы рынка (переход рынка из одного статуса в другой – открытие рынка (Market Open), аукционы (Opening/Closing/ Periodic Auction Calls), вычисления цен аукционов и публикация цены закрытия, закрытие рынка и т. д. Биржевой симулятор рынка содержит API, аналогичный реальной бирже, с элементами контроля отсылаемых ответов клиенту [15]. Такие симуляторы рынков позволяют иметь более полный контроль над генерируемыми событиями для системы Ticker Plant [16], таким образом увеличивая покрытие тестирования системы. С помощью симуляторов можно создать необходимую нагрузку, сравнимую с потоками данных, свойственных реальным высоконагруженным биржевым и брокерским системам.

На рис. 3 представлена схема верификации системы Ticker Plant при помощи симулятора.

Составление библиотеки сценариев и тестов для верификации корректности работы системы агрегации и распределения информации о котировках (Ticker Plant)

Исходя из основных требований к системе Ticker Plant, мы разработали библио-

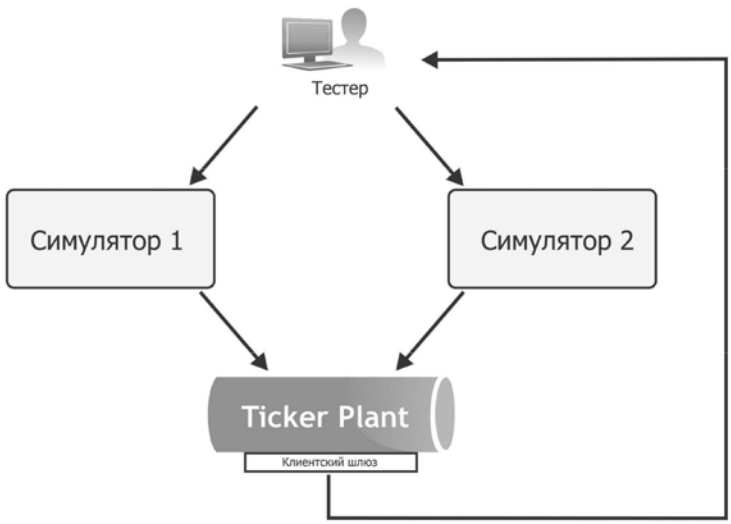


Рис. 3. Схема тестирования Ticker Plant с применением симулятора

теку сценариев тестирования, с помощью которых можно проверить работоспособность такой системы. В табл. 1 приведены области тестирования системы Ticker Plant и дано их краткое описание. Таблица также показывает оценочную шкалу областей тестирования по приоритетам. Например, если напротив той или иной области указана важность 1, это означает, что при возникновении ошибок в данной области тестируемая система Ticker Plant не сможет полноценно выполнять свою основную задачу. Важность 3 означает, что при наличии ошибок в этой области система может выполнять свою основную задачу, но с некоторыми отклонениями.

Анализ покрытия тестирования системы Ticker Plant при помощи инструментов для тестирования

Для того чтобы оценить и проанализировать, насколько полно инструметы тестирования покрывают области функциональности системы Ticker Plant, которую необходимо протестировать, нами был разработан специальный метод, основанный на переборе сценариев в процентном отношении к всевозможным сценариям, относящимся к той или иной функциональной области. Опираясь на опыт тестирования, полученный в компании Exactpro Systems [17], была определена важность областей системы Ticker Plant, обозначенная в правой колонке табл. 1.

В следующем разделе описывается опыт применения данного метода покрытия тестами для оценки полноты тестирования системы Ticker Plant как с помощью реальных тестовых площадок, так и с помощью биржевых симуляторов.

Сравнительная характеристика оценочных данных при тестировании системы Ticker Plant с помощью реальной тестовой биржи и симулятора. В табл. 2 приведены сравнительные характеристики, включающие в себя оценочные данные, полученные в ходе тестирования системы Ticker Plant с помощью реальной тестовой биржи и симулятора, а также детальное объяснение полученных данных.

Может сложиться ощущение, что полное покрытие тестирования системы Ticker Plant возможно исключительно с использованием симуляторов (как видно из табл. 2). Однако в реальности это не так. Основное препятствие состоит в том, что эмулировать торговую площадку со 100-процентной точностью невозможно. В особенности это касается непосредственного взаимодействия между системой Ticker Plant и различными рынками. Причина заключается в том, что любая спецификация о протоколе соединения с биржей содержит ограниченное описание поведения при различных ситуациях, поэтому даже если симулятор воспроизведет 100 % спецификации с точностью до нюансов, все равно останутся элементы, сделанные «на усмотрение» создателей симулятора.

Для наглядности приведем пример. Предположим, на бирже происходит сделка, и биржа рассылает клиентам два события: первое событие содержит объем и цену сделки; второе событие содержит изменение HIGH, LOW параметров цены на данном финансовом инструменте. Если какой-либо алгоритм при расчете индекса использует и первое, и второе события, то вполне вероятно, что для дальнейшей работы такого алгоритма будет важно, в каком порядке описанные выше события происходят. Маловероятно, что спецификация торговой площадки будет содержать требования к порядку сообщений и событий. Более того, раз данного требования нет в спецификации, биржа может легко поменять этот порядок. Таким образом, у нас есть пример сценария, который невозможно симулировать точно. В связи с этим существенный объем тестирования должен осуществляться на тестовой площадке, которая наиболее близко соответствует поведению реального рынка. Данное заключение также сделано, опираясь на опыт в тестировании систем Ticker Plant.

Более наглядно сравнительный анализ покрытия тестами при помощи различных инструментов тестирования показан на рис. 4.

Таблица 1

Области тестирования Ticker Plant

Область тестирования	Описание	Важность (1–3)
1. Техническое подключение к биржам – поток данных о котировках и сделках в режиме реального времени (streaming quotes – real time): UDP (User Datagram Protocol); TCP/IP (Transmission Control Protocol (TCP) и Internet Protocol (IP); HTTPS (HyperText Transfer Protocol Secure)	Возможности подключения к основным каналам (primary channel); подключение к запасным каналам (secondary channels); отказоустойчивость (fault tolerance); тестирование аварийного восстановления данных (disaster recovery)	1
2. Техническое подключение к биржам – каналы восстановления данных (Replay/Recovery channels)	Возможности подключения к основным каналам (primary channel); подключение к запасным каналам (secondary channels); отказоустойчивость (fault tolerance); тестирование аварийного восстановления данных (disaster recovery)	1
3. Тестирование интерфейса протокола от биржи к системе Ticker Plant (real time + replay/recovery)	Проверка административных сообщений, инициируемых Ticker Plant (Administrative messages: Login Request, Replay Request, Snapshot Request, Logout Request); проверка административных сообщений, инициируемых каналом (Administrative messages): Heartbeat, Login Response, Replay Response, Snapshot Response, Snapshot Complete); проверка сообщений приложения (Application messages: Time, System Event, Symbol Directory, Symbol Status, Add Order, Add Attributed Order, Order Deleted, Order Modified, Order Book Clear, Order Executed, Order Executed with Price/Size, Trade, Auction Trade, Off-Book Trade, Trade Break, Auction Info, Statistics); проверка обязательных полей (mandatory tags/values) и значений; проверка необязательных полей и значений (optional tags/values); проверка всевозможных вариантов и сочетаний значений тагов (Order Type, TIF и т. д.); проверка негативных значений (неподдерживаемые значения, отрицательные значения, специальные символы и т. д.)	1
4. Восстановление потери небольшого количества данных (Replay channel)	Возможности подключения к основному каналу (primary channel); подключение к запасным каналам (secondary channels); проверка последовательности полученных данных; проверка полученных данных на правильность; проверка объема буферизации данных	1
5. Восстановление потери большого количества данных (Recovery channel)	Возможности подключения к основному каналу (primary channel); подключение к запасным каналам (secondary channels); проверка последовательности полученных данных; проверка полученных данных на правильность; проверка объема переданных данных	1
6. Проверка справочных данных (Reference Data)	Возможность загрузить данные о финансовых инструментах, предоставляемых биржей; правильность обработки данных; использование справочных данных для подсчета различных показателей	1

Продолжение таблицы 1

Область тестирования	Описание	Важность (1–3)
7. Тестирование поведения системы Ticker Plant при отказе компонентов биржи (Failover при потоке данных с биржи)	Восстановление данных после отказа основного и/или запасного каналов; возможность переподключения; правильная последовательность сообщений; возможность дальнейшей обработки данных после восстановления	1
8. Тестирование при отказе компонентов системы Ticker Plant (Failover при потоке данных из системы Ticker Plant)	Проверка работоспособности компонентов системы Ticker Plant при отказе основного/запасного каналов; правильная последовательность сообщений; возможность дальнейшей обработки данных после восстановления	2
9. Тестирование полного трейдингового дня (цикла) работы системы Ticker Plant (Daily Life Cycle – DLC)	Проверка правильной последовательности старта компонентов системы Ticker Plant; проверка правильной последовательности выключения компонентов системы Ticker Plant	1
10. Тестирование полного трейдингового дня (цикла) на бирже (DLC)	Правильность последовательности приходящих сообщений в течение дня; добавление нового инструмента, изменение его параметров, удаление; изменение статусов инструментов; переход заявок из одной трейдинговой фазы в другую; переход заявок из одного дня в другой	1
11. Проверка всевозможных торговых статусов рынка	Проверка на присутствие данных видов статусов внутри системы Ticker Plant: например, Halt, Opening auction call, Pre-mandatory quote period, Continuous trading, End trade reporting, Closing auction call; проверка обрабатывания переходов из одного статуса инструментов в другой; проверка правильности обработки сообщений при массовой смене статусов на финансовых инструментах	1
12. Измерение ширины потребляемого канала передачи данных по сети (Bandwidth)	Проводятся нагрузочные тесты для измерения объема передаваемых данных в единицу времени	1
13. Измерение пропускной способности каналов в единицу времени (Throughput)	Проводятся нагрузочные тесты для подсчета среднего количества гарантированно доставленных сообщений через каналы передачи данных	1
14. Измерение задержек (латентности (Latency) системы)	Проверка компонентов системы на задержу передачи данных во времени	1
15. Проверка нагрузочной способности системы (Capacity) – нагрузка входным потоком данных	Проверка на работоспособность системы Ticker Plant при большом входном потоке данных	2
16. Проверка нагрузочной способности системы (Capacity) – нагрузка с клиентской стороны системы Ticker Plant	Проверка на обработку системой Ticker Plant большого количества запросов	2
17. Проверка системы Ticker Plant при неправильных запросах клиентов и ответная реакция на них для Replay/ Recovery каналов	Попытки подключения клиента (CompID) с недостаточным кличеством прав; попытки подключения с несуществующим пользователем; проверки максимального количества подключений; проверки максимального количества запросов	1

Окончание таблицы 1

Область тестирования	Описание	Важность (1–3)
18. Проверка правильности обработки системой Ticker Plant некорректных сообщений от биржи	Отклик на несуществующие сообщения (messages types); неправильное количество тегов; неправильный порядок тегов; неправильная контрольная сумма (checksum)	2
19. Проверка корректности работы системы в целом – от трейдингового клиента до системы Ticker Plant (E2E тестирование)	Добавление заявок; изменение заявок; удаление заявок; добавление агрессивных заявок (приводящих к сделке)	1
20. Сложные сценарии на функциональность самой биржи	Проверка на то, как система обрабатывает очередность событий (например, многоуровневые сделки с айсберг-заявками); вычисления цены аукциона; поведение Stop /Stop Limit ордеров в зависимости от фазы трейдингового дня; проверка поведения GTC ордеров в течение нескольких дней	2
21. Реконсиляционное тестирование	Проверка при сравнении потоков, идущих к системе и от системы (детальная проверка на то, что каждое событие, поступившее на вход системы, обработалось и отобразилось на выходном потоке)	1
22. Тестирование MBO (market by order) и MBP (market by price) сервисов	Функциональная проверка сервисов, предоставляемых системой Ticker Plant	1
23. Проверки расчета индексов	Проверка правильности расчетов индексов в зависимости от потока сделок	2
24. Правильность расчета статистических данных	Проверка расчета VWAP, Turnover, Trade High Low, 52 week Trade High Low и т. д.	2
25. Работа с историческими данными	Сбор исторических данных; сортировка в зависимости от типа данных; выдача результатов согласно запросам клиентов; возможность проигрывать записанные исторические данные	2
26. Проверка получения и передачи новостных каналов от бирж	Тестирование новостей (News); тестирование оповещений клиентов (Announcements)	3
27. Проверка действия биржевого оператора	Отмена сделок; изменение параметров сделки	2
28. Проверка сложных запросов клиента	Выдача данных для деривативного инструмента (опционные цепочки – option chains) на запрашиваемый базовый инструмент	2
29. Мониторинг системы	Проверка приложений, с помощью которых можно наблюдать за системой или управлять ее компонентами	1

Таблица 2

Сравнительные характеристики данных для тестовой биржи и симулятора

Область тестирования	Покрытие функциональности при тестировании с помощью реальной тестовой биржи, %	Покрытие функциональности при тестировании с помощью симулятора, %	Детализация оценочных данных
1. Техническое подключение к биржам – поток данных о котировках и торгах в режиме реального времени (streaming quotes – real time) UDP (User Datagram Protocol); TCP/IP (Transmission Control Protocol и Internet Protocol; HTTPS (Hypertext Transfer Protocol Secure)	100	40	При данной оценке мы исходим из того, что симулятор биржи по своему определению не способен полноценно воспроизвести все технические детали соединения с биржей. Если есть стандартная спецификация, то эмулировать шлюз можно примерно на 40 %, исходя из набора тестовых сценариев на соединение с биржей
2. Техническое подключение к биржам – каналы восстановления данных (Replay/Recovery channels)	100	40	При данной оценке мы исходим из того, что симулятор не может воспроизвести всех технических деталей соединения с биржей. Если есть стандартная спецификация, то эмулировать шлюз можно примерно с 40-процентным покрытием сценариев
3. Тестирование интерфейса протокола от биржи к системе Ticker Plant (real time + replay/recovery)	100	100	Данная функциональность (тестирование внешнего шлюза) изолирована от биржи, поэтому мы считаем, что использование симулятора и тестовой биржи обеспечивают одинаковое покрытие набора тестовых сценариев
4. Восстановление потери небольшого количества данных (Replay channel)	100	100	Данная функциональность (тестирование внешнего шлюза) изолирована от биржи, поэтому мы считаем, что использование симулятора и тестовой биржи обеспечивают одинаковое покрытие набора тестовых сценариев
5. Восстановление потери большого количества данных (Recovery channel)	75	100	Несмотря на то, что данная функциональность изолирована от биржи, ее проверка требует значительного потока котировок с биржи. Исходя из нашего опыта, такое возможно не всегда с тестовой биржей, и есть ряд значительных ограничений. Поток данных с тестовых площадок обычно соответствует ожиданиям, однако его практически невозможно контролировать, чего не скажешь о симуляторе, контроль над которым находится в руках тестировщиков

6. Проверка справочных данных (Reference Data)	100	5	Данное тестирование сосредоточено на верификации настроек системы, настроек финансовых инструментов и т. д. Необходимо тестировать в связке с реальными рынками
7. Тестирование поведения системы Ticker Plant при отказе компонентов биржи (Failover при потоке данных с биржи)	100	10	Тестирование с симулятором в данном случае возможно, но значительная часть тестов проверяет соединение между Ticker Plant и биржами. При данной оценке мы исходим из того, что симулятор не может воспроизвести всех технических деталей соединения с биржей. При наличии стандартной спецификации эмулирование шлюза возможно с 40-процентным покрытием сценариев
8. Тестирование при отказе компонентов системы Ticker Plant (Failover при потоке данных из системы Ticker Plant)	100	40	Возможно тестирование с симулятором, но значительная часть тестов верифицирует соединение между Ticker Plant и рынками. При данной оценке мы исходим из того, что симулятор априори не может воспроизвести всех технических деталей соединения с биржей. Если есть стандартная спецификация, то эмулировать шлюз можно примерно с 40-процентным покрытием
9. Тестирование полного трейдингового дня (цикла) работы системы Ticker Plant (Daily Life Cycle – DLC)	40	100	В связи с большими ограничениями в управлении торговым днем на тестовых площадках, симулятор предоставляет больше возможностей при воспроизведении подобных сценариев тестирования
10. Тестирование полного трейдингового дня (цикла) на бирже (DLC)	100	40	Несмотря на то что предыдущее обоснование применимо и в этом случае, в даном сценарии больший вес имеет понятие того, что входящие сообщения создаются при помощи тестового рынка, согласно настройкам справочных данных самой биржи. При этой оценке мы исходим из того, что симулятор априори не может воспроизвести всех технических деталей соединения с полного трейдингового дня (цикла) работы системы (DLC). Поэтому оценка покрытия составляет лишь 40 %
11. Проверка всевозможных торговых статусов рынка	50	100	В связи с ограничениями в управлении торговым днем на тестовых площадках, симулятор предоставляет больше возможностей при воспроизведении подобных сценариев
12. Измерение ширины потребляемого канала передачи данных по сети (Bandwidth)	75	100	Тестирование с симулятором возможно; для тестовых площадок имеется значительная зависимость от аппаратных средств и настроек их производительности. В данном случае симулятор имеет преимущество
13. Измерение пропускной способности каналов в единицу времени (Throughput)	75	100	Тестирование с симулятором возможно; для тестовых площадок имеется значительная зависимость от аппаратных средств и настроек их производительности. В данном случае симулятор имеет преимущество

Окончание таблицы 2

Область тестирования	Покрытие функциональности при тестировании с помощью реальной тестовой биржи, %	Покрытие функциональности при тестировании с помощью симулятора, %	Детализация оценочных данных
14. Измерение задержек (латентности (Latency) системы)	75	100	Тестирование с симулятором возможно; для тестовых площадок имеется значительная зависимость от аппаратных средств и настроек их производительности. В данном случае симулятор имеет преимущество
15. Проверка нагрузочной способности системы (Capacity) – нагрузка входным потоком данных	75	100	Тестирование с симулятором возможно; для тестовых площадок имеется значительная зависимость от аппаратных средств и настроек их производительности. В данном случае симулятор имеет преимущество
16. Проверка нагрузочной способности системы (Capacity) – нагрузка с клиентской стороны системы Ticker Plant	75	100	Тестирование с симулятором возможно; для тестовых площадок имеется значительная зависимость от аппаратных средств и настроек их производительности. В данном случае симулятор имеет преимущество
17. Проверка системы Ticker Plant при неправильных запросах клиентов и ответная реакция на них для Replay/Recovery каналов	100	100	Данная функциональность (тестирование внешнего шлюза) изолирована от биржи, поэтому мы считаем, что использование симулятора и тестовой биржи обеспечивают одинаковое покрытие
18. Проверка правильности обработки системой Ticker Plant некорректных сообщений от биржи	25	100	В данном случае симулятор может отправлять на вход Ticker Plant гибкий набор негативных сценариев тестирования, что обеспечивает большее покрытие. С другой стороны, абсолютное число сценариев, с технической точки зрения, имея при этом только спецификацию биржи, невозможно воспроизвести
19. Проверка корректности работы системы в целом – от трейдингового клиента до системы Ticker Plant (E2E тестирование)	90	85	Данное тестирование представляет собой набор всех трейдинговых сценариев тестирования. Это можно обеспечить как симулятором, так и тестовой площадкой. Поэтому оценочная характеристика примерно одинаковая

20. Сложные сценарии на функциональность самой биржи	50	100	Оба подхода предоставляют равные возможности при эмуляции сложных сценариев. Поскольку симулятор подконтролен тестировщикам, он дает больше возможностей при эмуляции по-настоящему нетривиальных событий
21. Реконсиляционное тестирование	100	50	В данном тесте происходит сравнение потока данных из биржи и из Ticker Plant. Тестирование с реальной тестовой площадкой предпочтительнее и является более правильным, поскольку она по умолчанию является независимым источником котировок
22. Тестирование MBO (market by order) и MBP (market by price) сервисов	100	100	Данная функциональность (тестирование внешнего шлюза) изолирована от биржи, поэтому мы считаем, что использование симулятора и тестовой биржи обеспечивают одинаковое покрытие
23. Проверки расчета индексов	50	100	Данная функциональность должна проверяться при полном контроле над рынком. Симулятор имеет явное преимущество, с чем и связана такая оценка покрытия
24. Правильность расчета статистических данных	50	100	Данная функциональность должна проверяться при полном контроле над рынком. Симулятор имеет явное преимущество, с чем и связана такая оценка покрытия
25. Работа с историческими данными	50	100	Необходима эмуляция искусственных исторических событий. Для повторяемости тестов необходим симулятор, а не тестовая площадка, на события в которой также могут влиять и другие пользователи
26. Проверка получения и передачи новостных каналов от бирж	50	100	В этом тесте необходима эмуляция искусственных новостей. Для повторяемости тестов и полного контроля необходим симулятор
27. Проверка действия биржевого оператора	100	30	Системы для управления рынками должны проверяться в связке с реальными системами, на которых будет впоследствии работать система. Преимущество у тестовой площадки
28. Проверка сложных запросов клиента	100	100	Данная функциональность (тестирование внешнего шлюза) изолирована от биржи, поэтому мы считаем, что использование симулятора и тестовой биржи обеспечивает одинаковое покрытие
29. Мониторинг системы	100	30	Системы для управления рынками должны проверяться в связке с реальными системами, с которыми будет впоследствии работать система. Преимущество у тестовой площадки

Рис. 4. Сравнительный анализ покрытия тестами при помощи тестовой биржи и симулятора

Анализ данных, приведенных в табл. 2 и на рис. 4, показывает, что тестирование с помощью симуляторов и реальных тестовых бирж (в разумный временной период) имеет свои плюсы и минусы. В целом, мы имеем приблизительно одинаковые показатели по оценочной методике.

