МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ
РОССИЙСКОЙ ФЕДЕРАЦИИ

РЯЗАНСКИЙ ГОСУДАРСТВЕННЫЙ
РАДИОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

Межвузовский сборник научных трудов

Рязань 2016

УДК 004

Информационные технологии, межвузовский сборник научных трудов. – Рязань: Рязанский государственный радиотехнический университет, 2016. – 204 с.

ISBN 978-5-7722-0301-9

Публикуются статьи о проблемах использования информационных технологий в науке и образовании.

Сборник рассчитан на научно-педагогических работников вузов и может быть использован студентами и аспирантами инфокоммуникационных и информационных специальностей.

ISBN 978-5-7722-0301-9

СОДЕРЖАНИЕ

Т.Г. АВАЧЕВА, Н.В. ДОРОШИНА, А.Н. КАБАНОВ
Рязанский государственный медицинский университет
им. ак. И.П. Павлова
Рязанский государственный радиотехнический университет

АДАПТИВНЫЙ МЕТОД АНАЛИЗА МНОГОМЕРНЫХ ВРЕМЕННЫХ РЯДОВ

На основе многомерноматричных представлений решена задача разбиения временных рядов на однородные группы, исключения аномальных результатов измерений, определения главных компонент однородных групп.

Постановка задачи. Пусть исходный многомерный процесс описывается уравнением [1]:

$$y(1,0;n) = x(0,1;n) \cdot b(2,0) + e(1,0;n)$$

где Y(1,0; n) ={y1(n), y2(n), …, yk(n)} - наблюдаемые временные ряды; x(0,1; n) = {x1[n], x2(n),…, xl(n)} - наблюдаемые параметры временных рядов; b(2,0) = $\{b_{i,j}\}$, i ={1..k}; j = {1..l} - оцениваемые коэффициенты временных рядов; e(1,0; n) = {e1(n), e2(n), …, ek(n)} - помеха типа белого шума. Необходимо оценить коэффициенты $\{b_{i,j}\}$ при наличии аномальных результатов измерений во временных рядах и выделить главные компоненты x(0,1) для однородных групп временных рядов.

Решение задачи.

1. Множество наблюдаемых процессов необходимо разбить на однородные группы по наблюдаемым данным{y1(n), y2(n), …, yk(n)}, {x1[n], x2(n), …, xl(n)}.

Рассмотрим построение кратчайшего остовного дерева путем распространения алгоритма Прима на многомерный случай. По заданному графу заполняется матрица весов W(N, N). Веса несуществующих ребер предполагаются сколь угодно большими. Образуется массив P(N) меток вершин графа (столбцов матрицы весов). Алгоритм решения задачи заключается в последовательном заполнении массива меток столбцов и состоит из следующих этапов.

Предварительный этап. Обнуляется массив P(N) меток столбцов таблицы. Произвольно выбранному столбцу присваивается значение метки, равная его номеру.

Этап, повторяющийся N-1 раз (общий этап). В строках, номера которых равны номерам помеченных столбцов, находится минимальный элемент среди элементов непомеченных столбцов. Столбец, в котором находится минимальный элемент, помечается меткой, номер которой равен номеру его строки. В случае если минимальных эле-

ментов несколько, то выбирается любой. После помечивания очередного столбца элементу, симметричному относительно главной диагонали (для многомерного графа с «транспонированными индексами»), присваивается сколь угодно большое значение.

Заключительный этап. Ребра, включенные в минимальное остовное дерево, определяются по меткам столбцов. Вес остовного дерева задается суммой весов входящих в него ребер.

Адаптивная кластеризация множества элементов производится путем удаления части ребер графа по критерию минимальной суммарной дисперсии классов. Для разбиения множества элементов на К классов удаляются (К-1) ребро.

2. После определения однородных групп необходимо исключить аномальные измерения **в отдельных** процессах. Это можно выполнить с помощью метода наименьших модулей (МНМ).

Для отдельного временного ряда уравнение имеет вид

$$Y[m] = X^T[m]b[n] + e[m];\ \ \mathrm{m} = \overline{\mathrm{n} - \mathrm{N}, \mathrm{n}},$$

где X[m]={x1[n], x2(n), …, xl(n)}- вектор наблюдаемых линейно независимых факторов;

b – вектор неизвестных и подлежащих оценке параметров;

e[m] – помеха типа белого шума.

В основу алгоритма положен итеративный метод решения на основе МНМ. При этом оценка $\hat{b}$, полученная на основе N результатов измерения, имеет вид

$$\hat{b}_N[n] = \widehat{A}_N^{-1}[n]Z_N[n], \text{ где } \widehat{A}_N[n] = \sum_{m=n-N}^{n} X[m]R[m]X^T[m];$$

$$Z_{\mathrm{N}}[\mathrm{n}] = \sum_{\mathrm{m=n-N}}^{\mathrm{n}} \mathrm{X[m]R[m]Y[m]},$$

где $R[m] = (\left|Y[m] - \hat{Y}[m]\right|)^{-1};\ \hat{Y}[m] = X^T[m]\hat{b}_N[n]$ - оценка m-го измерения выходного сигнала.

Начальные значения R[m]=1; $\mathrm{m} = \overline{\mathrm{n} - \mathrm{N}, \mathrm{n}}$ соответствуют определению параметров $\hat{b}$ по методу наименьших квадратов (МНК). Далее вычисления оценок проводятся итерационно до тех пор, пока изменения оценок за одну итерацию не достигнут заданной малой величины. При этом наименьший весовой коэффициент R[l*] указывает на наиболее грубое l*-измерение.

3. После исключения сбойных результатов определяем главные компоненты для однородных групп с помощью нейронных сетей [2, 3]. Входные и выходные данные нейронной сети имеют одинаковую

размерность. Скрытый слой определяется числом главных компонент M < N.

Входные данные	Выходные данные
X1	Хапр1
X2	Хапр2
X3	Хапр3
XN	ХапрN

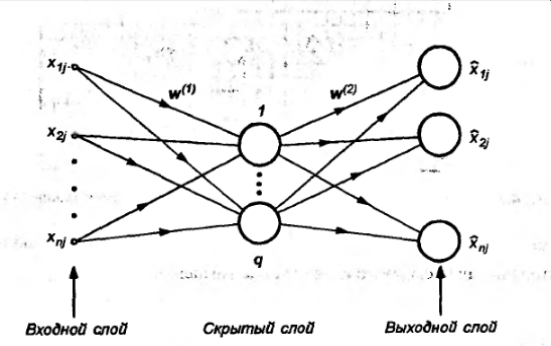


Рис.1 - Граф нейронной сети.

Для реализации метода можно использовать пакет Deductor. Для этого выбрать метод обработки «Нейросеть» и задать количество главных компонент (число нейронов в скрытом слое). Для анализа полученных результатов использовать диаграммы рассеяния и оператор «Что-если».

Выводы. Предложена адаптивная методика анализа многомерных временных рядов, позволяющая на основе последовательных процедур произвести очистку данных от аномальных измерений и выделить главные компоненты в однородных группах многомерных временных рядов. Методика применяется для выявления решающих факторов в медицинской диагностике и исследованиях материалов [4].

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Булаев М.П., Дорошина Н.В., Кабанов А.Н. Информатика и математическая статистика: Практикум / РГМУ, 2015. 104 с.

2. Паклин Н.Б., Орешков В.И. Бизнес-аналитика: от данных к знаниям. / СПб: Питер, 2013. 704 с.

3. Нейронные сети для обработки тестовых данных. Практикум/ М.П. Булаев, А.Н. Кабанов, И.С. Маркова. / РязГМУ, 2009. 92 с.

4. Авачева Т.Г., Авачев А.П. Информационный анализ АСМ-изображений…// Труды VII всерос. школы-семинара «Диагностика наноматериалов и наноструктур». Рязань, РГРТУ. 2014. С. 161-163.

М.А. БАКУЛЕВА, А.В. БАКУЛЕВ
Рязанский государственный радиотехнический университет

ПОСТРОЕНИЕ ЛИНЕЙНОЙ РЕГРЕССИОННОЙ МОДЕЛИ ВРЕМЕННОГО РЯДА НА ОСНОВЕ КРАТНОМАСШТАБНОГО ПРЕДСТАВЛЕНИЯ ДАННЫХ

Очевидно, что большинство финансовых показателей имеют, по сути, линейную природу взаимозависимости на достаточно малых интервалах наблюдений, поэтому при построении моделей прогнозирования рассматриваются модели линейного регрессионного анализа. Однако при построении регрессионных моделей временных рядов (показателей наблюдаемой величины в зависимости от времени) следует учитывать тот факт, что на протяжении некоторого промежутка времени Δt значения показателя могут не меняться или изменения настолько незначительны, что ими можно пренебречь в системе измерений предметной области. На сегодняшний день метод линейной регрессии активно используется в технических задачах, но пока недостаточно распространен в экономическом прогнозировании, поэтому в качестве анализируемой величины с промежутками «статичности» выбраны показатели экономической предметной области.

Для решения данной задачи необходимо, прежде всего, автоматизировать процесс обнаружения и локализации диапазона статичности данных.

В докладе предлагается подход, основанный на кратномасштабном представлении данных временного ряда (на основе вейвлет разложения в базисе Хаара) (рис.1)[1,2].

<table>
<tr><td>$w_{0,1}$</td><td rowspan="2">$w_{1,1}=\frac{w_{0,1}+w_{0,2}}{2}$</td><td rowspan="3">$w_{2,1}=\frac{w_{1,1}+w_{1,2}}{2}$</td><td></td></tr>
<tr><td>$w_{0,2}$</td><td></td></tr>
<tr><td>$w_{0,3}$</td><td>$w_{1,2}=\frac{w_{0,3}+w_{0,4}}{2}$</td><td></td></tr>
</table>

<table>
<tr><td>$w_{0,4}$</td><td></td><td></td><td></td></tr>
<tr><td>…</td><td>…</td><td>…</td><td>…
$w_{p,m}$
…</td></tr>
<tr><td>$w_{0,n-3}$</td><td rowspan="2">$w_{1,\frac{n}{2}-1}=\frac{w_{0,n-3}+w_{0,n-2}}{2}$</td><td rowspan="4">$w_{2,\frac{n}{4}}=\frac{w_{1,\frac{n}{2}-1}+w_{1,\frac{n}{2}}}{2}$</td><td></td></tr>
<tr><td>$w_{0,n-2}$</td><td></td></tr>
<tr><td>$w_{0,n-1}$</td><td rowspan="2">$w_{1,\frac{n}{2}}=\frac{w_{0,n-1}+w_{0,n}}{2}$</td><td></td></tr>
<tr><td>$w_{0,n}$</td><td></td></tr>
</table>

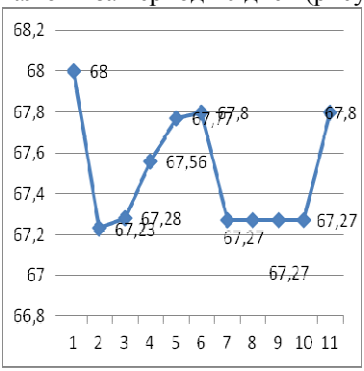
Рис. 1 – Кратномасштабное представление временного ряда.

На основе исследования особенностей кратномасштабного разложения в базисе Хаара можно заключить следующее:

Теорема 1. Для определения наличия статичности данных необходимо и достаточно, чтобы данные первого и второго уровней разложения совпадали.

Теорема 2. Диапазон статичности данных прямо пропорционален значению n^2, где n – последний уровень кратномасштабного разложения со значениями, удовлетворяющими Теореме 1.

На основе теоремы 1 и 2 разработан алгоритм обнаружения и локализации диапазона статичности данных. В качестве примера рассмотрим кратномасштабное представление данных значения курса валюты за период 10 дней (рисунок 2).



67,615

67,42 67,5175

67,785 67,5225

67,27 67,5275

67,27

Рис. 2 – Произвольная выборка курсовых значений за период 10 дней и кратномасштабное представление данного ряда.

В соответствии с теоремой 1 повторяющееся значение 67,27 указывает на наличие в исходной временном ряду интервала статичности, поэтому при построении линейной регрессии следует разбить интервал прогнозирования и строить линию регрессии в диапазоне [1;7].

Апробация предложенного алгоритма, разработанного в среде *Qt Creator* на тестовой выборке, позволяет заключить, что задача, поставленная в рамках данного исследования, решена [3,4].

Исследования проводились в рамках НИР 9-14Г при финансовой поддержке Министерства образования и науки РФ.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Бакулева М.А. Применение вейвлет-преобразований для представления данных хранилища Вестник РГРТА. — 2006. — № 18. — С.80—86

2. Бакулев А.В., Бакулева М.А. Применение вейвлет-преобразования для анализа данных хранилища Вестник РГРТУ. Научно-технический журнал. Выпуск 21. Рязань: РГРТУ, 2007. С. 57-60.

3. Бакулев А.В., Бакулева М.А., Телков И.А. Алгоритм автоматизации проектирования хранилищ данных. Вестник РГРТУ. Научно-технический журнал. Выпуск 23. Рязань: РГРТУ, 2008. С. 90-93.

4. Бакулев А.В., Бакулева М.А., Авилкина С.В. Математические модели и алгоритмы организации мобильных параллельных вычислений в среде многоядерных процессоров. European Researcher, 2012, Vol. (33), № 11-1, С. 1826-1834.

И.А. БАРАНКОВА
Московский государственный технический университет им. Баумана

ОБЗОР ИНФОРМАЦИОННЫХ СИСТЕМ АНАЛИЗА И МОДЕЛИРОВАНИЯ ДИНАМИЧЕСКОЙ СТРУКТУРЫ СОЦИАЛЬНЫХ СЕТЕЙ

В статье приведен обзор существующих систем анализа социальных сетей, рассмотрены основные классы пользователей таких систем и типы задач, которые они решают. Большинство рассмотренных систем предназначены для оперативного мониторинга и анализа и не затрагивают вопросы прогнозирования и управления.

В связи с усилением роли сетевых ресурсов как средства массового информирования и информационного управления возникает и становится актуальной задача мониторинга, систематизации и анализа содержимого веб-ресурсов и выявления скрытых и явных связей меж-

ду ними с целью определения источников влияния в рамках заданной проблемы или определённого вопроса. Особенно важным в контексте данной проблемы является анализ социальных сетей и их структуры.

Основными задачами информационно-аналитической работы с социальными сетями являются: мониторинг и анализ социальных сетей (для достижения понимания происходящих в социальных сетях процессов), прогнозирование и управление (для перевода социальной сети в требуемое состояние).

Мониторинг включает получение и структурирование первичных данных. Собираются тексты сообщений, связи между пользователями, ссылки на внешние ресурсы.

Анализ включает несколько этапов обработки первичных данных. Во-первых, вычисление базовых показателей, которое позволяет отвечать на количественные вопросы. Далее выявление статистических и структурных закономерностей в данных дает понимание природы исследуемой сети.

Прогноз возможен после идентификации математической модели информационного процесса. Могут использоваться статистические модели и модели динамических процессов на графах (распространение эпидемий, каскадное поведение).

Управление заключается в оказании целенаправленных воздействий на социальную сеть для перевода информационных процессов в желаемое состояние.

Необходимо отметить, что существующие системы анализа социальных сетей ориентированы на различные типы пользователей (коммерческие организации, государственные структуры, исследователей). Проведём анализ аналогичных систем для коммерческих организаций и государственных структур.

Достоинство систем, предназначенных для коммерческих организаций, в том, что они позволяют проводить целенаправленную работу с социальными сетями. В частности, с их помощью можно отслеживать упоминания брендов и продуктов по заданному набору ключевых слов, определять значимость обсуждений при помощи типичного набора показателей, поддерживать работу с пользователями социальных сетей при помощи унифицированного доступа к учетным записям организации и интегрировать данные социальных сетей с имеющимися данными организации (например, с данными продаж).

Ограничением таких систем является то, что они ориентированы на оперативное реагирование, а не на стратегическую бизнес-аналитику, прогнозирование и управление. Кроме того, они требуют

наличия дорогостоящей инфраструктуры и квалифицированных сотрудников.

Достоинством систем для государственных учреждений являются расширенные аналитические возможности и умение работать с различными открытыми источниками информации, а не только с социальными сетями (продукты *OSINT*). Однако такие системы являются закрытыми, их доступные описания неполны, туманны и мифологизированы.

Сравнение программных систем будем производить по следующим критериям:

- методы анализа;
- уровень анализа данных;
- объекты анализа;
- режим анализа;
- режимы сбора данных;
- объемы обрабатываемых данных.

Методы анализа – этот параметр описывает разнообразие применяемых в системе методов: статистические методы и методы анализа графов. Также удобно выделить отдельно методы семантического анализа и анализа тональности текстов (необходимо обратить внимание на поддержку системой языка, на котором общаются пользователи).

Уровень анализа данных – данный параметр характеризует сложность системы: она может осуществлять простой мониторинг ресурсов и социальных сетей, анализ веб-страниц и социальных сетей, прогнозирование процессов в социальных сетях, управление социальными сетями.

Объекты анализа – параметр, который характеризует степень детализации анализа: системы могут фокусироваться на анализе сети «в целом» (при помощи некоторых агрегированных глобальных показателей); подсетей и сообществ; отдельно взятых пользователей (при помощи локальных показателей); информационных сообщений (при помощи показателей упоминания некоторых информационных объектов); мнений (при помощи показателей тональности сообщения относительно некоторых информационных объектов); внешних узлов (или информационных ресурсов сети Интернет). Информационным объектом может быть некоторая персона, событие, организация и т.п.

Режимы анализа данных – параметр, который характеризует аналитические возможности: системы могут не предоставлять возможность анализа данных (отсутствует), или предоставлять возмож-

ность ретроспективного анализа данных и/или анализа данных в режиме реального времени.

Режимы сбора данных – параметр, характеризующий подсистему сбора данных: системы могут не предоставлять возможность сбора данных (отсутствует) или предоставлять возможность ретроспективного сбора данных и/или сбора данных в режиме реального времени. Системы могут осуществлять сбор всего объема данных или осуществлять сбор данных по определенной тематике.

Объемы обрабатываемых данных – параметр, характеризующий размер массива анализируемых данных: системы могут быть рассчитаны на модельные объемы данных или на промышленные объемы данных (программные продукты *BigData*).

Важнейшим является параметр «методы анализа» так как он характеризует эффективность аналитической подсистемы. Система с упрощёнными методами анализа (или слишком малым количеством методов) фактически непригодна для использования. Вторым по степени важности является параметр «объекты анализа», поскольку он характеризует степень детальности анализа (будет ли рассматриваться взаимодействие внутри кластеров (сообществ), кластеров между собой, будут ли анализироваться только информационные сообщения и др.). Параметр «режимы анализа данных» характеризует возможности системы анализа: будет ли осуществляться оперативный либо ретроспективный анализ первичных данных, поэтому также важен.

Параметр «уровень анализа данных» является менее важным, поскольку системы, как правило, осуществляют мониторинг и первичный анализ данных и не осуществляют прогнозирование и управление процессами системы. Параметр «режим сбора обрабатываемых данных» также менее важен, поскольку основное внимание уделяется сравнению аналитических возможностей различных систем.

Система анализа социальных сетей Radian 6

Система *Radian* 6 (www.radian6.com) предназначена для отслеживания в реальном времени упоминаний брендов с учетом тональности в социальных сетях (предоставляется панель управления мониторингом) и для участия в происходящих обсуждениях (предоставляется панель управления участием). Панель управления участием позволяет реагировать на активность в социальных сетях из одного места, используя имеющиеся учетные записи в блогах, площадках *Twitter* и *Facebook*.

Для ретроспективного анализа доступны данные, накопленные за последние 30 дней. Такое ограничение представляется существенным для анализа продолжительных кампаний в социальных сетях. За-

метим, что система *Radian* 6 в большей степени фокусируется на оперативном реагировании на происходящие события, нежели на бизнес-аналитике (стратегический уровень принятия решений), поэтому управляющие воздействия могут привести лишь к кратковременному всплеску продаж. Клиенты системы: более 50 компаний из *Fortune* 100 (*Pepsi*, *Dell*, *Kodak* и др. (см. полный список на www.radian6.com/about-us/customers/)).

Дополнительные возможности системы: управление рабочим процессом (совместная классификация и категоризация постов, назначение приоритетов и выполнение работ по плану), возможность работы в социальных сетях: интеграция с *Twitter* и *Facebook*.

Основные характеристики системы *Radian* 6 приведены в таблице 1.

Таблица 1. Описание системы Radian6.

Уровень анализа данных	Мониторинг и анализ
Методы анализа	Базовые методы анализа и поиска текстов на уровне ключевых слов, анализ тональности текстов (поддерживается в том числе русский язык), визуальный анализ (инфографика)
Объекты анализа	Сеть в целом, пользователи (см. примечание ниже), информационные сообщения, мнения – анализ при помощи простых агрегированных показателей
Режим анализа	Анализ в режиме реального времени, ретроспективный анализ с ограничением в 30 дней
Объём обрабатываемых данных	Более 10000 постов и твитов ежедневно
Сбор данных	Сбор данных в режиме реального времени

Примечание: пользователям системы предоставляется возможность настраивать (и сохранять) профили ранжирования по следующим показателям: по количеству постов заданной темы, по количеству комментариев заданной тематики, по количеству уникальных комментаторов, по количеству входящих ссылок, по количеству голосов, по количеству ответов на тематических форумах.

Система анализа социальных сетей ALTERIAN SM2

Основное решение компании *SDL* в области анализа социальных медиа. Система *Alterian SM*2 в связке с дополнительными приложениями и сервисами. Система *Alterian SM*2 - типичная для своего класса система, которая позволяет отслеживать упоминания брендов в социальных сетях с учетом тональности (определяется положительная,

отрицательная и нейтральная тональность). Кроме того утверждается, что система позволяет локализовать места обсуждений и определять демографические характеристики пользователей социальных сетей.

ALTERIAN SM2 позволяет анализировать блоговые площадки (Живой журнал, *TypePad*, *Twitter*, *Plurk*, *Identi.ca*), доски объявлений и форумы, вики сайты, сервисы обмена фотографиями и видео (*YouTube*, *Flickr*), социальные сети (*Ning*, *Facebook*, *LinkedIn*), сайты электронных объявлений(*Craigslist*), сайты обзоров потребителей (*Epinions*). Среди её клиентов: *MD Anderson Cancer Center*, *Pursuit*, *YouCast*, *Red Bricks Media* и др.

Основные характеристики системы *Alterian SM*2 приведеныв таблице 2.

Таблица 2. Описание системы Alterian SM2.

Уровень анализа данных	Мониторинг и анализ
Методы анализа	Базовые методы анализа и поиска текстов на уровне ключевых слов (поддерживается в том числе русский язык), анализ тональности текстов (русский язык не поддерживается), тематический анализ, визуальный анализ (инфографика)
Объекты анализа	Сеть в целом, пользователи – анализ при помощи простых агрегированных показателей
Режим анализа	Анализ в режиме реального времени, ретроспективный анализ (5 лет)
Объём обрабатываемых данных	Более 60 миллионов постов, комментариев и твитов ежедневно
Сбор данных	Сбор данных в режиме реального времени

Система анализа «Призма»

Система позиционируется как инструмент оперативного мониторинга и анализа политико-социальной активности населения в *интернет блогах и форумах*. Работает на базе информационно-аналитической системы «Медиалогия». Система предназначена для управления репутацией и рисками в социальных медиа.

Охват источников информации: сообщения более 40 млн. русскоязычных социальных медиа: блогов, микроблогов, форумов и социальных сетей.

Основные функции работы с информацией:

1. Анализ тональности высказываний по отношению к информационным объектам.

2. Кластеризация сообщений по сюжетам с возможностью найти каждое отдельное сообщение и его характеристики (в том числе тональность); ранжирование сюжетов по обсуждаемости.

3. Анализ динамики параметров обсуждений информационных объектов.

4. Определение интереса блогосферы к тем или иным информационным объектам, выявление аномального интереса.

5. Оценка реакции социальных медиа на события, связанные с определенными ведомствами, регионами, руководителями.

6. Предупреждение о возможных рисках – возможно, заключается в слежении за количественной и качественной динамикой обсуждения информационного объекта и предсказании дальнейшей динамики.

Возможной слабостью системы является описание взаимодействия на достаточно «низком», детальном уровне, без выстраивания общей картины взаимодействия пользователей.

Система «*PALANTIR*» (*корпорация PALANTIR*)

Система *Palantir* оценивает транзакции в интернете (финансовые транзакции через кредитные карты, звонки по сотовому телефону, записи об адресатах и темах писем электронной почты, покупки и использования авиабилетов, логи разыскиваемой человеком информации в интернете и т.д.) как часть «общего паттерна активности» пользователя. В такой системе выявляется информация о компьютере человека, организующего транзакцию, о других людях, с которыми он ведет дела, а также о том, как это все вписывается в общую историю транзакций.

Идея состоит в скрещивании алгоритмов искусственного интеллекта с опытом человека-аналитика, каждая информационное событие – часть «общего паттерна активности».

Основная цель создания системы – обнаружение и разоблачение злонамеренных организаций. Раскрытие связанных событий на основе огромного массива разрозненной информации.

Ниже перечислены функции системы:

1. Работа с произвольными типами неструктурированной и структурированной информации.

1. Человеко-читаемое представление данных (в основном графы связей и различные графики).

2. Способность выполнять статистическую, временную, геопространственную и реляционную обработку всех данных.

3. Способность строить социальные сети интересующих объектов/субъектов (т.е. отображать графовую структуру их взаимодействия/знакомств и т.п.).

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Губанов Д.А., Новиков Д.А., Чхартишвили А.Г. Социальные сети: модели информационного влияния, управления и противоборства. – М.: Изд-во физико-математической литературы, 2010. – 228 с.
2. Список систем анализа сетевых структур [Электронный ресурс].URL:http://en.wikipedia.org/wiki/Social_network_analysis_software (дата обращения 02.03.2016).
3. Список систем анализа социальных медиа [Электронный ресурс]. – URL: http://wiki.kenburbary.com (дата обращения 02.03.2016).

А.В. БЕЛЯКОВ
Национальный исследовательский университет «МИЭТ»

ИСПОЛЬЗОВАНИЕ КРАСНО-ЧЁРНЫХ ДЕРЕВЬЕВ ДЛЯ ХРАНЕНИЯ ДАННЫХ

Рассматриваются преимущества хранения данных в виде структур, называемых красно-чёрными деревьями.

При постоянном увеличении нагрузки на вычислительную технику важным аспектом в улучшении её производительности является усовершенствование структур, в виде которых хранятся данные в памяти устройства. Результатом работ в данной области стало красно-чёрное дерево, которое широко применяется на данный момент.

Под деревом здесь следует понимать двоичное дерево поиска - дерево, все вершины которого упорядочены, каждая вершина имеет не более двух потомков (назовём их левым и правым), и все вершины, кроме корня, имеют родителя. Вершины, не имеющие потомков, называются листами

Название красно-чёрных деревьев объясняется дополнительной характеристикой вершины – цветом. В устоявшейся терминологии эта характеристика может принимать два значения: чёрный или красный.

Данная структура данных должна удовлетворять следующим свойствам:

- узел (вершина) покрашен либо в красный, либо в чёрный цвет;
- листьями объявляются NIL-узлы. Листья покрашены в черный цвет;
- потомки красного узла обязательно чёрные;
- любая ветвь дерева, ведущая от корня к листу, содержит одно и то же количество чёрных узлов.

Последнее свойство означает сбалансированность красно-чёрного дерева по черным вершинам. Количество чёрных узлов также называют чёрной высотой дерева. Следует отметить, что сбалансированным называется дерево, для которого длины всех путей от корня к внешним вершинам равны между собой.

Так же из последнего условия следует вывод, что любая ветвь от корня до листа такого дерева не более чем в два раза длиннее самой короткой ветви. Так как вставка, удаление и поиск элемента требуют в худшем случае затраты по времени пропорциональные высоте дерева, то это условие позволяет быть красно-чёрным деревьям более эффективными по сравнению с обычными бинарными деревьями поиска.

Таблица 1. Временные затраты двух видов деревьев.

Операция	Бинарное дерево поиска		Красно-чёрное дерево	
	В среднем	В худшем случае	В среднем	В худшем случае
Поиск	O(log n)	O(n)	O(log n)	O(log n)
Вставка	O(log n)	O(n)	O(log n)	O(log n)
Удаление	O(log n)	O(n)	O(log n)	O(log n)

Из приведенной таблицы напрямую следует, что худшее время выполнения операций с рассматриваемой структурой не зависит от взаимного расположения данных до их добавления в красно-чёрное дерево, за счёт чего получается избегать временных затрат порядка O(n).

Популярность и повсеместное использование таким деревьям приносят достижение баланса между степенью сбалансированности дерева и затратами на поддержание этой сбалансированности. На практике часто выясняется, что при выполнении операции вставки/удаления в идеально сбалансированные деревья значительная часть времени тратится на поддержание требуемой степени сбалансированности.

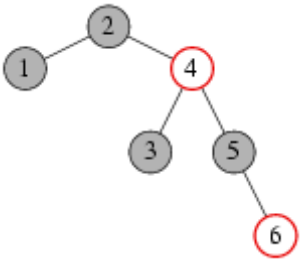


Рис. 1 – Пример чёрно-красного дерева.

Тёмным цветом на рисунке обозначены чёрные вершины, а белым – красные. Как мы видим, при графическом изображении «фиктивные листовые узлы» часто опускаются, из-за чего дерево выглядит противоречиво, однако на самом деле противоречий нет. Так в приведённом случае чёрная высота дерева равна трём, что достигается на каждой ветви структуры.

На практике красно-чёрные деревья активно используются в программировании: таких языках как c#, java и т.д. Именно с помощью этой структуры реализованы контейнеры set и map в библиотеке STL языка C++.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Кормен Т.Х., Лейзерсон Ч.И., Ривест Р.Л., Штайн К. Алгоритмы: построение и анализ. -3-е изд. –М.:Вильямс, 2005. -336-356 стр.
2. Материалы взяты с сайта algolist.manual.ru http://algolist.manual.ru/ds/rbtree.php
3. Материалы взяты с сайта http://math.msu.su http://math.msu.su/~vvb/2course/Borisenko/lecTree.html

А.С. БУРОБИНА, Е.С. ГЕРАЩЕНКО, В.Ю. ПОТАПОВА
Рязанский государственный радиотехнический университет

МИКРОКОНТРОЛЕР CORTEX M3

Рассматривается микроконтроллер cortex m3, его ядро, режим энергосбережения, отличия от других микроконтроллеров и применение.