Общая формула покрытия всех функциональностей соответствующим инструментом тестирования:

$$\sum_{N=1}^{29} \frac{CovN * 1 / PriorN}{\sum_{N=1}^{29} \frac{1}{PriorN}}, \qquad (1)$$

где *Summ* – сумма (табл. 2); *CovN** – относительное покрытие данной функциональности соответствующим инструментом тестирования (табл. 2); *PriorN* – приоритет данной функциональности (табл. 1).

Итоговый результат: тестирование при помощи симуляторов обеспечивает 76 % покрытия; тестирование при помощи тестовой площадки – 83 %.

В данной статье описана система агрегации и распределения информации о котировках (Ticker Plant), перечислены ее основные характеристики, составлена справочная библиотека скриптов для тестирования, максимально покрывающих функциональность обобщенной системы Ticker Plant. Эту информацию можно использовать как пособие для тестирование таких систем. Также в статье даны определения биржевого симулятора и реальной тестовой биржи. При помощи разработанной методики оценки покрытия сценариями для тестирования системы Ticker Plant проанализированы и обработаны два возможных пути тестирования: при помощи реальной тестовой биржи и симуляторов бирж. На основе двух сводных таблиц выведено заключение о преимуществах тестирования как с реальной тестовой биржей, так и с биржевым симулятором. Сделан вывод (подтвержденный также практическим путем) о том, что распределенные системы, предназначенные для обработки данных в режиме реального времени, нельзя протестировать на 100 %, используя один из описанных выше подходов. Желаемого качества системы можно добиться, только сочетая лучшие характеристики методов тестирования.

СПИСОК ЛИТЕРАТУРЫ

1. B2Bits <epam> [электронный ресурс] / URL: http://www.b2bits.com/trading_solutions/market-data-solutions.html

2. FIXprotocol [электронный ресурс] / URL: http://www.fixprotocol.org/

3. London Stock Exchange [электронный ре-

cypc] / URL: http://www.londonstockexchange.com/products-and-services/millennium-exchange/millennium-exchange-migration/mit303.pdf

4. **Karlton P., Kocher P.** The Secure Sockets Layer (SSL) Protocol Version 3.0 [электронный ресурс] / URL: http://tools.ietf.org/html/rfc6101

5. FTSE [электронный ресурс] / URL: http://www.ftse.com/Indices/Data_Licenses/Real-time_Constituent_Data.jsp

6. B2Bits <epam> [электронный ресурс] / URL: http://www.b2bits.com/trading_solutions/market-data-solutions.html

7. **Sherman M., Sood P., Wong K., Iakovlev A., Parashar N.** Building the Book: A Full-Hardware Nasdaq Itch Ticker Plant on Solarflare's AoE FPGA Board [электронный ресурс] / URL: http://www.cs.columbia.edu/~sedwards/classes/2013/4840/reports/Itch.pdf

8. Tokyo Stock Exchange [электронный ресурс] / URL: http://www.tse.or.jp/english/market/mkinfo/mains.html

9. Day Trading About.com [электронный ресурс] / URL: http://daytrading.about.com/od/daytradingmarketdata/a/MarketDataDefin.htm

10. Customer Development Service (CDS). London Stock Exchange [электронный ресурс] / URL: http://www.londonstockexchange.com/products-and-services/technical-library/customer/customerdevelopmentservice/customerdevelopmentservice.htm

11. Trading Floor Architecture. Cisco [электронный ресурс] / URL: http://www.cisco.com/en/US/docs/solutions/Verticals/Trading_Floor_Architecture-E.html

12. BSE India [электронный ресурс] / URL: http://www.bseindia.com/markets/MarketInfo/DispNoticesNCirculars.aspx?page=20120531-22&pagecont=0,31/05/2012,31/05/2012,,All,All,All,Scrip%20Name%20/%20Code

13. ММВБ [электронный ресурс] / URL: http://rts.micex.ru/s437

14. **Воронцов К.В., Пшеничников С.Б.** Имитационное моделирование торгов: новая технология биржевых тренажеров. Индикатор. 2002. № 2 (42) [электронный ресурс] / URL: http://www.forecsys.com/site/about/press/exchange_simulator/

15. Oslo Bors Stock Exchange [электронный ресурс] / URL: http://www.oslobors.no/ob_eng/Oslo-Boers/Trading/Delta/Millennium-Exchange/Guide-to-Testing-Services-updated-issue

16. **Zverev A., Bulda A.** Exchange Simulators for SOR. Algo Testing: Advantages vs. Shortcomings. Conf. ExTENT [электронный ресурс] / URL: http://www.slideshare.net/extentconf/exchsims-forsoralgotestingadvantagesvsshortcomings29102011111113011104phpapp02

17. Exactpro Systems [электронный ресурс] / URL: http://www.exactprosystems.com/

REFERENCES

1. *B2Bits <epam>*. Available: http://www.b2bits.com/trading_solutions/market-data-solutions.html

2. *FIXprotocol*. Available: http://www.fixprotocol.org/

3. *London Stock Exchange*. Available: http://www.londonstockexchange.com/products-and-services/millennium-exchange/millennium-exchange-migration/mit303.pdf

4. **Karlton P., Kocher P.** *The Secure Sockets Layer (SSL) Protocol Version 3.0*. Available: http://tools.ietf.org/html/rfc6101

5. *FTSE*. Available: http://www.ftse.com/Indices/Data_Licenses/Real-time_Constituent_Data.jsp

6. *B2Bits <epam>*. Available: http://www.b2bits.com/trading_solutions/market-data-solutions.html

7. **Sherman M., Sood P., Wong K., Iakovlev A., Parashar N.** *Building the Book: A Full-Hardware Nasdaq Itch Ticker Plant on Solarflare's AoE FPGA Board*. Available: http://www.cs.columbia.edu/~sedwards/classes/2013/4840/reports/Itch.pdf

8. *Tokyo Stock Exchange*. Available: http://www.tse.or.jp/english/market/mkinfo/mains.html

9. *Day Trading. About.com*. Available: http://daytrading.about.com/od/daytradingmarketdata/a/MarketDataDefin.htm

10. *Customer Development Service (CDS)*. London Stock Exchange. Available: http://www.londonstockexchange.com/products-and-services/technical-library/customer/customerdevelopmentservice/customerdevelopmentservice.htm

11. *Trading Floor Architecture. Cisco*. Available: http://www.cisco.com/en/US/docs/solutions/Verticals/Trading_Floor_Architecture-E.html

12. *BSE India*. Available: http://www.bseindia.com/markets/MarketInfo/DispNoticesNCirculars.aspx?page=20120531-22&pagecont=0,31/05/2012,31/05/2012,,All,All,All,Scrip%20Name%20/%20Code

13. *MMVB*. Available: http://rts.micex.ru/s437

14. **Vorontsov K.V., Pshenichnikov S.B.** Imitatsionnoye modelirovaniye torgov: novaya tekhnologiya birzhevykh trenazherov, *Indikator*, 2002, No. 2 (42). Available: http://www.forecsys.com/site/about/press/exchange_simulator/ (rus)

15. *Oslo Bors Stock Exchange*. Available: http://www.oslobors.no/ob_eng/Oslo-Boers/Trading/Delta/Millennium-Exchange/Guide-to-Testing-Services-updated-issue

16. **Zverev A., Bulda A.** Exchange Simulators for SOR. Algo Testing: Advantages vs. Shortcomings. *ExTENT*. Available: http://www.slideshare.net/extentconf/exchsims-forsoralgotestingadvantagesvsshortcomings29102011111111301104phpapp02

17. *Exactpro Systems*. Available: http://www.exactprosystems.com/

БУЛДА Алёна Юрьевна – *менеджер проектов ООО «ИТС-Эксперт»*.
156000, Россия, г. Кострома, Ленина, д. 20.
E-mail: alyona.bulda@exactprosystems.com

BULDA, AlyonaYu. *ITS-Expert, LLC.*
156000, Lenina 20, Kostroma, Russia.
E-mail: alyona.bulda@exactprosystems.com

БУЯНОВА Ольга Александровна – *аналитик по качеству ООО «ИТС-Эксперт»*.
156000, Россия, г. Кострома, Ленина, д. 20.
E-mail: olga.buyanova@exactprosystems.com

BUYANOVA, Olga A. *ITS-Expert, LLC.*
156000, Lenina 20, Kostroma, Russia.
E-mail: olga.buyanova@exactprosystems.com

ЗВЕРЕВ Алексей Валерьевич – *соучредитель ООО «ИТС-Эксперт»*.
115088, Россия, Москва, 2-й Южнопортовый проезд, д. 20А/4.
E-mail: natalia.goroshnikova@exactprosystems.com

ZVEREV, Alexey V. *ITS-Expert, LLC.*
115088, 2nd Yuzhnoportovy proyezd 20A/4, Moscow, Russia.
E-mail: natalia.goroshnikova@exactprosystems.com

УДК 519.7

В.В. Подымов, У.В. Попеско

ВЕРИФИКАЦИЯ ПРОГРАММНО-КОНФИГУРИРУЕМЫХ СЕТЕЙ ПРИ ПОМОЩИ СИСТЕМЫ UPPAAL

V.V. Podymov, U.V. Popesko

UPPAAL-BASED VERIFICATION OF SOFTWARE-DEFINED NETWORKS

В последние несколько лет активное развитие получили программно-конфигурируемые сети (ПКС) – особый вид компьютерных сетей, в которых все коммутирующие устройства имеют централизованное управление. В статье изучены задачи формального описания и верификации ПКС. Для описания ПКС использована библиотека элементов UML в редакторе диаграмм Dia. Для верификации ПКС использовано программно-инструментальное средство UPPAAL. Основной результат исследований – разработка транслятора, позволяющего по диаграмме сети получить ее модель для верификации в виде сети конечных временных автоматов. Корректность трансляции строго обоснована. Проведен ряд экспериментов, показывающих применимость предложенного метода верификации для проверки свойств поведения ПКС, специфицированных посредством формул темпоральной логики реального времени.

ПРОГРАММНО-КОНФИГУРИРУЕМАЯ СЕТЬ; ВЕРИФИКАЦИЯ; ВРЕМЕННЫЕ АВТОМАТЫ; ТЕМПОРАЛЬНАЯ ЛОГИКА; UPPAAL.

A lot of efforts were made in the last few years in the area of software-defined networks (SDN) – a special kind of computer networks in which the switching device control is fully centralized. This paper investigates the problems of formal description and verification of SDN as a real-time system. We provide a UML-based description of SDN, using the UML diagram editor Dia. To verify real-time properties of SDN, we use a well-known model-checking tool UPPAAL. The main result of the research is an approach for SDN verification, based on translation of SDN description into network of timed automata. Translation correctness is formalized and proved. A number of experiments were done to show that the approach can be used to verify real-time properties of SDN specified as TCTL formulae.

SOFTWARE-DEFINED NETWORKS; VERIFICATION; TIMED AUTOMATA; TEMPORAL LOGIC; UPPAAL.

Идея программно-конфигурируемых сетей (ПКС) сформулирована специалистами университетов Стэнфорда и Беркли в 2006 г. [1] В таких сетях уровень управления отделен от устройств передачи данных: коммутаторы не участвуют в определении маршрутов для пакетов, а только реализуют программу контроллера. Основным стандартом, применяемым для построения ПКС, является протокол OpenFlow [2]. С помощью этого протокола реализуется независимый от производителя интерфейс между логическим контроллером и сетевыми коммутаторами ПКС.

Сеть OpenFlow состоит из коммутаторов, управляемых централизованным контроллером. Пакет, передаваемый по сети, обрабатывается контроллером существенно медленнее, нежели коммутатором, поэтому одной из основных функций контроллера является организация работы коммутаторов так, чтобы они обрабатывали большую часть пакетов, и лишь в исключительных случаях пакеты обрабатывались бы на контроллере.

Организация работы коммутаторов заключается в установке правил в таблицы коммутации (flow tables), определяющих, как будут обрабатываться те или иные пакеты. Правило состоит из шаблона, идентифицирующего вид пакетов, целочисленного приоритета, устраняющего неоднозначность

в случае наложения шаблонов, целочисленного лимита времени, указывающего число секунд до истечения срока активности правила, и списка действий, описывающих обработку пакета. Также для каждого правила в этих таблицах коммутатор содержит счетчики для учета количества и размера обрабатываемых пакетов.

Коммутатор обрабатывает приходящий пакет в три этапа. Сначала он выбирает правило из своей таблицы коммутации так, чтобы заголовок пакета соответствовал шаблону этого правила. Если такового не найдено, коммутатор отправляет пакет контроллеру для дальнейшей обработки. Иначе выбирается совпадающее по шаблону правило с наибольшим приоритетом. На втором шаге обновляются счетчики для выбранного правила. И, наконец, к пакету поочередно применяются все действия, записанные в правиле. К допустимым действиям относятся отправка пакета на порт и переписывание заголовочного поля правила.

Контроллер устанавливает правила в таблицы коммутации, реагируя на события в сети. Такими событиями являются подключение нового коммутатора к сети, удаление ранее действующего коммутатора из сети, события для сбора статистики, истечение лимита времени правила коммутатора. Также контроллер оперирует функциями отправки сообщений коммутаторам: установкой правил в таблицу коммутации, удалением всех правил с заданным шаблоном из таблицы коммутации, отправкой пакета и действия для его обработки коммутатором.

В статье исследуется возможность верификации ПКС как распределенных систем реального времени. Для этого потребовалось решить следующие задачи: выбрать адекватное средство для построения сетей; выбрать подходящее средство для верификации сетей как систем реального времени; построить корректный транслятор из выбранного средства построения сетей во входной язык нужной системы верификации; провести экспериментальное исследование возможности применения выбранного средства верификации для проверки спецификаций ПКС.

OpenFlow как стандарт реализации ПКС включает в себя большое количество свойств и предписаний для коммутации пакетов в сети. Производители компонентов компьютерных сетей используют лишь часть из них. Мы также ограничимся в рассматриваемых вариантах. При моделировании правил таблиц коммутации не будем уделять внимание приоритетам и счетчикам, а из всех допустимых действий будем рассматривать только действия типа отправки пакета на порт коммутатора. Сбор статистических данных в сети также не будет нас интересовать. С другой стороны, в нашу модель будут добавлены временные характеристики, описывающие работу физических объектов сети. Посредством этого мы будем учитывать не только предписания стандарта OpenFlow, но и технические возможности коммутаторов и каналов.

Статья обладает следующей логической структурой. Сначала мы опишем характеристики ПКС, учитываемые нами в предлагаемой модели, и общую схему верификации свойств ПКС. Здесь же дадим базовые сведения о выбранном инструменте верификации UPPAAL. Далее определим формальные синтаксис и семантику предложенной нами модели ПКС. После этого опишем разработанный нами алгоритм трансляции ПКС в сети временных автоматов, позволяющий проверять свойства ПКС в рамках возможностей UPPAAL. Затем представим теорему, обосновывающую корректность алгоритма трансляции, и приведем экспериментальные результаты, демонстрирующие возможности применения нашего подхода к верификации временных свойств ПКС.

Схема верификации

Общая схема верификации свойств ПКС состоит в следующем: сначала строится модель ПКС, учитывающая выбранные нами временные характеристики сети; затем эта модель корректным образом транслируется в сеть временных автоматов; далее свойство ПКС представляется в виде формулы логики LTL_{-X} с возможностью использования временных ограничений; построенная

формула автоматически проверяется на модели средством UPPAAL. Остаток главы посвящен пояснению основных особенностей модели и краткому описанию средства UPPAAL.

Модель имеет три составляющие: модель коммутатора, модель контроллера и описание сетевых каналов. В модели коммутатора учтены и описываются следующие временные характеристики: *rule_imp* – задержка поиска и выполнения правила на коммутаторе (временной интервал); *rule_def* – задержка установки в коммутатор правила от контроллера; *rule_ar* – интенсивность поступления пакетов из внешней среды; *rule_con* – задержка доставки необработанного пакета контроллеру. Также описываются физические характеристики *port* – множество портов коммутатора, соединяющих его с другими коммутаторами и внешней средой, и *tab* – объем таблицы коммутации, т. е. максимальное число правил таблицы.

Сетевые каналы описываются тем, какие порты они соединяют, и тем, какие временные ограничения на доставку пакетов они имеют. Поведение контроллера определяется политикой коммутации пакетов в сети, которая для нашего класса задач представима в виде таблицы, отображающей соответствие между заголовком поступившего на контроллер пакета и правилом, которое необходимо установить на коммутатор, приславший этот пакет. Модель контроллера состоит из множества записей такой таблицы.

В следующих разделах приведено формальное описание всех составляющих предложенной нами модели ПКС и их поведения, пригодное для применения формальных математических методов анализа.

Для построения модели ПКС был выбран кроссплатформенный редактор диаграмм Dia [3]. Модель сети из рассматриваемого нами подкласса описывается с помощью объектов библиотеки элементов UML, предоставляемых редактором, как показано на рис. 1.

Каждый коммутатор изображается в виде прямоугольника с тремя секциями. В верхней секции обозначается имя коммутатора, в нижней содержатся правила таблицы коммутации, в средней – описанные выше физические характеристики коммутатора. Каждое правило таблицы коммутации имеет четыре поля: имя порта, на который поступил пакет, заголовок поступившего пакета, лимит времени жизни правила и порт, на который необходимо отправить пакет.

Для отображения топологии сети ком-

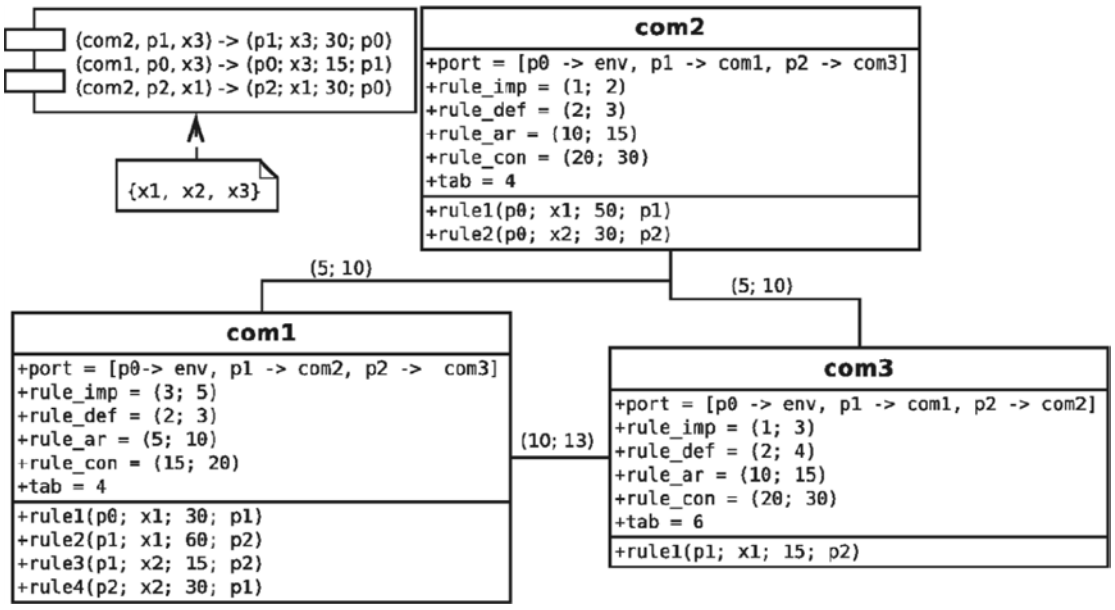


Рис. 1. Пример описания ПКС диаграммами Dia

мутаторы соединяются линиями, моделирующими дуплексные каналы сети. Контроллер изображается как прямоугольник с набором правил для коммутаторов. Также отдельно описывается множество всех возможных заголовков пакетов.

Для проверки спецификаций модели ПКС было выбрано программно-инструментальное средство UPPAAL [4]. Особенностью UPPAAL является возможность проверять свойства модели, в описание которой включено реальное время, это и определило наш выбор инструмента верификации. Для верификации UPAAL использует метод проверки свойств на моделях. Данный метод предполагает наличие модели, описывающей систему на определенном уровне абстракции, и позволяет проверить, удовлетворяет ли заданная модель системы формальным спецификациям. В средстве верификации UPPAAL в качестве модели используются сети конечных временных автоматов (параллельные композиции временных автоматов) [5], а формальные спецификации задаются формулами темпоральной логики TCTL [4].

Временные автоматы представляют собой конечные автоматы, работающие в реальном времени и осуществляющие синхронизацию посредством передачи сигналов через каналы связи. Особенностью таких автоматов является возможность использования таймеров. Значения таймеров можно указывать во временных ограничениях условий переходов между состояниями. Показания всех таймеров изменяются на одинаковые величины с течением времени.

Для верификации ПКС, описанных в терминах диаграмм Dia, средством UPPAAL нами предложен алгоритм трансляции диаграмм в сети временных автоматов. Сеть, получаемая при трансляции, состоит из автоматов, моделирующих коммутаторы, контроллер, внешнюю среду и каналы сети. Таким образом, в результате трансляции диаграмм мы получаем сеть, пригодную для верификации средством UPPAAL. Подобный подход к верификации распределенных систем реального времени применен в работе [6].

Формальный синтаксис ПКС

Перед описанием алгоритма трансляции необходимо предоставить формальное описание всех компонентов сети ПКС. С учетом выбранных нами ограничений ПКС может быть описана системой (H, con, Com, $Chan$), где H – множество заголовков пакетов сети, con – контроллер, $Com = (com_1, ..., com_n)$ – набор коммутаторов сети и $Chan = (c_1, ..., c_m)$ – набор каналов сети. Заметим, что во введенной формальной модели ПКС вместо пакетов рассматриваются только их заголовки, при этом содержащиеся в пакетах сообщения опускаются. Для простоты восприятия далее под пакетами в формальной модели будут подразумеваться их заголовки. Здесь и далее символы L и R будут использоваться для обозначения соответственно левой и правой границы интервалов, описывающих временные характеристики сети, такие как время доставки и обработки пакетов.

Коммутатор com_i включает в себя множество портов $Port_i$, начальную таблицу коммутации $Rule_i \subseteq Port_i \times H \times Real \times Port_i$ объема tab и временные характеристики L_i^{imp}, R_i^{imp}, L_i^{def}, R_i^{def}, L_i^{ar}, L_i^{con}, R_i^{con}, естественным образом соотносящиеся с характеристиками коммутатора, описанными в диаграмме (см., например, рис. 1). Правило (p, h, x, p') таблицы коммутации означает, что пакет h, пришедший на порт p, перенаправляется на порт p'. При этом x – время жизни правила; если $x \geq 0$, то правило назовем активным, иначе – истекшим. Шаблоном правила (p, h, x, p') будем называть пару (p, h).

Контроллер con представляет собой множество пар вида (i, r), означающих, что правило r может быть выслано коммутатору com_i. Канал c описывается тем, из какого порта какого коммутатора он исходит, и какими временными характеристиками задержки доставки пакетов (L, R) обладает. Заметим, что в предложенной модели каналы являются однонаправленными. Такое ограничение несущественно, т. к. дуплексный канал моделируется двумя однонаправленными.

Формальная семантика ПКС

Для доказательства корректности алго-

ритма трансляции необходимо ввести формальное описание работы ПКС. Данное описание является формализацией поведения ПКС в рамках стандарта OpenFlow с учетом введенных нами ограничений.

Состояние ПКС складывается из состояний контроллера и всех коммутаторов и каналов. Для удобства описания предполагаем, что каждый коммутатор com_i располагает следующими каналами: канал c_i^{ar} входного потока, по которому поступают пакеты из окружения, с временными характеристиками $L = L_i^{ar}$, $R = R_i^{ar}$, ведущий в специально выделенный под него порт; канал c_i^{tocon}, ведущий в контроллер из другого, специально выделенного порта, с характеристиками $L = L_i^{con}$, $R = R_i^{con}$; канал $c_i^{fromcon}$, ведущий от контроллера в третий, специально выделенный порт коммутатора, с характеристиками $L = R = 0$. Таким образом, состояние S ПКС включает в себя состояния s^{con} контроллера, $s_1^{com}, ..., s_n^{com}$ коммутаторов и $s_1^{ch}, ..., s_m^{ch}$, $s_1^{ar}, ..., s_n^{ar}$, $s_1^{tocon}, ..., s_n^{tocon}$, $s_1^{fromcon}, ..., s_n^{fromcon}$ каналов *ch* между коммутаторами, *ar* от входного потока, *tocon* к и *fromcon* от контроллера соответственно.

Коммутатор com_i может находиться в состояниях (*start*, *Rule*), (*select*, *t*, *Rule*, *h*, *p*), (*hit*, *Rule*, *h*, *p*), (*miss*, *Rule*, *h*, *p*), (*mod*, *t*, *Rule*). В состоянии *start* коммутатор прослушивает входящие каналы на предмет поступивших пакетов. В состоянии *select* коммутатор, получив на порт *p* пакет *h*, просматривает таблицу коммутации для принятия решения о перенаправлении пакета. В состоянии *hit* пакет засылается в канал, исходящий из порта *p*. В состоянии *miss* шаблон, состоящий из пакета *h* и порта *p*, на который он пришел, засылается в канал к контроллеру. Вообще говоря, коммутатор также должен посылать контроллеру свой идентификатор, однако в построенной формальной модели идентификатор может быть восстановлен по номеру порта, входящего в коммутатор. В состоянии *mod* происходит запись в таблицу коммутации правила, присланного контроллером. Во всех состояниях коммутатора *Rule* – текущая таблица коммутации, *t* – время нахождения в текущем управляющем состоянии, *h* и *p* – хранимые пакет и порт.

Каналы могут находиться в состояниях (*empty*), (*full*, *t*, *o*) и (*sent*, *o*), где *o* – пакет (для обычных и поточных каналов), шаблон (для каналов к контроллеру) либо правило (для каналов от контроллера). Компонента $t \in Real$ – время обработки сообщения каналом. Состояние *empty* соответствует пустому каналу. В состоянии *full* в канале содержится пока ещё не доставленный пакет. В состоянии *sent* канал содержит доставленный пакет и ожидает считывания этого пакета коммутатором или контроллером.

Контроллер может находиться только в состояниях (*idle*) (ожидание запросов от коммутаторов) и (*send*, *i*, *r*) (послать *i*-му коммутатору новое правило *r*).

Начальное состояние S_o системы строится из начальных состояний коммутаторов, контроллера и каналов. Начальное состояние *i*-го коммутатора – (*start*, $Rule_i$), контроллера – (*idle*), каналов – (*empty*).

Работа ПКС характеризуется последовательностью состояний, начинающейся в S_0 и строящейся по описанным далее правилам. Запись $s \to s'$ означает, что следующее состояние может быть получено из предыдущего заменой компоненты *s* на *s'*. Запись $s_1, s_2 \to s_1', s_2'$ означает то же самое для пары компонент. Построение вычисления ПКС происходит по следующим правилам.

1. Получение пакета: s_i^{com}, s_i^{ar} = (*start*, *Rule*), (*sent*, *h*) → (*select*, 0, *Rule*, *h*, *p*), (*empty*).

2. Применение активного правила $r = (p, h, x, p')$: s_i^{com} = (*select*, *Rule*, *t*, *h*, *p*) → (*hit*, *Rule*, *h*, *p'*), если $t \in [L_i^{imp}, R_i^{imp}]$, $r \in Rule$.

3. Обработка истекшего правила $r = (p, h, x, p')$: s_i^{com} = (*select*, *Rule*, *t*, *h*, *p*) → (*miss*, *Rule'*, *h*, *p*), если $t \in [L_i^{imp}, R_i^{imp}]$, $r \in Rule$ и $Rule' = Rule \setminus \{r\}$.

4. Сброс пакета: s_i^{com} = (*select*, *Rule*, *t*, *h*, *p*) → (*start*, *Rule*), если $t \in [L_i^{imp}, R_i^{imp}]$, $|Rule| = tab_i$, все правила из *Rule* активны и ни одно из них не имеет шаблона (*p*, *h*).

5. Несоответствие шаблонов: s_i^{com} = = (*select*, *Rule*, *t*, *h*, *p*) → (*miss*, *Rule'*, *h*, *p*), где *Rule'* получается из *Rule* удалением всех истекших правил.

6. Начало перезаписи таблицы: s_i^{com},

$s_i^{fromcon}$ = (*start*, *Rule*), (*sent*, *rule*) → (*mod*, 0, *Rule*), (*sent*, *rule*).

7. Успешная перезапись таблицы: s_i^{com}, $s_i^{fromcon}$ = (*mod*, *t*, *Rule*), (*sent*, (*p*, *h*, *x*, *p'*)) → (*hit*, *Rule'*, *h*, *p'*), (*empty*), если $t \in [L_i^{def}, R_i^{def}]$, |*Rule*| < tab_i и *Rule'* = *Rule* ∪ {(*p*, *h*, *x*, *p'*)}.

8. Неуспешная перезапись таблицы: s_i^{com}, $s_i^{fromcon}$ = (*mod*, *t*, *Rule*), (*sent*, *rule*) → (*start*, *Rule*), (*empty*), если $t \in [L_i^{def}, R_i^{def}]$ и |*Rule*| = *tab*.

9. Пересылка пакета коммутатору: s_i^{com}, s_j^{ch} = (*hit*, *Rule*, *h*, *p*), (*empty*) → (*start*, *Rule*), (*full*, 0, *h*), если канал c_j исходит из порта *p* коммутатора *i*.

10. Пересылка шаблона контроллеру: s_i^{com}, s_i^{tocon} = (*miss*, *Rule*, *h*, *p*), (*empty*) → (*start*, *Rule*), (*full*, 0, (*p*, *h*)).

11. Успешная обработка шаблона контроллером: s^{con}, s_i^{tocon} = (*idle*), (*sent*, (*p*, *h*)) → (*send*, *i*, *r*), (*empty*), если (*i*, (*p*, *h*, *x*, *p'*)) ∈ *con*.

12. Неуспешная обработка шаблона контроллером: s^{con}, s_i^{tocon} = (*idle*), (*sent*, (*p*, *h*)) → (*idle*), (*empty*), если (*i*, (*p*, *h*, *x*, *p'*)) ∉ *con*.

13. Пересылка правила от контроллера: s^{con}, $s_i^{fromcon}$ = (*send*, *i*, *r*), (*empty*) → (*idle*), (*full*, 0, *r*).

14. Появление пакета в сети: s_i^{ar} = = (*empty*) → (*full*, 0, *h*), если *h* ∈ *H*.

15. Сброс пакета в окружение: s_i^{com} = = (*hit*, *Rule*, *h*, *p*) → (*start*, *Rule*), если ни один канал c_j не исходит из порта *p* коммутатора com_i.

16. Доставка в канале: (*full*, *t*, *o*) → (*sent*, *o*), если *t* ∈ [*L*, *R*] для характеристик *L*, *R* соответствующего канала.

17. Продвижение времени: таймеры всех коммутаторов и каналов увеличиваются на одну и ту же положительную величину *d*, времена жизни правил уменьшаются на эту же величину, и при этом:

если какой-либо канал находится в состоянии (*full*, *t*, *o*), то $t + d \le R$ для характеристики *R* этого канала;

если s_i^{com} = (*select*, *Rule*, *t*, h, *p*), то $t + d \le R_i^{imp}$;

если s_i^{com} = (*mod*, *Rule*, *t*), то $t + d \le R_i^{def}$.

Заметим, что в каждый момент времени к текущему состоянию сети может быть применено в общем случае более одного правила. В такой ситуации недетерминированно может быть выбрано любое из правил. Роль окружения в нашей модели заключается в генерации пакетов для заданных портов коммутаторов и приеме предназначенных для сброса во внешнюю среду пакетов. Таким образом мы абстрагируемся от конкретного окружения.

Алгоритм трансляции

Алгоритм трансляции переводит ПКС в эквивалентную ей сеть временных автоматов. Уточнение понятия эквивалентности обсуждается в следующем разделе.

Сеть *N*, получаемая в результате трансляции ПКС ((*com*$_1$, ..., *com*$_n$), *con*, (*c*$_1$, ..., *c*$_m$), *H*), содержит автоматы *Hurry*, *Env*, *Stream*, *Chan*, A_{con} и A_i, $i \in \{1, ..., n\}$.

Каждый канал c_i моделируется булевыми переменными *full*[c_i] (пакет послан), *ready*[c_i] (пакет доставлен), таймером *t*[c_i] и переменными для хранения объектов, пересылаемых через канал. Сброс пакетов в окружение моделируется с помощью особого канала c_{env}. Также в сеть добавляются все каналы c_i^{ar}, c_i^{tocon}, $c_i^{fromcon}$.

Автомат *Hurry* обеспечивает срочные дуги (т. е. дуги, выполняющиеся немедленно по выполнении их предусловий) и состоит из одной вершины и петли, принимающей сигнал по срочному каналу *hurry*. К срочным дугам по умолчанию добавляется посылка сигнала по каналу *hurry*. Автомат *Env* удаляет пакеты из канала c_{env} и состоит из одной вершины и срочной петли с записью в переменную *full*[c_{env}] значения *false*. Автомат *Stream* генерирует пакеты, состоит из одной вершины и содержит набор срочных петель, недетерминированно засылающий пакеты в каналы c_i^{ar}. Автомат *Chan* обеспечивает доставку пакетов каналами, состоит из одной вершины и содержит петлю для каждого канала *c*, помеченную предусловием *full*[*c*]&&!*ready*[*c*]&&(*t*[*c*] ≥ *L*(*c*)) и записью в переменную *ready*[*c*] значения *true*, где *L*(*c*) – характеристика *L* канала *c*. Сама вершина автомата *Chan* при этом помечена инвариантом – конъюнкцией неравенств *t*[*c*] ≤ *R*(*c*).

Автомат A_{con} имеет два обычных состояния – *idle* и *send* – и одно срочное состоя-

ние *l*. Срочная дуга *idle* → *l* записывает в локальные переменные автомата шаблон, присланный коммутатором, и номер этого коммутатора и в зависимости от наличия отправляемого обратно правила переходит либо обратно в *idle*, либо в *send*. Срочная дуга *send* → *idle* присваивает переменной *full*[*c*] значение *true* и записывает в переменную канала выбранное правило.

Автомат A_i моделирует работу коммутатора com_i и состоит из обычных состояний *start*, *select*, *hit*, *miss*, *mod*, соединенных через срочные вершины. Срочная дуга *start* → *select* записывает в локальные переменные автомата доставленный через канал *c* пакет и порт, на который он пришел, и записывает в переменную *full*[*c*] значение *false*. Состояние *select*, помеченное инвариантом $t \le R_i^{imp}$, имеет одну исходящую дугу в срочное состояние *l*, помеченную предусловием $t \ge L_i^{imp}$. Здесь *t* – локальный таймер коммутатора. Из состояния *l* в зависимости от наличия шаблона происходит переход в состояния *hit* и *miss* с модификацией таблицы согласно правилам 2–5 семантики ПКС. Удаление многих правил из таблицы коммутации обеспечивается срочной вершиной *l'* с петлей, удаляющей из таблицы истекшие правила, и исходящей дугой, помеченной предусловием, утверждающим, что из таблицы удалены все истекшие правила. Срочная дуга *start* → *mod* записывает в локальные переменные автомата правило, доставленное каналом $c_i^{fromcon}$. Состояние *mod* помечено инвариантом $t \le R_i^{def}$, исходящие из него дуги – предусловием $r \ge L_i^{def}$. В зависимости от того, заполнена ли таблица коммутации, из состояния *mod* автомат может либо перейти в состояние *start*, либо перейти в состояние *hit* с записью правила в таблицу.

Описанный в этом разделе алгоритм трансляции будем обозначать записью *Alg*, а сеть временных автоматов, получаемую из ПКС-модели *N*, – записью *Alg*(*N*).

Корректность алгоритма трансляции

Важным результатом данной статьи является строгое обоснование корректности алгоритма *Alg*, – эквивалентности по прореживанию (stuttering equivalence) систем переходов [7], описывающих поведение исходной ПКС *N* и сети временных автоматов *Alg*(*N*), получаемой в результате применения алгоритма *Alg*. Обоснования такой эквивалентности достаточно для того, чтобы признать все формулы LTL_{-X}, а значит, и все формулы, которые могут быть проверены средством UPPAAL [4], равновыполнимыми.

Под корректностью алгоритма *Alg* понимается равновыполнимость формул, используемых в средстве UPPAAL, для исходной ПКС *N* и результирующей сети *Alg*(*N*). Ключевым понятием в доказательстве корректности алгоритма *Alg* является эквивалентность по прореживанию (stuttering equivalence) для систем переходов [7]. Неформально такая эквивалентность означает, что если в каждом вычислении двух данных систем склеить все одинаковые состояния, то мы получим одинаковые множества вычислений. Одинаковыми полагаются состояния с совпадающими множествами выполнимых формул. В работе [8] сформулирована следующая теорема.

Теорема 1. Если системы переходов M_1, M_2 эквивалентны по прореживанию и формула Ф логики LTL_{-X} истинна для M_1, то она также истинна для M_2.

Следующая теорема позволяет применить только что сформулированную к системам переходов, описывающим поведение ПКС *N* и сети *Alg*(*N*). Система переходов, описывающая поведение сети временных автоматов, обсуждается, например, в [5]. Пусть TS_A – система переходов, описывающая поведение системы *A*.

Теорема 2. Пусть *N* – произвольная ПКС. Тогда системы переходов TS_N и $TS_{Alg(N)}$ эквивалентны по прореживанию.

Теорему обосновывают следующие рассуждения. Состояниям контроллера и коммутаторов ставятся в соответствие состояния автоматов, в которые они транслируются, составляемые из одноименных управляющих состояний и из значений локальных переменных, представленных в описании алгоритма трансляции. Состояниям каналов ставятся в соответствие наборы значений переменных *full*, *ready* и переменных для хранения данных. Одному применению

семантических правил 1–17 соответствует последовательность переходов в сети временных автоматов через срочные состояния, и смена значений переменных происходит только в одном месте последовательности. Если к состоянию ПКС N применять правило из описания формальной семантики и к соответствующему состоянию сети $Alg(N)$ применять описанную выше последовательность переходов, то результирующее состояние ПКС N будет соответствовать результирующему состоянию сети $Alg(N)$.