Семейство ARM Cortex – новое поколение процессоров. В отличие от других ЦПУ ARM оно является завершенным процессорным ядром, объединяющим стандартное ЦПУ и системную архитектуру. Благодаря низкой стоимости процессоры данного типа могут конкурировать с традиционными 8 и 16-битными микроконтроллерами. Возможность оперировать с фрагментированными данными является отличительной чертой данного семейства и позволяет обеспечивать максимальную эффективность использования внутреннего статического ОЗУ.

Cortex–M3 – стандартизированное микроконтроллерное ядро, выполненное по Гарвардской архитектуре, которое помимо ЦПУ содержит систему прерываний, системный таймер SysTick, отладочную систему, карту памяти. Четырехгигабайтное адресное пространство разделено на четко распределенные области кода программы, статиче-

ского ОЗУ, устройства ввода-вывода и системных ресурсов. Использование нового набора команд Thumd-2 позволяет достигнуть на 70% большей производительности на мегагерц по сравнению с классическим ARM процессором, использующим набор команд Thumd.

Ядро имеет раздельные шины для команд и для данных. Оно предоставляет производительность порядка 1.25 MIPS/МГц за счет использования технологии с трехступенчатым конвейером, которая обеспечивает возможность предсказания перехода, однократного умножения и аппаратно реализованного деления. Ядро реализует 2 режима работы: потоковый режим и режим обработки, для каждого из которых можно сконфигурировать свои собственные стеки. Благодаря этому появляется возможность разработки более интеллектуального программного обеспечения и поддержки операционных систем реального времени.

Данный микроконтроллер может функционировать в режиме энергосбережения (SLEEP). Ядро Cortex M3 реализует 2 экономичных режима работы. В первом из них при переходе в режим SLEEP ядро Cortex приостановит выполнение программы и обработку отправленных прерываний. Существует два сценария завершения выполнения процедуры обработки прерывания:

- ЦПУ выходит из процедуры обработки прерывания и возвращается к дальнейшему выполнению фоновой программы;
- ядро после выхода из процедуры обработки прерываний автоматически перейдёт в режим SLEEP.

Во втором режиме работа ядра Cortex возобновляется с того же места, с которого оно было переведено в режиме экономичной работы. Это прерывание не приводит к переходу к процедуре обработки прерывания.

В микроконтроллерах, основанных на ядре Cortex M3, поддерживаются возможности по обеспечения безопасной работы. Основные из них направлены на выявление некорректного кода программы, ненадёжного источника питания и неправильного поведения микроконтроллера в целом. Например, для исключения возможности работы от ненадёжного источника питания предусмотрена встроенная схема, переводящая его в состояние сброса, если напряжение будет ниже минимально допустимого значения.

Для того, чтобы контролировать корректность выполнения программы, можно использовать два встроенных сторожевых таймера. Один из них – так называемый оконный таймер, который нужно обновлять в соответствии с определенной частотой. Второй – независимый сторожевой таймер, для синхронизации которого используется

обособленный генератор, не связанный с основной системной синхронизацией.

Микроконтроллеры на основе ядра Cortex M3 обеспечивают высокую надежность хранения данных благодаря наличию встроенной Flash-памяти, которая позволяет хранить данные в течение 30 лет при температуре 85°C. Данные значения являются лучшими показателями хранения памяти для микроконтроллеров данного сегмента.

Благодаря своей производительности, высокой плотности кода и крайне компактному размеру ядро Cortex M3 находит применение в самых различных областях производства:

- недорогие микроконтроллеры общего назначения. Благодаря малой потребляемой мощности и простоте применения ядро Cortex M3 успешно применяется в самых разных потребительских товарах, начиная от игрушек и заканчивая бытовыми электроприборами;
- автомобильная электроника. В данном сегменте основными преимуществами Cortex M3 перед другими процессорами является очень высокая производительность и малое время реакции на прерывания, благодаря чему он идеально подходит для применения в системах реального времени;
- передача данных. Так как устройства на основе Cortex M3 имеют малое энергопотребление и поддерживают команды работы с битовыми полями, содержащимися в наборе Thumb-2, их можно эффективно применять в коммуникационных приложениях, например, Bluetooth и ZigBee;
- автоматизация производства. Основным преимуществом применения ядра Cortex M3 в данной области служит наличие расширенных средств обеспечения отказоустойчивости. Кроме этого важными факторами для устройства управления промышленным оборудованием является малое время реакции на прерывание и лучшая в своём сегменте надёжность хранения данных.

Наиболее популярными на сегодняшний день устройствами на основе Cortex M3 являются продукты компании Apple, такие, как iPhone, iPad, iPad mini.

В них используется обновлённая версия микроконтроллера Apple M8, ядром которой является Cortex M3, что позволяет достичь заметного улучшения энергосбережения, благодаря возможности обрабатывать и хранить данные различных датчиков (акселерометры, гироскопы и т.д.), даже когда устройство находится в спящем режиме. Кроме этого, применение Cortex M3 увеличивает срок службы аккуму-

лятора за счёт увеличения интервалов опроса станций сотовых сетей и точек доступа.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Ознакомительное руководство по ARM-микроконтроллерам CortexM3 [электронный ресурс] http://www.gaw.ru/html.cgi-/txt/doc/micros/arm/cortex_arh/

2. Ядро CortexM3 компании ARM, Джозеф Ю., М: ДМК Пресс, 2015, стр. 27-28.

А.С. БУРОБИНА
Рязанский государственный радиотехнический университет

СОВРЕМЕННАЯ ТЕХНОЛОГИЯ ПИТАНИЯ СЕТЕВОГО ОБОРУДОВАНИЯ

Рассматривается технология передачи напряжения в вычислительных сетях с помощью Ethernet-кабеля для питания удалённого устройства.

Тенденции развития последних лет в мире IT-технологий нацелены на эволюцию и широкое применение сетевого оборудования. Точки доступа, сетевые камеры, маршрутизаторы – всё это плотно вошло в нашу жизнь. Но зачастую установка такого рода оборудования сталкивается с бытовой, но крайне важной проблемой – учёт того, где находится близлежащая электрическая розетка.

Для решения этой проблемы в июле 2003 года был принят стандарт IEEE 802.3af, регламентирующий способ электроснабжения устройств, подключенных к сети Ethernet. Технология, позволяющая передавать удалённому устройству вместе с данными электрическую энергию через стандартную витую пару в сети Ethernet, получила название Power over Ethernet (PoE).

Согласно стандарту обеспечивается ток до 400 мА с номинальным напряжением 48 В (от 36 до 57 В) для обеспечения максимальной мощности 15,4 Вт.

В 2009 году вышел стандарт IEEE 802.3at, запрещающий устройству-потребителю получать питание по всем четырём парам Ethernet-кабеля одновременно и предусматривающий подачу мощности до 25,5 Вт.

Технология PoE не оказывает влияния на качество передачи данных. Для её реализации используются свойства физического уровня Ethernet:

1. С использованием высокочастотных трансформаторов на обоих концах линии с центральным отводом от обмоток постоянное напряжение питания подаётся на центральные отводы вторичных обмоток этих трансформаторов, и так же с центральных отводов снимается на приёмной стороне. Использование центральных отводов позволяет передавать по одним и тем же проводникам и высокочастотные данные, и постоянное напряжение

2. С использованием свободных пар для подачи питания. Это возможно, так как современные кабельные сети Ethernet состоят из четырёх пар, две из которых не задействованы.

Описанные два типа технологий PoE получили название active и passive PoE соответственно. Основное отличие заключается в том, что устройства с active PoE способны отделять цифровые данные от электропитания и подавать их на два разных выхода, полностью реализуя стандарт IEEE 802.3af, а устройства с passive PoE поддерживают только электрические характеристики соответствия данному стандарту, но не протокольные. Это означает, что передача по одной паре и данных и напряжения невозможна.

Текущая спецификация технологии PoE описывает Ethernet-оборудование, которое можно разделить на 3 части:

- PoE источники (коммутатор, инжектор)
- PoE потребители (ip камера, точка доступа, сплиттер, коммутатор)
- PoE адаптеры (инжекторы и сплиттеры).

PoE источники полностью соответствуют стандартам 802.3af и 802.3at, то есть они способны различать какой потребитель к ним подключен. Для этого на витые пары сначала подаётся минимальное напряжение, чтобы определить имеет ли принимающее оборудование оконечное сопротивление в 25 кОм. Если нагрузка PoE является потребителем, то есть устройство, подключенное к источнику, может питаться от этого источника, то в кабель подаётся питание. Таким образом, исключается повреждение устройств, не поддерживающих данную технологию. Все PoE источники реализуют тип active PoE.

PoE потребители – это сетевые устройства, питание которых осуществляется по тому же кабелю (витой паре), что и передача данных.

PoE адаптеры бывают двух типов: питающие (инжектор) и принимающие (сплиттер). Работа питающих адаптеров заключается в объединении сигнала и напряжения в один кабель для дальнейшей передачи принимающему устройству. Они выдают напряжение в 48 вольт вне зависимости от того поддерживает потребитель питание по техно-

логии PoE или нет. Принимающие PoE адаптеры предназначены для разделения на выходе из кабеля сигнала и напряжения. Оба вида PoE адаптеров функционируют по технологии passive PoE.

PoE является отличным решением для инфраструктуры небольших предприятий и крупных сетей корпораций, так как со временем сеть разрастается разнородным оборудованием, которое нуждается в электроснабжении, старой системы электропроводки уже недостаточно, а прокладка новой требует времени и значительных затрат.

Использование же оборудования с поддержкой PoE (как инжекторов, так и коммутаторов) позволяет с легкостью запитать и подключить блоки питания IP-телефонов, точки доступа Wi-Fi, камеры и многое другое сетевое оборудование. Если же необходимо запитать оборудование, не поддерживающее PoE, на помощь придут адаптеры.

Следует учесть, что использование оборудования с различными типами технологии PoE для питания одного устройства недопустимо, так как:

- при использование активного PoE источника и пассивного принимающего PoE адаптера источник определит, что конечное устройство не принадлежит к классу PoE потребителей и напряжение не подаст
- при использовании пассивного питающего PoE адаптера и активного PoE потребителя велика вероятность повреждения потребителя из-за несоответствия типа передачи и напряжения.

Также при использовании технологии PoE в промышленных масштабах стоит принять во внимание возросшие потребности мощности электроснабжения и предусмотреть дополнительное охлаждение, чтобы избежать непредвиденных отключений или нарушений в работе оборудования.

Конечные потребители могут находиться на расстоянии 100 метров от PoE источника или инжектора.

PoE питает отдельное оборудование по звездообразной топологии и не допускает каскада в линейной топологии сети.

PoE является универсальным решением как при организации новых сетей, так и при модернизации существующих. Эта технология позволяет не только существенно сэкономить на стоимости силовых кабелей, но и сократить время инсталляции оборудования Ethernet.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. IT решения для дома и офиса [электронный ресурс] http://it-tek.ru/power-over-ethernet-poe.html

2. Концепции безопасности [электронный ресурс] http://kb-sb.ru/pub/10/56/

3. Power over Ethernet Technology Consortium («PoETeC») [электронный ресурс] http://www.0baset.org/.

Д.М. ВУКАЛОВ, К.Н. КОНОЧКИН
Национальный исследовательский университет «МИЭТ»

ПЕРВЫЙ ШАГ В РОБОТОТЕХНИКЕ С LEGO MINDSTORMS

Не так давно робототехника была занятием исключительно высококвалифицированных инженеров и программистов. А сейчас любой человек и даже ребенок, не имеющий специального образования, может полностью собрать и запрограммировать своего собственного робота с помощью наборов компании Lego.

Впервые такой набор был представлен в 1998 году. Через 8 лет (2006) в свет вышла модель LEGO Mindstorms NXT 1.0, в 2009 — LEGO Mindstorms NXT 2.0, а в 2013 — LEGO Mindstorms EV3. NXT является интеллектуальным, управляемым компьютером-роботом на базе элементов Lego и системы Mindstorms. Основа конструктора – программируемый блок NXT, интерактивные сервомоторы, несколько датчиков. Все датчики и моторы подсоединяются к NXT-блоку через порты входа и выхода посредством черных шестипроводных кабелей. Так же в набор входит ряд деталей для конструирования: балки, пластины, штифты, оси, втулки, колеса и др.

Рис. 1 – Микрокомпьютер, моторы и датчики серии NXT.

Программное обеспечение Lego Mindstorms NXT 2.0 может быть установлено на компьютер, имеющий операционную систему MS Windows или Mac OS.

Знакомство с программированием роботов в данной серии наборов начинается с языка NXT-G в графической среде программирова-

ния. Данный софт имеет интуитивно понятный интерфейс, создание программ управления роботами напоминает создание блок-схем и осуществляется с помощью специальных блоков, размещаемых на LEGO-балках вдоль оси последовательности действий. Также существует большое количество NXT-совместимых языков программирования:

- RobotC;
- NXT++;
- NXT_Phyton;
- SqLego;
- RoboRealm.

Пользователь может «перепрошить» своего робота с помощью USB-кабеля или Bluetooth-соединения и получить на выходе всевозможные конструкции:

- Роботы, которые перемещаются, видят, реагируют на окружающую среду;
- различные типы шасси (колёса, гусеницы);
- роботизированные руки, подобные тем, что используются в производстве;
- роботизированная система, реагирующая на движение;
- роботизированные животные, которые двигаются, как их живые аналоги.

Так же хочется отметить, что робот может управляться не только посредством «вшитого» в него кода, но и с помощью приложения для смартфона.

Особенностью данных систем является простота использования и наглядность работы. Десятилетний ребенок может создать собственного робота и прекрасно понимать, как он работает. Освоение робототехники в школьном возрасте может дать толчок для дальнейшего изучения таких важных в современное время дисциплин как программирование и конструирование роботов.

Однако, очень немногим школам так повезло. Во-первых, стоимость робототехнического комплекса на данный момент превышает стоимость средней компьютерной системы. Во-вторых, руководить занятиями робототехники должен высококвалифицированный педагог, хорошо разбирающийся и в техническом конструировании, и в микроэлектронике, и в программировании. Подготовка таких специалистов-педагогов сегодня только начинается. В-третьих, русскоязычные учебники данной тематики можно в буквальном смысле пересчитать по пальцам одной руки.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Jerry Lee Ford, Jr. Lego Mindstorms NXT2.0 for teens. 2010. - 337 стр.
2. Дженжер В.О., Денисова Л.В. Введение в программирование Lego-роботов на языке NXT-G. Учебное пособие для студентов и школьников. –М.:Национальный открытый университет «Интуит», 2014. - 87 стр.
3. Материалы взяты с сайта «Википедия» https://ru.m.wikipedia.org/wiki/Mindstorms_(серия_LEGO)

Е.С. ГЕРАЩЕНКО
Рязанский государственный радиотехнический университет

CWDM КАК ВАЖНЕЙШАЯ ТЕХНОЛОГИЯ В ПОСТРОЕНИИ СЕТЕЙ ГОРОДСКОГО СЕГМЕНТА

Рассматривается технология грубого спектрального мультиплексирования (CWDM) с точки зрения применимости в городском сегменте волоконно-оптических сетей.

В настоящее время размер пользовательских сетей, обеспечивающих потребителям доступ к самым разным сервисам, среди которых - Интернет и цифровое телевиденье, очень быстро увеличивается. Часто провайдеры сталкиваются с ситуацией, когда прокладывание новых кабелей может обернуться большими денежными и трудовыми затратами, например, в случае необходимости перестройки топологии сети. В подобных случаях всё чаще и чаще используются технологии класса WDM (wavelength-division multiplexing, мультиплексирование с разделением по длине волны), которые позволяют осуществить передачу нескольких информационных каналов по одному оптическому волокну, используя разные частоты для каждого из них, а значит, решить одну из самых значимых проблем в данной области – проблему нехватки пропускной способности. В сегменте городских сетей применяется CWDM (coarse WDM, грубое спектральное мультиплексирование).

Технология CWDM позволяет использовать до 18 оптических каналов, которые находятся на расстоянии 20 нм друг от друга в диапазоне от 1271 до 1611 нм. На практике наиболее часто используют диапазон 1471-1611 нм, так как в левой его части наблюдается достаточно сильное затухание.

С технической точки зрения, для того, чтобы организовать CWDM в сети, используются специальные устройства, например, модули XFP и SFP+ с поддержкой технологии CWDM для различных длин волн в выделенном диапазоне. В зависимости от класса модели они могут передавать данные на расстояние до 70 км.

Существует довольно важный аспект, который надо учитывать при применении подобных модулей. Они в основном являются однонаправленными, следовательно, необходимо применение мультиплексора на другом конце линии связи. Существуют две реализации мультиплексоров:

- одноволоконные;
- двухволоконные.

Выбор реализации для конкретного сетевого решения зависит от типа используемого оптического кабеля.

Двухволоконные мультиплексоры осуществляют прием и передачу сигналов на одной и той же длине волны по разным волокнам одномодового кабеля. После того, как мультиплексор получает сигналы от модулей, они передаются в отдельных оптических каналах по одному волокну оптического кабеля. По второму волокну осуществляется прием уже мультиплексированных сигналов, которые впоследствии демультиплексируются. Такой вариант позволяет осуществлять двунаправленную передачу данных – к модулю CWDM и от него.

Можно выделить следующие особенности технологии CWDM, которые являются её достоинствами:

- увеличивает срок службы волоконно-оптических сетей за счёт того, что использует сетку частот, которая не используется обычными приёмопередатчиками;
- не зависит от протокола передачи информации, следовательно, обеспечивает большую гибкость в плане передачи различных телекоммуникационных услуг в единой среде. Наиболее простой пример – объединение канала передачи интернет-трафика и цифрового телевидения;
- стоимость активных и пассивных компонентов оборудования сравнительно невелика за счёт использования большего частотного расстояния между каналами (например, по сравнению с родственной технологией DWDM);
- обеспечивает гибкость построения различных топологий, что позволяет конструировать и модифицировать сети волоконно-оптической связи;
- повышает эффективность использования волокна за счёт увеличения объёма передаваемого трафика в 8 раз.

Безусловно, в настоящее время существуют и другие варианты решения подобных проблем, которые, возможно, являются более эффективными. В частности, это DWDM (dense WDM, плотное спектральное мультиплексирование) и HDWDM (high dense WDM, высокоплотное спектральное мультиплексирование). Они развиваются достаточно интенсивно и являются весьма перспективными. Например, ведутся работы по созданию DWDM-системы с пропускной способностью 27 Тбит/с. Однако эти технологии требует больших материальных затрат. Кроме того, они рассчитаны на передачу на сверхдальние расстояния, например, в магистральных сетях, поэтому в рамках городских сетей использование CWDM является более компактным и экономически выгодным решением.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. CWDM – технология уплотнения оптических каналов. [Электронный ресурс] http://shop.nag.ru/catalog/article/id/4/catalog_id/43.

2. Технологии коммутации и маршрутизации современных сетей Ethernet. Базовый курс D-link. М., 2014. С. 21-22.

3. DWDM – система 27 Тбит/с. [Электронный ресурс] http://t8.ru/?page_id=4563.

4. Фриман Р. Волоконно-оптические системы связи. М.:Техносфера, 2003. С.285-289.

А.М. ГОСТИН, П.А. ГОСТИН, А.С. ШУМИЛОВА
Рязанский государственный радиотехнический университет

РАЗРАБОТКА ПАРСЕРА РАБОЧИХ УЧЕБНЫХ ПЛАНОВ С ПРИМЕНЕНИЕМ ПРИНЦИПОВ ФУНКЦИОНАЛЬНО-РЕАКТИВНОГО ПРОГРАММИРОВАНИЯ НА ЯЗЫКЕ PHP

Рассматриваются принципы функционально-реактивного программирования, реализованные на языке PHP, а также их практическое применение для создания парсера рабочих учебных планов.

В соответствии с федеральными образовательными стандартами нового поколения, каждый обучающийся в течение всего периода обучения должен быть обеспечен индивидуальным неограниченным доступом к электронной информационно-образовательной среде (ЭИОС) университета. Центральную часть ЭИОС должен занимать образовательный портал [1]. Для функционирования образовательного портала

необходимо внести информацию об образовательных программах, реализуемых в университете. Эти данные доступны в рабочих учебных планах, формируемых соответствующим подразделением ВУЗа (УМУ) в формате электронных таблиц. Для импорта данных в базу данных необходимо конвертировать файлы в универсальный текстовый формат CSV и разработать парсер, который получит всю необходимую информацию.

Разработка парсера усложняется тем, что из-за различных требований, предъявляемых нормативными документами каждый год к оформлению рабочих учебных планов, формат документов год от года меняется. Таким образом, разрабатываемый парсер должен позволять достаточно универсально обрабатывать входные данные и иметь простую и гибкую структуру для внесения изменений в будущем. В связи с этим, наилучшим средством для достижения поставленной задачи является функционально-реактивное программирование.

Функционально-реактивное программирование (*Functional reactive programming,* FRP) – это парадигма программирования, основанная на шаблоне проектирования наблюдатель (*Observer*) с одной стороны и принципах функционального программирования с другой.

В FRP существует 2 понятия: поведение (*behavior*) и событие (*event*). Поведение (или характеристика) - это меняющееся со временем значение, а событие – это поток происшествий в порядке их появления [2].

Основные принципы FRP:

- явные состояния,
- распространение изменений,
- работа не данными, а с источниками данных [3].

В данном контексте источники данных – это потоки событий, а сами данные – это состояние потока в определенное время. Изменение данных влечет за собой немедленное изменение других данных, которые зависят от первых. По итогу получается структурное дерево описывающее зависимость одних данных от других, что делает систему прозрачной и легкоусвояемой [3].

Таким образом, входными данными для парсера является поток данных (CSV-файл). Поток – это последовательность событий, отсортированная по времени. Парсер подписывается (*subscribing*) на поток, слушает (*listening*) его и реагирует на соответствующие события определенным поведением [2].

Для вызова обработчика используется метод *callback*, которому передается лямбда-функция для выполнения с параметром строки *$row* из потока.

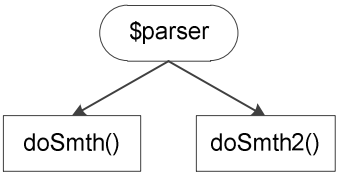
```
$parser->callback( function($row) {
  list($line, $line_str) = $row;
});
```

Потоки событий также можно фильтровать. Данная библиотека также позволяет отфильтровать события, оставляя только те, для которых функция возвращает истину (*true*).

```
$parser->filter( function() {
  return $condition;
})->callback( function($row) {
  doSmth();
});
```

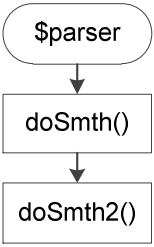
Кроме того, можно несколько раз подписываться на один поток, формируя дерево событий (рис. 1).

```
$parser->callback( function($row) {
  return doSmth();});
$parser->callback( function($row) {
  return doSmth2();});
```



а)

```
$parser->callback( function($row) {
  return doSmth();})
 ->callback(
 function($row) {
  return doSmth2();});
```



б)

Рис. 1 – Дерево событий.

В случае б) обработчику передается строка *$row* из потока, а затем происходит вызов функции *doSmth()*. Если функция вернет ложь (*false*), тогда обработка данной строки будет завершена.

Потоки могут быть использованы как входные параметры друг друга – к примеру, можно отфильтровать один поток, чтобы получить другой, который содержит только необходимые данные (рис. 2).

```
$rows_spec = $parser->filter( function() {
  return hasSpec();});
```

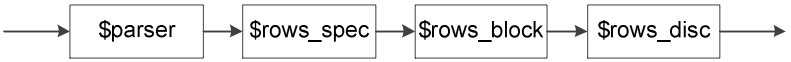


Рис. 2 – Поток событий парсера.

Преимущества данного решения:

- гибкость – код легко дополняется путем добавления новых операторов фильтрации и реализующих их поведение методов-обработчиков;
- работа с источниками данных и ориентированность на события позволяет строить простые для понимания деревья событий и использовать отфильтрованный поток в качестве новых данных.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Электронная информационно-образовательная среда университета / [Гостин А.М., Гуров В.С. и др.] // НИТ-2015. Рязань, 2015. – С. 3-8.

2. Брайко Ю. Вступление в Реактивное Программирование, которое вы пропустили [Электронный ресурс] – Электрон. текстовые дан. – Режим доступа: https://habrahabr.ru/post/279715/, свободный.

3. FRP (functional reactive programming) на Bacon.js [Электронный ресурс] – Электрон. текстовые дан. – Режим доступа: https://habrahabr.ru/post/198656/, свободный.

Д.С. ДЕМИДОВ
Рязанский государственный радиотехнический университет

ДЕТАЛИ РЕАЛИЗАЦИИ МОДЕЛИ МНОГОПОРОГОВОГО ДЕКОДЕРА С ПОМОЩЬЮ OPENCL

Рассматриваются детали использования вычислительных ресурсов GPU (технологии OpenCL) для моделирования работы системы передачи цифровых данных с многопороговым декодером.

Введение

В последнее время появилось достаточно много материалов, посвященных моделированию работы многопорогового декодера (МПД)[1]. В частности, большой интерес предоставляет использование вычислительных для этих целей ресурсов графических процессоров ПК. Исходя из этого интересен процесс адаптации модели системы передачи цифровых данных с МПД к многопоточной архитектуре GPU, которая позволяет параллельно запускать множество вычисли-

тельных потоков. Добиться эффективного использования вычислительных ресурсов GPUможно при помощи технологии OpenCL, которая является разработкой компании «**KhronosGroup**» [2].

CPU и GPU

Концептуальным проблемой адаптации вычислений с CPU, к GPU, являются базовые архитектурные различия этих двух видов процессоров. В частности, каждое ядро у многоядерного CPUявляется независимым, и даже в случае запуска параллельных вычислений на каждом из ядер вычисления будут производиться без жесткой привязки к процессам проходящим на других ядрах. Число ядер даже во флагманских моделях CPUот ведущих мировых производителей не превышает 8, максимум 16 единиц. Это обусловлено высокой стоимость одного ядра, которое является универсальным вычислительным устройством, способным выполнять различные операции, используемые для решения широкого спектра задач.

GPU же основано на абсолютно другой концепции, которая подразумевает одновременное выполнение множества параллельных потоков с абсолютно одинаковым набором инструкций. Количество вычислительных потоков (также называемых конвейерами), может достигать нескольких тысяч, на современных видеокартах. Конвейеры не являются аналогом ядер в CPU, а предназначены для проведения элементарных операций над обрабатываемыми объектами, к примеру, над изображениями. Универсальность CPU-ядер конвейерам просто не нужна, так приблизительный набор операции используемых для работы с компьютерной графикой известен.

Данная концепция архитектуры и принципа работы исходит из решения основной задачи GPU– обработки и отображение компьютерной графики. Именно по этой причине GPUи имеет множество параллельно работающих конвейеров, которые выполняют набор элементарных операций над каждым пикселем, обрабатываемого изображения. Различия CPU и GPU хорошо видны на рисунке 1.

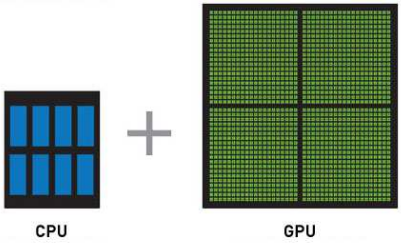


Рис.1 – Конвейеры GPU и ядра CPU.

Таким образом, главной задачей адаптации модели МПД является перевод вычислений с использования универсальных ядер CPUна использование узкоспециализированных конвейеров GPU.

Анализ работы CPU

При параллельном запуске на CPU, наборы инструкции на разных ядрах будут выполняться независимо друг от друга, если конечно какая-то синхронизация не была дополнительно запрограммирована (рис. 2). Таким образом общее время выполнения программы будет равняться максимальному времени, затраченному одним из ядер, на выполнение всех наборов инструкций.

$$T = \max(t_1, t_2, t_3, t_4) \qquad (1)$$

$$t_i = q_1^i + q_2^i + \ldots + q_n^i$$

где T – общее время выполнения вычислений, t_i- время выполнения всех наборов инструкций на i-ом ядре, q^i_j- время выполнения j-ой инструкции на i-ом ядре, n–число наборов инструкций.

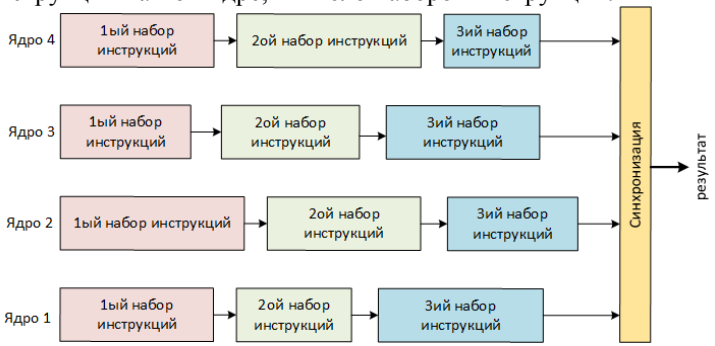


Рис.2 – Пример параллельных вычислений на CPU.

На рисунке 2 представлен идеальный случай, в котором каждое ядро выполняет полный набор инструкций, последовательно (в данном случае 1-ый, 2-ой и 3-ий набор инструкций). На практике же наборы инструкций могут быть «раскиданы» по ядрам CPUв любой последовательности, но коренным образом это ничего не изменит, так как формула (1) останется актуальной.

Анализ работы GPU

В работе же GPU лежат абсолютно другие принципы. Так на все конвейеры при параллельной работе могут выполнять только один набор инструкций. Более того переход к следующему набору инструкций невозможен пока не завершится выполнение текущего набора инструкций. Иными словами, если какой-то набор инструкций имеет в своей логике, к примеру, оператор ветвления, в котором побочная

ветвь выполняется допустим крайне редко, но имеет большую вычислительную сложность, то GPUне перейдет к следующему набору инструкций пока не дождется окончания выполнения всех ветвей. Данный алгоритм работы наглядно показан на рисунке 3[3].