Корректность алгоритма напрямую следует из приведенных теорем с учетом того, что формулы, проверяемые средством UPPAAL, могут быть переформулированы в терминах логики LTL_{-X}.

Экспериментальное исследование

На рис. 2–4 приведена сеть временных автоматов, полученная в результате трансляции из ПКС, изображенной на рис. 1. В целях удобочитаемости выражения автоматов написаны в синтаксисе, отличном от синтаксиса UPPAAL и при этом более простом для восприятия.

Запись вида (i = 1..3, 5, 7) означает перебор всех i из заданного диапазона и создание копий дуги для каждого значения i. Запись вида (i = 1..3 → Ф) означает «для всех i из заданного диапазона верна формула Ф».

Функция *get*(c) сохраняет содержимое канала c в локальные переменные (h, p, ...) и выполняет присваивание c = *false*. Функция *send*(c, o) копирует o в содержимое канала и выполняет присваивания c = *true*, *c_ready* = *false*, *c_t* = 0. Функция *set_rule*(i) копирует локальные переменные коммутатора, отвечающие компонентам правила, в i-ю позицию таблицы.

Предикат *to_deliver*(c) описывается как $c\&\&!c\_ready\&\&(c\_t \geq c\_L)$. Предикат *ok*($c$) описывается как $!c||c\_ready||(c\_t \leq c\_R)$.

Канал c[0] выделен под окружение, каналы c[1], c[2], c[3] – под потоки, прикрепленные к соответствующим коммутаторам. Автоматы A_2, A_3 отличаются от автомата A_1 только номерами задействованных каналов и константами, определяемыми количеством портов и объемом таблицы, и по этой причине опущены.

Заметим, что в наши задачи входила не верификация «реальных» сетей, содержащих сотни коммутаторов и тысячи правил в таблицах коммутации (очевидно, это невозможно на начальной стадии исследования), а описание главных идей верификации временных аспектов поведения ПКС. Ниже приведены проверенные с помощью средства UPPAAL свойства и их запись на языке запросов UPPAAL [4].

1. В работе системы не возникает блокировки:

$$A[]\ not\ deadlock$$

2. В сеть всегда будут поступать пакеты из внешней среды:

$$A<>\ forall(num : int\ [0,2])\ (channel\_h[stream.align[num]])$$

3. Допустим сценарий работы сети, в котором коммутатор не принимает ни одного пакета:

$$E[]\ com1.start$$

4. При любом сценарии работы сети контроллер обработает хотя бы один пакет:

$$E<>\ !con.idle$$

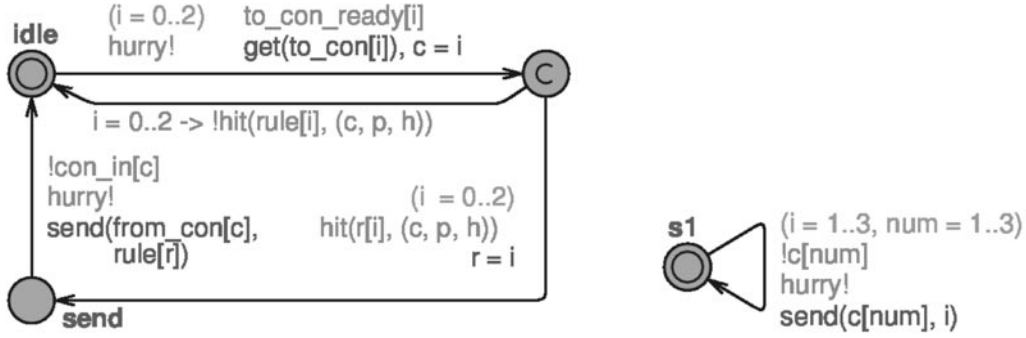


Рис. 2. Автоматы A_{con} (слева), *Stream* (справа)

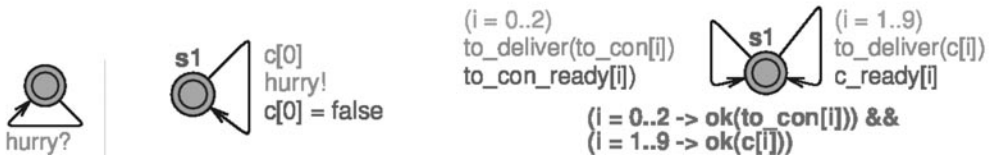


Рис. 3. Автоматы *Hurry* (слева), *Env* (в центре), *Chan* (справа)

5. Хотя бы один пакет будет успешно перенаправлен коммутатором (т. е. коммутатор выполняет свою главную функцию):

$$E<>\ com1.hit$$

Свойства 1, 2, 4, 5 соответствуют спецификации сетей OpenFlow, в то время как свойство 3 описывает нежелательные сценарии работы сети. Проверка показала, что свойства 1, 2, 4, 5 выполняются, а свойство 3 не выполняется. Это свидетельствует о том, что функционирование полученной сети временных автоматов соответствует нашим ожиданиям и что предложенная схема верификации ПКС с помощью средства UPPAAL позволяет проверять свойства поведения ПКС как системы реального времени.

В ячейках таблицы представлено время проверки описанных темпоральных свойств для некоторых моделей ПКС: 1 – три коммутатора в топологии кольца (см. рис. 1); 2 – четыре коммутатора в топологии звезды; 3 – четыре коммутатора в произвольной топологии; 4 – два коммутатора с изначально пустыми таблицами коммутации. Проверка проводилась на вычислительном устройстве со следующими характеристиками: INTEL Core i7 3820/1600 МГц/2Гб DDR3. Первое свойство не было проверено на моделях 1–3 в связи с нехваткой памяти.

В результате проведенных исследований мы подтвердили практическую осуществимость подхода, предложенного в статье [6], для проверки выполнимости свойств поведения моделей программно-конфигурируемых сетей в реальном времени. Предложенное нами средство анализа поведения ПКС может быть использовано для наглядного описания конфигурации и коммутационной политики сети в виде диаграмм. Разработанный нами алгоритм трансляции преобразует диаграмму в сеть временных автоматов, подаваемую на вход системы верификации

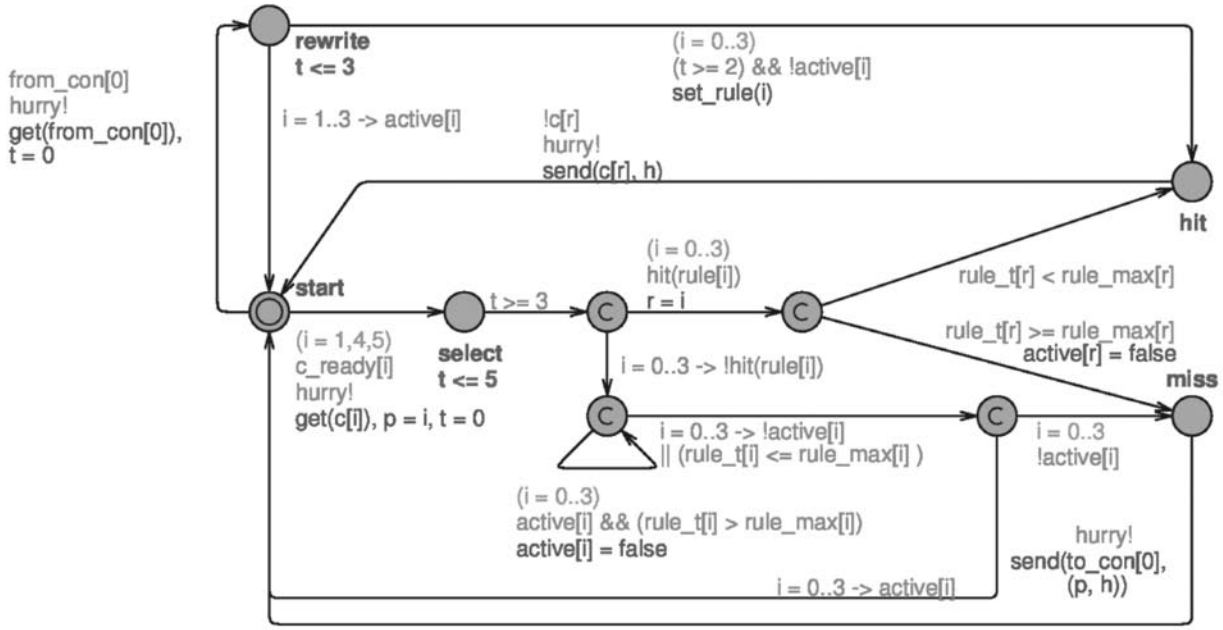


Рис. 4. Автомат A_1

Время проверки свойств ПКС

	Свойство				
	1	2	3	4	5
Модель 1	–	1 с	1 с	7 с	1 с
Модель 2	–	1 с	1 с	1 мин 2 с	1 мин 25 с
Модель 3	–	1 с	1 с	1 мин	1 мин 19 с
Модель 4	27 ч	1 с	1 с	1 с	1 с

UPPAAL. Корректность алгоритма трансляции строго обоснована, транслятор реализован на языке программирования Perl. Получаемая на выходе транслятора сеть временных автоматов позволяет проверять спецификации ПКС с помощью системы UPPAAL. Особенностью такой верификации является возможность учитывать временной аспект поведения ПКС как распределенных систем реального времени.

СПИСОК ЛИТЕРАТУРЫ

1. **Casado M., Garfinkel T., Akella A., Fredman M., Boneh D., McKeown N., Shenker S.** SANE: A Protection Architecture for Enterprise Networks // 15th Usenix Security Symp. Vancouver, Canada, 2006.

2. **McKeown N., Anderson T., Balakrishnan H., Parulkar G., Peterson L., Rexford J., Shenker S., Turner J.** Openflow: Enabling innovation in campus network // SIGCOMM Computer Communication Review. 2008. Vol. 38. No. 2. Pp. 69–74.

3. Dia [электронный ресурс]/ URL: http://dia-installer.de/doc/en/dia-manual.pdf (дата обращения 08.11.2013)

4. **Behrmann G., David A., Larsen K.** A tutorial on Uppaal // Lecture Notes in Computer Science. 2004. Vol. 3185. Pp. 200–236.

5. **Alur R., Dill D.** Automata for modelling real-time systems // Proc. of Internat. Colloquium on Algorithms, Languages, and Programming, LNCS. 1990. Vol. 443. Pp. 322–335.

6. **Волканов Д.Ю., Захаров В.А., Зорин Д.А., Коннов И.В., Подымов В.В.** Как разработать простое средство верификации систем реального времени // Моделирование и анализ информационных систем. 2012. № 6. Т. 19. С. 45–56.

7. **Browne M.C., Clarke E.M., Grumberg O.** Characterizing finite Kripke structures in propositional temporal logics // Theor. Comp. Sci. 1988. Vol. 59(1-2). Pp. 115–131.

8. **Clarke E.M., Grumberg O., Peled D.** Model Checking. The MIT Press, 1999.

REFERENCES

1. **Casado M., Garfinkel T., Akella A., Fredman M., Boneh D., McKeown N., Shenker S.** SANE: A Protection Architecture for Enterprise Networks. *15th Usenix Security Symposium*, Vancouver, Canada, August 2006.

2. **McKeown N., Anderson T., Balakrishnan H., Parulkar G., Peterson L., Rexford J., Shenker S., Turner J.** Openflow: Enabling innovation in campus network. *SIGCOMM Computer Communication Review*, 2008, Vol. 38, No. 2. Pp. 69–74.

3. Dia. Available: http://dia-installer.de/doc/en/dia-manual.pdf (Accesed 08.11.2013)

4. **Behrmann G., David A., Larsen K.** A tutorial on Uppaal, *Lecture Notes in Computer Science*, 2004, Vol. 3185. Pp. 200–236.

5. **Alur R., Dill D.** Automata for modelling real-time systems, *Proc. of Internat. Colloquium on Algorithms, Languages, and Programming,* LNCS, 1990, Vol. 443, Pp. 322–335.

6. **Volkanov D.Yu., Zakharov V.A., Zorin D.A., Konnov I.V., Podymov V.V.** Kak razrabotat prostoye sredstvo verifikatsii sistem realnogo vremeni [On the Designing of Model Checkers for Real-Time Distributed Systems], *Modelirovaniye i analiz informatsionnykh system* [*Modelling and Analysis of Information Systems*], 2012, No. 6, Vol. 19, Pp. 45–56.

7. **Browne M.C., Clarke E.M., Grumberg O.** Characterizing finite Kripke structures in propositional temporal logics, *Theor. Comp. Sci.*, 1988, Vol. 59(1-2), Pp. 115–131.

8. **Clarke E.M., Grumberg O., Peled D.** *Model Checking*. The MIT Press, 1999.

ПОДЫМОВ Владислав Васильевич – *аспирант факультета вычислительной математики и кибернетики Московского государственного университета имени М.В. Ломоносова.*
119991, Россия, Москва, ГСП-1, Ленинские горы, д. 1, стр. 52.
E-mail: valdus@yandex.ru

PODYMOV, Vladislav V. *Lomonosov Moscow State University.*
119991, GSP-1, Leninskie Gory, Moscow, Russia.
E-mail: valdus@yandex.ru

ПОПЕСКО Ульяна Владиславовна – *аспирант факультета вычислительной математики и кибернетики Московского государственного университета имени М.В. Ломоносова.*
119991, Россия, Москва, ГСП-1, Ленинские горы, д. 1, стр. 52.
E-mail: ulya_kiber@mail.ru

POPESKO, Uliana V. *Lomonosov Moscow State University.*
119991, GSP-1, Leninskie Gory, Moscow, Russia.
E-mail: ulya_kiber@mail.ru

УДК 536.421

А.В. Никешин, Н.В. Пакулин, В.З. Шнитман

АВТОМАТИЗАЦИЯ ТЕСТИРОВАНИЯ СООТВЕТСТВИЯ РЕАЛИЗАЦИЙ СТАНДАРТУ ПРОТОКОЛА БЕЗОПАСНОСТИ ТРАНСПОРТНОГО УРОВНЯ TLS

A.V. Nikeshin, N.V. Pakulin, V.Z. Shnitman

CONFORMANCE TESTING AUTOMATION FOR TRANSPORT LAYER SECURITY PROTOCOL TLS

Протокол TLS используется для защиты обмена данными между клиентом и сервером в различных прикладных сценариях: при обмене данными между браузером и веб-сервером, передаче электронной почты, создании виртуальных сетей, голосовой и видеосвязи и т. п. В настоящее время широко используются более десятка различных реализаций TLS. Обеспечение совместимости реализаций TLS является очень актуальной задачей: несовместимость двух реализаций может привести к различным последствиям, начиная с невозможности установить соединение вплоть до разглашения конфиденциальных данных.

Представлен подход к тестированию соответствия стандарту TLS, основанный на автоматизированном тестировании соответствия формальным спецификациям. Приведены результаты тестирования нескольких общедоступных реализаций TLS. Описаны направления дальнейших исследований.

ТЕСТИРОВАНИЕ; ВЕРИФИКАЦИЯ; ФОРМАЛЬНЫЕ МЕТОДЫ; ФОРМАЛЬНЫЕ СПЕЦИФИКАЦИИ; ТЕСТИРОВАНИЕ С ИСПОЛЬЗОВАНИЕМ МОДЕЛЕЙ; МОДЕЛИ ПРОГРАММ; TLS.

TLS protocol is widely used to protect data exchange between clients and servers in various scenarios: while browsing Internet, sending and receiving e-mails, establishing VPN, etc. There are dozens implementations in the market at the moment; such as, ensuring interoperability is highly important. Incompatibility between two implementations might result in disconnection or even disclosure of sensitive data.

Conformance testing is the primary tool to ensure interoperability between implementations of a protocol. The paper presents a model-based approach to conformance testing of TLS implementations. It discusses the formal model of TLS protocol, the structure of the test suite. We applied the test suite to a several popular implementations of TLS, and present brief results. The paper concludes with discussion of the directions for future research.

TESTING; VERIFICATION; FORMAL METHODS; FORMAL SPECIFICATION; MODEL BASED TESTING; SOFTWARE MODELS; TLS.

Среди всех современных протоколов, предназначенных для защиты передачи информации в открытых сетях, наиболее широко используется протокол Transport Layer Security (TLS)[1–4, 6]. Разработчики протокола TLS поставили перед собой цели обеспечить вычислительно эффективную криптографическую безопасность, совместимость реализаций и расширяемость протокола [4].

Протокол TLS применяется для защиты обменов между клиентом и сервером в различных прикладных сценариях: для защиты www-трафика (протокол HTTPS), отправки и получения электронной почты (с почтовыми протоколами SMTP, IMAP, POP3), при организации виртуальных защищенных сетей (OpenVPN), для защиты голосовой и видеосвязи (с SIP и основанными на нем приложениями, например VoIP), и многих др.

Как и у других популярных телекомму-

никационных протоколов, у протокола TLS в настоящее время широко используются более десятка различных реализаций. Несовместимость двух реализаций может привести к различным последствиям, начиная с невозможности установить соединение до разглашения конфиденциальных данных.

Наивный подход к решению задачи обеспечения совместимости заключается в том, чтобы каждую реализацию испытать на возможность взаимодействия с каждой. Очевидны недостатки этого подхода. Необходимо собрать все реализации TLS и провести попарные сценарии обменов данными. Так как для протоколов безопасности сценарии отказов не менее важны, чем сценарии нормальной передачи данных, необходимо разработать большое количество сценариев для каждой пары, причем может оказаться, что отдельные сценарии невозможно реализовать для тех или иных пар. Кроме того, такие испытания плохо поддаются автоматизации, многие сценарии приходится выполнять вручную.

В настоящей статье предлагается подход к оценке совместимости реализаций протокола TLS, основанный на оценке степени соответствия реализации стандарту протокола: если две реализации соответствуют спецификации протокола, то они совместимы. Необходимо отметить, что указанная гипотеза о совместимости соответствующих (conforming) реализаций принимается как постулат и в рамках данной работы не проверяется. Можно сказать, что мы доверяем разработчикам протокола TLS, которые заявили указанное свойство как одну из целей проектирования протокола [4].

Несмотря на популярность протокола TLS и ценность информации, которая защищается этим протоколом, открытых тестовых наборов для тестирования соответствия TLS, как это ни парадоксально, нет. Фактически, TLS является единственным из всех основных протоколов стека TCP/IP, для которого нет тестового набора для тестирования соответствия стандарту. Подробнее ситуация с тестами для TLS рассмотрена в следующем разделе.

Практическая значимость разработки такого тестового набора несомненна. С другой стороны, разработка тестового набора традиционным образом, как набора отдельных тестовых программ, каждая из которых проверяет определенное требование, является трудоемкой задачей. Прежде всего тесты должны проверять большое количество вариантов поведения протокола TLS в зависимости от выбора параметров TLS-соединения, таких как выбор криптографических алгоритмов, алгоритма сжатия, схемы построения общего секрета.

Альтернативой ручной разработке тестов как отдельных независимых программ является автоматизация построения тестов, основанная на модели протокола. Существует большое число методов, использующих модели протоколов для уменьшения ручного труда при создании тестов, эти методы получили совокупное название Model Based Testing или MBT, что переводится как «Тестирование, основанное на моделях» или «Тестирование с использованием моделей».

Цель данной работы – создание тестового набора для тестирования соответствия спецификации протоколу TLS, причем с самого начала было принято решение разрабатывать тестовый набор в рамках методологии тестирования, основанного на моделях.

В настоящей статье мы рассматриваем вопросы разработки модели протокола, тестов и результаты тестирования некоторых серверных реализаций TLS на соответствие стандарту протокола. Также в статье представлены направления дальнейших работ по развитию тестового набора для протокола TLS.