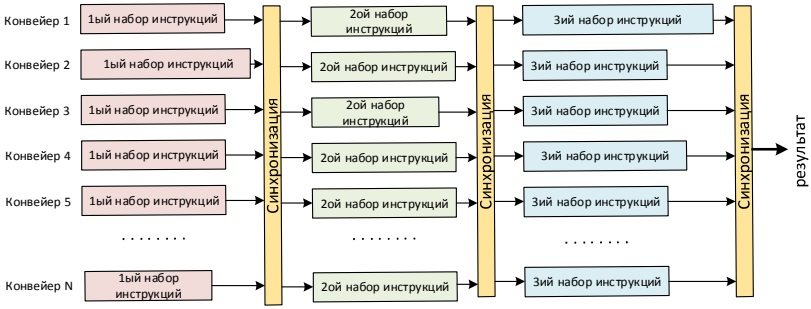


Рис. 3 – Пример параллельных вычислений на GPU.

Очевидно, что общее время выполнения вычислений будет вычисляться по формуле.

$$T = Q_1 + Q_2 + Q_3 \qquad (2)$$

$$Q_i = \max(q_i^1 + q_i^2 + ... + q_i^n)$$

Где T – общее время вычислений, Q_i– время, потраченное на вычисление i-ого набора инструкций, q^j_i–время затраченное на вычисление i-го набора инструкций на j-ом конвейере, n – число конвейеров.

Кроме того, необходимо учитывать вычислительную сложность наборов инструкций. Для наборов инструкций с высокой вычислительной сложностью и как следствие с большим временем, затрачиваемым на их выполнение, сложно добиться максимально производительности. Это связано с невозможностью запуска большого количества параллельно выполняемых инструкций. Данное ограничение вызвано невозможностью затраты слишком большого времени(более 10 секунд) на вычисления. Так как, в таком случае, внутренние средства ОС Windows могут решить, что выполняемый код может нанести вред операционной системе и самой GPU, и прервёт его выполнение. Таким образом необходимо избегать выполнения на GPUслишком объемных наборов инструкций.

Моделирование

Модель системы передачи цифровой информации содержит несколько сложных с алгоритмической точки зрения узлов:

– кодер;

– декодер;

– источник данных, канал передачи данных (который включает в себя генератор случайных чисел).

Все из перечисленных компонентов могут иметь в своей реализации оператор ветвления, который естественно изменит время выполнения i-го набора инструкций на j-ом конвейере q^j_i.

Начальная реализация использовала для всего процесса моделирования единый набор инструкций, который включал в себя все узлы. Что очевидно не оптимально с точки зрения формулы (2). Так же высокие объемы вычислений не позволяли получить максимальный эффект от параллельных вычислений. В связи с этим моделирование с использованием GPU - *AMD Radeon HD 6300M Series*доле следующие результаты, для МПД в различных конфигурациях, что отражено в таблице 1. Так же таблица 1 содержит результаты моделирования тех же конфигураций с использованием CPU*IntelCorei3*. Под скоростью моделирования понималось количество переданных через модель бит за одну секунду.

Таблица 1. Результаты моделирования.

Кодовое расстояние, d	Скорость моделирования, GPU, кбит/с	Скорость моделирования, CPU, кбит/с
11	3250	624
19	1950	450
23	3485	517
10	2011	450

Данные результаты требовали улучшения. В среднем по сравнению с результатами CPU мы получали прирост не более чем в 5 раз. При оптимизации была полностью переработана программная реализация модели. А именно:

- каждый узел вынесен в отдельный набор инструкций;
- уменьшено использование числа операторов ветвления;
- минимизировано число обращений к памяти;

В результате получилось значительно улучшить результаты. Несмотря на то, что не дулось полностью уйти от операторов ветвления в программной реализации. Это связано с тем, что алгоритм декодирования требует оператора ветвления на этапе подсчета значений на пороговом элементе. Уход от данного оператора ветвления означал бы несоответствие модели реальному алгоритму МПД. Оптимизированные результаты представлены в таблице 2.

Таблица 2 – Результаты моделирования после оптимизации.

D	Скорость моделирования, кбит/с
11	6180
19	4016
23	6321
10	3840

Представленные результаты дали увеличение скорости моделирования на один порядок по сравнению с CPU.

Вывод

Адаптация процесса моделирования для GPUявляется нетривиальной задачей, которая исключает механический перевод алгоритмов, используемых для реализации модели с использованием обычных языках высокого уровня, таких как C++, C# и т.д. Процесс адаптации требует нестандартных алгоритмических решений, однако, грамотное соблюдение особенностей GPU позволяет получить впечатляющие результаты. Очевидно, что данные результаты не являются окончательными и имеют большой потенциал для улучшения.

Работа выполнена при поддержке РФФИ (грант №14-07-00824).

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Демидов Д.С., Овечкин Г.В. Моделирование системы передачи данных с многопороговым декодером с использованием OpenCL // Фундаментальные исследования. – 2015. – № 12-2. – С. 243-247. URL: http://www.fundamental-research.ru/ru/article/view?id=39398 (дата обращения: 20.01.2016).

2. The OpenCL Specification. Version: 2.0. Document Revision: 22 / Khronos OpenCL Working Group, Editor: AaftabMunshi, 2014, 483 p.

3. RyojiTsuchiyama, Takashi Nakamura, TakuroIizuka, Akihiro Asahara, Jeongdo Son, Satoshi Miki, The OpenCL Programming Book, Fixstar, 2014, 324p.

М.И. ЕРЕМИН, А.Е. БОРЗЕНКО

Рязанский государственный радиотехнический университет

ИСПОЛЬЗОВАНИЕ СИТУАЦИОННЫХ МОДЕЛЕЙ ПРИ РАЗРАБОТКЕ ОБУЧАЮЩИХ СИСТЕМ

Рассматриваются теоретические и практические вопросы реализации программы графического редактора ситуационных обучающих сцен.

Методика вовлечения в учебную программу детального и глубокого изучения реальной, либо имитированной ситуации направлена на выявление её частных и/или общих характерных свойств. Это позволяет развивать аналитические способности учащихся, а также осуществлять системный подход к решению той или иной проблемы. Кроме того, это помогает находить варианты не только правильных, но и ошибочных решений, и тем самым отслеживать критерии верных решений. Данная методика, которая вначале применялась лишь в бизнес-школах, была разработана в 80-е годы XX века в Гарварде. В настоящее время она широко используется и в профессиональном образовании.

Опыт и навыки полученные в результате прохождения обучения с использованием ситуационных обучающих сцен позволяют предвидеть различные варианты развития ситуаций и как следствие значительно снизить возникновение внештатных ситуаций.

Под ситуацией понимается множество оперативных элементов, которые располагаются в определённых точках статической системы. Под текущей ситуацией принято считать совокупность всех сведений о структуре объекта и его функционировании в текущий момент времени [1-2].

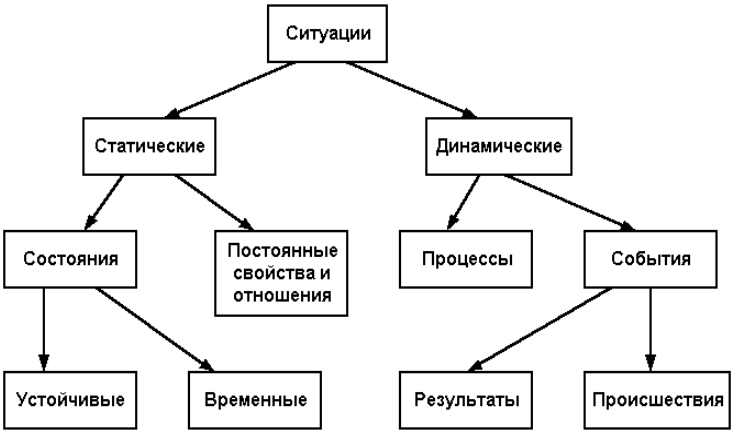


Рис. 1 – Классификация ситуаций.

Можно выделить два важных свойства ситуации: множественность и неоднородность исходных данных. Ситуация всегда представляет собой некую оценку (анализ, обобщение) множества данных, и эта оценка является субъективной, т.к. она зависит от средств и методов обобщения конкретного человека (человеко-машинной системы).

Ситуация системы есть оценка (анализ, обобщение) совокупности характеристик объектов и связей между ними, которые состоят из постоянных и причинно-следственных отношений, зависящих от произошедших событий и протекающих процессов.

Обобщенное описание (отображение) системы с помощью ситуаций называется ситуационной моделью (СМ).

Любую программу, где создается модель, или устройство, которое транслирует реальные объекты, можно назвать СМ, ситуационной моделью или ситуационной комнатой.

Ситуационные центры классифицируются следующим образом:

- по составу СМ;
- по масштабу;
- по размещению;
- по целевой направленности;
- по универсальности.

Одним из эффективных путей повышения эффективности обучения является использование интеллектуальных обучающих систем (ИОС). Такие системы предоставляют обучаемому дружественную среду для самостоятельной работы и экспериментов в рассматриваемой предметной области [2, 3].

Компьютерные комплексы, используемые в ИОС, обеспечивают процесс обучения, в максимальной степени приближенный к реальным ситуациям и позволяющий приобрести правильные и устойчивые знания путем демонстрации физических закономерностей сложных процессов и явлений. На основе математической модели, заложенной в комплекс, рассчитываются координаты и операции всех объектов динамической сцены. Параметры передаются по сети системам визуализации всех каналов, которые воспроизводят ситуацию в реальном режиме времени. Подобный режим работы предполагает довольно жёсткие требования к системам визуализации, работающим в составе вычислительно-имитационного комплекса. При этом особое значение имеет хорошее качество визуализации виртуальной модели объекта и окружающей среды и работа системы в режиме реального времени.

Довольно большое внимание при работе ИОС уделяется организации интерфейса «Обучаемый – ИОС». Наиболее сложным моментом является разработка понятного диалога, содержащего компоненты, связанные с разработкой, редактированием и генерацией сценариев диалога в не штатных ситуациях, где необходимо привлечение графической информации, в том числе и когнитивной компьютерной графики. С учетом механизма адаптации под конкретного обучаемого формирование сценария диалога и декомпозиция процесса взаимодействия

«Обучаемый – ИОС» придаёт интерфейсу свойства интеллектуальности. Интеллектуальный интерфейс позволяет решать такие сложные задачи, как обнаружение и идентификация ситуации, оценка ее опасности, выдача практических рекомендаций и их корректировка. Особенно сложно это в нестандартных и экстремальных ситуациях, когда интеллектуальная обучающая система испытывает сложности во время логического вывода [3-5].

В условиях неопределенности и неполноты исходной информации обучаемому приходится принимать нестандартные решения, основываясь на фактических характеристиках внешних условий и динамики ситуации. Интеллектуальный интерфейс в такой ситуации снабжает обучаемого всеми необходимыми доступными данными, включая результаты имитационного моделирования взаимодействия исследуемого динамического объекта с внешней средой для различных вариантов при выборе решения.

Несмотря на ряд преимуществ, такая технология обучения, имеет и свои недостатки. Они заключается в том, что обучение, как правило, происходит, на тех материалах, которые имеются у преподавателя, что зачастую сводит к нулю эффективность анализа, так как конкретные сведения о самой ситуации могут у него отсутствовать.

В заключении следует отметить, что и ситуационные системы, и интеллектуальные обучающие системы продолжают активно развиваться с внедрением большого количества новых технологий [1, 4 ,5].

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1.Поспелов Д.А, Ситуационное управление: теория и практика. М.: Наука, 1986.

2.Клыков Ю.И, Ситуационное управление большими системами. М.: Энергия, 1974.

3.Падучева Е.В, Семантические типы ситуаций и значение всегда //Семантика и информатика. — 1985. — Вып. 24. С. 96-116.

4.Поспелов Д.А , 1990. Справочник по искусственному интеллекту. / Под ред Поспелова Д.А. т.2. 1990.

5.Козелецкий Ю, Психологическая теория решений . М. Прогресс, 1979.

С.К. ЕРМОЛЕНКО
Рязанский государственный радиотехнический университет

ОБЗОР АЛГОРИТМОВ СЕГМЕНТАЦИИ СТРУКТУР КОМПЬЮТЕРНЫХ СЕТЕЙ

В статье представлен обзор алгоритмов сегментации компьютерных сетей. Особенностью обзора является широкое использование для анализа работ российских и зарубежных авторов, изданных за последнее время, что должно показать текущий уровень исследований и помочь определить потенциальные точки их дальнейшего развития. Классифицированы основные алгоритмы сегментации компьютерных сетей. Выделены особенности, достоинства и недостатки алгоритмов сегментации, обусловлена их трудоемкость.

Задача сегментации графов на непересекающиеся подмножества вершин актуальна во многих областях современной науки и техники. Разбиение графа используется при сегментации изображений для их последующего анализа и интеллектуальной обработки. Широкое применение задача сегментации нашла также в социальных, информационных и биохимических сетях, где часто требуется выявлять в структуре сети сильно связанные сообщества. Однако одной из самых распространенных областей науки, в которой проблема разбиения графа имеет важное значение, являются компьютерные сети. Для улучшения маршрутизации и разбиения на широковещательные домены сеть разделяется на отдельные непересекающиеся подмножества элементов (компьютеров или роутеров), причем количество связей внутри этих подмножеств должно превышать количество внешних связей с другими подмножествами. Оптимальность разбиения зависит от выбора алгоритма сегментации. В данной статье будут рассмотрены комбинаторные алгоритмы сегментации структур компьютерных сетей, так как они обеспечивают более сбалансированное разбиение и меньшее информационное взаимодействие полученных подсетей по сравнению с другими алгоритмами и используют информацию о связности вершин в графе.

Постановку задачи сегментации графов можно сформулировать следующим образом. Представим компьютерную сеть в виде неориентированного взвешенного связанного графа $G = (V, E)$, каждой вершине $v \in V$ и каждому ребру $e \in E$ которого приписан вес. Задача оптимального разделения графа состоит в разбиении его вершин на непересекающиеся подмножества с максимально близкими суммарными весами вершин и минимальным суммарным весом ребер, проходящих между полученными подмножествами вершин. Следует отметить возможную противоречивость указанных критериев разбиения графа –

равновесность подмножеств вершин может не соответствовать минимальности весов граничных ребер и наоборот. В большинстве случаев необходимым является выбор того или иного компромиссного решения.

Задача разбиения графа относится к классу NP-полных, верхняя оценка числа разбиений определяется числом Белла.

Рассмотрим следующие комбинаторные алгоритмы сегментации структур компьютерных сетей:

- алгоритм рекурсивного деления пополам;
- «жадный» алгоритм разбиения графов;
- спектральный алгоритм Ньюмана;
- алгоритм Гирвана – Ньюмана;
- усовершенствованный алгоритм сегментации структур корпоративных сетей по критерию минимальной стоимости.

Алгоритм рекурсивного деления пополам относится к числу самых простых и наименее трудоемких алгоритмов сегментации. В его основе лежит метод бинарного деления, при котором на первой итерации граф разделяется на две равные части, на втором шаге каждая из полученных частей также разбивается на две части, аналогичное разбиение производится и на всех последующих итерациях. Применительно к данному алгоритму сначала находится диаметр графа, определяемый как наибольшее расстояние между вершинами графа. Расстояние между двумя вершинами графа определяется как наименьшее число граней, которые нужно пройти от узла i к узлу j. Начиная с одной из вершин, половина вершин присваивается одной из подобластей, а половина - другой подобласти. Затем рекурсивная процедура применяется к каждой из подобластей. Время работы алгоритма оценивается как $O(N)$, где N – число вершин графа или маршрутизаторов в компьютерной сети [1].

Для нахождения диаметра графа, начиная с исходной вершины v_s, принятой за потенциальный корень, каждой соседней вершине присваивается метка 1, а соседям соседей присваивается метка 2. Последняя из помеченных вершин имеет метку m, которая имеет смысл расстояния от корневой вершины. Далее следующая вершина v_i обозначается как исходная, процесс повторяется. Вершина v_i с максимальным значением m считается корнем, при этом m является диаметром графа. Реализация алгоритма нахождения диаметра графа также требует $O(N)$ операций.

Пример работы алгоритма рекурсивного деления пополам показан на рис. 1. Исходный граф разбит на $k = 3$ части. Диаметр графа равен $m = 3$, вершина v_0 является корнем графа.

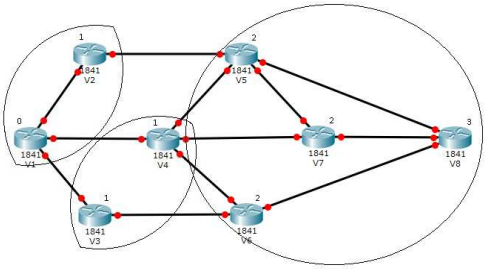


Рис. 1 – Алгоритм рекурсивного деления пополам.

Вершина v_4 с меткой 1 будет являться границей раздела исходного графа на 2 части. Произведем разбиение графа. Далее перейдем к первому подграфу и применим к нему процедуру разбиения. Диаметр подграфа будет равен $m = 2$, вершина v_2 будет являться корнем подграфа. В соответствии с вычисленными метками вершин данный подграф разделится ещё на 2 области: в первой области будут расположены вершины v_1 и v_2, во второй - вершины v_3 и v_4. В результате работы алгоритма исходный граф разобьется на 3 подграфа.

Однако алгоритм рекурсивного деления пополам имеет существенный недостаток: в нем мало учитывается связность вершин, и получаемое в итоге разбиение оказывается неоптимальным.

«Жадный» алгоритм разбиения заключается в принятии локально оптимальных решений на каждом этапе, допуская, что конечное решение также окажется оптимальным [1].

Процедура начинается с вершины, имеющей наименьшую степень. Маркируются все соседние вершины, а затем вершины, соседние с соседями. Первые N/k маркированных вершин присваиваются одной подобласти, и процедура применяется к оставшейся части графа до тех пор, пока все вершины не являются маркированными. Данный алгоритм имеет схожие черты с алгоритмом рекурсивного деления пополам, хотя и не является таковым. Трудоёмкость алгоритма оценивается как $O(N)$ [1].

Пример работы «жадного» алгоритма разбиения показан на рис. 2. Разобьем исходный граф (рис. 1) на $k = 3$ части. Разбиение будем начинать с вершины v_3, так как она имеет самую низкую степень, равную 2. Произведем маркировку всех вершин графа и выделим $8/3 = 2$ вершины с наименьшими пометками в отдельное подмножество.

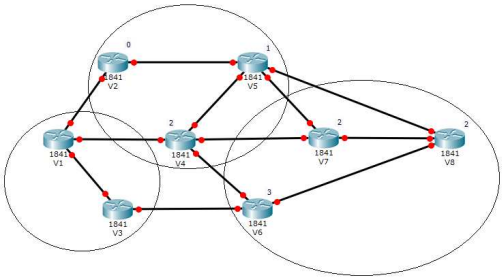


Рис. 2 – Результат разбиения исходного графа на 3 части.

Повторим процедуру для оставшегося подграфа. Разбиение начнем с вершины v_2, с самой низкой степенью, равной 2. Произведем маркировку всех вершин графа и выделим 6/2 = 3 вершины с наименьшими пометками в отдельное подмножество. Результат разбиения графа показан на рис. 2.

Преимуществом данного алгоритма является простота и высокая скорость работы, однако итоговое разбиение графа получается неоптимальным, что является весомым недостатком при сегментации компьютерных сетей.

Спектральный алгоритм Ньюмана основан на использовании собственных векторов и собственных значений и производит разбиение исходного графа с минимизацией количество внешних ребер [2].

В отличие от ранее рассмотренных алгоритмов спектральный алгоритм Ньюмана приводит к связным подобластям при условии, что исходный граф также является связным, при этом число ребер графа, разрезаемых при сегментации области, оказывается примерно в два раза меньшим по сравнению с остальными алгоритмами. Трудоёмкость нахождения собственных векторов оценивается как $O(N\log N)$.

Алгоритм Гирвана-Ньюмана основывается на понятии центральности вершин и производит разбиение графа в соответствии с данной величиной. Центральность – мера важности вершины в графе (применительно к сетевым сообществам) [3]. Для определения данной величины рассчитывается смежность ребер графа – мера важности, пропорциональная числу кратчайших путей от всех вершин до всех, проходящих через ребро.

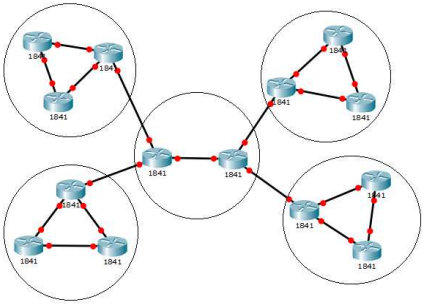


Рис. 3 – Разбиение графа алгоритмом Гирвана – Ньюмана.

Алгоритм Гирвана - Ньюмана выполняет иерархическую кластеризацию дивизионным методом. Результаты разбиения структуры сети при помощи данного алгоритма показаны на рис. 3. Однако существенным недостатком данного способа разбиения является высокая вычислительная сложность.

Усовершенствованный алгоритм сегментации структур компьютерных сетей по критерию минимальной стоимости [5].

Математическую модель корпоративной сети представим в виде неориентированного взвешенного связного графа $G = (V, E, W)$, где V – множество вершин (узлов связи или маршрутизаторов), $|V| = N$, E – множество ребер (каналов или линий связи), $|E| = M$, W – множество весов ребер (стоимость каналов связи между узлами).

В соответствии с алгоритмом [4] используем следующие понятия. Обозначим через G_1 такой связный подграф графа G, для которого $G_1 = (\mathrm{V}, E_1, W_1)$, где V – множество вершин подграфа (узлов связи или маршрутизаторов), $|V| = N$, E_1 – множество ребер подграфа (каналов или линий связи), $|E_1| = M_1$, W_1 – множество весов ребер подграфа (стоимость каналов связи между узлами). Степень узла связи $\deg(v_i)$ подграфа $G_1 \subset G$ – число неповторяющихся каналов связи $e_{i,j} \in E$, инцидентных узлу v_i. Узел связи v_i называется листом дерева (подграфа G_1), если его степень $\deg(v_i) = 1$. Потомком называется часть минимального покрывающего дерева, имеющая узел-предок. Предком называется узел связи, имеющий одного и более потомков. Величиной связности сегментов Q называется отношение количества связей внутри сегмента к числу внешних связей у данного сегмента.

Усовершенствование алгоритма сегментации структур компьютерных сетей [4] заключается в введении нового порядка распределения свободных узлов по сегментам и объединения сегментов. Перед каждым очередным изменением структуры уже существующих сегментов просматривается весь граф, а лишь затем производится при-

соединение узлов или объединение сегментов. Исходный узел для расчетов требуется только для осуществления алгоритма Прима, для алгоритма сегментации такой узел не нужен. В результате удается получить минимально возможные веса внутренних каналов сегментов по отношению к весам межсегментных каналов связи.

Итоговое разбиение графа G на сегменты приведено на рис. 4.

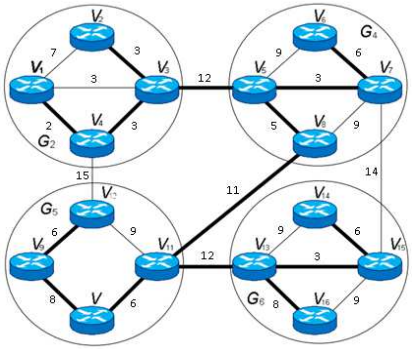


Рис. 4 – Итоговое разбиение графа *G* на сегменты.

Рассмотренный алгоритм наиболее отчетливо проявляется при исследовании частичносвязных топологий компьютерных сетей, на древовидных и на полносвязных топологиях результаты работы предложенного алгоритма менее заметны. Действительно, при любых изменениях в каналах связи сети древовидной структуры минимальный остов не изменится, а в случае применения алгоритма к полносвязным топологиям, предполагающим высокий уровень связности между узлами, уменьшается вклад значения величины связности Q в результирующее разбиение сети на подсети. Анализ трудоемкости алгоритма сегментации [5] показывает его эффективность по сравнению с известными алгоритмами и составляет величину $O(kN\log N)$.

В таблице приведен сравнительный анализ рассмотренных алгоритмов. В представленной таблице используются следующие обозначения: N – число узлов связи или маршрутизаторов в сети, M – число каналов или линий связи в сети, k – число полученных сегментов при разбиении.

Таблица 1. Сравнительный анализ алгоритмов сегментации

Наименование алгоритма	Метод решения	Преимущества	Недостатки	Трудоемкость
Алгоритм рекурсивного деления пополам	Граф разделяется на две равные части, на втором шаге каждая из получен-	Самый простой и наименее трудоемкий	Мало учитывается связность	$O(N)$

	ных частей также разбивается на две части, аналогичное разбиение производится и на всех последующих итерациях		вершин, и получаемое в итоге разбиение оказывается неоптимальным	
«Жадный» алгоритм разбиения	Процедура начинается с вершины, имеющая наименьшую степень. Маркируются все соседние вершины, а затем вершины, соседние с соседями. Первые N/k маркированных вершин присваиваются одной подобласти, и процедура применяется к оставшейся части графа до тех пор, пока все вершины не являются маркированными	Простота и высокая скорость работы	Итоговое разбиение графа получается неоптимальным	$O(N)$
Спектральный алгоритм Ньюмана	Производит разбиение исходного графа, минимизировав при этом количество внешних ребер	Приводит к связным подобластям при условии, что исходный граф также является связным, при этом число ребер графа, разрезаемых при сегментации оказывается в два раза меньше, чем в остальных алгоритмах	Высокая трудоемкость	$O(N \log N)$ $O((k\text{-}1)\ N \log N)$
Алгоритм Гирвана-Ньюмана	Основывается на понятии центральности вершин и производит разбиение графа в соответствии с данной величиной	Отсутствие необходимости в предварительном обучении	Высокая вычислительная сложность	$O(NM\text{^}2)$
Усовершенствованный алгоритм сегментации структур ком-	Введение нового порядка распределения свободных узлов по сегментам и объединение сег-	Минимальная стоимость, повышает эффективность процессов маршрутизации, надеж-	Эффективен при работе с частичносвяз-	$O(kN \log N)$

пьютерных сетей	ментов. Перед каждым очередным изменением структуры уже существующих сегментов просматривается весь граф, а лишь затем производится присоединение узлов или объединение сегментов	ность и безопасность передачи данных, использование дополнительной информации о структуре базовой сети и связности каналов связи	ными топологиями	

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Беликов Д.А., Говязов И.В., Данилкин Е.А., Лаева В.И., Проханов С.А., Старченко А.В. Высокопроизводительные вычисления на кластерах. Томск: Изд-во ТГУ. 2008.

2. M. E. J. Newman and M. Girvan, Finding and evaluating community structure in networks. Phys. Rev. E 69, 026113. 2004.

3. M. E. J. Newman, Detecting community structure in networks. Eur. Phys. J., 2004, pp. 321–330.

4. Перепелкин Д.А. Алгоритм формирования оптимальных структур сегментов корпоративных сетей с учетом данных о структуре базовой сети и связности каналов связи // Вестник Рязанского государственного радиотехнического университета. 2014. № 50-1. С. 59-64.

5. Перепелкин Д.А., Цыганов И.Ю.. Усовершенствованный алгоритм сегментации структур корпоративных сетей по критерию минимальной стоимости // Вестник Рязанского государственного радиотехнического университета. 2015. № 53. С. 48-51.

С.И. ЗАИКИН
Московский государственный технический университет им. Баумана

КОНЦЕПЦИЯ ОБЕСПЕЧЕНИЯ ОТКАЗОУСТОЙЧИВОСТИ ИТ-ИНФРАСТРУКТУРЫ

Рассматривается роль отказоустойчивости ИТ-инфраструктуры в бизнесе, а также принципы и методы построения отказоустойчивых систем.

Основной задачей любого бизнеса является максимизация прибыли, для чего необходимо обеспечить непрерывность работы организации. В современном мире практически повсеместно используется вычислительная техника, которая подвержена поломкам из-за которых возникают задержки в производстве и даже полная остановка ра-

боты компаний. Возникла необходимость создания отказоустойчивых систем, способных работать непрерывно в течение всего года. Для грамотного создания подобных систем разрабатываются как нормативные документы внутри компаний, так и существуют ГОСТы различных стран, в пределах которых ведется бизнес-деятельность.

Отказоустойчивость в бизнесе

Одним из основных принципов, на котором базируется архитектура современных отказоустойчивых систем, является понятие планирования непрерывности бизнеса. Обеспечение непрерывности бизнеса является одним из важнейших стратегических направлений развития любой компании. Это обусловлено необходимостью сохранять устойчивость и стабильность функционирования компании и ее информационной системы в различных условиях неблагоприятного воздействия внешних и внутренних факторов техногенного и/или природного характера.

«Планирование непрерывности бизнеса» (Business Continuity Planning или) - Это деятельность, направленная на снижение рисков прерывания бизнеса и негативных последствий таких сбоев, восстановление бизнеса до приемлемого уровня в определенной последовательности и установленные сроки, начиная с момента прерывания. Процедура планирования непрерывности бизнеса подразумевает оценку рисков разнообразных организационных процессов, создание политик, планов и процедур для минимизации этих рисков. Главная цель BCP – поддержание основных бизнес функций компании.

Создание плана обеспечения непрерывности бизнеса позволит минимизировать риски, возникающие при:

- ошибках/отказе оборудования (например, выход из строя дискового массива);
- отказе в системе электропитания или коммуникаций;
- ошибках прикладного ПО или повреждении баз данных;
- человеческих ошибках, саботаже, террористических атаках или забастовке;
- воздействии вредоносного ПО (вирусы, троянские кони и т.п.);
- атаках хакеров;
- неконтролируемом доступе к информации.

Говоря об обеспечении непрерывности ИТ-поддержки бизнеса, стоит упомянуть также термин «Планирование деятельности после катастроф» (Disaster Recovery Planning или DRP) – заблаговременное составление плана действий, направленных на уменьшение ущерба и обеспечение бесперебойного выполнения критически важных функций организации в случае катастрофы. Термин катастрофа (disaster) в рам-

ках ИТ означает незапланированное прерывание обычных бизнес процессов, вызванное прерыванием в работе ИТ инфраструктуры.