Тестирование TLS

В настоящее время используются более десяти семейств реализаций протокола TLS. Безусловно, все они тестировались в процессе разработки. Мы проанализировали тестовые наборы для открытых реализаций TLS и пришли к следующим выводам:

реализации тестируются по принципу «белого ящика», при этом существенно используются внутренние особенности реализации;

тестирование нацелено на выявление

ошибок в «недрах» реализации, соответственно, тестовые наборы не содержат специально выделенного подмножества тестов соответствия стандарту.

Тестовые наборы для проприетарных реализаций недоступны для использования вне компании-разработчика, поэтому мы ничего не можем сказать про их устройство.

Тестовый набор для самой популярной свободной реализации SSL/TLS – проект OpenSSL – содержит большой набор тестов для проверки различных аспектов поведения библиотеки OpenSSL. Однако эти тесты непереносимы на другие реализации, т. к. существенно используют особенности реализации OpenSSL.

С некоторой натяжкой можно сказать, что в OpenSSL частично реализовано тестирование совместимости: имеется возможность запустить приложение, реализующее клиентскую сторону TLS, в режиме тестирования. При запуске необходимо указать адрес и порт, на которых находится тестируемая реализация сервера. Результатом выполнения тестов будет вердикт о совместимости целевой реализации с OpenSSL.

Необходимо отметить, что при этом нет возможности как-либо соотнести результат работы тестов со стандартом. Результат говорит только о совместимости с OpenSSL, но ничего не говорит о том, какие пункты стандарта проверялись. При этом нельзя считать OpenSSL эталоном, т. к. не исследовалось, как данная реализация сама по себе соотносится со стандартом. Кроме того, очень важно тестировать поведение реализации в случае ошибок и отказов другой стороны. При тестировании на соответствие OpenSSL проверяется только поведение в нормальных сценариях.

Тестовый набор для реализации TLS в проекте открытой реализации языка Java OpenJDK насчитывает порядка 140 классов. Из них для тестирования соответствия стандарту предназначены только четыре, остальные проверяют различные компоненты реализации TLS в Java Secure Socket Extension.

В работе [7] описана верификация самого протокола TLS на исполнимой модели. Модель разрабатывалась на функциональном языке F# и представляет собой упрощенную реализацию TLS, которая, тем не менее, была способна взаимодействовать с полноценными (mainstream) реализациями TLS. Разработанная исполнимая модель была верифицирована на соответствие требованиям безопасности, другими словами, верифицировался сам протокол TLS. Авторы упомянули, что идет разработка соответствующего тестирующего программного обеспечения для протокола TLS на базе разработанной модели. Но подробности не приведены, других работ этих авторов на тему тестирования реализаций TLS обнаружить не удалось.

Подход, используемый в работе. В данной работе используется подход к автоматизации тестирования, основанный на технологии автоматизированного тестирования UniTESK [8], которая предоставляет средства автоматизации тестирования соответствия формальным спецификациям.

Требования, представленные в тексте стандарта, формализуются как набор булевских предикатов и императивных конструкций, заданных средствами исполнимого формализма. Для упрощения разработки и анализа модели в данной работе в качестве формализма использовалось расширение языка Java[9]. Указанный подход уже применялся нами в аналогичных проектах [10–14].

В UniTESK тест представляет собой конечный автомат. С каждым переходом автомата сопоставлено определенное тестовое воздействие. При выполнении перехода это воздействие подается на тестируемую реализацию, регистрируются реакции реализации и автоматически выносится вердикт о соответствии наблюдаемого поведения спецификации. Вердикт выносится на основании модели, которая проверяет, допустима ли зарегистрированная реакция в текущем состоянии тестируемой реализации. Так как тестирование проводится по схеме «черного ящика», т. е. внутреннее состояние реализации недоступно тесту, то модель также обновляет модельное состояние в соответствии с требованиями стандарта. Тем самым модель выступает как

эталонная реализация протокола.

Спецификация автомата теста на конкретном формализме называется *тестовым сценарием*. В данной работе для записи тестовых сценариев используется то же расширение языка Java [9], что и для разработки моделей. Тестовые воздействия, реализующие дуги в автомате теста, представляют собой методы на языке Java, в которых тестовые воздействия оказываются на объект модели протокола. Эти методы разрабатываются вручную, но обход осуществляется в полностью автоматическом режиме. Алгоритм обхода не зависит от протокола, тестируемой реализации или конкретного теста; он реализован как библиотечный компонент UniTESK, разработчикам тестов не требуется адаптировать его под целевой протокол.

Модель протокола TLS. Построение тестовых воздействий и верификация полученных ответов реализации опирается на модель протокола.

Стандарт протокола не требует, чтобы реализации протокола использовали в точности те структуры данных, которые введены в стандарте для описания протокола. Требуется только соответствие внешне наблюдаемого поведения требованиям. Однако в модели мы не гонимся за производительностью, поэтому набор внутренних переменных модели состоит из тех же самых переменных, что указаны в стандарте: параметры сессии TLS и по четыре состояния чтения-записи на сессию.

Помимо состояний в модель протокола входят спецификационные методы – исполнимые модели стимулов и реакций TLS-сервера. Каждый спецификационный метод содержит предусловие, в котором проверяется правильность структуры тестового сообщения и его своевременность, и на основании этого делается вывод о том, должен ли на него быть ответ, сообщение об ошибке или реализация должна его проигнорировать.

Параметрами спецификационных методов служат сообщения TLS в модельном представлении, т. е. объекты Java. Наборы полей соответствующих классов следуют структуре сообщений, описанных в стандарте TLS. В состав тестового набора входят кодеки (codecs) для преобразования сообщений из объектов в байтовые массивы для отправки в целевую систему и для обратного преобразования полученных сообщений.

В отличие от большинства телекоммуникационных протоколов, кодеки для TLS не являются контекстно-независимыми. Для корректного кодирования и декодирования сообщений кодек должен иметь доступ к текущему состоянию приема и отправки, прежде всего к ключам шифрования и криптографической контрольной суммы.

Другая особенность кодеков для TLS отражает тот факт, что протокол TLS двухуровневый. Сообщения служебных протоколов (например, TLS Handshake) и данные прикладных протоколов преобразуются в двоичный вид и фрагментируются, а полученные фрагменты обрабатываются протоколом TLS Records – упаковываются и шифруются. Фактически, именно в кодеках реализуется модель протокола TLS Records и его взаимодействие со служебными протоколами.

Тестовый набор. Тестовый стенд состоит из двух сетевых узлов: инструментального и целевого. На целевом узле функционирует тестируемая реализация. На инструментальном узле исполняется обход тестового автомата и верификация наблюдаемых реакций. Инструментальный и целевой узлы могут располагаться в разных сегментах сети.

Стимулами в разработанном тестовом наборе являются сообщения от инструментального узла, а реакциями – сообщения со стороны тестируемого узла. Основная часть требований спецификации TLS проверяется в модели протокола. Часть требований проверяется в декодерах сообщений – прежде всего требования к структуре сообщений и форматам данных.

Процесс тестирования начинается с создания TCP соединения к целевому узлу и инициализации начальных значений переменных модели. Все обмены сообщениями происходят по созданному соединению. Из модельного представления тестового сооб-

щения TLS строится реализационное, которое и отправляется в сеть. Сообщения TLS, полученные из установленного соединения, декодируются, и строятся модельные представления этих сообщений. Последовательность блоков данных в полученных сообщениях рассматривается как последовательность реакций целевой системы.

В модели полученные сообщения проверяются на соответствие требованиям спецификации. Проверка разделена на несколько стадий. Сначала проверяется допустимость такого сообщения от реализации и его своевременность, затем – структура самого сообщения (присутствующие поля и их значения должны соответствовать текущему обмену). После проверки всех требований результат передается тестовому сценарию, где в зависимости от плана сценария принимается решение о продолжении или завершении информационного обмена. В случае выявления нарушения требований принимается решение о критичности ошибки и возможности отправки следующих запросов.

Первыми по установленному TCP соединению передаются сообщения протокола TLS Handshake. В случае успешного завершения обменов по этому протоколу в модели создается новая TLS-сессия и устанавливаются параметры следующих состояний чтения-записи (pending read-write states). Кроме того, модель протокола поддерживает сокращенный вариант обмена, в котором используется уже существующая TLS-сессия.

При создании TLS-сессии клиент и сервер согласовывают различные параметры: применяемые алгоритмы сжатия, шифрования и цифровой подписи, схемы генерации разделяемых секретов, методы аутентификации. Для TLS Handshake разработано несколько тестовых сценариев, в которых перебираются различные сочетания криптографических алгоритмов и соответствующих им схем построения секретов. Тест для Handshake разделен на несколько автоматов в силу особенности алгоритма обхода в UniTESK – длина тестовой последовательности (а значит, и время выполнения теста) быстро растут по мере увеличения числа вариантов тестовых воздействий (дуг в автомате). Для получения приемлемого времени работы теста иногда имеет смысл разделить тестовый автомат на несколько, которые в совокупности дают то же покрытие требований, что и «большой» тест, но исполняются за меньшее время.

Протокол TLS Handshake содержит львиную долю функциональности протокола TLS. Функции остальных служебных протоколов значительно проще, поэтому для них не требуется большое количество сценариев. Протокол смены состояния и протокол передачи прикладных данных тестируются в одном сценарии, т. к. невозможно передавать данные через соединение TLS, не осуществив смену состояния. Протокол оповещений покрыт в текущей реализации тестового набора не полностью. Часть из ошибочных ситуаций, предусмотренных стандартом протокола, не тестируются.

Апробация. Тестовый набор апробирован на примере тестирования нескольких открытых серверных реализаций TLS. В роли TLS-сервера реализация не генерирует запросы, а лишь поддерживает информационный обмен, инициированный другим узлом. Стимулами являются тестовые сообщения от инструментального узла.

Для тестирования на соответствие стандарту были выбраны следующие реализации:

почтовый сервер Postfix 2.9.3 с открытой реализацией протокола TLS OpenSSL 1.0.1c под управлением операционной системы Red Hat Enterprise Linux 5.5;

реализация TLS в виртуальной машине Java 1.7.0_05 (Java Secure Socket Extension, JSSE);

тестовый сервер TLS интернет-ресурса https://www.mikestoolbox.net.

В первой реализации было выявлено восемь отклонений от спецификации TLS. В остальных – по три отклонения от спецификации. Несмотря на некоторые особенности и нарушения, реализации в целом соответствуют спецификации. Однако третья реализация (https://www.mikestoolbox.net) нарушает критичное требование проверки версии протокола TLS в сообщении

ClientKeyExchange с использованием алгоритма RSA. Подробный разбор части результатов тестирования можно найти в [14].

Направления дальнейшего развития

Разработанный тестовый набор для протокола TLS покрывает существенную часть функциональных требований сервера TLS, изложенных в стандартах протокола.

Мы видим два направления дальнейшего развития разработанного тестового набора.

С одной стороны, необходимо расширить тестовый набор тестами для клиента TLS. Основная трудность здесь заключается в том, что необходимо разработать средства для взаимодействия теста с тестируемым клиентом: необходимо выдавать стимулы для установления связи с сервером или отправки данных, а также для сбора реакций клиента. Если для реализации TLS в Java реализовать подобное средство не составит труда, то тестирование клиента в браузере требует разработать и установить расширения, которые позволят воздействовать на браузер извне.

Кроме того, для протоколов безопасности помимо тестирования соответствия есть еще одна важная задача: тестирование устойчивости. Протокол TLS используется для защиты общедоступных серверов, поэтому реализация TLS должна быть «готова» к приему произвольных сообщений.

Спецификация TLS определяет более 40 ошибочных ситуаций, однако их список не является исчерпывающим. Помимо тестирования на выделенные в спецификации особые случаи необходимо тестировать устойчивость к ошибкам в сообщениях, не указанным в спецификации протокола.

На практике тестирование устойчивости к некорректным сообщениям осуществляется преимущественно посредством случайного тестирования (fuzz testing, random testing). При случайном тестировании генерируются более-менее произвольные последовательности байтов в качестве целых пакетов или отдельных полей.

Построение случайных сообщений легко реализовать. Существует большое число исследовательских и коммерческих генераторов случайных сообщений, например IxDefend компании Ixia, Mu Dynamix одноименной компании, EMRET компании Microsoft, Dranzer организации CERT и т. д. Однако вероятность построения сообщения, которое способно преодолеть хотя бы часть механизмов валидации (сжатие, шифрование и контрольные суммы TLS Records, согласованные значения полей сообщений TLS Handshake) настолько мала, что случайное тестирование, дающее сколько-нибудь значимую долю покрытия кода реализации, займет неприемлемо длительное время.

Мы предполагаем дополнить имеющийся тестовый набор средствами мутационного тестирования (mutation testing), при котором ошибочные сообщения строятся как искажения корректных сообщений. При мутационном тестировании необходимо сначала построить корректное сообщение, а затем трансформировать («мутировать») его в некорректное. При мутационном тестировании число тестовых воздействий существенно меньше, чем при случайном тестировании, а покрытие реализации выше. Применение мутационного тестирования нацелено на выявление нарушений функционирования из-за переполнений буфера, разрушения стека, индексирования за пределами массива.

Применение мутационного тестирования к TLS связано с нетривиальной задачей построения корректного сообщения, т. к. содержимое сообщения связано с внутренним состоянием протокола: сообщение должно быть корректно зашифровано и подписано, в противном случае оно будет отброшено на ранних этапах обработки. Мы предполагаем использовать мутации совместно с тестированием на основе моделей: благодаря наличию модели можно целенаправленно привести реализацию в необходимое состояние и подготовить разумные данные для корректного сообщения.

Тестирование соответствия спецификации или стандарту является в настоящее время основным механизмом обеспечения совместимости между различными реализациями: если две реализации корректно

реализуют протокол, то они обязательно смогут взаимодействовать. Такой подход используется для протоколов всех уровней в стеке TCP/IP: существуют тесты для канального уровня (Ethernet), сетевого (IPv4 и IPv6), транспортного (TCP и UDP), многих протоколов прикладного уровня (DNS, DHCP, SMTP/POP/IMAP и т. п.).

Однако в сфере TLS сложилась парадоксальная ситуация. Протокол TLS используется на сотнях тысяч серверов по всему миру, сотни миллионов людей пользуются им ежедневно для защиты соединений с веб-серверами, отправки и получения электронной почты, существует более десятка реализаций, но при этом нет ни одного достаточно содержательного тестового набора, с помощью которого можно было бы удостовериться в соответствии реализации стандарту TLS.

В данной статье представлен подход к разработке тестового набора для тестирования соответствия реализации протокола TLS на соответствие стандарту. Подход использует технологию UniTESK к автоматизации тестирования: тестовая последовательность строится динамически как обход некоторого автомата теста, для построения тестовых воздействий и вынесения вердикта о корректности наблюдаемого поведения реализации используется модель протокола.

Разработан тестовый набор, покрывающий большинство требований, изложенных в стандарте протокола TLS [4]. Тестовый набор апробирован на трех общедоступных реализациях TLS/SSL, во всех реализациях обнаружены отклонения от спецификации, в одном случае нарушение расценивается как критическое.

В качестве дальнейшего направления работ рассматривается расширение тестового набора тестами на некорректные сообщения. Такие тесты предполагается автоматизировать с использованием методов мутационного тестирования.

Работа выполнена при финансовой поддержке Российского фонда фундаментальных исследований, проект 13-07-00869.

СПИСОК ЛИТЕРАТУРЫ

1. **Freier A., Karlton P., Kocher P.** The Secure Sockets Layer (SSL) Protocol Version 3.0. Aug. 2011.

2. **Dierks T., Allen C.** IETF RFC 2246 // The TLS Protocol Version 1.0, Jan. 1999.

3. **Dierks T., Rescorla E.** IETF RFC 4346 // The Transport Layer Security (TLS) Protocol Version 1.1. Apr. 2006.

4. **Dierks T. Rescorla E.** IETF RFC 5246 // The Transport Layer Security (TLS) Protocol Version 1.2. Aug. 2008.

5. **Turner S., Polk T.** IETF RFC 6176 // Prohibiting Secure Sockets Layer (SSL) Version 2.0. March 2011.

6. Проект SSL Pulse консорциума Trustworthy Internet Movement [электронный ресурс] / URL: https://www.trustworthyinternet.org/ssl-pulse/

7. **Bhargavan K., Fournet C., Corin R., Zalinescu E.** Cryptographically verified implementations for TLS // Proc. of the 15th ACM Conf. on Computer and Communications Security. ACM New York, 2008.

8. **Bourdonov I., Kossatchev A., Kuliamin V., Petrenko A.** UniTesK Test Suite Architecture // Proc. of FME. Springer-Verlag, 2002. LNCS 2391. Pp. 77–88.

9. **Bourdonov I.B., Demakov A.V., Jarov A.A., Kossatchev A.S., Kuliamin V.V., Petrenko A.K., Zelenov S.V.** Java Specification Extension for Automated Test Development // Proc. of PSI-2001. Novosibirsk. Springer-Verlag, 2001. LNCS 2244. Pp. 301–307.

10. **Пакулин Н.В.** Формализация стандартов и тестовых наборов протоколов Интернета. Дис. ... канд. физ.-мат. наук. М., 2006.

11. **Пакулин Н.В., Хорошилов А.В.** Разработка формальных моделей и тестирование соответствия для систем с асинхронными интерфейсами и телекоммуникационных протоколов // Программирование. 2007. № 5. С. 1–29.

12. **Никешин А.В., Пакулин Н.В., Шнитман В.З.** Разработка тестового набора для верификации реализаций протокола безопасности IPsec v2 // Труды Института системного программирования РАН. 2010. Т. 18. С. 151–182.

13. **Никешин А.В., Пакулин Н.В., Шнитман В.З.** Верификация функций безопасности протокола IPsec v2 // Программирование. 2011. № 1. С. 36–56.

14. **Никешин А.В., Пакулин Н.В., Шнитман В.З.** Разработка тестового набора для верификации реализаций протокола безопасности TLS // Труды Института системного программирования РАН. 2012. Т. 23. С. 387–404.

REFERENCES

1. **Freier A., Karlton P., Kocher P.** *The Secure Sockets Layer (SSL) Protocol Version 3.0.* Aug. 2011.

2. **Dierks, T., Allen C.** IETF RFC 2246. *The TLS Protocol Version 1.0.* Jan. 1999.

3. **Dierks, T., Rescorla E.** IETF RFC 4346. *The Transport Layer Security (TLS) Protocol Version 1.1.* Apr. 2006.

4. **Dierks, T. and E. Rescorla** IETF RFC 5246. *The Transport Layer Security (TLS) Protocol Version 1.2.* Aug. 2008.

5. **Turner S., Polk T.** IETF RFC 6176. *Prohibiting Secure Sockets Layer (SSL) Version 2.0.* March 2011.

6. *Project «SSL Pulse» of Trustworthy Internet Movement Consortium*. Available: https://www.trustworthyinternet.org/ssl-pulse/

7. **Bhargavan K., Fournet C., Corin R., Zalinescu E.** Cryptographically verified implementations for TLS, *Proceedings of the 15th ACM Conference on Computer and Communications Security*, ACM New York, USA, 2008.

8. **Bourdonov I., Kossatchev A., Kuliamin V., Petrenko A.** UniTesK Test Suite Architecture, *Proceedings of FME*. Springer-Verlag, 2002. LNCS 2391, Pp. 77–88.

9. **Bourdonov I.B., Demakov A.V., Jarov A.A., Kossatchev A.S., Kuliamin V.V., Petrenko A.K., Zelenov S.V.** Java Specification Extension for Automated Test Development. *Proceedings of PSI-2001*, Novosibirsk, Russia, Springer-Verlag, 2001, LNCS 2244, Pp. 301–307.