Различие между терминами BCP и DRP в том, что BCP направлено на поддержку бизнес активности и бизнес функций, в то время как DRP - на поддержание бесперебойной работы информационных систем и сохранности данных. В то же время, BCP ориентировано на предупреждение проблем, а DRP - на их решение. Т.е. DRP направлено на восстановление ИТ инфраструктуры после сбоев, в то время как BCP охватывает вопросы восстановления бизнеса.

Для качественного выполнения плана восстановления необходимо знать допустимое время восстановления (Recovery Time Objective), а также конечные цели восстановления (Recovery Point Objective). Стратегия технического восстановления после инцидента основывается на комбинации этих требований.

Допустимая точка восстановления (RPO) определяется допускаемым уровнем потери данных в случае прерывания операций. Она показывает точку во времени, с которой можно восстановить данные. Допустимое время восстановления (RTO) определяется количеством времени неработоспособности сервиса в случае прерывания операций. Оно показывает раннюю точку времени, после которой операции могут быть продолжены. Обе концепции основываются на временных параметрах. На рисунке показаны взаимосвязи RPO и RTO.

Маленькое время до точки восстановления означает высокую стоимость реализации стратегии по резервированию. RPO равное нескольким минутам влечет применение отказоустойчивых кластерных технологий (дублирование / зеркалирование).

Маленькое время восстановления может означать необходимость иметь альтернативный Hot-Site, т.е. выделенные помещения с проложенной ЛВС, установленным и настроенным оборудованием и ПО. Маленькое RTO означает низкую толерантность к происшествиям. Толерантность к происшествиям означает интервал времени, в течение которого могут быть недоступны IT-сервисы и который может принять бизнес.

Кроме RPO и RTO существуют несколько важных дополнительных параметров, которые необходимо учитывать в стратегии восстановления. Они включают:

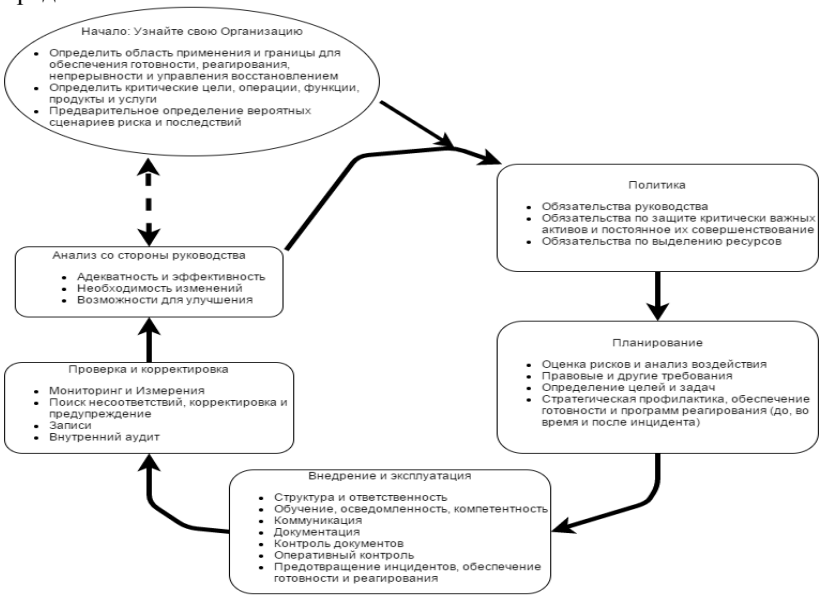
• окно недоступности сервиса (Interruption window) – ожидаемое время от начала происшествия до восстановления сервиса;

• уровень предоставления сервиса (Service delivery objective, SDO), который может быть достигнут на альтернативном оборудовании до возврата на основное;

• максимальное альтернативное время (Maximum tolerable outages) – время, в течение которого возможна работоспособность сервиса на альтернативном оборудовании. После этого времени, возможно возникновение проблем, особенно если альтернативный уровень SDO ниже основного.

Стандартизация отказоустойчивости ит-инфраструктуры

В настоящее время не только бизнес заинтересован в обеспечении отказоустойчивой ИТ-инфраструктуры, но и само государство. Наиболее продвинутой в данной области оказалось США, разработав стандарт ANSI/ASIS.SPC.1: 2009. Данный стандарт описывает общую методику организации отказоустойчивой ИТ-Инфраструктуры. В частности, предлагает использовать модель PDCA (англ. «Plan-Do-Check-Act» - планирование-действие-проверка-корректировка) для обеспечения наилучшего управления процессом организации отказоустойчивости. Наилучшим образом данный документ характеризует представленная ниже схема:



ЕС так же имеет свой стандарт управления рисками, применительно к ИТ-инфраструктуре, ETSI EG 203 251. В отличие от американского стандарта, организация ETSI, разработавшая EG 203 251 сделала акцент не столько на общей организации процесса риск-

менеджмента в ИТ-инфраструктуре, сколько тестовым мероприятиям при организации отказоустойчивой инфраструктуры.

Данный документ включает в себя основную терминологию, и теоретические знания, а также сами методы тестирования инфраструктуры на предмет устойчивости при наступлении инцидента. Так же в стандарте приводится детальная декомпозиция процесса организации риск-менеджмента инфраструктуры, которую с успехом можно применить и к области ИТ, однако прямых указаний к способам построения отказоустойчивой архитектуры системы нет.

В России, подобных стандартов пока нет, при проектировании отказоуйсточивой ИТ-инфраструктуры на территории РФ чаще всего пользуются стандартами, косвенно относящимся к данной тематике:

При проектировании вычислительных центров:

- Правила устройства электроустановок (ПУЭ)
- Инструкция по проектированию зданий и помещений для электронно¬вычислительных машин

При проектировании отказоустойчивых программных и технических средств:

- ГОСТ 28806-90
- ГОСТ 27.002-89

Однако все эти документы не дают полного представления о том, как должна выглядеть отказоустойчивая ИТ-инфраструктура, и какими конкретными методами можно ее реализовать, поэтому каждая организация вынуждена реализовывать свою политику в данном вопросе, основываясь на собственном опыте, либо опыте компаний-интеграторов подобных решений.

Методы построения отказоустойчивой ит инфраструктуры

Для обеспечения должного уровня отказоустойчивости системы существует ряд основных методик, комбинация которых позволяет достичь оптимального баланса между стоимостью и качеством решения.

Каждая ИТ-инфраструктура уникальна и невозможно вывести универсальный подход, подходящий всем без исключения, каждый раз необходимо просчитывать полный набор вариантов, проводить аудит инфраструктуры. Для удешевления сопровождения и будущего проектирования составляется документ, в котором определяют политику отнесения подсистем ИТ-инфраструктуры к определенному классу надежности или критичности, а также комплекс мер, применяемых для обеспечения должного уровня надежности каждого класса систем.

Наиболее распространённая классификация групп надежности:

- группа А – критичные;

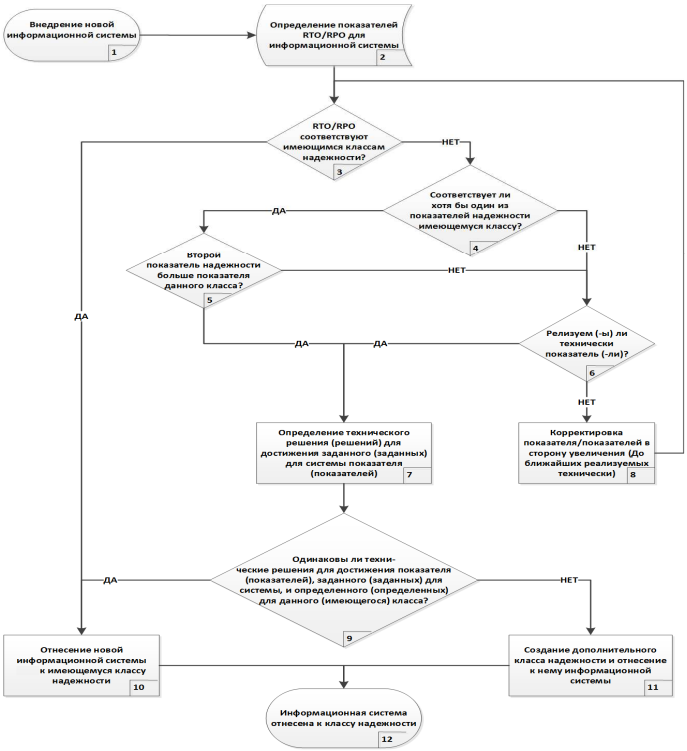
- группа Б – ответственные;
- группа С – некритичные.

При этом чаще всего для них определяют следующие показатели:

- коэффициент доступности;
- требования к изоляции и оснащению контуров;
- временные показатели восстановления

Методика распределения по группам надежности должна быть направлена на сохранение количества групп и основана на выявлении/сравнении технических решений, обеспечивающих показатели, определенные в рамках BCP для той или иной информационной системы. Вместе с тем, возможность расширения групп надежности также предусмотрена, что обусловлено тем, что в некоторых случаях отнесение информационной системы к существующей группе может быть невозможно либо экономически не обосновано.

Процедура отнесения системы к классу надежности:



Способы реализации

Существует несколько методов реализации отказоустойчивости системы:

- low level fault tolerant services - система должна состоять из более менее независимых систем, каждая из которых должна быть отказоустойчивой. Наилучший вариант, когда вся инфраструктура опирается на отказоустойчивые сервисы.
- Избыточность - когда в вашей системе присутствует избыточное количество необходимых вам компонент. И при падении одного из таких избыточных компонент, все должно продолжать работать. При таком подходе проектирования можно выделить две стратегии: active-active и active-passive.
  - Active-Active – работа одновременно с двумя идентичными компонентами. При этом если один из таких компонентов откажет, то клиент вообще не заметит, что в система случилась какая-либо проблема. В то же время стоит отметить, что удваивается количество трафика и время на его обработку, а также требуется дополнительная серверная архитектура, которая постоянно находится в рабочем режиме и потребляет ресурсы. Так же данный подход не всегда возможно реализовать ввиду различных бизнес ограничений(например система транзактов не может обрабатывать запрос одновременно от двух компонент).
  - Active-Passive - Данная стратегия, представляет собой только один постоянно рабочий компонент, в случае падения которого, автоматически поднимается второй, восстанавливает состояние и берет на себя всю работу. Однако данный подход скорее всего потребует больше усилий и средств, по сравнению с active-active, так как тут нужен надежный способ проверки, что активный компонент функционирует, и нужно уметь восстанавливать состояние на момент падения или постоянно его синхронизировать. Еще данное решение всегда будет иметь некую задержку в работе при падении компонента. Из преимуществ данного подхода можно выделить меньшее потребление ресурсов, а также возможность реализации решения на более слабом железе, за счет того, что не требуется поддерживать его в работающем состоянии постоянно.
- Балансировка нагрузки – при условии, что существует несколько идентичных компонент, между ними можно более менее распределить всю нагрузку. В отличие от вышеописан-

ной active-active стратегии, здесь каждую задачу выполняет только один компонент. Данный механизм идеально подходит для stateless компонент, иначе для отказоустойчивости вам постоянно придется синхронизировать состояние. В данном решении очень важно иметь как минимум N+1 redundancy, т.е. если для пиковых нагрузок необходимо N работающих компонент, то в системе должно присутствовать N+1 таких компонент, так как иначе, если упадет один из элементов, то на все остальные нагрузка увеличится, и вся система откажет.

- Изоляция различных систем – принцип вынесения различных подсистем на разное железное либо виртуальное оборудование для исключения взаимопроникающих помех. В классической реализации имеет очень высокую стоимость, а также очень неэффективное использование доступных вычислительных ресурсов. Отличным вариантом оптимизации может стать вынесение таких изолированных подсистем в виртуальные среды. Виртуализация – процесс представления набора вычислительных ресурсов, или их логического объединения, который даёт какие-либо преимущества перед оригинальной конфигурацией. Это новый виртуальный взгляд на ресурсы, не ограниченных реализацией, географическим положением или физической конфигурацией составных частей». Перенос системы в виртуальную среду позволяет эффективно распределять ресурсы. При использовании кластера гипервизоров (программа или аппаратная схема, обеспечивающая или позволяющая одновременное, параллельное выполнение нескольких систем на одном и том же хост-компьютере.) есть возможность избавиться от единой точки отказа в виде аппаратного сервера. В случае отказа одного из серверов, или каких-либо плановых отключений – замены железа, установка обновлений ОС с перезагрузкой, и т.д. – виртуальные машины могут быть перемещены на работоспособный узел достаточно быстро, или даже незаметно для пользователей. Таким образом, время восстановления системы после сбоя будет измеряться минутами, а плановых остановов серверов пользователи не заметят вообще.

Все перечисленные методы обеспечивают высокий уровень надежности, возможность восстановить работоспособность системы и избежать потери данных при отказе отдельных элементов системы. Ключевым различием при выборе конкретной стратегии в данной ситуации становится стоимость готового решения. При этом дать точную

оценку не представляется возможным, так как зачастую существует множество факторов, как в области архитектуры целевой системы, так и в области организации бизнеса в компании, влияющих на конечный результат. С развитием технологий в области ИТ, наиболее перспективным подходом на текущий момент можно назвать виртуализацию системы, так как данный метод позволяет не только обеспечить высокий уровень надежности, но и обеспечивает наилучшее распределение вычислительных ресурсов в организации.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Галкин В. А. Технологии распределенных объектных систем / Галкин В. А. // Проблемы построения и эксплуатации систем обработки информации и управления : сб. ст. / МГТУ им. Н. Э. Баумана. - 2000. - Вып. 1. - С. 63-68.

2. Гонсалес Х.К., Галкин В. А. Постановка задачи о разработке автоматизированной системы обработки технических данных конечного интернет-пользователя республики Эквадор. Инженерный вестник (МГТУ им. Н.Э.Баумана). Электронный журнал. 2013 .- № 10

3. Галкин В.А., Григорьев Ю.А. «Телекоммуникации и сети», М.:МГТУ им. Н.Э.Баумана, 2003 – 608 с.

4. Акамова Н.В., Голяев С.С., Правосудов Р.Н. ОБЕСПЕЧЕНИЕ ОТКАЗОУСТОЙЧИВОСТИ IT-СЕРВИСОВ // Современные проблемы науки и образования. – 2014. – № 6.

5. Казаков В.Г., Федосин с.А. Технологии и алгоритмы резервного копирования / Всероссийский конкурсный отбор обзорно-аналитических статей по приоритетному направлению "Информационно-телекоммуникационные системы", 2008. - 49 с.

М.А. ИВАНЧИКОВА
Рязанский государственный радиотехнический университет

ИССЛЕДОВАНИЕ СРЕДСТВ ПРОЕКТИРОВАНИЯ ПРОГРАММНО-КОНФИГУРИРУЕМЫХ СЕТЕЙ

Рассматривается технология и основные средства проектирования программно-конфигурируемых сетей на основе протокола OpenFlow.

Стремительное развитие вычислительной производительности, объемов памяти и возможностей высокоскоростной передачи данных привело к необходимости разработки новых технологий построения сетевых инфраструктур.

В обычном маршрутизаторе Internet одновременно реализуются и управление и передача данных (рисунок 1). Уровень управления представлен встроенным контроллером, уровень передачи данных – таблицей коммутации и коммутационной матрицей. Контроллер обладает некоторыми интеллектуальными функциями, позволяющими ему самому принимать решения о передаче данных на основе информации о структуре сети. Но непосредственно управлять принятием решения нельзя – можно лишь конфигурировать контроллер, задавая определенные наборы правил и приоритетов. Это значительно ограничивает функциональность коммутатора и всей сети. Проблему разделения уровня управления и передачей данных исследователи Стэнфорда и Беркли предложили решить в рамках подхода, получившего название программно-конфигурируемые сети (ПКС).

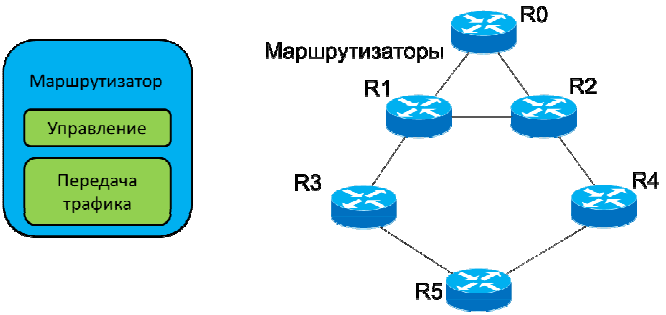


Рис. 1 – Представление сети и функций классического маршрутизатора.

Программно-конфигурируемые сети (Software Defined Networks, SDN) – развивающаяся архитектура сети, где функция управления сетью разделена с функцией передачи данных и полностью программируема [1]. Основная идея ПКС состоит в том, чтобы, не изменяя существующего сетевого оборудования отделить управление этим оборудованием (маршрутизаторами и коммутаторами) за счет создания специального программного обеспечения, которое может работать на обычном отдельном компьютере и находится под контролем администратора сети (рисунок 2).

Теоретически ПКС дает возможность абсолютной гибкости в управлении трафиком, а так же обеспечивает легкую балансировку трафика без задействования отдельного специализированного оборудования. Самым используемым протоколом для поддержания обмена между контроллерами и коммутаторами ПКС является протокол OpenFlow.

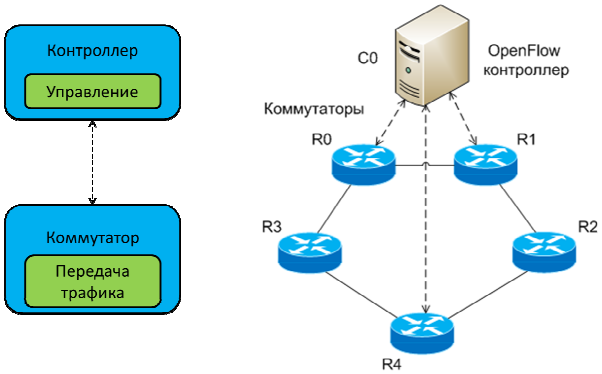


Рис. 2 – Представление сети и функций коммутатора с использованием SDN.

Для проектирования и анализа инфраструктуры программно-конфигурируемых сетей с поддержкой протокола OpenFlow наиболее часто используют эмулятор MiniNet. Он моделирует компьютерные сети, создавая виртуальные узлы, коммутаторы, контроллеры и каналы связи. Следовательно, топологии, построенные и работающие в MiniNet, могут быть применимы в реальных сетях, состоящих из реальных аппаратных устройств. MiniNet предоставляет пользователям готовые решения для получения сведений о поведении и производительности ПКС архитектур, реализованных на различных экспериментальных сетевых топологиях. Появляется возможность комплексного тестирования сложных топологий без запуска физической сети. Эмулятор поддерживает не только базовый набор встроенных топологий, но и произвольные пользовательские топологии. Их создание возможно с помощью сценариев, написанных на языке Python.

Так же для программирования контроллеров OpenFlow сетей используют язык Frenetic [2]. Это новый, объектно-ориентированный язык, встроенный в Python, который включает в себя:

1) ограниченный, но высокоуровневый и декларативный язык сетевых запросов,

2) функциональную библиотеку политик управления сети.

Frenetic предоставляет программисту интерфейс для классификации сетевого трафика, а так же набор правил фильтрации, группировки и передачи пакетов. Объединяя возможности Python и Frenetic, был создан язык Pyretic [3]. Pyretic одновременно является объектно-ориентированным языком программирования, встроенным в Python, и системой исполнения, которая реализует программы, написанные на языке Python на сетевых коммутаторах. Он позволяет сетевым про-

граммистам и операторам создавать емкие модульные сетевые приложения, предоставляя им мощные абстракции. За счет этого основное внимание уделяется способу задания сетевых политик. Например, с использованием Pyretic программисты могут задать правила передачи трафика в виде компактных абстрактных функций, которые на входе получают пакет, а на выходе возвращают набор новых пакетов. Pyretic облегчает модульное проектирование за счет использования двух правил взаимодействия операторов: параллельного и последовательного, позволяющих программистам комбинировать множество правил без опасения возникновения конфликта между ними. Разработчики приложений так же могут создавать динамические правила, которые меняют свое поведение в течение времени. Таким образом, Pyretic предоставляет ПКС-программистам краткий модуль сетевого приложения как абстракцию высокого уровня.

Работа выполнена при финансовой поддержке стипендии президента РФ СП-505.2016.5, программы УМНИК Фонда содействия развитию малых форм предприятий в научно-технической сфере.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. «Software-defined networking: the new norm for Networks», ONF White Paper. April 2012.

2. N. Foster, M. J. Freedman et al, «Languages for software-defined networks», IEEE Communications Magazine. 2013. Vol. 51, Issue 2. pp. 128-134.

3. J. Reich, C. Monsanto et al, «Modular SDN Programming with Pyretic», USENIX ;login: Magazine. Vol. 38, Issue 5. 2013. pp.128-134.

М.А. ИВАНЧИКОВА
Рязанский государственный радиотехнический университет

МОДЕЛИРОВАНИЕ ПРОЦЕССОВ БЫСТРОЙ ПЕРЕМАРШРУТИЗАЦИИ ТРАФИКА МЕЖДУ ЦЕНТРАМИ ОБРАБОТКИ ДАННЫХ

В работе проведено исследование и сравнение известных алгоритмов быстрой перемаршрутизации трафика между центрами обработки данных (ЦОД).

Для повышения эффективности функционирования сети между ЦОД важной задачей является выбор алгоритма маршрутизации, который будет обеспечивать передачу данных между площадками по оптимальным маршрутам в условиях динамических отказов узлов и ли-

ний связи. В настоящее время широкое применение получили алгоритмы адаптивной маршрутизации [1]. Однако известные алгоритмы имеют высокую трудоемкость вычисления оптимальных маршрутов и при обновлении маршрутной информации выполняют полный пересчет таблиц маршрутизации. В связи с этим необходима разработка эффективных моделей и алгоритмов быстрой перемаршрутизации трафика между ЦОД [2].

Математическую модель сети между ЦОД представим в виде неориентированного взвешенного связного мультиграфа **G** = (**DC**(**S**, **ND**(**SND**)), **E**(**W**, **Z**)), где **DC** – множество вершин (площадок ЦОД), |**DC**| = *N*, **E** – множество ребер (каналов или линий связи), |**E**| = *M*, **W** – множество весов ребер (стоимость каналов связи между ЦОД), **Z** – множество провайдеров связи в ЦОД, |**Z**| = *m*, **S** – множество весов вершин (стоимость подключения каналов связи к ЦОД), |**S**| = *Nm*, **ND –** множество сетевых устройств в ЦОД, |**ND**| = *n*, **SND** – множество весов сетевых устройств (стоимость подключения каналов связи к сетевым устройствам в ЦОД), |**SND**| = *nm*.

На основе предложенной математической модели разработан алгоритм быстрой перемаршрутизации трафика, вычисляющий оптимальные маршруты до каждого ЦОД без полного повторного построения дерева оптимальных маршрутов в условиях динамических отказов узлов и каналов связи в сети между ЦОД, обслуживаемых разными телекоммуникационными провайдерами.

Для подтверждения правильности предложенного алгоритма разработано программное обеспечение моделирования процессов маршрутизации трафика между ЦОД. Средствами разработки программного обеспечения моделирования сетей ЦОД и процессов маршрутизации выбраны среда визуального программирования Microsoft Visual Studio, платформа Microsoft .Net 4.0, язык программирования C#. Интерфейс программной системы и дерево оптимальных маршрутов представлены на рисунке 1.

Рассмотрим примеры работы предложенного алгоритма быстрой перемаршрутизации трафика в сети между ЦОД в случае динамических отказов узлов и линий связи. При отказе или перегрузке узла DC5 необходимо пересчитать оптимальный маршрут до узла DC7. Так как каналы связи инцидентные узлу DC5 ранее входили в дерево оптимальных маршрутов, то произойдет парный переход $e_{5,7}^{1} - e_{6,7}^{1}$. Результат работы алгоритма при отказе узла связи DC5 показан на рисунке 2.

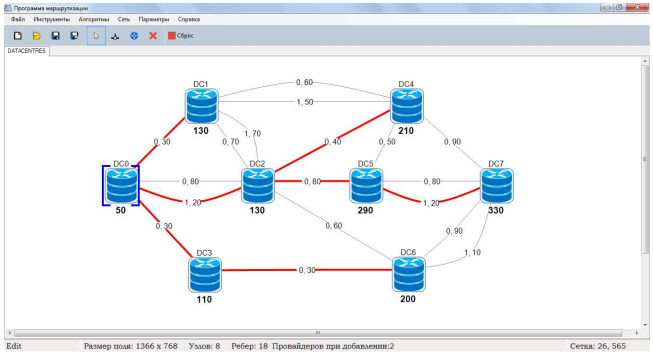


Рис. 1 – Интерфейс программной системы и дерево оптимальных маршрутов сети между ЦОД.

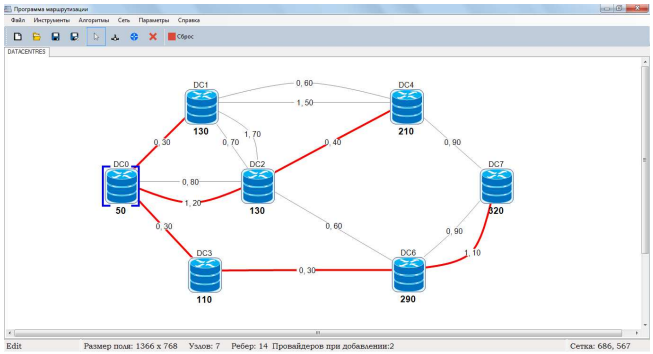


Рис. 2 – Отказ узла связи DC5 в сети ЦОД.

При отказе или перегрузке канала связи $e_{2,4}^{0}$ необходимо пересчитать оптимальный маршрут до узла DC4, так как канал $e_{2,4}^{0}$ входил в дерево оптимальных маршрутов. Произойдет парный переход $e_{2,4}^{0}$ – $e_{1,4}^{1}$. Результат работы алгоритма при отказе канала связи $e_{2,4}^{0}$ показан на рисунке 3.

В программной системе проведено исследование различных топологий ЦОД, состоящих из 10, 20, 50 и 100 узлов, обслуживаемых 1, 2 и 5 провайдерами связи, при 100 изменениях в структуре и параметрах линий связи сети. На рисунке 4 представлены графики зависимости числа парных переходов от количества ЦОД в сети 2 провайдеров при 100 случайных изменениях в сети.

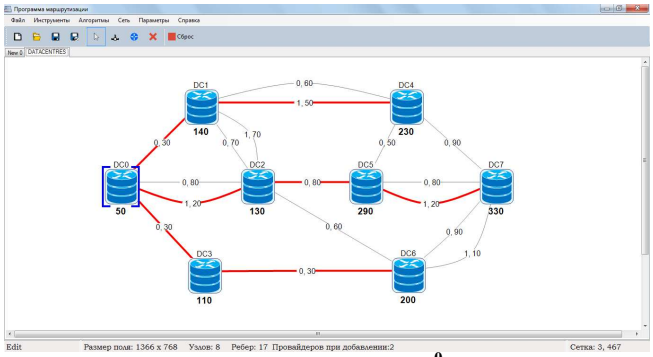
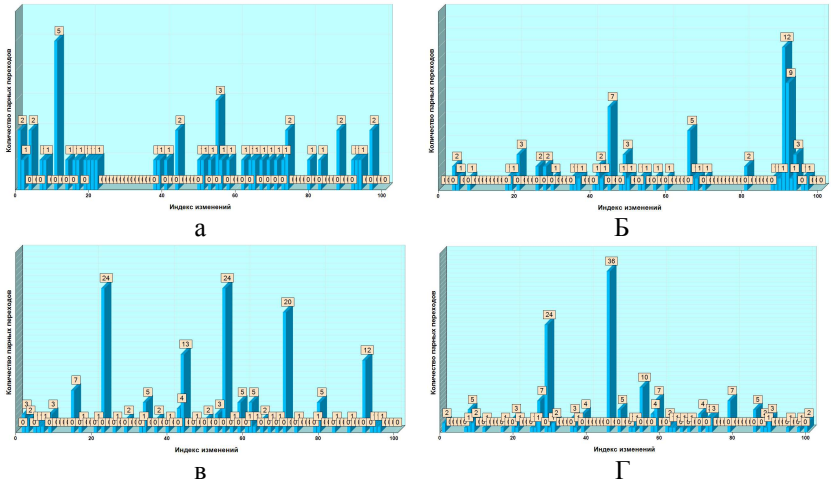


Рис. 3 – Отказ канала связи $e_{2,4}^{0}$ в сети ЦОД.



а Б

в Г

Рис. 4 – Графическое представление числа изменений в сети ЦОД 2 провайдеров:
а – 10 ЦОД, б – 20 ЦОД, в – 50 ЦОД, г – 100 ЦОД.

Рассчитаны показатели минимального и максимального числа парных переходов, значения оценок математического ожидания и среднеквадратичного отклонения, которые доказали эффективность и надежность предлагаемого алгоритма при применении в реальных сетях. На основе результатов исследования, можно сделать вывод, что при увеличении числа узлов в сети ЦОД или при увеличении количества обслуживающих провайдеров значение оценки математического ожидания числа парных переходов уменьшается, что позволяет сказать

об эффективности предложенного алгоритма по сравнению с известными аналогами.

Работа выполнена при финансовой поддержке стипендии президента РФ СП-505.2016.5, программы УМНИК Фонда содействия развитию малых форм предприятий в научно-технической сфере.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Andersen D., Balakrishnan H., Kaashoek F., Morris R. Resilient Overlay Networks. Proc. 18th ACM Symposium on Operating Systems Principles, 2001. pp. 131-145.

2. Корячко В.П., Перепелкин Д.А., Иванчикова М.А. Алгоритм парных переходов каналов связи при динамическом изменении нагрузки в корпоративных сетях нескольких провайдеров связи с различными зонами покрытия // Вестник Рязанского государственного радиотехнического университета. 2014. № 48. С. 68-76.