10. **Pakulin N.V.** *Formalizatsiya standartov i testovyih naborov protokolov Interneta* [*Formalisation of specification and test suite for Internet protocols*]. Dis. ... kand. fiz.-mat. nauk [PhD thesis], Moscow, 2006. (rus)

11. **Pakulin N.V., Khoroshilov A.V.** Razrabotka formalnyih modeley i testirovanie sootvetstviya dlya sistem s asinhronnyimi interfeysami i telekommunikatsionnyih protokolov [Development of formal models and conformance testing for systems with asynchronous interfaces and telecommunications protocols], *Programmirovanie* [*Programming and Computing Software*], 2007, Vol. 33, Iss. 6, Pp. 316–335. (rus)

12. **Nikeshin A.V., Pakulin N.V., Shnitman V.Z.** Razrabotka testovogo nabora dlya verifikatsii realizatsiy protokola bezopasnosti IPsec v2 [Development of a test suite for the verification of implementations of the IPsec v2 security protocol], *Trudyi Instituta sistemnogo programmirovaniya RAN* [*Proceedings of IPS RAS*], 2010, Vol. 18, Pp. 151–182. (rus)

13. **Nikeshin A.V., Pakulin N.V., Shnitman V.Z.** Verifikatsiya funktsiy bezopasnosti protokola IPsec v2 [Development of a test suite for the verification of implementations of the IPsec v2 security protocol], *Programmirovanie* [*Programming and Computing Software*], 2011, Vol. 37, Iss. 1, Pp. 36–56. (rus)

14. **Nikeshin A.V., Pakulin N.V., Shnitman V.Z.** Razrabotka testovogo nabora dlya verifikatsii realizatsiy protokola bezopasnosti TLS [Development of a test suite for the verification of implementations of the TLS security protocol], *Trudyi Instituta sistemnogo programmirovaniya RAN* [*Proceedings of IPS RAS*], 2012, Vol. 23, Pp. 387–404. (rus)

НИКЕШИН Алексей Вячеславович – *младший научный сотрудник Института системного программирования РАН.*
109004, Россия, Москва, ул. Александра Солженицына, д. 25.
E-mail: alexn@ispras.ru

NIKESHIN, Alexey V. *Institute for System Programming of the Russian Academy of Sciences.*
195251, Alexander Solzhenitsyn Str. 25, Moscow, Russia.
E-mail: alexn@ispras.ru

ПАКУЛИН Николай Витальевич – *старший научный сотрудник Института системного программирования РАН, кандидат физико-математических наук.*
109004, Россия, Москва, ул. Александра Солженицына, д. 25.
E-mail: npak@ispras.ru

PAKULIN, Nikolay V. *Institute for System Programming of the Russian Academy of Sciences.*
195251, Alexander Solzhenitsyn Str. 25, Moscow, Russia.
E-mail: npak@ispras.ru

ШНИТМАН Виктор Зиновьевич – *заместитель директора Института системного программирования РАН, доктор физико-математических наук, профессор.*
109004, Россия, Москва, ул. Александра Солженицына, д. 25.
E-mail: vzs@ispras.ru

SNITMAN, Victor Z. *Institute for System Programming of the Russian Academy of Sciences.*
195251, Alexander Solzhenitsyn Str. 25, Moscow, Russia.
E-mail: vzs@ispras.ru

УДК 004.05

Н.В. Пакулин

ДИНАМИЧЕСКАЯ ВЕРИФИКАЦИЯ ГИБРИДНЫХ СИСТЕМ

N.V. Pakulin

DYNAMIC VERIFICATION OF HYBRID SYSTEMS

В последние десять лет практически повсеместно аналоговые контуры управления вытесняются цифровыми. В настоящее время ведутся работы по созданию цифровых систем управления критическими системами: «интеллектуальные подстанции» в электроэнергетике, «интегрированная модульная авионика» в авиации, «интеллектуальные фабрики» в промышленности и т. д.

Внедрение крупномасштабных систем с цифровыми каналами управления, отказы в которых могут повлечь тяжелые последствия, вплоть до катастрофических, требует создания новых методов анализа и верификации, в частности моделирования таких систем и верификации корректности и надежности гибридных систем на моделях.

Рассмотрены методы верификации моделей гибридных систем и предложена архитектура тестового стенда для проведения динамической верификации гибридной системы.

ГИБРИДНЫЕ СИСТЕМЫ; КИБЕРФИЗИЧЕСКИЕ СИСТЕМЫ; ВЕРИФИКАЦИЯ НА МОДЕЛЯХ; ДИНАМИЧЕСКАЯ ВЕРИФИКАЦИЯ; ТЕСТИРОВАНИЕ; UNITESK.

In the recent decade we face aggressive replacement of analogue control loops with digital ones. There are ongoing projects for digital control even for critical systems: «Smart Grids» in power industry, «Integrated Modular Avionics» in aerospace, «Smart Fabrics» in manufacturing, etc. Introduction of large scale digital control channels raises the risks of faults that might result in heavy losses. Those risks call for new methods of analysis and verification, including modeling hybrid systems and model-based verification. The paper overviews a number of existing approaches to verification of hybrid systems and introduces an architecture of a test bed for dynamic verification of models of hybrid systems.

HYBRYD SYSTEMS; CYBER-PHYSICAL SYSTEMS; MODEL-BASED VERIFICATION; DYNAMIC VERIFICATION; TESTING; UNITESK.

В последние десять лет практически повсеместно аналоговые контуры управления вытесняются цифровыми: цифровые каналы связывают сенсоры и актуаторы с ЭВМ, на которых выполняются управляющие программы. Получившиеся системы, состоящие из дискретных и непрерывных процессов, получили название *гибридных*. Гибридные системы – это сложные взаимодействующие физические процессы, которые в реальном времени управляются сетью специализированных ЭВМ. Принципиальное отличие гибридных систем от традиционных встроенных систем заключается в масштабах: и объект управления, и управляющая система представляют собой сложные распределенные системы.

По всему миру ведутся работы по созданию цифровых систем управления критическими системами: в авиации разрабатывают и внедряют подход интегрированной модульной авионики (Integrated Modular Avionics – IMA), в электроэнергетике разрабатывают «интеллектуальные подстанции» (Smart Substation), автоматизация промышленных производств приняла новые формы «интеллектуальных фабрик» (Smart Manufacturing) и т. д.

Верификация на этапе готовых изделий и систем, безусловно, необходима, но, принимая во внимание стоимость и возможную тяжесть последствий отказов и нештатного поведения системы, верификацию необходимо вводить в цикл разработки гибридной системы как можно раньше, как минимум, на этапе проектирования.

Проектирование гибридных систем нередко включает в себя разработку модели

системы. Используются различные подходы и инструменты к моделированию, но всех их объединяет тот факт, что разрабатываются исполнимые модели, на которых можно исследовать эволюцию системы. В данной статье именно эти модели рассматриваются как объекты верификации.

В статье изучаются различные подходы к верификации моделей гибридных систем. Один из наиболее распространенных подходов к моделированию сложных систем – моделирование т. н. гибридными автоматами, представляющими собой переосмысление традиционных конечных автоматов на случай непрерывных воздействий.

В случае гибридной системы подсистемы могут представлять собой дискретные устройства, аналоговые устройства, физические процессы и явления, и гибридные подсистемы, поэтому в общем случае можно считать, что все компоненты гибридной системы являются гибридными подсистемами.

Несмотря на то что для гибридных автоматов разработаны различные методы верификации, нельзя сказать, что задача верификации моделей гибридных систем полностью решена. С одной стороны, композиция гибридных компонентов в систему приводит к возникновению автоматов такой сложности, перед которыми «пасуют» разработанные методы верификации. С другой, – на практике встречаются модели, которые не укладываются в концепцию гибридного автомата: численные модели, разработанные в специализированных математических пакетах, а то и вовсе программы на языках программирования общего назначения (Fortran, C++ и т. п.).

Для верификации таких моделей в статье предлагается использовать тестирование. В противовес статической верификации гибридных автоматов, при которой автомат не исполняется, тестирование требует исполнения модели, поэтому в данной статье такой подход назван *динамической верификацией*.

Динамическая верификация должна проводиться в специализированном, контролируемом окружении. Такое окружение называется *тестовым стендом* (test harness). Задача стенда заключается в том, чтобы построить систему взаимодействующих компонентов, каждый из которых является некоей гибридной системой. Построение такого стенда позволяет проводить верификацию сложных систем на ранних этапах разработки, до того как подсистемы воплощены «в металле». В частности, такой стенд позволит проводить анализ вариантов реализации системы в целом и ее отдельных подсистем, проверять реализуемость системных требований и достаточность выделяемых ресурсов.

Статья представляет проект по разработке архитектуры тестового стенда и методов тестирования для проведения динамической верификации *моделей* гибридных систем. В отличие от существующих подходов к тестированию гибридных систем в данной работе гибридная система представлена не «железом», а моделями компонентов.

Необходимо отметить, что при верификации сложной гибридной системы может оказаться, что не для всех компонентов системы существуют модели. Часть компонентов может быть представлена готовыми изделиями. Это означает, что архитектура тестового стенда для динамической верификации должна поддерживать включение немодельных компонентов наряду с модельными.

Методы моделирования гибридных систем

Методы моделирования гибридных систем можно условно разделить на два класса: методы, используемые в научной среде для разработки средств анализа и верификации гибридных систем, и методы, которые используются в инженерной практике для численного моделирования систем и взаимодействия между ними.

Гибридные автоматы. В настоящее время в научной среде наиболее широко используются методы, основанные на концепции гибридного автомата [3–5] или производные от него. В данной статье мы не будем приводить строгое определение гибридного автомата из-за его громоздкости, ограничимся качественным описанием; читатель может найти математическое определение в указанных статьях.

Говоря неформально, гибридный автомат представляет собой расширенный конечный автомат, в состояниях которого определены режимы, связанные с конечным набором дискретных и вещественных переменных. С каждым режимом определены ограничения, задающие эволюцию системы, а дуги автомата задают дискретные переходы.

В качестве примера рассмотрим термостат, обеспечивающий поддержание температуры в интервале 68–70 градусов. Эта система моделируется автоматом с двумя состояниями ВКЛ и ОТКЛ и одной вещественной переменной T, представляющей температуру. Изменение температуры в режиме ВКЛ может определяться, например, дифференциальным уравнением $dT/dt = k(100 - T)$, где k — константа, параметр модели. Допускается также недетерминированная спецификация режима: например, вместо конкретного дифференциального уравнения режим может задаваться неравенством $k_1 < dT/dt < k_2$, то есть ограничение на динамику вещественной переменной. Исходя из требований к термостату формулируется условие дискретного перехода: при $T > 70$ автомат переходит в состояние ОТКЛ. В этом состоянии динамика температуры описывается отдельным режимом, который может быть никак не связан с режимом в состоянии ВКЛ. Переход из состояния ОТКЛ в состояние ВКЛ осуществляется по достижению ограничения $T < 68$.

Определение гибридного автомата играет в теории гибридных систем ту же базовую роль, какую играют конечные автоматы в теории дискретных систем. На базе гибридных автоматов были разработаны подходы к моделированию составных систем и нотации для записи таких моделей. Можно указать методы иерархического моделирования гибридного поведения Shift [6] и Ptolemy [7], гибридные автоматы ввода-вывода [8], гибридные модули [9], подход к моделированию параллельных составных гибридных систем Charon [10]. Кроме того, развиваются новые направления в математической логике для описания непрерывных процессов, получившие название *дифференциальной динамической логики* (см., например, [11]).

Гибридные автоматы служат основой для различных методов автоматизированного анализа и верификации моделей гибридных систем (подробнее см. далее). По этой причине в следующем пункте обсуждается вопрос извлечения гибридного автомата из численной модели.

Методы численного моделирования. *Simulink/Stateflow*. Мировым лидером среди производителей программных продуктов, применяемых в практических задачах моделирования гибридных систем, является компания MathWorks. Ее продукты Simulink [12] и Stateflow [13] являются стандартом де-факто для разработки моделей гибридных систем.

Simulink представляет собой среду графического программирования моделей динамических систем. Блоки моделей могут соответствовать отдельным физическим процессам, в частности поддерживается композиция моделей в модели более высокого уровня, а также математическим преобразованиям ($y = f(x)$) и операторам (прежде всего, оператору дифференцирования для графического задания обыкновенных дифференциальных уравнений). Важной отличительной чертой Simulink является интеграция с пакетом программ численного решения уравнений Matlab. Благодаря этому в Simulink можно импортировать математические модели сложных физических процессов, описанные на Matlab, и использовать мощные средства численного решения дифференциальных уравнений. На базе Simulink поставляются продукты для моделирования отдельных классов динамических систем, таких как механические системы или электротехнические. Simulink в чистом виде не поддерживает моделирование гибридных систем, т. к. в нем нет удобных средств описания дискретных, автоматных модулей. Для решения задачи моделирования дискретной составляющей гибридной системы используется продукт Stateflow. Эта программа предоставляет средства для описания конечных автоматов и поддерживает интеграцию с Simulink для спецификации непрерывной части модели.

В качестве внутреннего представления моделей Stateflow/Simulink используется язык Matlab.

Существуют аналоги Simulink/Stateflow, разработанные по модели открытого кода (open source): пакет Scicos [14], основанный на Scilab [15], открытой альтернативе языка Matlab.

Во второй половине 1990-х гг. в Европе был создан новый язык для описания динамических систем, получивший название *Modelica* [16]. Этот язык объединил лучшие концепции из представленных в то время на рынке европейских пакетов численного моделирования.

Modelica представляет собой открытый (non-proprietary), объектно-ориентированный язык, в котором эволюция переменных объекта описываются посредством уравнений. Язык позволяет специфицировать линейные, алгебраические и обыкновенные дифференциальные уравнения. На языке Modelica разработано множество моделей, поддерживается открытый репозиторий моделей Modelica Standard Library, содержащий порядка 1280 моделей компонентов и 910 функций из различных предметных областей. Язык Modelica предназначен для свободного обмена моделями между различными пакетами численного моделирования. В настоящее время нотация Modelica поддерживается более чем двумя десятками коммерческих и открытых программных продуктов [17].

Несмотря на поддержку автоматных концепций на уровне модели, в настоящее время отсутствуют средства преобразования произвольных моделей Simulink/Stateflow в гибридные автоматы в силу богатства выразительных средств языка Matlab и несоответствия концептуального представления автоматов в Stateflow дискретным переходам в гибридных автоматах [18,19].

Modelica не содержит отдельных средств для описания дискретных компонентов гибридной системы. Эта задача решается введением булевских или целочисленных переменных в модель. Задача преобразования модели на языке Modelica в гибридный автомат к настоящему времени не решена.

Доменно-специфичные методы моделирования. Наследованные модели. Несмотря на существование мощных средств численного моделирования общего назначения, таких как Simulink и Matlab, широко встречаются модели сложных систем или процессов, представленные как программы на языке программирования общего назначения (Fortran, C, Ada). Важно отметить, что методы разработки таких программ, методы моделирования физических процессов очень сильно зависят от предметной области, в силу чего данные модели называются *доменно-специфичными*.

Кроме того, среди моделей на языках программирования могут встречаться коды, которым уже не один десяток лет, разработанные на Fortran или Cobol, исполнявшиеся в свое время на мэйнфреймах. Такие модели можно назвать «унаследованными» (legacy) моделями.

Для таких моделей задача построения аналогичного гибридного автомата может быть решена только вручную путем анализа и обратной инженерии программного модуля. Общий подход для анализа таких моделей не существует.

Методы верификации гибридных систем

Задача верификации гибридных систем имеет ряд существенных отличий от задачи верификации программного обеспечения общего назначения.

Прежде всего, при верификации необходимо учитывать непрерывный характер физических процессов, протекающих в системе, что требует разработки специальных методов анализа по сравнению с традиционным ПО, в котором внешние события поступают преимущественно в виде дискретных событий.

В силу того что гибридные системы в подавляющем большинстве представляют собой системы «объект управления – система управления», для них актуальной является задача верификации безопасности (safety verification): верификация того факта, что при функционировании системы не может случиться «что-нибудь плохое».

Верификация гибридных автоматов. Исследования в области верификации гибридных автоматов идут преимущественно в об-

ласти верификации безопасности.

При верификации безопасности гибридной системы M задаются множество начальных состояний I и множество «безопасных» состояний S (включая ограничения на переменные состояния, в т. ч. вещественные). Задача заключается в том, чтобы установить, что при всех возможных эволюциях системы, начинающихся в состояниях из I, система остается в «безопасных» состояниях, либо построить пример достижения «небезопасного» состояния.

Необходимо отметить, что данная задача в общем случае алгоритмически неразрешима [4, 20]. Однако для простых случаев временны́х автоматов (timed state machine) и линейных гибридных автоматов существуют эффективные алгоритмы [21].

Один из подходов к верификации безопасности гибридных систем основывается на методе проверки на моделях (model checking). В рамках этого подхода итеративно вычисляются состояния, достижимые из текущего множества достижимых состояний, и проверяется их безопасность. Ограничения на непрерывные переменные задают криволинейный объем в многомерном пространстве. Так как анализ криволинейных поверхностей в общем случае сложен, большинство инструментов анализа используют линейные неравенства для задания ограничений на безопасные состояния. В этом случае множества допустимых значений переменных ограничены многогранниками.

Сложность методов, предложенных к настоящему времени, растет экспоненциально с увеличением числа переменных. В настоящее время наиболее совершенным инструментом этого класса считается SpaceEx [22], в котором удалось провести верификацию системы управления вертолетом, в модели которой было 28 вещественных переменных, а динамика описывалась линейными дифференциальными уравнениями [22].

Логический подход к верификации гибридных систем заключается в доказательстве теорем о том, что некоторое свойство φ выполняется в начальных состояниях I и во всех состояниях, достижимых из них. Основная проблема в данной области заключается в том, что современные инструменты решения булевских уравнений (SMT solvers) либо вовсе не поддерживают арифметику с вещественными числами, либо поддержка сильно ограничена (например, числа с фиксированной точкой конечной точности).

В качестве примера инструмента, позволяющего доказывать безопасность гибридных систем, можно указать KeYmaera [11, 23].

Интенсивные исследования ведутся в области абстракции, когда гибридный автомат сводится к более простому [24, 25]: ограничения заменяются на линейные неравенства, дифференциальные уравнения $dx/dt = f(x)$ заменяются на дифференциальные неравенства $l \le dx/dt \le m$, где I и m – нижняя и верхняя граница f. Для повышения степени приближения поведения упрощенного автомата к поведению исходного автомата, состояния и режимы в абстрактном разбивают на более мелкие. Цель абстрагирования гибридного автомата: получить «похожий» [26] линейный гибридный автомат, для которого есть эффективные средства анализа.

Тестирование с использованием моделей. Тестирование с использованием моделей (model-based testing) объединяет методы активного изучения поведения целевой системы (тестирования), при котором используются модели тестируемой системы. Модели могут использоваться для генерации тестовых данных, оценки корректности наблюдаемого поведения, оценки полноты тестирования.

Разработаны подходы, использующие гибридные автоматы для тестирования. В частности, инструмент CHARON [27] извлекает из описания системы в виде гибридного автомата тестовые данные и генерирует код для проведения тестирования. Требования безопасности задаются в виде предикатов темпоральной логики. Для темпоральных формул строится монитор, который в динамике верифицирует наблюдаемые трассы. Инструмент использовался как для верификации собственно модели, так и для реализации – робота Sony AIBO. Основной

недостаток CHARON и подобных ему инструментов заключается в том, что модель должна быть разработана в определенном формализме. Инструменты не интероперабельны: нет возможности обмениваться моделями для проведения верификации, всякий раз необходимо переписывать модель под конкретный инструмент.

Верификация численных моделей. Системы численного моделирования Simulink/Stateflow не содержат специализированных средств тестирования; методики тестирования численных моделей также не разработаны. В документации на Simulink/Stateflow предлагается использовать компонент SignalBuilder для генерации входных сигналов тестируемого компонента. Однако при этом отсутствуют средства определения корректности наблюдаемого поведения целевой системы (оракул) и средства оценки полноты достигнутого покрытия тестовых ситуаций.