А.П. КАЛИСТРАТОВ
Московский государственный технический университет им. Баумана

ОБЗОР ПРОТОКОЛОВ, ИСПОЛЬЗУЕМЫХ ПРИ СБОРЕ ТЕХНИЧЕСКИХ ДАННЫХ УДАЛЕННЫХ УСТРОЙСТВ

Рассматриваются протоколы сбора и передачи технических данных удаленных устройств по сети Интернет с целью формирования требований к проектируемой системе обработки технических данных конечного интернет-пользователя.

Системы сбора технических данных используются для постоянного наблюдения за целевыми объектами (компьютерными сетями или конечными пользователями) с целью поиска некорректно работающих или неисправных узлов. При обнаружении неисправности система посылает сигнал оператору, который формализует и передает информацию ответственным за устранение неисправности лицам. Сбор технических данных относится к одной из задач управления компьютерными сетями, т.е., поддержания определенного уровня доступности информации.

Системы сбора технических данных используют программы-агенты, размещаемые на целевых машинах, которые передают информацию на центральный сервер [1.25]. Управление программами-агентами также выполняется с центрального сервера. Часто в программы-агенты добавляется функционал, позволяющий не только со-

бирать необходимую техническую информацию, но и осуществлять частичное или полное управление целевыми машинами.

Технология VPN (virtual private network, виртуальная частная сеть) позволяет организовать частную IP сеть поверх публичной сети Интернет таким образом, что пользователи могут обмениваться информацией так, как будто бы они были физически подключены к данной частной сети.

Teamviewer – бесплатное ПО, позволяющее осуществлять контроль над целевой машиной, используя VPN соединение. Несмотря на то, что мониторинг целевой машины не является основной сферой применения данного ПО, сбор данных ведется одинаково эффективно как при установке на пользовательские машины, так и при установке на серверное оборудование [3]. Фактически, Teamviewer использует свой проприетарный протокол передачи данных, полностью покрывающий все взаимодействие пользователя с сервером.

Процесс логина состоит из следующих этапов:

1) Клиент обменивается пакетами cmd_ping с ping-сервером (pingx.teamviewer.com)

2) Клиент обменивается пакетами cmd_mastercommand с сервером логина (masterx.teamviewer.com).

После подключения к серверу KeepAlive, сессию можно считать установленной, и клиент переходит в статус online. Фактически, через обмен данными с master серверами устанавливаются исходящие подключения, а через обмен данными с Keepalive – входящие.Также Keepalive сервер регулярно отсылает ICMP-пакеты клиенту с целью подтверждения статуса online и при отсутствии ответа – помечает его в системе как неактивный и запрещает входящие подключения.

После выполнения этих двух шагов (при условии регулярного обмена данными с сервером keepalive) клиент получает статус Ready to connect.Для установления соединения «сервер» отсылает пакет cmd_mastercommand. Master-сервер Teamviewer отвечает на данные пакеты строкой, включающей в себя либо ответ типа NOROUTE (keepalive_lost, т.е., клиент не отвечает на запросы сервера, либо Id-NotFound, когда клиента нет среди зарегистрированных в системе на данный момент), либо CONNECT, позволяющий «серверу» запрашивать маршрутизацию до клиента у промежуточного router-сервера с помощью команды RequestRoute2. Также в этот момент сервер keepalive отсылает пакет CMD_RequestConnect клиенту с целью установления соединения между «сервером» и клиентом.

В целом, установление соединения между заданными «сервером» и клиентом с помощью протокола Teamviewer включает в себя

master сервер, keepalive сервер и вспомогательный сервер маршрутизации внутри созданного VPN. После установления соединения между всеми участниками подключение можно считать запущенным и становится возможным передача информации напрямую от клиента к «серверу».

OCS Inventory – система управления конфигурациями и сбора данных о компьютерах в сети. Система может собирать данные как о установленных на клиентские компьютеры программах, так и об оборудовании, которым оснащены клиентские машины. Данная система построена на архитектуре «клиент-сервер», используя протокол http для передачи данных [5].

Архитектура сервера OCSI состоит из следующих подсистем:

1) Сервер базы данных, который хранит информацию о клиентских машинах, существующих конфигурациях и доступном ПО.

2) Коммуникационный сервер, функционирующий в качестве мультиплексора данных при обмене данными между сервером БД и агентском ПО на клиентских компьютерах.

3) Сервер развертывания, хранящий все необходимое ПО как для агентской системы, так и для установочных пакетов дополнительного ПО.

4)Консоль управления, фактически являющаяся административной панелью, с помощью которой администратор системы может просматривать конфигурацию всех подсистем, информацию о всех клиентских компьютерах и вносить необходимые изменения с помощью веб-браузера.

ПО на клиентской машине формирует xml файлы, содержащие в себе всю необходимую информацию, после чего в определенный момент времени (по расписанию) отсылает сгенерированные файлы на сервер методом POST. При первом запуске агент отсылает на сервер, прописанный в конфигурации, запрос $prolog_request:

<!ELEMENT REQUEST (QUERY | DEVICEID) >
<!ELEMENT QUERY (#PCDATA)>
<!ELEMENT DEVICEID (#PCDATA)>

В ответ сервер отсылает агенту ответ prolog_reply, содержащий в себе данные о сервере и подключении (частота, аутентификация, регистрационные данные). После чего агент ожидает сообщения inventory_request, содержащего в себе информацию об ожидаемом ответе. Стоит заметить, что после получения запроса от сервера клиент не генерирует новый xml файл с сырыми данными, а использует полученный в виде «бланка» для создания ответа. XML файл полностью инвентаризованной системы весит примерно 5КБ, что делает возмож-

ным одновременный сбор данных с множества клиентских устройств без возникновения значительной нагрузки на сеть.

Особенностью OCSI является возможность ручного импорта данных, собранных с клиентов, не подключенных к сети. Это позволяет преодолеть трудности, связанные с инвентаризацией клиентов вне локальной сети. Коммуникационный сервер может сканировать порты клиентских машин с помощью утилиты nmap для определения типа клиента (ПК, принтер, активное сетевое оборудование). Это позволяет не отправлять серверу развертывания адреса тех машин, которые не поддерживают установку клиента.

Для понижения нагрузки на коммуникационный сервер также реализован механизм peer-to-peer обнаружения, т.е., коммуникационный сервер выбирает определенные клиентские машины, базируясь на некоторых критериях, и активирует в их агентах функцию IpDiscover, передавая таким образом часть своих функций.

SNMP – протокол прикладного уровня, использующийся для сбора и передачи информации об управляемых сетевых устройствах для удобства управления данными устройствами [4]. Протокол SNMP поддерживается почти всеми современными устройствами SOHO и SMB сегмента. Это позволяет организовывать мониторинг и управление в больших разнородных сетях с помощью одного набора серверного ПО, использующего данный протокол. Агентское ПО, работающее на управляемых устройствах, обрабатывает данные, полученные от внутреннего ПО устройств и отправляет его на сервер после получения запроса на отправку данных. В некоторых случаях агент может самостоятельно инициировать отправку данных, например, в случае возникновения какого-то важного события: перезагрузки или выхода из строя компонента устройства. После получения сообщения от агента сервер выполняет какое-либо заранее определенное действие, например, оповещение оператора.

Так как SNMP поддерживается множеством устройств, для стандартизации агентских сообщений применяются MIB (management information base) – базы информации менеджмента. Обмен данными, основанными на данных из этих баз гарантирует, что информация, отправленная агентом, будет правильно интерпретирована сервером. MIB имеют некоторую общую структуру, единую для всех устройств, поддерживающих SNMP. Также был сформированы единые принципы внесения записей в MIB, под названием SMI (structure of management information, структура информации управления) и ASN (abstract syntax of notation, абстрактный синтаксис нотаций). Единицей данных в

SNMP является PDU (protocol data unit, единица информации протокола).

В общем случае взаимодействие сервера и агента посредством протокола SNMP можно представить следующим образом:
Сервер-SNMP-UDP-IP-Ethernet-IP-UDP-SNMP-клиент.[5]Кратко логика обмена данными между клиентами и сервером посредством протокола SNMP может быть описана следующим образом:

1. Сервер обращается к клиенту на управляемом устройстве, запрашивая информацию, которая характеризует состояние устройства, на котором работает агент.

2. Сервер может указать агентскому ПО выполнить изменение конфигурации управляемого устройства, задав значение переменной.

3. Клиент может сам сообщать серверу о произошедших событиях посредством отправки сообщений типа trap.

4. Вся информация об управляемых устройствах хранится в древовидной структуре базы MIB, а агенты только передают цифровые коды, что экономит трафик.

Проанализировав описанные системы удаленного доступа и сбора данных, можно выделить следующие требования к разрабатываемой системе:

С пользовательской точки зрения процесс сбора данных должен быть незаметен, соответственно, нагрузка на канал передачи данных должна быть небольшой, а сбор данных должен быть автоматизирован. Отправка данных должна происходить либо в момент бездействия системы, либо в момент минимальной загрузки системных ресурсов пользователем.

Учитывая тот факт, что разрабатываемая система рассчитана на большое число пользователей и отсутствие постоянного обслуживания персоналом, клиентская часть ПО должна быть кросс-платформенной, а протокол передачи данных должен поддерживать обратную совместимость с предыдущими версиями.

Подсистема администрирования должна иметь веб-интерфейс для того, чтобы облегчить задачу управления сбором данных. В случае, если визуализировать данные не будет оправдано, система должна поддерживать загрузку конфигурационных файлов для упрощения настройки и отладки.

Система должна разделять и идентифицировать установленное ПО и установленное оборудование в клиентских компьютерах для того, чтобы предоставить как можно больше информации о клиентах и их деятельности.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Постников В.М., Черненький В.М. Методы принятия решений в системах организационного управления. М.: Изд-во МГТУ им. Н.Э. Баумана, 2014. 205 с.

2. Гонсалес Х.К., Галкин В. А. Постановка задачи о разработке автоматизированной системы обработки технических данных конечного интернет-пользователя республики Эквадор. Инженерный вестник (МГТУ им. Н.Э.Баумана). Электронный журнал. 2013 .- № 10

3. Галкин В.А., Григорьев Ю.А. «Телекоммуникации и сети», М.:МГТУ им. Н.Э.Баумана, 2003 – 608 с.

4. Олифер В.Г., Олифер Н.А. Компьютерные сети. Принципы, технологии, протоколы: учебник для вузов. 3-е издание. СПб: Питер, 2008. 960 с.

5. RFC. Structure and Identification of Management Information for TCP/IP-based Internets. Available at: http://tools.ietf.org/html/rfc1155. Accessed: 15.03.2016

Е.А. КЛИМЕНКО, А.Н. САПРЫКИН
Рязанский государственный радиотехнический университет

РЕАЛИЗАЦИЯ ТЕХНОЛОГИЙ ПРОГРАММНО-КОНФИГУРИРУЕМЫХ СЕТЕЙ С ПРИМЕНЕНИЕМ ПРОТОКОЛА OPENFLOW В ЭМУЛЯТОРЕ MININET

Рассматриваются перспективы построения компьютерных сетей с применением подхода централизованного программного управления на основе протокола OpenFlow.

Принципы функционирования современных компьютерных сетей были заложены еще в 60-70 годах прошлого века. Главной целью при создании сетей ставилась их живучесть и децентрализованность, которая позволяла бы при нарушении связей в сети находить альтернативные маршруты [1].

Данная концепция привела к тому, что в сетях находится большое количество маршрутизаторов, которые независимо друг от друга вычисляют маршруты прохождения пакетов. При этом каждое устройство требует отдельной настройки администратором сети, на что тратится много времени. Еще одним крупным недостатком существующих сетей является то, что их функционирование сильно зависит от функционала, заложенного производителем в соответствующее оборудование. Из-за этого часто наблюдаются проблемы применения в од-

ной сети устройств от различных вендоров, так как каждый из них отвечает за реализацию сетевых стандартов и реализовывать их может по-своему.

Еще одной проблемой традиционных IP-сетей является то, что они изначально не были рассчитаны на современные объемы информации, которые с каждым днем только возрастают. Современные исследователи говорят о том, что для того, чтобы покрыть потребности пользователей, требуется увеличить плотность покрытия базовыми станциями в 20 раз. Однако, это приведет к большим финансовым затратам из-за необходимости закупки дорогостоящего оборудования, к сложности управления такой инфраструктурой.

К тому же практически весь функционал в современных сетях реализован аппаратно (например, файерволы), что усложняет модернизацию сетей.

Для решения возникших проблем была предложена концепция программно-конфигурируемых сетей [2]. В таких сетях управления передачей трафика отделено от непосредственно самой передачи. В традиционных коммутаторах и маршрутизаторах размещаются микросхемы, реализующие пересылку пакетов между портами, и устанавливается специальное программное обеспечение, которое определяет правила данной пересылки. Для осуществления маршрутизации устройства реализуют набор протоколов (OSPF, RIP, BGPи т.д.), но при этом функционируют достаточно независимо друг от друга.

В программно-конфигурируемых сетях уровень управления выносится из маршрутизаторов и коммутаторов на отдельное устройство-контроллер, которое определяет топологию сети, находит в ней оптимальные маршруты, применяет политики маршрутизации, заданные провайдером. Маршрутизация осуществляется путем посылки подчиненным устройствам по защищенному каналу специальных управляющих сообщений, инструктирующих коммутаторы (в программно-конфигурируемых сетях маршрутизаторы не используются, так как их роль выполняет контроллер) добавить в таблицу коммутации новую запись, изменить либо удалить старую, либо выполнить какие-то другие действия.

При этом коммутаторы должны соответствовать некоторой общей спецификации.

Для реализации данных концепций была разработана технология OpenFlow, которая описывает правила функционирования устройств программно-конфигурируемой сети [3]. В таких сетях администраторы управляют потоками данных, могут задавать приоритеты трафика и путей его прохождения. Также важным пунктом таких сетей является

наличие на контроллере сетевой операционной системы, которая позволяет запускать на контроллере различные сервисы, которые в настоящее время реализуются аппаратно. Например, QoS – сервисы, протоколы маршрутизации, балансировщики нагрузки.

При таком построении сети в задачи коммутаторов входит пересылка пакетов по известным маршрутам. Каждый коммутатор поддерживает таблицу потоков, безопасный канал с контроллером и способен понимать управляющие сообщения от контроллера [4].

Таблица потоков является ключевым элементом OpenFlow-коммутатора [2, 5, 6]. Она содержитзаписи о потоках, каждая из которых состоит из полей, изображенных на рисунке 1.

Поля сравнения	Приоритет	Счетчики	Инструкции	Временные метки

Рис. 1 - Поля записи таблицы потоков.

Поля сравнения описывают поля, содержащиеся в заголовке пакета – содержат IPи MAC-адреса отправителя и получателя, различные метаданные.

Приоритет – обозначает приоритет конкретной записи и учитывается при поиске соответствия для нового пакета. Для нескольких записей с одинаковым заголовком пакета будет выбрана запись с максимальным значением приоритета.

Счетчики - хранят статистические данные о работе записи(например, сколько пакетов было обработано в соответствии с данной записью).

Инструкции – набор действий, выполняемых над пакетом в процессе обработки.

Временные метки – значения временных интервалов до удаления записи из таблицы потоков.

Соответствующая запись определяется по полям сравнения и приоритету – среди записей, у которых совпали поля, выбирается уникальная с наибольшим приоритетом.

Также коммутаторы поддерживают групповые таблицы и измерительные(meter) таблицы.

Групповые таблицы позволяют производить групповую обработку пакетов.

Измерительные таблицы отвечают за QoS-операции. Например, за ограничение пропускной способности.

Наша работа ставит перед собой задачу реализовать маршрутизацию пакетов по заданной топологии сети с учетом пропускных способностей каналов, распределением нагрузки между несколькими каналами и ее балансировкой.

Реализация будет оформлена в виде OpenFlow-приложения, запускаемого на контроллере. По заданному графу сети она будет находить альтернативные маршруты от узла-отправителя к узлу-получателю и использовать те из них, которые удовлетворяют заданным ограничениям.

Реализация планируется на языке Pythonкак приложение для контроллера POX.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Смелянский Р.Л. Программно-конфигурируемые сети- новый подход к построению компьютерных сетей [Электронный ресурс] URL: http://arccn.ru/knowledge-base?pdf=510b72827f553.pdf

2. Барсков А. SDN: кому и зачем это надо? [Электронный ресурс] URL: http://www.osp.ru/lan/2012/12/13033012/

3. Воган-Николс С. OpenFlow: сеть нового поколения? [Электронный ресурс] URL: http://www.osp.ru/os/2011/08/13011110/

4. OpenFlow [Электронный ресурс] URL: https://ru.wikipedia.org/wiki/OpenFlow

5.OpenFlow Switch Specification [Электронный ресурс] URL: https://www.opennetworking.org

6. Гончаров Ф.О. Программно-конфигурируемые сети [Электронный ресурс] URL: http://cs.mipt.ru/fileadmin/-template/publikacii/2014_g/ii.pdf

Д. А. КОЛЧАЕВ, А. Е. БОРЗЕНКО
Рязанский государственный радиотехнический университет

МОДЕЛИРОВАНИЕ ИЗОБРАЖЕНИЯ ОТ ЛАЗЕРНОГО ЛОКАТОРА ПО ЦИФРОВОЙ КАРТЕ МЕСТНОСТИ

Рассматриваются теоретические и практические вопросы моделирование изображения от лазерного локатора по цифровой карте местности.

Лазерной локацией называют область оптикоэлектроники, занимающуюся обнаружением и определением местоположения различных объектов при помощи электромагнитных волноптического диапазона, излучаемых лазерами. Объектами лазерной локации могут быть танки, корабли, ракеты, спутники, промышленные и военные сооружения [2,3].

Полученная информация от лазерного локатора может использоваться для систем наведения и целеуказания в тех случаях, когда полу-

чение информации от других источников технического зрения затруднено, для формирования трехмерной карты поверхности, определения размеров объектов и их расположения на этой карте.

Актуальной проблемой является получение изображения от лазерного локатора, так как на текущий момент отечественные лазерные локаторы находятся на стадии разработки. Моделирование изображения от лазерного локатора позволит выявить проблемы, особенности, а также эффективность и дальнейшие пути развития лидара до получения опытного образца лазерного локатора.

Целью работы является создание алгоритма, позволяющего промоделировать изображение от лазерного локатора при различных входных технических параметрах.

Лазерный локатор представляет собой моностатическую оптическую систему, т.е. систему с совмещенным источником и приемником излучения (передатчик и приемник излучения устанавливаются на едином поворотном устройстве, которое осуществляет развертку лазерного луча в плоскости сканирования) [4].

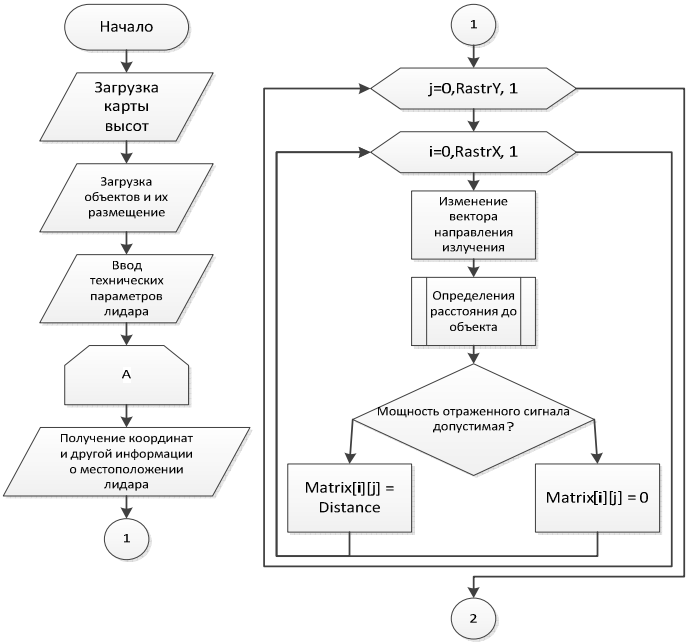


Рис. 1 – Укрупненная схема алгоритма моделирования изображения от лазерного локатора.

Развертка лазерного луча в лазерном сканере осуществляется с помощью механического устройства, в качестве которого могут выступать: качающееся зеркало, вращающаяся призма, вращающееся зеркало. Существует также способ формирования развертки с помощью акустооптического дефлектора - устройства для управления отклонением светового луча в пространстве на основе дифракции или рефракции.

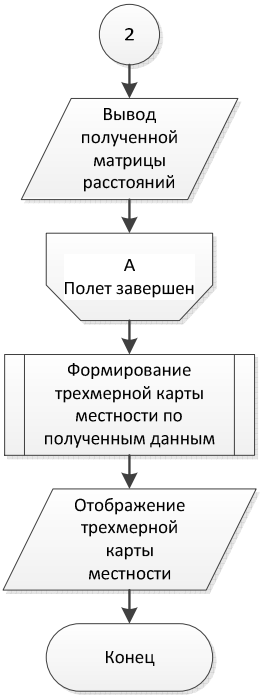


Рис. 2 – Продолжение схемы алгоритма моделирования изображения от лазерного локатора.

Для создания развертки по нескольким осям применяется метод, при котором комбинируется механическое устройство (качающееся зеркало) и акустооптический дефлектор [1,5].

Задача моделирования включает в себя несколько стадий. Начальная стадия предполагает загрузку карты высоты, объектов расположенных на этой местности, а также ввод технических параметров лазерного локатора. Следующая стадия заключается в получении информации о местоположении локатора в текущий момент времени, формировании основного растра, определении дальности до объектов

и записи полученных результатов в матрицу дальностей. Последняя стадия включает в себя анализ полученных данных и формирование трехмерной карты местности по изображениям, полученным от лазерного локатора в различные моменты времени. Полученная укрупненная схема алгоритма представлена на рисунке 1.

Полученный алгоритм позволяет промоделировать изображения от различных лазерных локаторов как в статическом, так и в динамическом состоянии, оценить качество и точность формируемого изображения, подобрать наилучшие входные параметры для получения достоверного изображения. В дальнейшем, планируется, разработать программу для моделирования изображения от лидара, используя разработанный алгоритм.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Кащеев Б.Л. Дистанционные методы и средства исследования процессов в атмосфере Земли / Под общ. ред. Б.Л. Кащеева, Е.Г. Прошкина, М.Ф. Лагутина. — Харьков: Харьк. нац. ун-т радиоэлектроники; Бизнес Информ, 2002. — 426 с.

2. Федоров Б.Ф.Лазеры. Основы устройства и применение. – М.:ДОСААФ, 1988, -190 с.

3. Филиппов В.Л. Атмосфера и моделирование оптико-электронных систем в динамике внешних условий. Монография [Электронный ресурс] / В.Л. Филиппов, В.П. Иванов, В.С. Яцык. — Электрон. дан. — Казань: КФУ (Казанский (Приволжский) федеральный университет), 2015. — 632 с. — Режим доступа: http://e.lanbook.com/books/element.php?pl1_id=72850 — Загл. с экрана.

4. Свободная энциклопедия [Электронный ресурс] // Режим доступа: http://lidar.pro/wiki/Воздушный_лазерный_сканер

5. Velodynelidar [Электронный ресурс] // Режим доступа: http://velodynelidar.com/

А.В. КОМИССАРОВ
Корпорация «Фазотрон-НИИР» АО ОП НИИ «Рассвет»

ПЕРСПЕКТИВНАЯ СИСТЕМА ЗАЩИТЫ ТОВАРОВ ОТ ПОДДЕЛКИ

Рассматривается перспективная система защиты товаров от подделки, основанная на применении современных информационных технологий.

В современном мире остро стоит проблема борьбы с контрафактной продукцией. С каждым годом объем поддельных товаров не только не уменьшается, но и постоянно растет. Подобный рост связан, прежде всего, со значительным увеличением объемов интернет торговли, полностью контролировать которую практически не возможно.

Одним из современных методов защиты товаров от подделки является использование специальных микросхем - радиочастотных меток (RFID) [1], в память которых записывается идентификационная информация о товаре. Покупатель товара, оснащенного подобным чипом, с помощью специального программного обеспечения, установленного на мобильное устройство, может определить его подлинность. Кроме указанных функций, такие радиочастотные метки часто снабжаются специальными проводниками для контроля целостности упаковки от вскрытия. Одним из мировых лидеров в производстве подобных систем защиты товаров от подделки является норвежская фирма Thinfilm [2].

Однако подобные системы защиты обладают следующими существенными недостатками:

- Возможность копирования содержимого метки с целью повторного использования;
- Обязательность наличия NFC датчика в мобильном устройстве - покупателя;
- Необходимость подключения к интернету для проверки - подлинности;
- Низкая скорость работы при высоких нагрузках на сеть;
- Отсутствие механизма проверки соответствия товара;
- Отсутствие централизованного контроля.

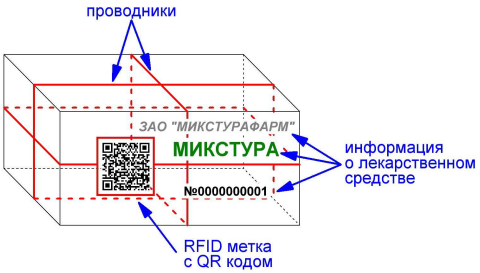


Рис.1 – Пример системы защиты для лекарственного средства.

Разрабатываемая перспективная система защиты товаров от подделки призвана повысить уровень безопасности за счет устранения приведенных выше недостатков. Она включает в себя программную и

аппаратную часть. При этом аппаратная часть включает в себя специальную RFID метку с проводниками (рис.1), для контроля целостности упаковки с нанесенным на нее матричным QR кодом [3]. В памяти RFID метки и в QR коде содержится информация о продукте (фирма производитель, название, дата производства, место реализации и т.п.) и цифровая подпись этих данных (рис.2).

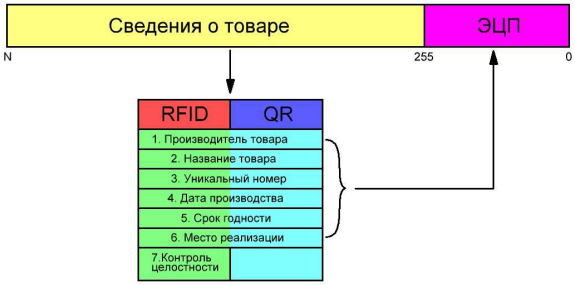


Рис.2 – Содержимое памяти RFID и QR.

Программная часть включает в себя кроссплатформенное мобильное приложение, с помощью которого, при использовании встроенного NFC датчика или цифровой фотокамеры можно осуществить проверку подлинности (а при наличии NFC и целостности упаковки) товара.

Проверка подлинности товара осуществляется за счет использования криптографических методов защиты информации. Данные методы включают в себя процедуры хеширования и вычисления ЭЦП. В качестве криптографических алгоритмов будут использоваться отечественные стандарты: ГОСТ Р34.11-2012, ГОСТ Р34.10-2012, ГОСТ 28147-89 [4,5].

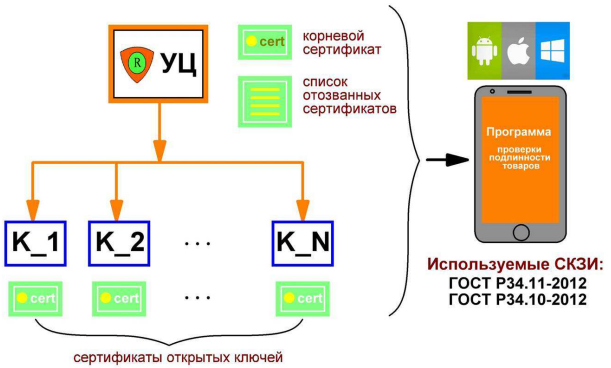


Рис.3 – Инфраструктура системы защиты товаров от подделки.

Для осуществления функционирования системы защиты и для обеспечения централизованного контроля создается удостоверяющий центр (УЦ), который осуществляет выдачу секретных ключей фирмам-производителям защищаемого товара (K_1,…,K_N) и формирует сертификаты открытых ключей для проверки подлинности товаров с помощью мобильного программного обеспечения [6]. Данные сертификаты, вместе с корневым сертификатом удостоверяющего центра и списком отозванных сертификатов входят в состав программного обеспечения. При появлении новой фирмы-производителя, ее открытый сертификат добавляется в приложение путем его обновления (рис.3).

Из приведенного описания следует, что разрабатываемая система защиты товаров от подделки устраняет практически все приведенные выше недостатки и обеспечивает:

- Контроль целостности упаковки (за счет использование микросхем RFID со специальными проводниками);
- Контроль подлинности товара (за счет применения криптографических методов защиты информации);
- Контроль от копирования с целью повторного использования (за счет указания в описании товара точного места его реализации);
- Возможность контроля подлинности при отсутствии датчика NFC (за счет использования камеры для считывания QR кода);
- Работу без интернета (все механизмы контроля подлинности и целостности включены в мобильное ПО);
- Высокую скорость работы (все вычисления происходят на мобильном устройстве без доступа к базам данных и т.п.);
- Возможность осуществления контроля за соответствием товара (при проверке товара с помощью мобильного ПО выводится основная информация о товаре);
- Наличие централизованного контроля (при возникновении определенных ситуаций можно осуществить быстрый отзыв сертификата производителя).

В заключении хочется отметить, что разрабатываемая система защиты товаров от подделки объединяет в себе самые современные методы идентификации и проверки подлинности информации, тем самым обеспечивается более высокий уровень доверия покупателей к приобретаемой продукции.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. [Электронный ресурс] https://ru.wikipedia.org/wiki/RFID
2. [Электронный ресурс] http://thinfilm.no/

3. [Электронный ресурс] https://ru.wikipedia.org/wiki/QR%D0%-BA%D0%BE%D0%B4

4. А.П. Алферов, А.Ю. Зубов, А.С. Кузьмин, А.В. Черемушкин, Основы криптографии. – М.: Гелиос АРВ, 2005.

5. О.Н. Жданов, В.А. Чалкин, Эллиптические кривые: Основы теории и криптографические приложения. – М.: УРСС, 2013.

6. А.В. Черемушкин, Криптографические протоколы. Основные свойства и уязвимости. – М.: Издательский центр «Академия», 2009.

К.И. КОРАБЕЛЬНИКОВА
Рязанский государственный радиотехнический университет

ДИАГНОСТИКА НЕПОЛАДОК ПОДКЛЮЧЕНИЯ К СЕТИ ИНТЕРНЕТ

Рассматривается проблема диагностики неполадок при подключения конечного пользователя к сети Интернет. Исследуются возможности автоматизации определения неполадок средствами маршрутизаторов D-Link без необходимости обращения к техническим специалистам и опроса пользователя.

Повсеместное использование сети Интернет (в частности, для доступа к Всемирной Паутине) и автоматизация настройки подключения приводит к тому, что у многих пользователей отсутствует понимание базовых принципов работы сети Интернет и телекоммуникационного оборудования. Из-за этого в ситуации отказа или неполадок оборудования пользователю бывает сложно определить причину сбоя, и специалист технической поддержки также не всегда может вынести верный вердикт из за отсутствия прямого доступа к устройству. Кроме того, многие устройства настроены таким образом, что нужные данные защищены системой безопасности или не могут быть получены без понимания конкретной архитектуры системы. Помимо этого, пользователь может неправильно выполнить действия или интерпретировать данные и требования, необходимые для стабильного доступа в Интернет.

Всё вышеперечисленное говорит о том, что наиболее эффективным способом диагностики проблем и неполадок подключения к сети Интернет является исполнение диагностирующего сценария домашним маршрутизатором и анализ полученного результата. Такое решение позволит получить доступ к любой информации, поступающей на устройство, и избежать ошибок, связанных с так называемым «человеческим фактором».

На данный момент телекоммуникационное оборудование производства компании D-Link поддерживает некоторые механизмы определения проблем подключения, таких как отсутствие физического соединения с сетью (если кабель не подключен или повреждён), неудачная аутентификация на сервере (если указан неправильный логин или пароль) и так далее. Однако необходима несколько более чёткая дифференциация случаев отсутствия подключения.

Примем для рассмотрения следующие типы проводного подключения по технологии Ehternet:

1. Статическое подключение IPoE;
2. Динамическое подключение IPoE по протоколу DHCP;
3. Подключение по протоколу PPPoE;
4. Статическое подключение IPoE с использованием туннельного протокола PPTP;
5. Статическое подключение IPoE с использованием туннельного протокола L2TP;
6. Динамическое подключение IPoE по протоколу DHCP с использованием туннельного протокола PPTP;
7. Динамическое подключение IPoE по протоколу DHCP с использованием туннельного протокола L2TP.

На оборудовании D-Link данные соединения реализуются стандартными средствами операционной системы Linux с использованием специализированных программных пакетов:

- пакет udhcp для реализации подключения по протоколу DHCP;
- пакет pppd для реализации технологии PPP (в частности, подключения по протоколу PPPoE и участия в подключениях по туннельным протоколам);
- пакеты и плагины pptp, l2tpd, pppol2tp для реализации подключений по туннельным протоколам (включение того или иного пакета зависит от аппаратной платформы конкретного устройства).

Реализация диагностического сценария на обозначенном оборудовании предполагает анализ и внесение изменений в исходный код этих программных продуктов, а также создание последовательности действий, осуществляющей начальную проверку общих условий, выбор наблюдаемого объекта подключения (интерфейса, на котором происходит соединение) и анализ полученных данных.

Перед рассмотрением непосредственно нюансов диагностики различных типов подключения следует отметить, что процесс обменом информации с сетью можно разделить на несколько фаз:

- фаза обнаружения сервера;
- фаза аутентификации;
- фаза обмена данными с сетью.

Различные типы могут не включать в себя некоторые из этих фаз, а могут включать некоторые из них несколько раз. Действия, выполняемые сетевым устройством во время одной и той же фазы, значительно отличаются для подключений разных типов. Фаза обмена данными идентична для любых подключений и представляет собой обмен пакетами по протоколу IP. Для рассмотрения принимаются сети, работающие по протоколу IPv4, в сетях протокола IPv6 могут отличаться инструменты выявления проблем, но общий порядок остаётся прежним. Указанные действия рекомендуется выполнять в указанной последовательности.

К рассмотрению не относятся вопросы неисправности клиентского устройства и некорректной настройки серверного оборудования.

Статическое подключение IPoE (Internet Protocol over Ethernet) является простейшим способом подключения к сети Интернет. Из всех вышеперечисленных это единственный способ, не требующий дополнительных данных для успешного установления соединения и, соответственно, взаимодействия с каким-либо сервером, предоставляющим такие данные (обычно это сервер провайдера, предоставляющего услуги подключения). Это означает, что в данном типе подключения существует только одна фаза — фаза обмена данными. Поскольку эта фаза одинакова для всех типов подключения, рассматриваемая здесь последовательность действий должна быть использована и при диагностике для других типов в том случае, если не обнаружено ошибок, относящихся к другим фазам.

Для данного типа подключения применяется следующая последовательность действий, каждое из которых способно выявить определённую проблему:

Таблица 1. Диагностика статического подключения.

Действие	Диагностируемая проблема
Проверка уникальности IP-адреса в локальной сети с помощью утилиты arping с ключом –D	Устройство не получает некоторые адресованные ему пакеты, поскольку в локальной сети присутствует другое устройство с таким же адресом

Действие	Диагностируемая проблема
Проверка доступности шлюза в локальной сети с помощью утилиты arping	Адресат, выбранный в качестве шлюза во внешнюю сеть, отсутствует в локальной сети (отключен или неисправен)
Попытка передачи ICMP-пакетов через шлюз	Адресат, выбранный в качестве шлюза во внешнюю сеть, не способен передавать пакеты (чаще всего это означает, что выбранный адрес принадлежит не искомому шлюзовому устройству, а какому-либо из устройств домашней группы)
Попытка обратиться к общеизвестным сетевым адресам (например, к серверу Google DNS по адресу 8.8.8.8	Вероятно, произошёл обрыв на линии провайдера
Проверка доступности указанных адресов DNS-серверов	Указанные DNS-серверы отсутствуют в сети (отключены или неисправны)
Запрос данных по протоколу DNS с указанных адресов DNS-серверов с помощью утилиты host	Указанные адреса не принадлежат DNS-серверам

Проверку статического подключения можно разделить на локальный (действия 1-3) и глобальный (4-6) участки. На локальном участке уточняются вопросы, связанные с локальной сетью и шлюзом, на глобальном рассматриваются вопросы доступа к удалённым сервисам.

Динамическое подключение по протоколу DHCP (Dynamic Host Configuration Protocol) характеризуется наличием DHCP-сервера, который предоставляет клиентскому устройству данные, необходимые для корректного обмена данными. Соединение инициируется со стороны клиента и в общих чертах представляет из себя следующую последовательность:

1. DHCPDISCOVERY: клиент запрашивает сервер в локальной сети.
2. DHCPOFFER: сервер оповещает клиента о готовности произвести подключение и предлагает настройки.
3. DHCPREQUEST: клиент оповещает сервер, что готов или не готов использовать эти настройки (при получении DHCPOFFER сразу

от нескольких серверов клиент принимает настройки одного и отказывает всем остальным).

4. DHCPACK: сервер оповещает клиента, что соединение установлено ИЛИ DHCPNAK: сервер оповещает клиента.

Обмен пакетами DHCPDISCOVERY и DHCPOFFER относятся к фазе обнаружения сервера, обмен DHCPREQUEST и DHCPACK/DHCPNAK можно отнести к фазе аутентификации. Аутентификация пользователя производится по MAC-адресу в момент.

Последовательность действий для диагностики соединения выглядит следующим образом:

Таблица 2. Диагностика подключения к серверу DHCP.

Действие	Диагностируемая проблема
Отправка пробного запроса DHCPDISCOVERY и ожидание DHCPOFFER	В сети отсутствует DHCP-сервер (отключен или неисправен)
Анализ ответа, пришедшего на DHCPREQUEST	Произошла ошибка аутентификации, наиболее вероятная причина в том, что при подключении указан неправильный MAC-адрес

Помимо прямых подключений следует рассмотреть также и подключения, осуществляемые по принципу «точка-точка». В настоящее время эти виды подключений реализуются с помощью семейства протоколов PPP (Point-to-Point Protocol). Эти протоколы отвечают за установление прямой связи между двумя узлами сети с возможностью аутентификации соединения, шифрования и сжатия данных. Отличительной особенностью данной технологии с точки зрения диагностики является документированный в RFC 1661 список кодов ошибок, используемый при её реализации и идентичный для любых соединений, использующих внутри себя семейство протоколов PPP. С их помощью можно различить следующие неполадки:

- неправильно указаны логин и пароль;
- неправильно указан логин в случае, когда не требуется аутентификация паролем;
- сервер требует аутентификации паролем в случае, если таковая опция отключена при запросе соединения;
- при попытке подключения выбран неверный алгоритм шифрования и т. д.

Помимо этого данный список позволяет определить, какая из сторон прервала соединение, и отследить ситуацию, когда подключе-

ние невозможно из-за программной неисправности клиентской или серверной части реализации.

Семейство протоколов PPP участвует в установлении соединений по различным технологиям, среди которых соединение по протоколу PPPoE и туннельные соединения по технологиям PPTP и L2TP.

Протокол PPPoE (Point-to-Point Protocol over Ethernet) — это подвид потокола PPP, который отличается способом обнаружения сервера: общение с сервером PPPoE в первой фазе напоминает обмен пакетами с сервером DHCP. Можно провести следующие параллели:

Таблица 3. Соответствие пакетов протоколов DHCP и PPPoE.

Пакет DHCP	**Пакет PPPoE**
DHCPDISCOVERY	PADI (PPPoE Active Discovery Initiation)
DHCPOFFER	PADO (PPPoE Active Discovery Offer)
DHCPREQUEST	PADR (PPPoE Active Discovery Request)
DHCPACK и DHCPNAK	PADS (PPPoE Active Discovery Session-confirmation) *Причины возможного отказа записываются в пакет в виде тегов с заранее определёнными числовыми значениями*

Последовательность диагностики на фазе обнаружения сервера выглядит следующим образом:

Таблица 4. Диагностика подключения к серверу PPPoE.

Действие	**Диагностируемая проблема**
Отправка пробного запроса PADI и ожидание PADO	В сети отсутствует PPPoE-сервер (отключен или неисправен)
Анализ пакета PADS	При подключении указано несуществующее название сервиса (0x0201 Service-Name-Error), либо произошла ошибка подключения со стороны сервера (0x0202 AC-System-Error)

Фаза аутентификации, реализуемая с помощью технологии PPP, анализируется в соответствии с методом, описанным ранее.

Туннельные соединения типов PPTP (Point-toPoint Tunneling Protocol) и L2TP (Layer 2 Tunneling Protocol) реализуются по схожим технологиям, производя инкапсуляцию кадров PPP в пакеты другого типа. Основное отличие с точки зрения диагностики заключается в том, что сервер туннельного соединения не занимается выдачей адресов (в случае сетей, построенных на стеке протоколов TCP/IP, речь идёт об IP-адресах). Поэтому непосредственно установлению соединения предшествует процедура получения адреса по одной из вариаций IPoE: необходимо либо задать адрес самостоятельно (статический IPoE), либо получить настройки от DHCP-сервера. В общих чертах процедура выполнения соединения происходит по следующему сценарию:

Таблица 5. Этапы установления туннельных подключений.

Этап подключения	Метод диагностики
В случае использования DHCP-сервера клиент выполняет подключение к нему и получает настройки. В противном случае он использует статические настройки, заданные пользователем.	Диагностика соединения с сервером DHCP
Клиент находится в общей сети (чаще всего — в локальной сети провайдера) и должен иметь доступ к данным присутствующих в ней DNS-серверов, а также иметь возможность подключиться к серверу, осуществляющему подключение к туннелю.	Диагностика статического подключения с некоторыми корректировками на глобальном уровне
Клиент выполняет подключение к серверу туннельного соединения по протоколу PPTP или L2TP. Сервер устанавливает туннельную сессию.	Диагностика подключения к серверу PPTP/L2TP

Этап подключения	Метод диагностики
Сервер аутентифицирует клиента и устанавливает сессию подключения «точка-точка» с использованием технологии PPP.	Диагностика подключения по технологии PPP
Клиент подключён к глобальной сети.	Диагностика статического подключения на глобальном уровне

Туннельные протоколы предлагают широкий спектр используемых пакетов. Их список можно разделить на две категории: пакеты передачи данных и пакеты установления соединения. Пакеты передачи данных инкапсулируют в себя кадры PPP и используются для передачи непосредственных данных удалённому хосту. Пакеты этого типа не используются для управления соединением, поэтому их можно исключить из рассмотрения

Пакеты установления соединения используются для создания туннельного подключения и поддержания его в активном состоянии. Пакеты этого типа в протоколах PPTP и L2TP могут быть сопоставлены с пакетами протоколов DHCP и PPPoE, поэтому метод диагностики неполадок на этом этапе схож с уже разобранными методами и не требует подробного освещения. Однако существенным отличием является возможность проверки активности соединения средствами самого протокола в виде отправки проверочных пакетов: по протоколу L2TP собеседники обмениваются пакетами Hello, а по протоколу PPTP — PPTP_ECHO_RQST и PPTP_ECHO_RPLY. Отсутствие ответа на соответствующие сообщения может говорить о выходе собеседника из строя, вследствие чего активная сторона завершает соединение в одностороннем порядке и делает попытку переподключения.

Автоматизация диагностики подключения устройств к различным сетям выглядит перспективным направлением исследований и разработки из-за большого количества возможных неисправностей, весьма информативной документации по способам подключения и отсутствия возможности получить некоторую информацию извне. Помимо этого, важным вопросом остаются также нюансы диагностики в других сетях, архитектура которых отлична от общеиспользуемого стека технологий TCP/IP.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Таненбаум Э. Компьютерные сети [Текст] / Э. Таненбаум, Д. Уэзеролл — 5-е изд. — Санкт-Петербург: Питер, 2014. — 960 с.

2. Диагностика локальных сетей и Интернет [Электронный

ресурс]. — Режим доступа: http://book.itep.ru/5/dia_5.htm. — 5 Диагностика локальных сетей и Интернет. — (Дата обращения: 12.01.2016).

3. ietf.org/rfc/rfc2131.txt [Электронный ресурс]. — Режим доступа: http://www.ietf.org/rfc/rfc2131. — Dynamic Host Configuration Protocol. — (Дата обращения: 10.02.2016).

4. ietf.org/rfc/rfc1661.txt [Электронный ресурс]. — Режим доступа: http://www.ietf.org/rfc/rfc1661. — The Point-to-Point Protocol. — (Дата обращения: 12.02.2016).

5. Перевод RFC 1661 [Электронный ресурс]. — Режим доступа: на русском http://rfc.com.ru/rfc1661.htm. — Протокол точка-точка Point-to-Point Protocol (PPP). — (Дата обращения: 12.01.2016).

6. ietf.org/rfc/rfc2516.txt [Электронный ресурс]. — Режим доступа: http://www.ietf.org/rfc/rfc2516. — A Method for Transmitting PPP over Ethernet (PPPoE). — (Дата обращения: 20.03.2016).

7. Перевод RFC 2516 [Электронный ресурс]. — Режим доступа: на русском http://rfc.com.ru/rfc2516.htm. — Метод передачи PPP через Ethernet (PPPoE). — (Дата обращения: 20.03.2016).

8. ietf.org/rfc/rfc2637.txt [Электронный ресурс]. — Режим доступа: http://ietf.org/rfc/rfc2637.txt. — Point-toPoint Tunneling Protocol. — (Дата обращения: 27.03.2016).

9. ietf.org/rfc/rfc2661.txt [Электронный ресурс]. — Режим доступа: http://www.ietf.org/rfc/rfc2661.txt. — Layer Two Tunneling Protocol "L2TP". — (Дата обращения: 27.03.2016).

В. Ю. ЛИКУЧЕВ
Рязанский государственный радиотехнический университет

АНАЛИЗ РЕЙТИНГА СОВРЕМЕННЫХ ЯЗЫКОВ ПРОГРАММИРОВАНИЯ, СОСТАВЛЕННОГО ПО КРИТЕРИЮ ПОПУЛЯРНОСТИ

В статье проведен анализ рейтинга популярности современных языков программирования, а также выяснение причин большей популярности одних языков программирования относительно других.

Одним из актуальных вопросов, которым задаются студенты технических специальностей, в особенности, связанных с информатикой и вычислительной техникой, является вопрос о выборе конкретных языков программирования для их дальнейшего углубленного изучения. Дело в том, что в рамках учебной программы студенты полу-

чают лишь базовые знания по разработке программного обеспечения на тех или иных языках программирования, которых может не хватать в дальнейшей практике после окончания учебного заведения. Поэтому появляется необходимость дополнительного изучения языков программирования, выбор которых может осуществляться исходя из возможных запросов работодателей, а также оценки популярности языков. Эти критерии являются подчас взаимосвязанными.

Очевидно, что хороший специалист по разработке программного обеспечения должен быть знаком с несколькими языками программирования в мере достаточной для оперирования ими при решении различных задач. Ведь каждый язык обладает отличными от других свойствами и преимуществами, и кроме того, имеет определенную сферу применения. Однако, нередки случаи, когда в одной и той же области, например при разработке серверной части сайтов, почти одинаково хорошо применимы сразу несколько языков программирования. В таких случаях разработчик, в особенности тот, кто служит в корпоративном секторе и не находится в жестких рамках, связанных с указанием конкретных условий разработки, обязан сам выбирать языковое средство для выполнения поставленной задачи. И тут программист, руководствуясь совершенно субъективными понятиями, может выбрать тот язык, который по его мнению обладает исключительными преимуществами, или даже тот, с которым лучше всего знаком.

Распространенным критерием оценки актуальности имеющихся навыков программирования на тех или иных языках, а также при принятии стратегического решения о том, какой язык программирования выбрать в начале построения новой системы программного обеспечения, является критерий популярности языков. Появляется все больше сервисов, занимающихся оценкой популярности и построением рейтингов языков на основе этого критерия, причем оценка ведется разными методами. Из этого следует то, что данный критерий слишком неоднозначен.

Особым вниманием пользуется ежемесячный рейтинг от голландской компании «TIOBE», построенный на основании индекса популярности, который определяется частотой пользовательских запросов с упоминанием названия языка в наиболее посещаемых порталах. Запрос имеет вид: +"<language> programming".

Авторы проекта утверждают, что данный индекс характеризует популярность того или иного языка, так как подразумевается взаимосвязь между количеством запросов и количеством разработчиков на этом языке, соответствующих вакансий на рынке и курсов.

Рассмотрим рейтинг, составленный по данным на март 2016 года (рис.1).

Анализируя рейтинг, можно сделать вывод о том, что определенное положение в нем того или иного языка действительно зависит от регулярно сменяющихся тенденций в свере разработки программного обеспечения. Так например язык Objective-C, который использовался при создании большинства приложений для операционной системы IOS, значительно ухудшил свою позицию по сравнению с предыдущими годами, когда он занимал лидирующие места в рейтинге популярности. Это очевидно связанно с решением компании Apple заменить Objective-C языком Swift, который в свою очередь по этой же причине поднялся в рейтинге.

Mar 2016	Mar 2015	Change	Programming Language	Ratings	Change
1	2	^	Java	20.528%	+4.95%
2	1	v	C	14.600%	-2.04%
3	4	^	C++	6.721%	+0.09%
4	5	^	C#	4.271%	-0.65%
5	8	^	Python	4.257%	+1.64%
6	6		PHP	2.768%	-1.23%
7	9	^	Visual Basic .NET	2.561%	+0.24%
8	7	v	JavaScript	2.333%	-1.30%
9	12	^	Perl	2.251%	+0.92%
10	18	^^	Ruby	2.238%	+1.21%
11	13	^	Delphi/Object Pascal	2.005%	+0.85%

Рис. 1 - Рейтинг TIOB за март 2016 года.

Однако, не все так очевидно с другими языками, особенно с теми, которые занимают самые высокие позиции. Так, нельзя однозначно сказать, почему на данный момент язык Java опережает по популярности C++ (Java, кстати сказать, стал языком 2015 года по версии этой же компании), ведь оба языка одинаково широко применяются во многих областях, связанных с разработкой программного обеспечения. Кроме того, они имеют ряд схожих особенностей, среди которых поддержка объектно-ориентированного программирования, многопоточность, кроссплатформенность, отсутствие конкретной целевой ниши и даже схожий синтаксис. Одновременно, эти два языка имеют свои преимущества и недостатки относительно друг друга. К примеру, в Java поддержано автоматическое управление памятью («уборка мусора»), что позволяет сократить число характерных ошибок, распространенных при работе с C++. В свою очередь, C++ считается более производительным языком.

Но даже факт различия не дает ответа на вопрос об относительной популярности.

Проанализируем теперь число предлагаемых вакансий на места разработчиков программного обеспечения того или иного вида с помощью американского сервеса indeed. com (таблица 1).

Таблица 1. Рейтинг вакансий.

Место	Вид программного обеспечения
1	Web-приложения
2	мобильные и браузерные игры
3	мобильные приложения
4	приложения для ПК
5	драйверы

Данный рейтинг дает оценку количества предлагаемых вакансий на места разработчиков сетевых приложений, программ, требующих установки на компьютер, мобильных приложений, мобильных и браузерных игр, встроенных и поддерживающих систем, например драйверов и контроллеров. Как видно, большинство разрабатываемого обеспечения приходится на долю сетевых и мобильных приложений, где как раз таки широко применяется язык Java. Взять например платформу Android, для которой на Java написано большинство приложений, или крупные сетевые ресурсы, серверная часть которых реализована опять таки на Java. Напротив, приложения для персональных компьютеров, а также драйверы чаще реализуются на C++.

Однако, все равно нельзя сделать однозначных выводов, почему именно Java чаще используется при разработке более востребованных приложений, а не C++, который не менее подходит для решения данных задач.

Решающим фактором, как отмечалось в самом начале статьи, может быть субъективность при выборе языка программирования. Следовательно, чем больше специалистов, отдающих предпочтение тому или иному языку, тем популярнее он становится. Это в свою же очередь побуждает разработчиков самого языка всесторонне улучшать его возможности, что не может не сказаться на перспективности изучения именно этого языка программирования.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. [Электронный ресурс] www.tiobe.com

2. [Электронный ресурс] www.indeed.com

3. Павловская Т. А. «C/C++. Программирование на языке высокого уровня»,СПб.: Питер, 2014. –461с.

М.П. МЯЛКИН
Московский государственный технический университет им. Баумана

ПРОБЛЕМЫ РАЗРАБОТКИ МНОГОАГЕНТНЫХ СИСТЕМ С ПРИМЕНЕНИЕМ APACHE STORM

Рассматриваются проблемы, возникающие при разработке многоагентных систем, приводятся обзор фреймворка Apache Storm и его связь с многоагентными системами.

Целью исследования многоагентных систем является поиск методов, которые позволяют построить сложные системы на основе автономных агентов. Агенты в свою очередь обладают только локальными знаниями и ограниченными возможностями, однако в совокупности определяют необходимое поведение всей глобальной системы. В первую очередь необходимо проанализировать, каким образом можно разбить описание того, что должна делать целая система на поведение отдельных объектов. При этом необходимо учитывать, что в основном, поведение многоагентных систем основано на принципах функционирования таких систем, как колонии муравьев, экономика или иммунная система.

В основе многоагентных систем лежит понятие агентов [1]. Агент действует отдельно от других объектов и обрабатывает часть заданий системы. До появления искусственных многоагентных систем можно было представить человека, работающего в компании, в роли агента, а саму компанию в роли многоагентной системы.

Разработка искусственных многоагентных систем началась в 1960-х годах, при этом в ее основе лежали такие области, как параллельное вычисление, искусственный интеллект, распределенное решение задач. В данное время мультиагентные системы являются одним из приоритетных направлений в области интеллектуальных систем.

По сравнению с классическим способом решения задач, когда вычисление наилучшего варианта решения происходит по заранее определенному алгоритму, в мультиагентных системах решение формируется путем взаимодействия агентов. Агенты совершают действия автономно и рационально (для их действий не требуется вмешательство со стороны внешней среды) и стремятся достичь поставленной цели, совершая направленные действия. Общение между агентами происходит путем обмена сообщениями.

Данный подход чрезвычайно эффективен в случаях, когда заведомо наилучшего решения не существует либо оно трудно

вычислимо. Для достижения необходимого решения задачи агентам необходимо поставить цель, которую необходимо достичь, а сценарий достижения цели будет определять сам агент.

При проектировании многоагентной системы необходимо решить две основные проблемы.

1. Проблема проектирования агента (agent design). Исходными данными в этом случае являются внешняя среда, в которой находится агент и задача, которую агенту необходимо решить. Целью проектирования является конструирование поведения агента таким образом, чтобы под воздействием окружающей среды он смог решить поставленную ему задачу.

2. Проблема социального взаимодействия (social design). Необходимо спроектировать агента таким образом, чтобы он обладал свойством социальности, т.е. не только обменивался данными с другими агентами, но и был способен кооперироваться и координировать свои действия с ними. Для достижения свойства социальности агент должен иметь представление о целях других агентов и о том, как другие агенты собираются достичь эти цели.

Проблемы проектирования многоагентных систем

При проектировании прежде всего необходимо определить, что такое агент и какие сложности возникают при проектировании агентов. В первую очередь необходимо сказать, что универсального определения агента не существует, по данному определению ведется большое количество споров [4]. При этом наиболее распространенным определением является следующее: «Агент – это компьютерная система, которая расположена в некоторой внешней среде и имеющая возможность действовать автономно в этой среде для достижения поставленной цели» [4].

На рисунке 1 представлен общий вид агента, который совершает воздействие на окружающую среду в ответ на приходящее к нему воздействие.

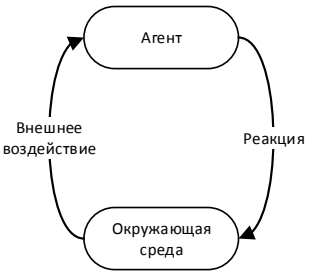


Рис. 1 – Взаимодействие агента и окружающей среды.

Необходимо понимать, что агент не может иметь полного контроля над окружающей средой. Контроль может быть только частичным - агент может лишь влиять на окружающую среду.

Окружающая среда, с которой взаимодействует агент, является недетерминированной, т.е. она может по-разному воспринимать в точности одинаковые реакции агента в разные моменты времени.

Обычно у агента есть некоторый набор действий, которыми он может повлиять на окружающую среду. Этот набор определяет *эффекторную возможность агента*, т.е. возможность изменять его окружающую среду. Важно понимать, что не все действия агента могут быть выполнены в разных ситуациях. У каждого действия есть условия, которые определяют, когда действие может выполняться, что можно представить в виде набора декларативных правил «если УСЛОВИЕ, то ВЫПОЛНИТЬ ДЕЙСТВИЕ». Такие правила обрабатываются машинами вывода агента. И эти машины вывода могут вычислить из текущих условий необходимую последовательность действий, которая приведет агента к назначенной цели.

Основная задача агента – решить, какое из действий он должен выполнить в текущий момент времени для того, чтобы максимально приблизиться к поставленной цели.

На рисунке 2 представлена структура не обучающегося агента.

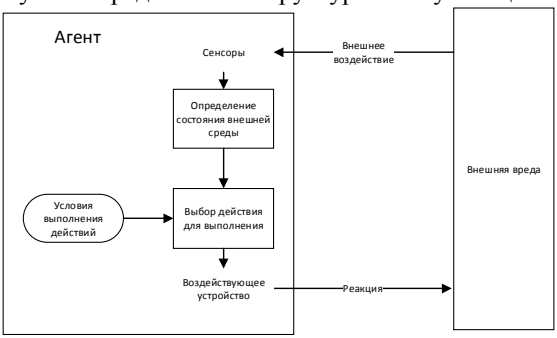


Рис. 2 – Структура простого объекта.

Рассмотрим подробнее свойства окружающей среды. Окружающая среда может быть:

- *Доступной или недоступной* – окружающая среда доступна, если агент может получать полную, точную, своевременную информацию о состоянии окружающей среды. С точки зрения агентов доступная среда предпочтительнее, потому что именно благодаря информации о состоянии внешней

среды агент решает, какое действие совершить в текущий момент. Однако, почти всегда окружающая среда не может считаться доступной.

- *Детерминированной или недетерминированной*. Окружающая среда будет считаться детерминированной, если любое производимое над ней действие будет приводить к точно определенному результату. В недетерминированной окружающей среде одно и то же действие, выполненное в разное время, может привести к разным последствиям и, соответственно, увести от желаемого результата. Поэтому дополнительная сложность реализации агентов в недетерминированных средах – необходимость постоянного отслеживания состояния окружающей среды и изменение действий со стороны агента по необходимости.
- *Статичной или динамичной*. Статичная окружающая среда изменяется только от действий агента. В динамичной окружающей среде происходят процессы, которые меняют её независимо от агентов. В динамичной окружающей среде перед тем, как выполнить какое-либо действие агент должен обязательно проверить ее состояние, а не надеяться на то, что оно остается неизменным в течение некоторого времени.
- *Дискретной или непрерывной*. В дискретной окружающей среде может быть только ограниченное количество состояний. Если окружающая среда является непрерывной, то при переводе ее состояния в дискретный вид (для обработки агентом) происходит некоторая потеря точности, что может повлечь за собой неправильное определение действия агента на окружающую среду.

Наиболее сложными окружающими средами являются недоступные, недетерминированные, динамичные, непрерывные среды, которые называют *открытыми*. При этом свойства окружающей среды определяют, в том числе, сложность разработки агента.

Важным вопросом, который необходимо рассмотреть, является природа *взаимодействия* между агентом и окружающей средой.

Взаимодействие агентов со средой происходит постоянно, при этом агенты являются реактивными системами. Это значит, что при взаимодействии со средой необходимо помнить, что локальные действия агента могут привести к глобальным последствиям.

Кроме этого существует, так называемая, проблема «честности». Она состоит в том, что выбор действия, считающегося разумным в

локальном контексте агента в текущее время, может привести к совсем другим, непредсказуемым результатам в контексте всей истории системы. То есть агент совершает действие, которое повлияет на систему в длительной перспективе, однако, как это действие повлияет на состояние системы, понять сложно.

Другим аспектом взаимодействия между агентом и средой является концепция реального времени. Можно выделить несколько типов взаимодействий реального времени:

- Решение о том, какое нужно предпринять действие, должно укладываться в некоторые временные рамки.
- Агент должен выполнить действие так быстро, как это возможно.
- Агент должен повторять действия так часто, как это возможно.