В компании Rockwell Collins совместно с университетом Иллинойса разработан набор инструментов [28], которые преобразуют модели Simulink/Stateflow в модели на языке Lustre [29], для которых существуют эффективные методы статического анализа. Разработанная цепочка инструментов использовалась в проектах по верификации бортового авиационного оборудования.

В системе инструментов, основанных на Modelica, нет развитых средств верификации и системного тестирования [30].

Европейский институт стандартизации телекоммуникаций (ETSI) ведет работу по стандартизации расширения языка TTCN3, предназначенного для спецификации тестов для систем с непрерывными (continuous) входами и выходами [31]. Расширение позволяет записывать темпоральные предикаты на ожидаемое поведение системы с непрерывными интерфейсами, а также задавать генераторы входных данных по сложным законам.

Системное тестирование. Помимо анализа отдельных модулей и компонентов для гибридных систем особое значение имеет анализ требований безопасности на системном уровне. На текущем уровне развития методов статического анализа модели, полученные в результате композиции моделей отдельных компонентов, не поддаются анализу из-за больших размеров. Единственным практическим средством исследования гибридных систем остается системное тестирование. В системном тестировании требования к системе формулируются в виде некоторых предельных значений параметров (например, значения ускорений при маневрах) или средних значений (средний расход топлива).

Проведение системных тестов для больших гибридных систем требует построения специализированных тестовых стендов. Пожалуй, наибольшее распространение этот подход получил в авиастроении, где цена отказа системы чрезвычайно велика. Крупные испытательные стенды для бортовой электроники и оборудования получили даже собственное имя – «Железные птицы» (Iron Birds). В рамках такого стенда объединяют реальное оборудование, управляющие программы, модели окружения (атмосфера, система позиционирования, каналы связи с наземными службами и т. д.) Часть реализаций авионических компонентов может быть заменена на модели. Общая динамика полета самолета описывается аэродинамической моделью самолета, которая служит моделью системного уровня.

Аналогичные проекты по системному тестированию реализуются также в других предметных областях: электроэнергетике, железнодорожном транспорте, космонавтике.

Использование программно-аппаратных стендов для верификации гибридных систем. Использование реалистичных программно-аппаратных моделей оборудования и коммуникационных сетей открывает следующие уникальные возможности в плане анализа надежности и верификации гибридных систем:

1. «Объем» верификации компонентов гибридной системы в условиях модельных испытаний существенно выше, чем при верификации на реальных объектах, поскольку имеются специальные технические возможности для поддержки дополнительных точек воздействия и точек контроля испытываемых систем.

2. Использование моделей оборудования и сетей позволяет гибко настраивать конфигурацию целевой системы для различных сценариев использования гибридной системы, состава и настроек оборудования, особенностей физических процессов. Настраиваемые модели позволяют верифицировать различные конфигурации объектов в разнообразных режимах функционирования, включая отказы и прочие нештатные ситуации.

3. Динамическая верификация модели гибридной системы позволяет проводить испытания отказоустойчивости и надежности систем в экстремальных режимах, при которых ошибки этих систем могут привести к катастрофическим последствиям. Подобные испытания невозможны на реальном оборудовании в силу разрушительности возможных последствий испытаний.

Без верификации на адекватных моделях анализ функциональной корректности и надежности гибридных систем трудно признать достоверным. Такой анализ может давать некорректные или плохо интерпретируемые результаты, т. к. отсутствует возможность испытывать целевую систему в условиях, приближенных к реальным. Для получения достоверных выводов о функциональной корректности и информационной безопасности необходимы реалистичные информационные потоки внутри системы, эксплуатационные настройки систем управления, работающие «по-боевому» средства обнаружения аварий и т. д.

Важность проведения всесторонних испытаний гибридных систем хорошо осознают за рубежом. С 2004 г. в США функционирует государственный стенд для испытаний систем управления объектами электроэнергетики (National SCADA Test Bed) [1], созданный по инициативе Министерства энергетики США. Стенд активно используется для оценки надежности и защищенности программных комплексов и оборудования различных производителей: Siemens, ABB и др. Однако испытания проводятся на реальном электротехническом оборудовании подстанций тестовой сети Национальной лаборатории Айдахо. С одной стороны, наличие оборудования позволяет обойти проблему адекватности моделей реальным процессам. С другой стороны, на оборудовании проводится только определенный спектр испытаний, «вблизи» от штатных режимов функционирования оборудования, не содержащих риск разрушения оборудования.

В 2008–2010 гг. в США велись работы по созданию виртуального стенда для силовой электротехнической аппаратуры [2], но опубликованы только промежуточные результаты, окончательные результаты в открытых источниках не представлены. Кроме того, в проекте развивался подход ограниченной виртуализации стенда – фактически, только программы управления и сетевые обмены выполнялись в виртуальном окружении, а объекты управления были реальными физическими приборами.

Стенды, имитирующие реальное оборудование, широко используются в аэрокосмической отрасли. В качестве примеров можно указать проекты по созданию «космических челноков» в США и КК «Буран» в СССР. Модели бортовых систем разрабатываются ведущими авиастроительными компаниями, включая Локхид-Мартин, Боинг и Аэрбас. В России в настоящее время монтируются стенды магистрального самолета нового поколения МС-21. В упомянутых стендах широко используются модели физических процессов. В частности, в ГосНИИАС разработана архитектура смешанного программно-аппаратного стенда, в котором реальное бортовое оборудование получает данные от виртуальной модели самолета через цифровые каналы связи с датчиками.

Несмотря на наличие программно-аппаратных стендов и прогресс в этой области в последние годы, недостаточно разработаны методики проведения испытаний моделей гибридных систем: критерии покрытия тестовых ситуаций, методики перебора тестовых воздействий, критерии оценки корректности поведения системы (тестовые оракулы). Особенность тестирования гибридных систем по сравнению с традиционным тестированием программного обеспечения заключается в том, что в гибридных системах значительно многооб-

разнее представлены входные воздействия – помимо штатных сигналов, которые сами по себе представлены непрерывными физическими величинами, гибридные системы подвержены шумам в датчиках, отказам телеметрии, резким (в течение миллисекунд) изменениям окружения (возгорание, взрыв, механическое разрушение и т. п.).

Особенности тестирования гибридных систем

Гибридные системы характерны следующими особенностями:

- Системы тесно связаны функционально. Большинство подсистем нельзя изолировать друг от друга, во время функционирования они взаимодействуют друг с другом прямо или косвенно. Это взаимодействие является существенным для корректного функционирования крупномасштабных систем, связанных посредством непрерывных физических процессов.
- Значительная часть подсистем основана на аналоговых физических процессах.
- Существуют требования промышленной безопасности. Условно говоря, «чтобы все работало и не взрывалось».
- У многих подсистем есть четко выделенные режимы, в которых алгоритмы могут существенно отличаться друг от друга.

Указанные особенности гибридных систем влияют на процесс системного тестирования:

- Для тестирования гибридных систем необходимо моделировать окружение. Изолированное тестирование отдельных модулей выполняется разработчиком, при системном тестировании интересует согласованность работы отдельных систем и их влияние на поведение системы в целом. Особенность заключается в том, что окружение, скорее всего, представляет собой набор физических процессов или явлений.
- Системное тестирование проверяет выполнение интегральных характеристик функционирования системы. Соответственно, требуется верифицировать интегральное поведение совокупности систем (выдерживание курсового угла, скорости относительно воздуха и т. д.), поэтому модель поведения отдельной системы может быть не востребована при ее тестировании.
- Внешне наблюдаемые параметры меняются непрерывно, требуются средства для спецификации ограничений на непрерывные параметры.
- Для тестирования необходим стенд, который предоставляет реалистичное окружение системы, включая модели или реализации связанных систем, модель системного уровня в целом, модель окружения и т. д.
- Существенный интерес представляет тестирование целевых систем в условиях отказов или ошибочных данных от связанных систем.

Динамическая верификация гибридных систем

Ручной перебор возможных сценариев воздействия на целевую систему в случае гибридных систем еще более трудоемкий, чем при тестировании обычных (дискретных) программных систем. Помимо дискретных входных событий такая система подвергается воздействию непрерывных физических процессов через сигналы от датчиков, а кроме дискретных выходных сообщений действует на физические процессы посредством актуаторов.

В данной работе предлагается для автоматизации тестирования гибридных систем применять подход тестирования с использованием моделей.

Тестирование с использованием моделей (model based testing) в широком смысле охватывает любые способы использования моделей для автоматизации тестирования программных и/или аппаратных систем.

Среди задач тестирования, подлежащих автоматизации, мы выделяем следующие:

спецификацию конфигурации тестируемой системы;

оценку корректности поведения тестируемой системы;

оценку качества тестирования;

генерацию тестовых воздействий и тестовых последовательностей;

формирование необходимого окружения тестируемой системы.

Задача спецификации конфигурации системы заключается в задании параметров

подсистем и компонентов, описании связей межу ними и задании начального состояния.

Задача оценки корректности поведения заключается в вынесении вердикта о том, является ли наблюдаемое в ходе тестирования поведение тестируемого объекта корректным или нет. Как правило, вердикт выносится на основании проверки соответствия наблюдаемого поведения требованиям к тестируемой системе. В случае, когда эти требования представлены в виде некоторой формальной модели, проверка корректности поведения может выполняться автоматически.

Две последующие задачи напрямую связаны с целеполаганием проводимого тестирования. Примеры целей тестирования могут быть такие:

проверка того, что тестируемая система соответствует функциональным требованиям;

формирование заглушек для эмуляции окружения недоступного окружения тестируемой системы;

проверка корректности интеграции тестируемой системы с другими системами или аппаратурой;

проверка системных требований к комплексу систем;

проверка обеспечения заданных характеристик при максимальной нагрузки;

проверка устойчивости к сбоям в аппаратуре и окружении.

Задача оценки качества тестирования заключается в количественной оценке объема проведенного тестирования относительно заданных целей. Как правило, эта задача решается путем введения прямой или опосредованной метрики, в которой происходит оценка. При наличии модели, представляющей требования к системе в целом, определение метрики на основе этой модели является широко распространенной практикой. Альтернативный подход представляют собой метрики на основе покрытия исходного кода тестируемой системы. Еще одним примером метрик, определяемых на основе моделей, являются метрики, формируемые на основе выделения классов эквивалентности в модельных пространствах, представляющих в той или иной степени пространство перебора тестовых ситуаций.

Задача генерации тестов состоит в формировании сценария тестирования, обеспечивающего достижение целевого покрытия. Часто эту задачу подразделяют на генерацию единичного тестового воздействия и генерацию тестовой последовательности. В первом случае требуется подобрать параметры для единичного тестового воздействия, в то время как во втором – необходимо сформировать последовательность воздействий для достижения заданной цели. Примером использования моделей для генерации единичных воздействий является автоматическая генерация значений параметров, попадающая в заданный класс эквивалентности, тогда как генерация тестовых последовательностей часто основывается на обходе графовых структур, FSM, LTS и т. д. Генерация тестов на основе моделей может выполняться непосредственно в процессе тестирования, а может происходить заранее. Необходимость формирования окружения тестируемой системы возникает, когда при тестировании часть реального окружения тестируемой системы недоступна и, соответственно, для выполнения тестов требуется его эмуляция, которая может быть реализована посредством тестовых заглушек или симуляции поведения отсутствующих компонентов.

Архитектура тестового стенда. Тестирование моделей гибридных систем, состоящих из подсистем, которые в свою очередь являются гибридными системами, требует специализированного окружения. Это окружение предоставляется тестовым стендом. Задача стенда заключается в том, чтобы построить систему взаимодействующих компонентов, каждый из которых является некоей гибридной системой. Построение такого стенда позволяет проводить верификацию сложных систем на ранних этапах разработки, до того, как подсистемы воплощены «в металле». В частности, такой стенд позволит проводить анализ вариантов реализации системы в целом и ее отдельных подсистем, проверять реализуемость системных требований и достаточность выделяемых ресурсов.

Модели подсистем могут быть реализованы с использованием различных подходов, специфицированы в различных формализмах и выполняться в разных окружениях. Задача тестового стенда – объединить модели в общую взаимодействующую систему, обеспечить возможность подавать внешние воздействия на получившуюся систему, получить внешне наблюдаемое поведение системы и обеспечить внутренние обмены данными между компонентами системы. При этом стенд может быть распределенным, т. е. различные компоненты выполняются на различных компьютерных узлах и, возможно, под управлением различных операционных систем.

При такой постановке задачи ключевым моментом является подсистема интеграции разнородных моделей в единую систему. Поддерживая интеграцию моделей подсистем в целостную систему, стенд позволяет проводить динамическую верификацию крупной гибридной системы.

«Сердцем» тестового стенда является модель системного уровня. На вход она получает информацию о внешне наблюдаемых параметрах (например, тяге двигателя, положении внешних элементов конструкции), состоянии окружения (например, температуре среды), и рассчитывает эволюцию системы в целом.

Фактически, тестовый стенд объединяет модели подсистем в одну модель системы, а также модели окружения в одно, целостное окружение. Модели подсистем могут разрабатываться на различных языках, в рамках различных подходов и парадигм, исполняться в различных условиях.

Представляется непрактичным требовать от разработчиков моделей придерживаться единой методологии и использовать одни и те же инструменты для моделирования отдельных подсистем. Но требуется обеспечить возможность связать гетерогенные модели в одну систему.

В данной работе используется система обменов, в которой общие данные разделены на именованные переменные, причем в рамках одного вычислительного узла эти переменные разделяются посредством общей памяти (shared memory), а синхронизация значений переменных осуществляется по протоколу UDP через выделенную сеть. Запись в переменную в произвольном компоненте приводит к относительно немедленной передаче нового значения во все остальные компоненты, использующие эту переменную. Для простоты выбрана тривиальная дисциплина разрешения коллизий: предполагается, что для каждой переменной есть ровно один компонент, который может изменять значение этой переменной. Все остальные компоненты могут только читать. Разработаны адаптеры для доступа к разделяемым переменным из языков C/C++ и Python. Ведется работа по созданию адаптеров для среды численного моделирования SciCos/SciLab.

Состав подсистем, связи между ними, начальное состояние специфицируется посредством языка описания архитектуры AADL [34]. Спецификация AADL должна быть согласована с описанием взаимосвязей между подсистемами в системе обменов данными между компонентами гибридной системы. Каналы, которые связывают компоненты стенда между собой, представляются в стенде переменными: передача сообщения по каналу в модели AADL равносильна записи этого блока в разделяемую переменную. Нерешенным пока остается вопрос синхронизации модельного времени между компонентами стенда.

Требования системного уровня должны представляться моделью системного уровня. Эта модель используется для вынесения вердикта об интегральном поведении системы. Возможно, не так важно взаимодействие между компонентами, сколько выполнение требований о работе системы в целом. Модель программируется на языке высокого уровня (C или C++), прототипируется на динамическом языке (Python) либо разрабатывается специализированными средствами моделирования, такими как Simulink или SciLab.

Тестирование системы должно проводиться в рамках модели окружения. Так, для самолета такой моделью является атмосфера, а для трансформаторной подстанции – генераторные мощности, ЛЭП и потребители. В предлагаемом подходе окружение

представляется в виде гибридной системы, которая замыкает интегральную модель целевой системы. Важно обеспечить согласованность модели окружения, системной модели и моделей компонентов. Под согласованностью понимаются следующие свойства:

• модели датчиков в компонентах целевой системы берут значения показаний (температура, давление, влажность окружающей среды и т. п.) из некоторых разделяемых переменных; соответственно, модель окружения должна записывать значения физических величин в те же самые переменные;

• физические процессы в целевой системе влияют на состояние окружения; например, температура воздуха повышается по мере работы трансформатора – модель окружения должна учитывать эти эффекты; т. к. обмен информацией между моделями осуществляется через разделяемые переменные, то должны быть предусмотрены соответствующие переменные в моделях, и имена переменных должны совпадать;

• физические процессы в окружении влияют на состояние целевой системы; например, трансформатор нагревается быстрее при повышении температуры окружающего воздуха – модель системы должна учитывать эти эффекты; аналогично должны быть предусмотрены переменные для соответствующих параметров окружения;

• физические процессы в окружении влияют на целевую систему в целом: например, боковой ветер сносит самолет целиком; в модели системного уровня должны быть предусмотрены подобные глобальные эффекты и переменные для получения соответствующих параметров из модели окружения.

Отдельная важная задача – проверка требований промышленной безопасности. Такие требования предполагается специфицировать как формулы темпоральной логики и проверять средствами Data Stream Mining [33]. Переменные, используемые в формулах, берутся из AADL спецификации конфигурации системы.

Задачу определения покрытия в рамках предлагаемой архитектуры предполагается решать как покрытия требований системного уровня в терминах покрытия модели системного уровня, покрытия моделей отдельных подсистем и AADL спецификации композиции подсистем.

Генерация тестовых последовательностей осуществляется средствами библиотеки PyTESK, разработанной в ИСП РАН, которая реализует алгоритмы генерации тестовой последовательности по частично заданному автомату. Специфика применения библиотеки в рамках представленной архитектуры заключается в том, что помимо дискретных событий в спецификацию тестового автомата добавляются переходы, срабатывающие по заданным ограничениям на отсчеты непрерывной физической системы. Другими словами, автомат тестирования представляет собой гибридный автомат.

Крупномасштабные системы с цифровым управлением непрерывных физических процессов – гибридные или киберфизические системы — все шире используются в промышленности, транспорте, градостроительстве. По мере роста масштабов таких систем возрастает риск аварий и техногенных катастроф. Дальнейшее развитие гибридных систем требует создания методов верификации и валидации таких систем на ранних этапах проектирования.

В статье приведен обзор академических и индустриальных подходов к моделированию и верификации гибридных систем и представлен новый подход к верификации таких систем. Подход объединяет применение гибридных автоматов для моделирования теста и модели системного уровня, и инженерных (численных) моделей для моделирования компонентов системы и окружения.

Численное моделирование используется для описания физических процессов, протекающих в объектах управления, а также в датчиках и актуаторах. Численные модели должны описывать как воздействие системы на датчики, так и эффекты от воздействий на динамические системы со стороны системы управления через актуаторы. Численные модели используются при вы-

несении вердикта о корректности поведения гибридной системы: например, перегрев, выгорание предохранителей и т. п. В качестве инструментов предполагается использовать свободно-распространяемые пакеты численного моделирования SciLab или Octave [32].

В данной статье не рассматриваются вопросы построения адекватных моделей физических процессов, протекающих в объектах управления гибридной системы. Мы умышленно ограничиваемся задачей тестирования модели системы в модельном окружении.

Модели событийных интерфейсов гибридной системы описывают преобразование непрерывных данных – показаний датчиков и преобразование управляющих сообщений – в действия физических объектов. Традиционные подходы к моделированию протоколов используют различные формализмы: конечные автоматы, контрактные спецификации, взаимодействующие процессы, системы переходов, темпоральные логики и т. д. В представленном подходе предполагается использовать один из этих подходов (или несколько) для описания границы «физический процесс» – «цифровой протокол». В качестве инструментальных средств предполагается использовать средства библиотеки PyTESK.

В рамках представленного исследования ведутся работы по следующим направлениям:

интеграция численных моделей физических процессов и дискретных моделей управляющих событий;

спецификация тестов для гибридных систем;

прототип архитектуры тестового стенда для проведения динамической верификации гибридной системы.

Работы ведутся при поддержке гранта РФФИ 12-01-31453 мол_а.

СПИСОК ЛИТЕРАТУРЫ

1. National SCADA Test Bed Program [электронный ресурс] / URL: http://www.inl.gov/scada/

2. **Bergman D.C., Jin D., Nicol D.M., Yardley T.** The Virtual Power System Testbed and Inter-Testbed Integration // In Proc. of the 2nd Conf. on Cyber Security Experimentation and Test, 2009.

3. **Alur R., Courcoubetis C., Henzinger T., Ho P.** Hybrid automata: An algorithmic approach to the specification and verification of hybrid systems // In Hybrid Systems. Springer-Verlag, 1993. LNCS 736. Pp. 209–229.