Если время не является проблемой для решения задачи, то агент может выбирать самое наилучшее решение из всех возможных, несмотря на рамки времени. Время в таком случае будет зависеть от количества действий, доступных агенту (агент перебирает любые цепочки из этих действий и выполняет единственную, которая ведет к наилучшему результату). Однако, для любых реалистичных сред такое поведение недопустимо.

Разумные агенты

Чтобы понять, какие агенты являются интеллектуальными, рассмотрим свойства, которыми должен обладать интеллектуальный агент:

- *Реактивность* – возможность поддерживать постоянное взаимодействие со средой и своевременно реагировать на изменения среды.
- *Проактивность* – возможность проявлять инициативу, т.е. управление не происходит исключительно событиями извне, агент может сам генерировать задачи и совершать действия, которые помогут их достичь.
- *Социальность* – возможность кооперироваться с другими агентами.

Как показывает практика, несложно достичь реактивности и проактивности независимо друг от друга, гораздо сложнее построить систему, которая будет соблюдать баланс этих свойств. При проектировании системы проблема соблюдения такого баланса является ключевой и требует решения в первую очередь.

Концепция BigData и Apache Storm

В настоящее время количество данных продолжает непрерывно

увеличиваться. Во многих случаях обработка этих данных стандартными способами будет занимать большое количество времени из-за резкого увеличения количества данных. Причем данные являются разнообразными и не всегда структурированными, что вносит дополнительную сложность их обработки. В любой системе обрабатывать данные нужно как можно быстрее. Все эти проблемы являются ключевыми в понятии BigData. Для многоагентных систем также актуальны такие проблемы в связи с типом и количеством информации, поступающей в систему из окружающей среды.

Apache Storm – это фреймворк, позволяющий распределенно обрабатывать большое количество информации, поступающей постоянно. Данный фреймворк может использоваться во многих случаях: аналитика в реальном времени, машинное обучение, постоянные вычисления и т.д. [5] В частности, его можно применить и к концепции многоагентных систем.

Рассмотрим работу Apache Storm подробее. В основе работы Storm лежит постоянная обработка данных, поступающих из внешней среды. Узлы объединяются в топологию, которая определяет, как данные будут передаваться внутри системы. Пример топологии изображен на рисунке 3.

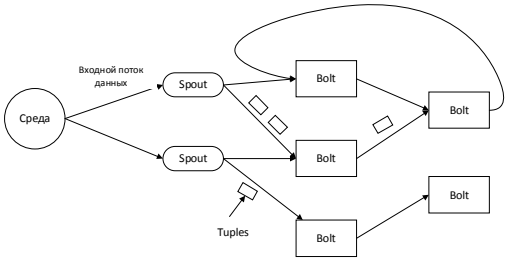


Рис. 3 – Пример топологии Apache Storm.

Данные берутся из потоков, называемых *Stream*. Stream – последовательность кортежей (*Tuple*). В кортеже находятся примитивные типы данных: числа, байты, строки, булевы значения. После того как данные получаются из входного потока, они перемещаются по системе в виде кортежей. В узлах происходит обработка данных из потока, изменение потока, его перенаправление и т.д.

Существует 2 вида узлов топологии:

- Spout - источник потоков, который только порождает потоки, беря данные из внешней среды. В качестве среды могут выступать различные источники данных: соединение с

сервером, база данных и т.д. На данный момент существует много библиотек, позволяющих обрабатывать разные внешние источники «из коробки».

- Bolt - принимает кортежи из входящих в него потоков, обрабатывает данные из них и может порождать новые потоки. Позволяет группировать, разделять потоки, фильтровать потоки. Может иметь доступ к базам данных.

Топология (Topology) представляет собой граф из Spout'ов и Bolt'ов. Ребра графа показывают направление движения кортежей в системе. В каждом узле топологии, будь то Bolt или Spout может работать несколько потоков, что позволяет обрабатывать данные параллельно. Топология, узлы и связи между ними указываются до запуска системы.

У Apache Storm существует два режима работы:

- Локальный – запускает топологию локально на 1 машине.
- Распределенный – запускает указанную топологию на кластере машин, связанных друг с другом. В данном режиме машины распределяются по обязанностям:
  - Nimbus – машина-мастер. Занимается распределением кода по всему кластеру машин, определение машинам их задач. Также проводит мониторинг падений отдельных машин.
  - Supervisor – является частью топологии, запускает определенное количество потоков и обрабатывает данные, поступающие на его узлы.
  - Zookeeper – координатор между Nimbus и Supervisor.

Состояния каждых процессов машин хранится на диске, поэтому при сбое вся система вернется в исходное состояние спустя очень малое количество времени.

Архитектура кластера Apache Storm представлена на рисунке 4.

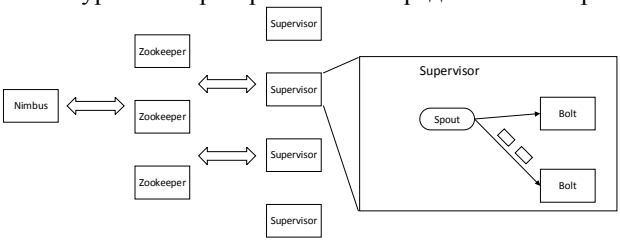


Рис. 4 – Архитектура кластера Apache Storm.

При реализации многоагентных систем на Apache Storm каждый узел топологии будет представлять собой агента, который

взаимодействует со связанными узлами.

Перечислим достоинства Apache Storm:

- *Простота использования*. Имеет всего 2 вида узлов, топология легко определяется, запуск кластера происходит автоматически.
- *Высокая производительность*. Возможность обрабатывать до 1 миллиона кортежей на одном узле.
- *Масштабируемость*. Возможность указывать количество узлов определенного типа и количество процессов в каждом узле. Если в системе наблюдается дисбаланс, то стоит изменить количество узлов или процессов обработки и кластер автоматически перенастроит конфигурацию на лету.
- *Устойчивость к сбоям*. Если процесс или поток падает, то он будет перезапущен. Если отказывает узел, то Nimbus развернет этот узел на другой машине. Все состояния хранятся на дисках, поэтому при восстановлении они будут также восстановлены, никакая информация не будет утеряна.
- *Гарантированная обработка данных*. С помощью механизма подтверждения приема кортежа Storm гарантирует, что все данные будут корректно обработаны и не будут потеряны.
- *Поддержка большого количества языков*. Существует поддержка Python, Ruby, Java, NodeJS, Scala, Perl и т.д.
- *Свободно распространяемый, открытый исходный код.*

Недостатки Apache Storm заключаются в следующем:

- *Статичная архитектура топологии.* Нет возможности изменять архитектуру динамически, в процессе работы системы – это значит, что агенты смогут влиять только на определенные части системы.
- *Не слишком подходит для описания многоагентных систем.* Не предоставляет возможностей работать с этой концепцией по умолчанию.

Применение Apache Storm для разработки многоагентных систем

Для того, чтобы создать многоагентную систему, необходимо определить, что будет являться агентами, как они будут принимать информацию из окружающей среды и влиять на нее. А также как агенты будут взаимодействовать между собой.

В кластере Apache Storm агентами будут являться отдельные Bolt-узлы. Они способны принимать информацию, обрабатывать ее и отправлять наружу системы, то есть влиять определенным способом на

окружающую среду.

Информация из окружающей среды в систему поступает через Spout-узлы. Для того, чтобы каждый агент мог реагировать нужным образом на изменение окружающей среды, Spout-узлы необходимо соединить с каждым Botl-узлом. Таким образом, данные будут поступать на каждого агента. Агент может влиять на окружающую среду, передавая обработанную информацию на внешние носители, например, базу данных. Для Apache Storm уже существует достаточное количество библиотек, позволяющих работать с различными внешними системами.

Чтобы агенты взаимодействовали друг с другом, необходимо связать Bolt-узлы двусторонними связями, где это необходимо. Нужно помнить, что связи не могут перестраиваться и быть изменены во время работы системы. Поэтому, нужно сразу определить, какие агенты будут взаимодействовать между собой. Можно эмулировать изменение связей в процессе работы системы. В таком случае понадобится связать каждого агента со всеми другими агентами в системе и добавить в него логику обработки сообщений от других участников в установленный момент времени.

Например, многоагентную систему можно реализовать с помощью Apache Storm, как показано на рисунке 5.

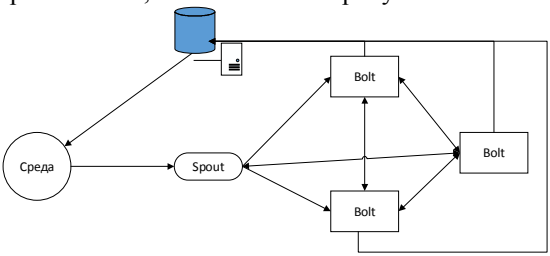


Рис. 5 – Структура многоагентной системы на Apache Storm.

Заключение

В статье рассмотрены основы многоагентных систем, описаны виды агентов и их свойства. Выделены проблемы, которые предстоит решить при реализации многоагентных систем: необходимость корректной настройки взаимодействия агента со средой (особенно если она является открытой), настройка взаимодействия агентов между собой для достижения общих целей, настройка баланса между проактивностью и реактивностью агентов. Рассмотрен фреймворк для распределенной обработки большого количества информации Apache Storm, выделены его достоинства и недостатки, его связь с многоагентными системами и пример реализации многоагентных

систем на данном фреймворке.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Мультиагентные технологии. Режим доступа: [Электронный ресурс] http://www.intuit.ru/studies/courses/10618/1102/lecture/17391 (дата обращения 28.02.16)

2. M. Wooldridge, P. E. Dunne. The complexity of agent design problems: Determinism and history dependence [Annals of Mathematics and Artificial Intelligence]. Springer, 2005. P. 343-371.

3. M. Wooldridge. An introduction to multiagent systems. Wiley, 2012. P. 484.

4. J. Ingham. What is an Agent? Technical report. #6 1999. P. 16.

5. Apache Storm. Available at: http://storm.apache.org/index.html, accessed 01.03.16.

6. Гапанюк Ю.Е., Терехов В.И., Черненький В.М. Структура гибридной интеллектуальной информационной системы на основе метаграфов. М., «Нейрокомпьютеры: разработка, применение». № 4, 2016.

А.С. ОВЧИННИКОВА
Рязанский государственный радиотехнический университет

ПОСТРОЕНИЕ МОДЕЛИ ГАУССОВЫХ СМЕСЕЙ ДЛЯ РЕШЕНИЯ ЗАДАЧИ РАСПОЗНАВАНИЯ ДИКТОРОВ

Рассматривается порядок формирования модели голоса диктора с использованием гауссовых смесей (Gauss Mixture Model, GMM).

При работе системы распознавания диктора создается образец голоса человека в виде последовательности векторов его признаков. Векторы признаков строятся посредством вычисления одного или нескольких видов коэффициентов, с помощью которых образец голоса может быть представлен в компактном виде при сохранении индивидуальных особенностей. Наиболее распространенными типами коэффициентов являются мел-частотные кепстральные коэффициенты (Mel-Frequency Cepstral Coefficients, MFCC), коэффициенты линейного предсказания (Linear Prediction Coefficients, LPC) и их усложненная реализация - перцепционные коэффициенты линейного предсказания (Perceptual Linear Predictive, PLP) [1]. После вычисления векторов признаков (один вектор описывает один фрейм записи длиной 10-40 мс) на их основе формируется модель диктора – непосредственно образец

его голоса, с которым затем будет сравниваться модель распознаваемого диктора. Одним из способов построения модели является модель гауссовых смесей (Gauss Mixture Model, GMM).

Сначала осуществляется векторное квантование множества полученных векторов: множество разбивается на конечное количество кластеров. В каждом кластере выбирается вектор «среднего значения» для векторов этого кластера, который можно будет использовать как описание их множества. Для выполнения квантования используют алгоритм K-средних (K-means). На первой итерации алгоритма выбираются произвольные начальные точки (векторы). Строится диаграмма Вороного: такое разбиение пространства на области, что в каждой области находится одна из выбранных точек и множество других точек пространства, наиболее близких к ней (по сравнению с другими начальными точками) [2]. На следующей итерации для полученных областей переопределяются центральные точки (они могут не совпадать с существующим вектором пространства) как центры масс этих областей. Снова выполняется разбиение Вороного для новых центральных точек. Алгоритм повторяется до схождения (момента, когда результат перераспределения центров и их областей на текущей итерации будет равен результату предыдущего шага).

Полученное разбиение векторов на области-кластеры и центральные точки (кодовая книга) используются для построения модели диктора.

Модель гауссовых смесей представляет собой сумму $p(\vec{x} \mid \lambda) = \sum_{i=1}^{M} \omega_i b_i(\vec{x})$, в которой $\vec{x}$ – вектор значений признаков, M – количество кластеров, $b_i(\vec{x})$ – функция плотности вероятности i-ого компонента, ω_i – его вес. Модель также можно представить в виде:

$$\lambda = \{\omega_i, \vec{\mu}_i, \Sigma_i\}, i = \overline{1, M},$$

где $\vec{\mu}_i$ – вектор математического ожидания компонента i, Σ_i – матрица ковариации, по диагонали которой располагаются дисперсии элементов вектора, а остальные элементы заполняются попарными ковариациями элементов вектора $\vec{x}$. Ковариация двух величин $\mathrm{cov}(a,b) = M\left[(a - Ma)(b - Mb)\right]$, где Ma и Mb – значения математических ожиданий величин *a* и *b*.

На практике [3] для определения весов компонентов ω_i и матриц ковариации Σ_i используется EM-алгоритм (Expectation-Maximization

algorithm) [4]. На начальной итерации на шаге Expectation вектор математического ожидания $\vec{\mu}_i$ инициализируется центральным вектором i-ого кластера (кодовым). Ковариационная матрица вычисляется из попарных ковариаций значений векторов, принадлежащих этому кластеру. Вес принимает значение доли векторов этого кластера от общего количества векторов. После инициализации параметров вычисляются значения апостериорных вероятностей компонента i при x_t – один из T векторов признаков:

$$p(i \mid x_t, \lambda) = \frac{\omega_i p_i(x_t)}{\sum_{n=1}^{N} \omega_n p_n(x_t)}.$$

На шаге Maximization заданные значения переопределяются следующим образом:

$$\omega_i = \frac{1}{T} \sum_{t=1}^{T} p(i \mid x_t, \lambda);$$

$$\mu_i = \frac{\sum_{t=1}^{T} p(i \mid x_t, \lambda) x_t}{\sum_{t=1}^{T} p(i \mid x_t, \lambda)};$$

$$\Sigma_i = \frac{\sum_{t=1}^{T} p(i \mid x_t, \lambda)(x_t - \mu_i)(x_t - \mu_i)^T}{\sum_{t=1}^{T} p(i \mid x_t, \lambda)}.$$

Осуществляется возврат к шагу Expectation с новыми значениями. Алгоритм повторяется до его сходимости. Полученных на последней итерации набор $\lambda = \{\omega_i, \vec{\mu}_i, \Sigma_i\}, i = \overline{1, M}$ будет представлять собой модель гауссовых смесей диктора.

Гауссова смесь позволяет наиболее точно представить распределение мел-частотных кепстральных характеристик, полученных при вычислении MFCC, как совокупность гауссовых нормальных распределений случайной величины [5]. Полученная таким образом модель диктора может использоваться как конечный шаблон его голоса. При запросе на аутентификацию диктора строится его модель по аналогичному алгоритму и сравнивается с моделью-шаблоном.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Hermansky Hynek. Perceptual linear predictive (PLP) analysis of speech // the Journal of Acoustical Society of America. 1990. Vol. 2. No. 4. P: 578 – 589.
2. Препарата Ф., Шеймос М. Вычислительная геометрия: Введение // М.: Мир. 1989. С: 295.
3. Садыхов Р.Р., Ракуш В.В. Модели гауссовых смесей для верификации диктора по произвольной речи // Минск, БГУИР. 2003. С: 97-98.
4. Hastie T., Tibshirani R., Friedman J. The EM algorithm // The elements of Statistical Learning. New York: Springer. 2001. P: 236-243.
5. Сорокин В.Н., Вьюгин В.В., Тананыкин А.А. Распознавание личности по голосу: аналитический обзор // Информационные процессы. 2012. т.12, №1. С: 16-19.

Е.А. ОКУНЦЕВ
Рязанский Государственный Радиотехнический Университет

ОБЗОР ЛИТЕРАТУРЫ ТЕХНОЛОГИИ СОЗДАНИЯ СИСТЕМЫ «УМНЫЙ ДОМ»

Рассматривается литература содержащая описание и технологии производства «умного дома», а так же между литературными источниками производиться сравнение в содержательности и близости к современным реалия.

Сфера автоматизации домов и зданий постоянно развивается. Системы «Умных домов» постоянно обновляются, усовершенствуются, а на место старых технологий, все чаще приходят новые инновационные разработки. В настоящее время существует и постоянно добавляется большое количество систем автоматизации, начиная от бюджетных модулей или плагинов, которые легко переоснащают ранее выпущенные системы управления, добавляя им новые возможности, и заканчивая профессионально разработанными комплексами, требующих квалифицированных специалистов и установщиков оборудования, а также инженерного проектирования на ранних стадиях строительства дома. Термин «Умный Дом» или другими словами «Автоматизация дома / Домашняя автоматизация» охватывает много различных областей, в которые входят Освещение, Теплоснабжение и Кондиционирование (HVAC), управление Энергоснабжением, системы Безопасности и Видеонаблюдения, Мультимедиа-системы и многие другие.

Соответственно появляется и литература рассматривающая эти системы, но большинство из них является в большей степени теоретическим описанием системы «умного дома», но никаких новых технологических способов создания данной системы не описывается. Рассмотрим некоторые литературные источники.

Первая книга на которой хотелось бы остановиться «Практические советы и решения по созданию «Умного дома»« Марка Эдварда Сопера - в этой книге автор останавливается в основном на двух темах: Во-первых, автор рассказывает, что было бы очень хорошо всё во круг себя в доме автоматизировать, что все уже начинают автоматизировать, а ты можешь не успеть. Во-вторых, перечисляет достаточно большое количество готовых устройств (большинство из которых принадлежат двум фирмам: smarthome и x10). В целом же книга представляет с собой теоретические основы технологии «умный дом», стандарты передачи данных и советы по автоматизации с помощью готовых и достаточно дорогих устройств.

Вторая же книга о которой я расскажу «Умный дом. Объединение в сеть бытовой техники и системы коммуникаций в жилищном строительстве» Вернер Харке. В этом литературном источники описана концепция создания «умного дома» в инженерном плане, но в основном для архитекторов, планировщиков, специалистов в области безопасности, отопления и частично специалистов в области радио- и телекоммуникаций, а также IT-специалистов, осуществляющих объединение это все в единое целое. Так же автор позаботился о читателях и вставил в конце источника подробный словарь и список адресов для реализации собственного проекта.

В целом же большая часть литературных источников по этой тематике достаточно однообразны и рассказывают в основном общие характеристики технологии и занимаются рекламой технологий занятых в этой сфере фирм, а так же популяризация самой технологии, но для специалистов там найдется лишь пара интересных глав, но есть и нацеленные на создание специалистов в этой области.

Например, книга Марии Устелемовой «Основы построения системы «умный дом»«, которая читателя, обладающему начальными знаниями о технологии, полностью вводит в тему «умного дома» , а так же подробно останавливается на отличиях от обычного дома и рассматривает экономические преимущества от использования «умного дома». Стоит отметь, что автор так же рассматривает эту технологию в рамках современных наук и предлагает свою концепцию уже более широкой системы - «умный город», как развитие системы «умного дома» , в тоже время рассматриваются этапы внедрения этой автома-

тизированной системы управления, ожидаемые результаты и препятствия к осуществлению. Также в данной книге рассмотрена статистика, касательно экономической эффективности от внедрения системы «умный город». Хотелось бы еще заметить, что в этом литературном источники так же имеются два теста с восемью заданиями по разделам данной книги и вопросы для экзамена, поэтому нельзя не отметить образовательный потенциал данной книги.

Самым же интересным встретившимся мне материалом в процессе изучения литературного пространства были методические указания для студентов и аспирантов Владимирского государственного университета имени А.Г и Н.Г. Столетовых под авторством А.Н. Стариков, С.И. Рощина, А.В. Власов. В данном литературном источнике описан полный спектр всех современных технологий входящих в состав «умного дома», но самое главное практическое подкрепление к подробно теоретическому материалу изложенному в этих методических указаниях к почти всем технологиям сейчас использующихся в проектировании дома, как автоматизированной системы. В практических работах, которые грамотно представлены авторами данного методического материала существует необходимая современность примеров и устройств объединяемых в единую систему, а так же сравнение всех рассматривающихся технологий, а так же рекомендации к их примирению на довольно интересных примерах. Общее впечатление от изучения этих методических указаний очень хорошее, особенно на фоне книг для мягко говоря любителях в области подобных технологий, оно подкрепляется необходимым по объему и пониманию описанием всего изложенного в источнике: начиная от постановки проблемы и заканчиваю преимуществами от разных вариантов решения этой проблемы. Поэтому рекомендовал бы всем, кому интересна тематика «умного дома», ознакомиться с данным материалом.

В заключение хотелось бы сказать, что разработка «умных жилищ» идет полным ходом и с применением самых современных технологий, литературное пространство по этому вопросу достаточно узкое и большинстве представляет любительский уровень, что достаточно плохо, так как в нашем обществе существует запрос на специалистов в этой области, и это мы можем заметить, взглянув на себя или кого-то рядом и увидев, что времени и сил на домашние хлопоты остается все меньше и меньше, потребности в уровне комфорта и безопасности повышаются и повышаются. А вырастить достойного специалиста можно, лишь опираясь на качественную теоретическую и литературную базу.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Сопер, М.Э. Практические советы и решения по созданию «Умного дома» / М.Э. Сопер // НТ Пресс. 2007.

2. Харке, В. Умный дом. Объединение в сеть бытовой техники и системы коммуникаций в жилищном строительстве / В. Харке // Техносфера. 2006.

3. Элсенпитер, Р.К. Умный дом строим сами / Р.К. Элсенпитер, Т.Д. Велт. // КУДИЦ-Образ. 2005.

4. Богданов, С.В. Умный дом / С.В. Богданов. // Наука и техника. 2005.

5. Тесля, Е.В. «Умный дом» своими руками. Строим интеллектуальную цифровую систему в своей квартире / Е.В. Тесля. // Питер. 2008.

6. Кашкаров, А.П. Электронные схемы для «умного дома» / А.П. Кашкаров. // НТ Пресс. 2007.

7. Гололобов, В.Н. «Умный дом» своими руками / В.Н. Гололобов // НТ Пресс. 2007.

8. Стариков А.Н. «Умный дом»: методические указания для слушателей курсов повышения квалификации / А.Н. Стариков, С.И. Рощина, А.В. Власов, Владим. гос. ун-т имени Александра Григорьевича и Николая Григорьевича Столетовых. - Владимир : Изд-во ВлГУ, 2014.

9. Устелемова, М.С. Основы построения системы «умный дом» / М.С. Устелемова // Интернет-Университет Информационных Технологий. 2010.

Д.А. ПЕРЕПЕЛКИН, В.С. БЫШОВ
Рязанский государственный радиотехнический университет

СИСТЕМА МОДЕЛИРОВАНИЯ БАЛАНСИРОВКИ НАГРУЗКИ В ПРОГРАММНО-КОНФИГУРИРУЕМЫХ СЕТЯХ

Разработана система моделирования процессов балансировки нагрузки в программно-конфигурируемых сетях. В работе приведено исследование и сравнение известных алгоритмов балансировки нагрузки: k-кратчайших путей (алгоритм Йена) с использованием модуля Traffic Engineering (TE) и алгоритма парных перестановок маршрутов.

Современные протоколы динамической маршрутизации, такие как OSPF и IS-IS [1] позволяют обойти сильно загруженные каналы связи, чтобы улучшить качество обслуживания сетевых сервисов и

приложений. Однако при частом обновлении маршрутной информации необходимы дополнительные вычислительные ресурсы для активации нового маршрута и его установки. Протоколы многопутевой маршрутизации, как правило, используют в своей работе алгоритм *k*-кратчайших путей (алгоритм Йена). Многопутевая маршрутизация позволяет сбалансировать нагрузку на сеть путем распределения сетевого трафика по резервным маршрутам. Развитие в последнее время технологии программно-конфигурируемых сетей [2] позволяет формулировать различные задачи оптимальной маршрутизации и балансировки сетевого трафика [3].

В данной работе основное внимание будет сосредоточено на сравнительном анализе модуля TE (Traffic Engineering, TE), работающего с маршрутами, вычисленными с помощью алгоритма Йена и балансировки нагрузки, включающей в себя три коэффициента: α, β, γ, дающих больше возможностей управления нагрузкой, основанной на работе алгоритма парных перестановок маршрутов (АППМ) [4].

В рамках проведенных исследований разработано программное обеспечение моделирования процессов многопутевой адаптивной маршрутизации и балансировки нагрузки. Приложение разработано на языке программирования C# с использованием платформы .NET Framework 4.5. Система позволяет построить модели компьютерных сетей, найти оптимальные и резервные маршруты, выполнить балансировку нагрузки. Сравнительный анализ рассматриваемых алгоритмов выполнялся на структуре компьютерной сети, показанной на рис. 1. На рис. 2-6 представлены результаты работы рассмотренных алгоритмов балансировки нагрузки.

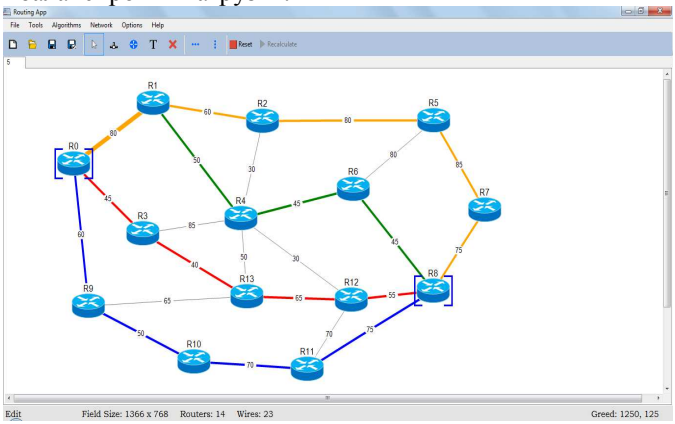


Рис. 1 – Интерфейс программной системы и найденные маршруты. с помощью алгоритма Йена.

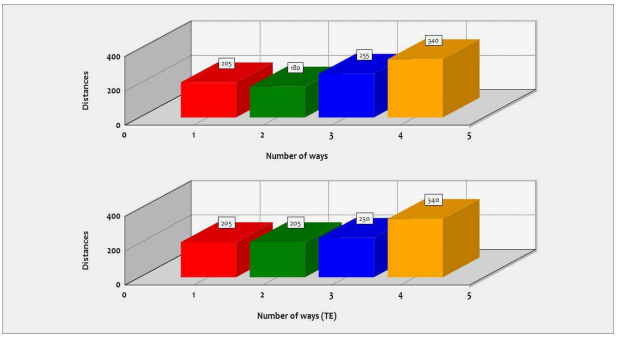


Рис. 2 – Результаты работы модуля TE на основе маршрутов, вычисленных с помощью алгоритма Йена ($\alpha = 0.7$).

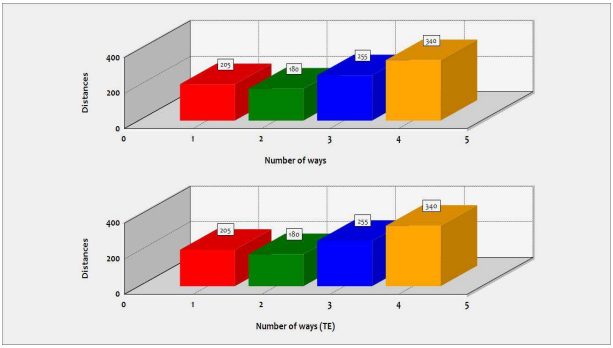


Рис. 3 – Результаты работы модуля TE на основе маршрутов, вычисленных с помощью алгоритма Йена ($\alpha = 0.8$).

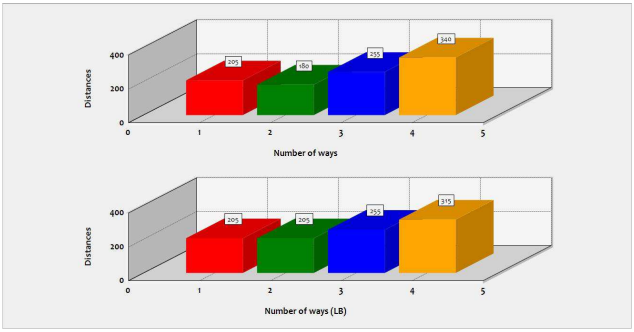


Рис. 4 – Результаты работы АППМ с балансировкой нагрузки ($\alpha = 0.7, \beta = 0.7, \gamma = 0.7$).

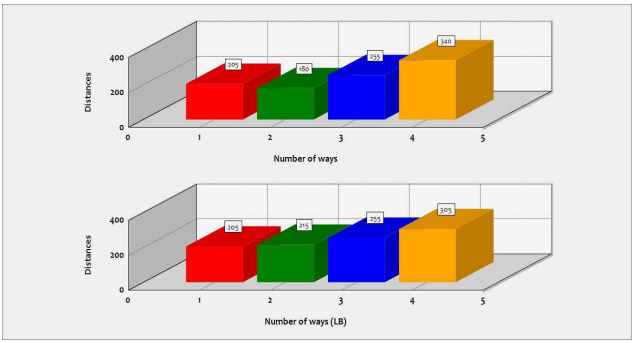


Рис. 5 – Результаты работы АППМ с балансировкой нагрузки ($\alpha = 0.8, \beta = 0.7, \gamma = 0.7$).