4. **Alur R., Courcoubetis C., Halbwachs N., Henzinger T., Ho P., Nicollin X., Olivero A., Sifakis J., Yovine S.** The algorithmic analysis of hybrid systems // Theoretical Computer Science. 1995. Vol. 138. Pp. 3–34.

5. **Henzinger T.** The theory of hybrid automata // In Proc. of the 11th IEEE Symp. on Logic in Computer Science. 1996. Pp. 278–293.

6. **Deshpande A., Gollu A., Varaiya P.** SHIFT: a formalism and a programming language for dynamic networks of hybrid automata // In Hybrid Systems III. Springer, 1996. LNCS 1567.

7. **Eker J., Janneck J., Lee E., Liu J., Liu X., Luvig J., Neuendorffer S., Sachs S., Xiong Y.** Taming Heterogeneity – the Ptolemy Approach // Proc. of the IEEE. 2003. No. 91(1). Pp. 127–144.

8. **Lynch N., Segala R., Vaandrager F., Weinberg H.** Hybrid I/O automata // In Hybrid Systems III: Verification and Control. 1996. LNCS 1066. Pp. 496–510.

9. **Alur R., Henzinger T.** Modularity for timed and hybrid systems // In 8th Internat. Conf. on Concurrency Theory. Springer-Verlag, 1997. LNCS 1243. Pp. 74–88.

10. **Alur R., Dang T., Esposito J., Hur Y., Ivancic F., Kumar V., Lee I., Mishra P., Pappas G., Sokolsky O.** Hierarchical modeling and analysis of embedded systems // Proc. of the IEEE. 2003. No. 91(1).

11. **Platzer A.** Differential dynamic logic for hybrid systems // J. Autom. Reasoning. 2008. No. 41(2). Pp. 143–189.

12. Simulink: software for numerical simulation of continuous processes [электронный ресурс] / URL: http://www.mathworks.com/products/simulink/

13. Stateflow: framework for simulation of event-driven systems [электронный ресурс] / URL: http://www.mathworks.com/products/stateflow/

14. SciCos: open-source visual constructor of models for numerical simulation [электронный ресурс] / URL: http://www.scicos.org/

15. SciLab: open-source software for numerical computations [электронный ресурс] / URL: http://www.scilab.org/

16. **Tiller M.** Introduction to Physical Modelling with Modelica // The Springer International Series in Engineering and Computer Science. 2001. Vol. 615. 346 p.

17. Modelica: modelling suite [электронный ресурс] / URL: http://modelica.org/tools

18. **Karsai G., Sztipanovits J., Ledeczi A., Bapty T.** Model-integrated development of embedded software // Proc. of the IEEE. 2003. No. 91(1). Pp. 145–164.

19. **Agrawal A., Simon G., Karsai G.** Semantic Translation of Simulink/Stateflow models to Hybrid Automata using GReAT // Proc. of Internat. Workshop on Graph Transformation and Visual Modelling Techniques. Electronic Notes in Theoretical Computer Science. 2004.

20. **Henzinger T., Kopke P., Puri A., Varaiya P.** What's decidable about hybrid automata // In Proc. of the 27th ACM Symp. on Theory of Computing. 1995. Pp. 373–382.

21. **Alur R., Henzinger T., Ho P.-H.** Automatic symbolic verification of embedded systems // IEEE Transactions on Software Engineering. 1996. No. 22(3). Pp. 181–201.

22. **Frehse G., Le Guernic C., Donze A., Cotton S., Ray R., Lebeltel O., Ripado R., Girard A., Dang T., Maler O.** SpaceEx: Scalable verification of hybrid systems // In Proc. 23rd Internat. Conf. on Computer Aided Verification. Springer, 2011. LNCS 6806. Pp. 379–395.

23. **Platzer A.** Logical Analysis of Hybrid Systems - Proving Theorems for Complex Dynamics. Springer, 2010.

24. **Alur R., Henzinger T., Lafferriere G., Pappas G.** Discrete abstractions of hybrid systems // Proc. of the IEEE. 2000. No. 88(7). Pp. 971–984.

25. **Clarke E.M., Fehnker A., Han Z., Krogh B.H., Stursberg O., Theobald M.** Verification of hybrid systems based on counterexample-guided abstraction refinement // In Tools and Algorithms for the Construction and Analysis of Systems. 9th Internat. Conf. 2003. LNCS 2619. Pp. 192–207.

26. **Girard A., Pappas G.** Approximation metrics for discrete and continuous systems // IEEE Transactions on Automatic Control. 2007. No. 52(5). Pp. 782–798.

27. **Tan Li, Kim J., Sokolsky O., Lee I.** Model-Based Testing and Monitoring for Hybrid Embedded Systems // Proc. of the 2004 IEEE Internat. Conf. on Information Reuse and Integration. 2004. Pp. 487–492.

28. **Miller S.** Bridging the Gap Between Model-Based Development and Model Checking // Lecture Notes in Computer Science. 2009. Vol. 5505. Pp 443–453.

29. **Halbwachs N., Caspi P., Raymond P., Pilaud D.** The Synchronous Dataflow Programming Language Lustre // Proc. of the IEEE. 1991. No. 79(9). Pp. 1305–1320.

30. **Lind I., Andersson H.** Model Based Systems Engineering for Aircraft Systems – How does Modelica Based Tools Fit? // In Proc. 8th Modelica Conf. 2011. Pp. 856–864.

31. ES 202 786. TTCN-3: Extensions: Support of interfaces with continuous signals. 2012. 45 p.

32 Octave: open-source software for numerical computations[электронный ресурс] / URL: http://www.gnu.org/software/octave/

33. **Babcock B., Babu S., Datar M., Motwani R., Widom J.** Models and issues in data stream systems // In Proc. of the 21st ACM SIGMOD-SIGACT-SIGART Symp. on Principles of Database Systems. ACM, New York, 2002. Pp. 1–16.

34. **Feiler P.H., Gluch D. P., Hudak J.J.** The architecture analysis & design language (AADL): An introduction. Carnegie-Mellon Univ Pittsburgh PA Software Engineering Inst, 2006. No. CMU/SEI-2006-TN-011.

REFERENCES

1. *National SCADA Test Bed Program*. Available: http://www.inl.gov/scada/

2. **Bergman D.C., Jin D., Nicol D.M., Yardley T.** The Virtual Power System Testbed and Inter-Testbed Integration, *In CSET'09 Proceedings of the 2nd Conference on Cyber Security Experimentation and Test,* 2009.

3. **Alur R., Courcoubetis C., Henzinger T., Ho P.** Hybrid automata: An algorithmic approach to the specification and verification of hybrid systems, *In Hybrid Systems*, Springer-Verlag, 1993, LNCS 736, Pp. 209–229.

4. **Alur R., Courcoubetis C., Halbwachs N., Henzinger T., Ho P., Nicollin X., Olivero A., Sifakis J., Yovine S.** The algorithmic analysis of hybrid systems, *Theoretical Computer Science*, 1995, Vol. 138, Pp. 3–34.

5. **Henzinger T.** The theory of hybrid automata, *In Proceedings of the 11th IEEE Symposium on Logic in Computer Science*, 1996, Pp. 278–293.

6. **Deshpande A., Gollu A., Varaiya P.** SHIFT: a formalism and a programming language for dynamic networks of hybrid automata, *In Hybrid Systems III*, Springer, 1996, LNCS 1567.

7. **Eker J., Janneck J., Lee E., Liu J., Liu X., Luvig J., Neuendorffer S., Sachs S., Xiong Y.** Taming

Heterogeneity – the Ptolemy Approach, *Proceedings of the IEEE*, 2003, No. 91(1), Pp. 127–144.

8. **Lynch N., Segala R., Vaandrager F., Weinberg H.** Hybrid I/O automata, *In Hybrid Systems III: Verification and Control*, 1996, LNCS 1066, Pp. 496–510.

9. **Alur R., Henzinger T.** Modularity for timed and hybrid systems, *In 8th International Conference on Concurrency Theory*, Springer-Verlag, 1997, LNCS 1243, Pp. 74–88.

10. **Alur R., Dang T., Esposito J., Hur Y., Ivancic F., Kumar V., Lee I., Mishra P., Pappas G., Sokolsky O.** Hierarchical modelling and analysis of embedded systems, *Proceedings of the IEEE*, 2003, No. 91(1).

11. **Platzer A.** Differential dynamic logic for hybrid systems, *J. Autom. Reasoning*, 2008, No. 41(2), Pp.143–189.

12. *Simulink: software for numerical simulation of continuous processes*. Available: http://www.mathworks.com/products/simulink/

13. *Stateflow: framework for simulation of event-driven systems*. Available: http://www.mathworks.com/products/stateflow/

14. *SciCos: open-source visual constructor of models for numerical simulation*. Available: http://www.scicos.org/

15. *SciLab: open-source software for numerical computations*. Available: http://www.scilab.org/

16 **Tiller M.** Introduction to Physical Modeling with Modelica, *The Springer International Series in Engineering and Computer Science*, 2001, Vol. 615, 346 p.

17. *Modelica: modelling suite*. Available: http://modelica.org/tools

18. **Karsai G., Sztipanovits J., Ledeczi A., Bapty T.** Model-integrated development of embedded software, *Proceedings of the IEEE*, 2003, No. 91(1), Pp.145–164.

19. **Agrawal A., Simon G., Karsai G.** Semantic Translation of Simulink/Stateflow models to Hybrid Automata using GReAT, *Proceedings of International Workshop on Graph Transformation and Visual Modelling Techniques. Electronic Notes in Theoretical Computer Science*, 2004.

20. **Henzinger T., Kopke P., Puri A., Varaiya P.** What's decidable about hybrid automata, *In Proceedings of the 27th ACM Symposium on Theory of Computing*, 1995, Pp. 373–382.

21. **Alur R., Henzinger T., Ho P.-H.** Automatic symbolic verification of embedded systems, *IEEE Transactions on Software Engineering*, 1996, No. 22(3), Pp. 181–201.

22. **Frehse G., Le Guernic C., Donze A., Cotton S., Ray R., Lebeltel O., Ripado R., Girard A., Dang T., Maler O.** SpaceEx: Scalable verification of hybrid systems, *In Proceedings 23rd International Conference on Computer Aided Verification*, Springer, 2011, LNCS 6806, Pp. 379–395.

23. **Platzer A.** *Logical Analysis of Hybrid Systems – Proving Theorems for Complex Dynamics*. Springer, 2010.

24. **Alur R., Henzinger T., Lafferriere G., Pappas G.** Discrete abstractions of hybrid systems, *Proceedings of the IEEE*, 2000, No. 88(7), Pp. 971–984.

25. **Clarke E.M., Fehnker A., Han Z., Krogh B.H., Stursberg O., Theobald M.** Verification of hybrid systems based on counterexample-guided abstraction refinement, *In Tools and Algorithms for the Construction and Analysis of Systems, 9th International Conference*, 2003, LNCS 2619, Pp. 192–207.

26. **Girard A., Pappas G.** Approximation metrics for discrete and continuous systems, *IEEE Transactions on Automatic Control*, 2007, No. 52(5), Pp. 782–798.

27. **Tan Li, Kim J., Sokolsky O., Lee I.** Model-Based Testing and Monitoring for Hybrid Embedded Systems, *Proceedings of the 2004 IEEE International Conference on Information Reuse and Integration*, 2004, Pp. 487–492.

28. **Miller S.** Bridging the Gap Between Model-Based Development and Model Checking, *Lecture Notes in Computer Science*, 2009, Vol. 5505, Pp. 443–453.

29. **Halbwachs N., Caspi P., Raymond P., Pilaud D.** The Synchronous Dataflow Programming Language Lustre, *Proceedings of the IEEE*, 1991, No. 79(9), Pp. 1305–1320.

30. **Lind I., Andersson H.** Model Based Systems Engineering for Aircraft Systems – How does Modelica Based Tools Fit? *Proceedings 8th Modelica Conference*, 2011. Pp. 856–864

31. *ES 202 786*. TTCN-3: Extensions: Support of interfaces with continuous signals.

32. *Octave: open-source software for numerical computations*. Available: http://www.gnu.org/software/octave/

33. **Babcock B., Babu S., Datar M., Motwani R., Widom J.** Models and issues in data stream systems, *In Proceedings of the 21st ACM SIGMOD-SIGACT- SIGART Symposium on Principles of Database Systems*, ACM, New York, USA, 2002, Pp. 1–16.

34. **Feiler P.H., Gluch D.P., Hudak J.J.** *The architecture analysis & design language*, An introduction. Carnegie-Mellon Univ Pittsburgh PA Software Engineering Inst, 2006, No. CMU/SEI-2006-TN-011.

ПАКУЛИН Николай Витальевич – *старший научный сотрудник Института системного программирования РАН, кандидат физико-математических наук.*
109004, Россия, Москва, ул. Александра Солженицына, д. 25.
E-mail: npak@ispras.ru

PAKULIN, Nikolay V. *Institute for System Programming of the Russian Academy of Sciences.*
195251, Alexander Solzhenitsyn Str. 25, Moscow, Russia.
E-mail: npak@ispras.ru

НАУЧНОЕ ИЗДАНИЕ

«НАУЧНО-ТЕХНИЧЕСКИЕ ВЕДОМОСТИ САНКТ-ПЕТЕРБУРГСКОГО ГОСУДАРСТВЕННОГО ПОЛИТЕХНИЧЕСКОГО УНИВЕРСИТЕТА. ИНФОРМАТИКА. ТЕЛЕКОММУНИКАЦИИ. УПРАВЛЕНИЕ»

«ST. PETERSBURG STATE POLYTECHNICAL UNIVERSITY JOURNAL. COMPUTER SCIENCE. TELECOMMUNICATIONS AND CONTROL SYSTEMS»

№ 2 (193) 2014

Учредитель – Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования «Санкт-Петербургский государственный политехнический университет»

Журнал зарегистрирован Федеральной службой по надзору в сфере информационных технологий и массовых коммуникаций (Роскомнадзор). Свидетельство о регистрации ПИ № ФС77-51457 от 19.10.2012 г.

Телефон редакции (812)552-62-16, 297-18-21

E-mail: infocom@spbstu.ru

Компьютерная верстка *А.Н. Смирнов*

Директор Издательства Политехнического университета *А.В. Иванов*

Лицензия ЛР № 020593 от 07.08.97

Подписано в печать 30.04.2014. Формат 60×84 1/8. Бум. тип. № 1.
Печать офсетная. Усл. печ. л. 23,99. Уч.-изд. л. 23,99. Тираж 1000. Заказ

Санкт-Петербургский государственный политехнический университет
Издательство Политехнического университета
член Издательско-полиграфической ассоциации университетов России
Адрес университета и издательства: 195251, Санкт-Петербург, ул. Политехническая, д. 29.

УСЛОВИЯ ПУБЛИКАЦИИ СТАТЕЙ

в журнале «Научно-технические ведомости Санкт-Петербургского государственного политехнического университета. Информатика. Телекоммуникации. Управление»

1. ОБЩИЕ ПОЛОЖЕНИЯ

Журнал «Научно-технические ведомости Санкт-Петербургского государственного политехнического университета. Информатика. Телекоммуникации. Управление» является периодическим печатным научным рецензируемым изданием. Зарегистрировано Федеральной службой по надзору в сфере информационных технологий и массовых коммуникаций (Роскомнадзор). Свидетельство о регистрации ПИ № ФС77-51457 от 19 октября 2012 г. С 2008 года выпускается в составе сериального периодического издания «Научно-технические ведомости СПбГПУ» (ISSN 1994-2354).

Издание с 2002 года входит в Перечень ведущих научных рецензируемых журналов и изданий (перечень ВАК) и принимает для печати материалы научных исследований, а также статьи для опубликования основных результатов диссертаций на соискание ученой степени доктора наук и кандидата наук по следующим основным научным направлениям: ИНФОРМАТИКА, ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА, РАДИОТЕХНИКА И СВЯЗЬ, ЭЛЕКТРОНИКА, ИЗМЕРИТЕЛЬНАЯ ТЕХНИКА, УПРАВЛЕНИЕ В СОЦИАЛЬНЫХ И ЭКОНОМИЧЕСКИХ СИСТЕМАХ. Научные направления журнала учитываются ВАК Минобрнауки РФ при защите докторских и кандидатских диссертаций в соответствии с Номенклатурой специальностей научных работников.

Сведения о публикациях представлены в РИНЦ, в Реферативном журнале ВИНИТИ РАН, в международной справочной системе «Ulrich`s Periodical Directory».

Периодичность выхода журнала – 6 номеров в год.

2. ТРЕБОВАНИЯ К ПРЕДОСТАВЛЯЕМЫМ МАТЕРИАЛАМ

2.1. Оформление материалов

1. Рекомендуемый объем статей для авторов с ученой степенью доктора наук, званием профессора, соискателей ученой степени доктора наук (докторантов) 12–20 страниц формата А-4 с учетом графических вложений. Количество графических вложений (диаграмм, графиков, рисунков, таблиц, фотографий и т. п.) не должно превышать 4.

2. Рекомендуемый объем статей для преподавателей, авторов без ученой степени, соискателей ученой степени кандидата наук – 8–15 страниц формата А-4; аспирантов – 8 страниц формата А-4 с учетом графических вложений. Количество графических вложений (диаграмм, графиков, рисунков, таблиц, фотографий и т. п.) не должно превышать 3.

3. Авторы должны придерживаться следующей обобщенной структуры статьи: вводная часть (0,5–1 стр., актуальность, существующие проблемы); основная часть (постановка и описание задачи, изложение и суть основных результатов); заключительная часть (0,5–1 стр., предложения, выводы), список литературы (оформление по ГОСТ 7.05.-2008).

4. Число авторов статьи не должно превышать трех человек.

5. Набор текста осуществляется в редакторе **MS Word**, формул – в редакторе **MathType**. Таблицы набираются в том же формате, что и основной текст.

6. Шрифт – TNR, размер шрифта основного текста – 14, интервал – 1,5; таблицы большого размера могут быть набраны 12 кеглем. Параметры страницы: поля слева – 3 см, сверху, снизу – 2,5 см, справа – 2 см, текст размещается без переносов. Абзацный отступ – 1 см.

2.2. Предоставление материалов

Вместе с материалами статьи должны быть обязательно предоставлены:

- номер УДК в соответствии с классификатором (в заголовке статьи);
- аннотация на русском и английском языках;
- ключевые слова (5–7) на русском и английском языках;
- сведения об авторах на русском и английском языках: ФИО, место работы, должность, ученое звание, ученая степень, контактные телефоны, e-mail;
- аспиранты представляют документ отдела аспирантуры, заверенный печатью;
- акт экспертизы о возможности опубликования материалов в открытой печати.

С авторами статей заключается издательский лицензионный договор.

Предоставление всех материалов осуществляется в электронном виде через личный кабинет **ЭЛЕКТРОННОЙ РЕДАКЦИИ** по адресу **http://journals.spbstu.ru**

2.3. Рассмотрение материалов

Предоставленные материалы (п. 2.2) первоначально рассматриваются редакционной коллегией и передаются для рецензирования. После одобрения материалов, согласования различных вопросов с автором (при необходимости) редакционная коллегия сообщает автору решение об опубликовании статьи. В случае отказа в публикации статьи редакция направляет автору мотивированный отказ.

При отклонении материалов из-за нарушения сроков подачи, требований по оформлению или как не отвечающих тематике журнала материалы не публикуются и не возвращаются.

Редакционная коллегия не вступает в дискуссию с авторами отклоненных материалов.

Публикация материалов аспирантов очной бюджетной формы обучения осуществляется бесплатно в соответствии с очередностью.

При поступлении в редакцию значительного количества статей их прием в очередной номер может закончиться ДОСРОЧНО.

Более подробную информацию можно получить:
на сайте журнала http://ntv.spbstu.ru
по телефону редакции +7(812) 552-62-16 с 10^{00} до 18^{00} Галина Александровна
или по e-mail: infocom@spbstu.ru