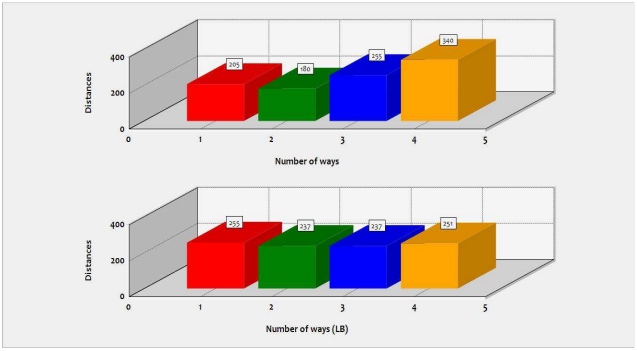


Рис. 6 – Результаты работы АППМ с балансировкой нагрузки ($\alpha = 0.8, \beta = 0.9, \gamma = 0.9$).

В таблице 1 представлены основные данные рассматриваемой топологии компьютерной сети: N – количество маршрутизаторов, M – количество каналов связи, D – диаметр графа, CD – общая длина, т.е. общая маршрутная метрика всех доступных маршрутов между маршрутизатором-источником и маршрутизатором-получателем, P – номер маршрута, Ds – длина маршрута, I – информация, проходящая через данный маршрут, AV – среднее значение каналов, входящих в машрут, SD – среднеквадратичное отклонение каналов, входящих в маршрут, MxVL – максимальное значение канала в маршруте, MnVL – минимальное значение канала в маршруте, J – отклонение значения длины текущего маршрута от длины оптимального маршрута, α – коэффицент балансировки нагрузки, β – отклонение длины минимального маршрута от длины максимального маршрута, γ – отклонение метрики канала, находящемся в минимальном маршруте, от метрики ка-

нала, находящемся в максимальном маршруте, которые состоят в отношении парного перехода.

Таблица 1. Первоначальная информация о топологии.

Topology	P	Ds	I,%	AV	SD	Mx VL	Mn VL	J,%
N = 14	1	205	28	51,3	9,60	65	40	47
M = 23	2	180	32	45	3,54	50	40	
D = 4	3	255	23	63,8	9,60	75	50	
CD = 980	4	340	17	68	16,3	85	40	

Результаты работы алгоритма Йена совместно с модулем ТЕ и алгоритма парных перестановок с балансировкой нагрузки представлены в таблицах 2 и 3 соответственно.

Таблица 2. Результаты работы алгоритма Йена с модулем ТЕ.

A	P	Ds	I,%	AV	SD	Mx VL	Mn VL	J,%
0,7	1	205	29	51,3	9,6	65	40	39
	2	205	29	51,3	11,4	70	40	
	3	230	26	57,5	8,4	70	50	
	4	340	17	68	16,3	85	40	
0,8	1	205	28	51,3	9,6	65	40	47
	2	180	32	45	3,6	50	40	
	3	255	23	63,8	9,6	75	50	
	4	340	17	68	16,3	85	40	
0,9	1	205	28	51,3	9,6	65	40	47
	2	180	32	45	3,6	50	40	
	3	255	23	63,8	9,6	75	50	
	4	340	17	68	16,3	85	40	

Таблица 3. Результаты работы АППМ с балансировкой нагрузки.

A	β	Г	P	Ds	I,%	AV	SD	Mx VL	Mn VL	J,%
0,7	0,7	0,7	1	205	29	51,3	9,6	65	40	34
			2	205	29	51,3	11,4	70	40	
			3	255	23	63,8	9,6	75	50	
			4	315	19	63	17,3	85	40	
0,7	0,9	0,9	1	205	29	51,3	9,6	65	40	34
			2	205	29	51,3	11,4	70	40	
			3	255	23	63,8	9,6	75	50	
			4	315	19	63	17,3	85	40	
0,8	0,9	0,9	1	255	24	63,8	14,8	80	40	7
			2	237	26	59,3	13,5	80	45	
			3	237	26	59,3	13,6	75	42	
			4	251	24	50,2	32,4	85	1	
0,9	0,7	0,9	1	250	24	62,5	15,6	84	40	11

			2	225	27	56,3	19,8	90	40	
			3	255	24	63,8	9,6	75	50	
			4	250	24	50	32,6	85	1	
0,9	0,9	0,7	1	234	26	58,5	17,4	84	40	15
			2	225	27	56,3	19,8	90	40	
			3	255	24	63,8	9,6	75	50	
			4	266	23	53,2	30,6	85	1	
0,9	0,9	0,9	1	250	24	62,5	15,6	84	40	4
			2	240	26	60	17,7	90	45	
			3	240	26	60	12,7	75	45	
			4	250	24	50	32,5	85	1	

Анализ полученных результатов показывает, что применение АППМ с балансировкой нагрузки в отличие от классического ТЕ-модуля позволяет балансировать нагрузку не только между смежными интерфейсами, но и проложенными маршрутами за счет контроля над параметрами α, β, γ. С добавлением двух новых коэффициентов (β и γ) появляется дополнительная возможность контроля работы алгоритма: большая часть загруженных каналов и, следовательно, маршрутов в компьютерной сети будут сбалансированы (разгружены), тогда как в случае классического подхода, возможно, будет только уменьшена метрика максимально-загруженного канала в маршруте, а отклонение от длины оптимального маршрута изменится незначительно.

В соответствии с таблицами 2 и 3 на основании полученных результатов можно сделать следующие выводы: после введения двух коэффициентов β и γ со значением 0.7, отклонение длины максимального маршрута от оптимального маршрута сразу же уменьшилось на 5%. При α >= 0.8 классический метод перестает срабатывать, т.к. загрузка всех каналов связи в топологии находится в пределах указанного значения коэффициента, следовательно, отклонение от длины оптимального маршрута остается прежним. В АППМ при α >= 0.8 вступают в работу введенные коэффициенты β и γ, благодаря которым удается уменьшить значение отклонения длины от оптимального маршрута, что говорит о целесообразном их использовании.

Как видно из таблицы 3 при установке коэффициента α больше 0.8, разрешается загрузка каналов связи более чем на 80%, и алгоритм балансировки нагрузки выравнивает почти все маршруты, отклонение длины которых будет стремиться к 0. К сожалению, данное действие не оставляет в каналах связи дополнительного резерва, тем самым увеличивая шансы потерь пакетов данных и вероятность отказа данных каналов. Поэтому желательно варьировать коэффициент α от 0.7 до 0.8, так как дальнейшее увеличение α не существенно повлияет на результаты работы. Коэффициенты β и γ позволяют выравнивать длины

смежных маршрутов и веса смежных каналов соответственно, поэтому их следует устанавливать в пределах от 0.8 до 0.9.

Подводя итог, можно сделать вывод о том, что варьирование коэффициента α от 0,7 до 0,8 и коэффициентов β и γ от 0.8 до 0.9 обеспечит требуемое качество обслуживание и оптимизацию загрузки каналов связи в компьютерной сети.

Работа выполнена при финансовой поддержке гранта Президента РФ для молодых ученых - кандидатов наук МК-6016.2016.9 и гранта РФФИ 16-47-620300 р_а.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Sridharan A., Gu´erin R., and Diot C., Achieving near-optimal traffic engineering solutions for current ospf/is-is networks, in *IEEE/ACM Trans. Netw.*, vol. 13, 2005, pp. 234–247.

2. McKeown N., Anderson T., Balakrishnan H. et al. Openflow: Enabling Innovation in Campus Networks // ACM SIGCOMM Computer Communication Review. – 2008. – Vol. 38, № 2. – P. 69–74.

3. Корячко В.П., Перепелкин Д.А. Разработка и исследование математической модели многопутевой адаптивной маршрутизации в сетях связи с балансировкой нагрузки // Электросвязь. 2014. № 12. С. 27–31.

4. Перепелкин Д.А., Перепелкин А.И. Алгоритм адаптивной ускоренной маршрутизации в условиях динамически изменяющихся нагрузок на линиях связи в корпоративной сети // Информационные технологии. 2011. № 3. С. 2–7.

Д.А. ПЕРЕПЕЛКИН, И.П. КАПАКЛЫ
Рязанский государственный радиотехнический университет

АДАПТИВНАЯ МАРШРУТИЗАЦИЯ В ПРОГРАММНО-КОНФИГУРИРУЕМЫХ СЕТЯХ

Рассмотрены основные алгоритмы адаптивной маршрутизации, применяемые в программно-конфигурируемых сетях.

Программно-конфигурируемые сети (Software Defined Networks, SDN) – это разделение плоскости передачи и управления данными, позволяющее осуществлять программное управление потоками данных, которые могут быть физически или логически отделены от аппаратных коммутаторов и маршрутизаторов. Появление новой возмож-

ности использования высокопроизводительных процессоров для управления современными сетевыми коммутаторами, позволило объединить компьютеры и сетевое оборудование на основе новой технологии OpenFlow, реализующей принципиально новый механизм обработки динамических сетевых нагрузок.

Традиционные коммутаторы реализуют как быструю переадресацию пакетов (data path – тракт передачи данных), так и высокоуровневую маршрутизацию (control path – тракт управления данными). OpenFlow обеспечивает независимость функций высокоуровневого управления от аппаратного обеспечения, за счет чего ускоряются процессы пересылки и маршрутизации. В сети OpenFlow вся интеллектуальная часть реализуется на центральном сервере или контроллере, что упрощает выполнение сложных операций. Идея заключается в использовании для управления такими сетями стандартных серверов или контроллеров, работающих отдельно от сетевых устройств, благодаря чему сетевые администраторы будут получить детализированный контроль над трафиком.

Как видно из рис.1 механизм работы OpenFlow маршрутизатора достаточно простой. OpenFlow маршрутизатор состоит из одной или нескольких flow-таблиц (flow-table), групповой таблицы(group-table) и openflow канала (OpenFlow channel) для доступа к удаленному контроллеру (controller). Маршрутизатор обменивается сообщениями с контроллером при помощи протокола OpenFlow. Используя данный протокол, контроллер может добавлять, обновлять и удалять flow-записи (flow-entries) в flow-таблицах, как реактивно (reactive – ответы на входящие пакеты), так и проактивно (proactive). Flow-таблица содержит набор flow-записей, каждая запись определяется полями сравнения, счетчиками (counters) и набором инструкций (instructions), которые применяются к пакету с совпавшими полями.

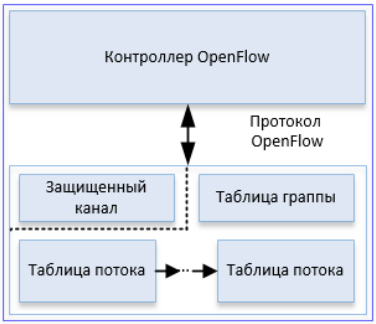


Рис. 1 – Структура управления в OpenFlow.

Сравнение начинается с первой flow-таблицы и может продолжаться в других таблицах. Поля сравнений flow-записей сравниваются с заголовком пакета в порядке приоритета (priority – одно из полей сравнения). Если найдена совпадающая flow-запись, к пакету применяются инструкции (instructions) ассоциированные с данной записью. Если не найдено ни одной записи, результат зависит от конфигурации маршрутизатора, пакет может быть сброшен (dropped), либо передан контроллеру по OpenFlow каналу для анализа и принятия решения. Инструкции в соответствующих записях содержат либо действия (actions), которые применяются к пакету и определяют непосредственно сами правила пересылки, либо определяют дальнейшую обработку пакета, в последующих flow-таблицах. Конвейерная обработка (pipeline processing) позволяет передавать пакеты из одной таблицы в другую, дополнительно пересылая специальные метаданные, например, числовые переменные, связанные с предыдущей обработкой. Такой подход позволяет создавать «ветвление» в обработке трафика. Обработка заканчивается, когда соответствующие инструкции не содержат команд по пересылки в следующую таблицу, в этот момент выполняется модификация пакета и выполнение «накопленных» действий. В большинстве современных коммутаторов Ethernet используются таблицы потоков, которые описывают, как наиболее эффективно доставить пакет от отправителя к пункту назначения. У каждого поставщика таблица потоков своя, однако, можно выделить набор функций, общих для большинства коммутаторов старшего класса, например, качество обслуживания и отчетность по трафику.

Протокол OpenFlow предоставляет единый API, с помощью которого администраторы могут программировать работу сети, а также задавать правила маршрутизации пакетов, балансировки нагрузки и управления доступом. В качестве правил выбора оптимального маршрута для сети, часто используют следующие алгоритмы маршрутизации: Дейкстры, Йена и Беллмана-Форда. На основе серии модельных экспериментов обоснована эффективность применения технологии программно-конфигурируемых сетей для адаптивного управления трафиком. Для выполнения эксперимента были использованы следующие открытые программные средства: SDN (операционная система программно-конфигурируемой сети в который встроены все существующие контроллеры OpenFlow), POX (контроллер OpenFlow).

На рис. 2 представлена топология, состоящая из шести маршрутизаторов (s0, s1, s2, s3, s4, s5), объединённых каналами передачи данных, и два конечных устройства, подключенных к двум из маршрутизаторов.

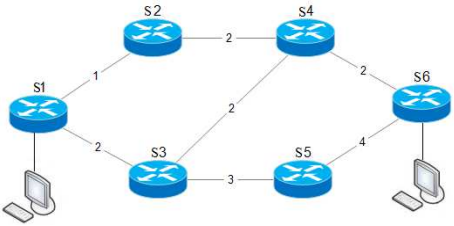


Рис. 2 – Модель сети для алгоритма Дейкстры.

Алгоритм Дейкстры находит кратчайшие пути от одной из вершин графа до всех остальных и широко применяется в протоколах маршрутизации OSPF и IS-IS. Алгоритм Дейкстры, реализован программно на контроллере POX. Для передачи пакета из первого хоста h1, подключённого к коммутатору s0, контроллер проверяет существует ли правило, по которому будет отсылаться пакет. В случае если такое правило не будет найдено, то пакет будет обработан и контроллер создаст для него маршрут по алгоритму Дейкстры, по которому будет отсылаться все последующие пакеты. Для алгоритма Дейкстры, маршрут создаётся исходя из параметров сети (задержка, пропускная способность, процент потерь пакетов). На рисунке 3 представлены результаты исследования, где вычислен оптимальный маршрут, исходя из задержки между маршрутизаторами, для передачи пакетов из первого хоста h1 ко второму h2 и обратно.

```
src= 1  dst= 6  first_port= 1  final_port= 1
Selected Shortest Path: [1, 2, 4, 6]
[(1, 1, 3), (2, 2, 1), (4, 3, 1), (6, 3, 1)]
src= 6  dst= 1  first_port= 1  final_port= 1
Selected Shortest Path: [6, 4, 2, 1]
[(6, 1, 3), (4, 1, 3), (2, 1, 2), (1, 3, 1)]
```

Рис. 3 – Результат эксперимента по алгоритму Дейкстры.

Алгоритм Йена определяет *k*-кратчайших путей минимальной стоимости во взвешенном графе с неотрицательными весами. Пример реализации алгоритма Йена в программно-конфигурируемых сетях рассмотрим на структуре, представленной на рисунке 4.

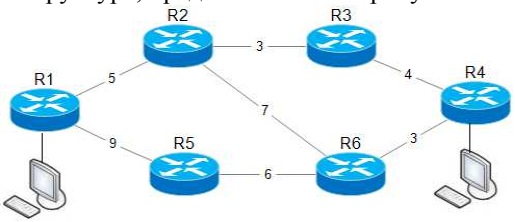


Рис. 4 – Модель сети для алгоритма Йена.

Так как алгоритм предполагает, что один из кратчайших путей уже найден каким-либо способом, то любой другой путь должен отличаться от уже найденного хотя бы одним ребром. Тогда, построим новый граф, который получим из исходного, выбросив одно из ребер найденного кратчайшего пути и уже в новом графе осуществим поиск кратчайшего маршрута.

Правило, по которому будет просчитываться маршрут для передачи пакета из первого хоста ко второму, аналогичен примеру, описанному выше. Как видно из эксперимента, контроллер вычислил три оптимальных маршрута (см. рис.5).

```
YenKSP is called
src= 1  dst= 4  first_port= 1  final_port= 2  max_k= 3
YenKSP->
[1, 5, 6, 4]
[1, 2, 6, 4]
[1, 2, 3, 4]
Chosen Path= [1, 2, 3, 4]
[(1, 1, 2), (2, 1, 2), (3, 1, 2), (4, 1, 2)]
Dijkstra's algorithm
src= 4  dst= 1  first_port= 2  final_port= 1
[(4, 2, 3), (6, 2, 1), (5, 2, 1), (1, 3, 1)]
```

Рис. 5 – Результат эксперимента по алгоритму Йена.

Алгоритм Беллмана-Форда – алгоритм поиска кратчайшего пути во взвешенном графе с отрицательными весами ребер. За время T алгоритм находит кратчайшие пути от одной вершины графа до всех остальных. В отличие от алгоритма Дейкстры, алгоритм Беллмана–Форда допускает ребра с отрицательным весом. При передаче пакета первый пакет будет передан маршрутизатору, который проверит существует ли для него правила передачи пакетов, если ничего не будет найдено, то маршрутизатор создаст для него маршрут по алгоритму Беллмана-Форда, по которому будут проходить все остальные пакеты.

```
[bell1               ] Switch 00-00-00-00-00-06 processed unicast ARP (0x0806
) packet, send to recipient by switch 00-00-00-00-00-01
[bell1               ] Switch 00-00-00-00-00-01 received PacketIn of type 0x0
800, received from 00-00-00-00-00-01.1
[bell1               ] Path from 12:ed:0d:00:dd:9b to 46:65:c6:b7:b8:cc over
path 00-00-00-00-00-01->00-00-00-00-00-02->00-00-00-00-00-04->00-00-00-00-00-06
[bell1               ] Switch 00-00-00-00-00-01 processed unicast 0x0800 type
 packet, send to recipient by switch 00-00-00-00-00-06
[bell1               ] Switch 00-00-00-00-00-06 received PacketIn of type 0x0
800, received from 00-00-00-00-00-06.1
[bell1               ] Path from 46:65:c6:b7:b8:cc to 12:ed:0d:00:dd:9b over
path 00-00-00-00-00-06->00-00-00-00-00-04->00-00-00-00-00-02->00-00-00-00-00-01
```

Рис. 6 – Результат эксперимента по алгоритму Беллмана-Форда.

Таким образом, были рассмотрены алгоритмы адаптивной маршрутизации, в результате эксперимента было продемонстрировано эффективность использования алгоритмов Дейкстры, Йена, Беллмана-Форда в программно-конфигурируемых сетях на основе протокола OpenFlow.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Black book ixia. Edition 10. SDN/OpenFlow. Jule 2010.
2. OpenFlow Tutorial. http://archive.openflow.org/wk/index.php/-OpenFlow_Tutorial.
3. Стивен Воган-Николс. OpenFlow: сеть нового поколения? http://www.osp.ru/os/2011/08/13011110/

А.А. ПЕТУХОВ, К.Н. ТУРБИН
Рязанский государственный радиотехнический университет

ТЕХНОЛОГИИ СОЗДАНИЯ ВИРТУАЛЬНОЙ РЕАЛЬНОСТИ

Технологии создания виртуальной реальности (ВР) позволяют создать достоверную имитацию действительности для человека. В этой статье речь пойдёт об истории развития этих технологий, областях применения ВР, а также актуальных системах ВР, представленных на рынке.

Виртуальная реальность (или искусственная реальность) — созданное техническими средствами виртуальное окружение (объекты и субъекты), передаваемое человеку через его ощущения и восприятие: зрение, слух, осязание, обоняние и другие. Очевидно, что в этом случае необходимо использовать различные по принципу функционирования инструменты воздействия на различные органы чувств человека. В общем случае для создания виртуальной реальности, в значительной степени, имитирующей действительность, необходимо использовать видимый свет для зрительного восприятия, звуковые волны для акустического восприятия, ускорение тела человека для воздействия на вестибулярный аппарат и непосредственное механическое воздействие для осязания. Способов как-либо имитировать запахи в реальном времени на данный момент не существует.

Важным аспектом виртуальной реальности является моделирование не только воздействия, но и реакции не него. Очевидно, что такое моделирование должно происходить в реальном времени с минимальными задержками. При этом, в отличие от близкой по сути технологии дополненной реальности, виртуальная реальность должна полностью создавать виртуальное окружение вокруг пользователя, не основываясь на окружающей пользователя действительности.

Задача реализация виртуальной реальности, как упоминалось выше, решается с использованием специальных технических средств, называемых *системами виртуальной реальности*. Такой системой яв-

ляется в принципе любое устройство, которое более полно (по сравнению с привычным компьютерным монитором и звуковыми колонками) позволяет отображать информацию, воздействуя в большей степени количественно (на большее количество различных органов чувств) либо качественно (с большей степенью достоверности) на органы чувств человека.

Первая система виртуальной реальности появилась в 1962 году, когда Мортон Хейлинг представил публике мультисенсорный симулятор, который получил название «*Сенсорама*». Погружение в виртуальную реальность осуществлялось посредством демонстрации коротких фильмов, искусственных запахов, ветра (для его создания использовался фен) и записанного шума мегаполиса. Уже в 1967 году был сконструировал первый шлем, изображение на котором генерировалось при помощи компьютера.

Технология постепенно развивалась и совершенствовалась вплоть до начала второй декады XXI века, когда произошёл бум технологий создания виртуальной реальности. Его причиной в первую очередь являлось развитие игровой индустрии, которая, являясь экономическим «двигателем» развития графических процессоров, вывела их на такой уровень, когда стало возможным в реальном времени создавать фотореалистические изображения достаточного качества для использования его в виртуальной реальности.

Помимо использования в игровой сфере, новые технологии виртуальной реальности могут оказаться полезными и в других областях. В первую очередь это киноиндустрия. В настоящее время активно разрабатываются новые форматы фильмов, пригодных для просмотра в режиме виртуальной реальности.

Более полезным с практической точки зрения является применение технологий виртуальной реальности в области медицины. Уже сейчас существуют технологии, позволяющие роботу, установленному в операционной, повторять все движения хирурга и создавать для него эффект присутствия. Врач способен при этом наблюдать не только конкретный оперируемый участок, но и всю картину в целом.

Технологии виртуальной реальности также находят своё применение и в авиации. Например, для обучения пилотов, вместо дорогостоящих симуляторов, которые помимо всего прочего ещё и подходят строго для обучения управлением одной моделью самолёта, можно использовать сравнительно недорогую систему виртуальной реальности.

Виртуальную реальность также можно задействовать в тех сферах деятельности, где необходима оценка человека, однако само его

присутствие небезопасно. Речь идёт о выполнении опасных работ, вроде ликвидации пожаров или разборе завалов. Окружение робота, который посылается для выполнения такой работы, естественным образом может восприниматься его оператором, который в этот момент может находится в безопасном месте.

В настоящее время на рынке активно появляются новые технологии, представляющие из себя различные вариации систем виртуальной реальности.

Шлемы виртуальной реальности

Google Cardboard

Эксперимент компании *Google* в области технологий виртуальной реальности, представляет собой шлем из картона, пары двояковыпуклых линз из стекла или пластика диаметром 34 мм, и фокусным расстоянием 45 мм и смартфона на базе операционной системы *Android*, *iOS* или *Windows Phone* со специальным программным обеспечением. На основе получаемых данных со встроенных в смартфон магнитометра, акселерометра и камеры ПО разделяет изображение на экране смартфона на стереопару, тем самым достигается эффект виртуальной реальности. Важной особенностью *Cardboard* является его доступность — конструкцию можно собрать самостоятельно, купив нужную пару линз и вырезав из картона заготовку, используя шаблоны, свободно распространяемые *Google*.

Samsung Gear VR

Устройство симуляции виртуальной реальности, разработанное компанией *Samsung*. Так же как и *Google Cardboard* использует в качестве устройства обработки и вывода графичекой и звуковой информации смартфон, закрепленный в держателе шлема. Но в отличие от шлема компании *Google*, *Gear VR* снабжается линзами, сенсорной панелью управления, регулятором фокуса и громкости, акселерометром, магнитометром и гироскопом. *Samsung Gear VR* совместим со смартфоном *Samsung Galaxy Note*, в видеорежиме 2560 x 1440 с частотой обновления экрана 60 Гц.

Oculus Rift

Шлем виртуальной реальности, требующий для работы соединения с компьютером. Первая система виртуальной реальности, предназначенная для широких масс. По сарвнению с *Google Cardboard* и *Samsung Gear VR*, имеет более высокое качество изображения, меньший отклик и более широкое поле зрения. и более высокую цену. В потребительской версии на каждый глаз выводится изображение с разрешением более 2160×1200 пикселей и частотой 90 Гц. Положение

шлема в пространстве оценивается не только по данным встроенных сенсоров, но и при помощи внешней инфракрасной камеры.

Прочие технологии

Virtuix Omni

Всенаправленная беговая дорожка. Первая массово-доступная система виртуальной реальности, реализующая возможность естественного для человека передвижения в виртуальном пространстве. Система представляет из себя комплект из специальной вогнутой пластиковой площадки и особой обуви с гладкой подошвой. Сам человек удерживается специальной конструкцией. Таким образом, при ходьбе человек будет фактически оставаться на месте, но будет иметь возможность поворачиваться и «шагать» в любую сторону, при этом будучи надёжно зафиксированным вертикально.

MotionParallax3D

Класс устройств виртуальной реальности, позволяющих сформировать у пользователя иллюзию объемного предмета за счет отображения на экране специальной проекции виртуального объекта, сгенерированной в зависимости от положения пользователя относительно экрана. В настоящее время на рынке присутствует множество вариантов реализации данной технологии. Для её реализации достаточно использовать обычный монитор с особыми метками на его корпусе и специальный трекер — устройство, которое крепится на голове пользователя, и передаёт на компьютер информацию о направлении взгляда пользователя относительно поверхности монитора. В итоге, создаётся ощущение, что монитор является «окном» в виртуальную реальность, расположенную с другой стороны экрана. К преимуществам данной технологии можно отнести относительную дешивизну. Главным недостатокм является тот факт, что эффект виртуальной реальности ограничен рамками экрана и виден только одному человеку.

Заключение

Второе десятилетие XXI века тесно связано с бумом развития технологий виртуальной реальности. На этот рынок выходят крупнейшие корпорации вроде Google, Facebook или Samsung, а также значительное множество независимых разработчиков аппаратных и программных средств. Технологии виртуальной реальности имеют огромный потенциал в самых различных сферах жизни человека и являются одной из самых перспективных отраслей информационных технологий в целом.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Myron W. Krueger, Artificial Reality II / Myron W. Krueger. – Addison-Wesley Professional,1991. – 304 p.

2. Google Cardboard - Technical Specification version 2.0 – 2015

3. Сергей Грицачук. До и после Oculus Rift [Электронный ресурс] / 2013. – Режим доступа: http://www.it-world.ru/tech4human/newtech/50400

4. Сергей Уваров. Обзор шлема виртуальной реальности Samsung Gear VR [Электронный ресурс] / 2015. – Режим доступа: http://www.ixbt.com/mobile/samsung-gear-vr.shtml

5. Scott Stein. Samsung Gear VR review [Электронный ресурс] / 2014. – Режим доступа: http://www.cnet.com/products/samsung-gear-vr/

6. About Virtuix Omni [Электронный ресурс] / Режим доступа: http://www.virtuix.com/about/

7. Технология MotionParallax3D [Электронный ресурс] / Режим доступа: http://nttl.ru/ru/technology/

Н.А. ПОДГОРНОВА

Рязанский государственный радиотехнический университет

ПРИМЕНЕНИЕ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ ДЛЯ ОБРАБОТКИ И ПРОГНОЗИРОВАНИЯ ДЕНЕЖНЫХ ПОТОКОВ ПРЕДПРИЯТИЯ

Рассматривается практическое применение метода прогнозирования денежных потоков на основе нейронных сетей. Для сравнительного анализа и построения прогнозной оценки с помощью нейронной сети используются в качестве обучающих выборок 3 вида временных рядов ежедневных изменений денежного притока и оттока предприятия, предварительно очищенные от высокочастотного шума с помощью вейвлет-преобразования с использованием различных программных систем.

Многие реальные экономические процессы не могут адекватно быть описаны с помощью традиционных статистических моделей, т. к. являются существенно нелинейными. Жесткие статистические предположения о свойствах временных рядов ограничивают возможности методов математической статистики. В данной ситуации адекватным аппаратом для решения задач диагностики и прогнозирования могут служить специальные искусственные сети [1], реализующие идеи предсказания при наличии обучающих последовательностей, отличающиеся высокой скоростью обучения и универсальными аппроксимирующими возможностями [4].

Наиболее адекватным и эффективным инструментарием прогнозирования денежных потоков, обладающие стохастическим движением и нелинейными зависимостями, являются нейронные сети, обладающие свойством самообучения, позволяющие выявить спрятанные в потоке информации закономерности, способные автоматически на-

строить связь между входными и исходными данными с определенной степенью точности. В ходе исследования НМ для эффективного решения проблемы аппроксимации выбран многослойный персептрон и в качестве алгоритма обучения сетей и построения прогноза метод обратного распространения ошибки, как мощное средство для обучения многослойных нейронных сетей прямого распространения, поиска закономерностей, прогнозирования, качественного анализа.

В настоящее время разработано большое количество программных пакетов, обеспечивающих возможности решения задач с помощью нейросетевых технологий. Но основное применение в настоящее время получили программные реализации различных нейросетевых парадигм – нейропакеты (нейроэмуляторы), к классу которых относится пакет *Statistica Neural Networks* (*Statsoft*), являющийся богатой и мощной средой анализа нейросетевых моделей и более совершенным нсйросетевым продуктом по сравнению с раннее выпущенными. Система *STATISTICA Neural Networks* соответствует самым современным технологиям и показывает наилучшие рабочие характеристики среди нейросетевых пакетов, представленных сейчас на рынке программного обеспечения.

Проектирование нейронных сетей начинается с определения требований к сети, включая требования к предварительной обработке данных и требования по интеграции. При работе с обучающей выборкой используемые фактические данные в привычном для пользователя виде и формате, чаще всего непригодны для использования нейросетью и требуют предварительной обработки данных.

Для исследования применяется данные предприятия г. Рязани ОАО «В» – электросбытовое предприятие. Количество наблюдений, составляющий временной ряд предприятия «В», $N = 175$.

При постановке задачи трудно сразу определить правильный способ предобработки исходной информацией. Поэтому для обоснования способа необходимо провести серию экспериментов. В каждом из экспериментов используется одна и та же обучающая выборка и разные способы предобработки входных данных сети. В частности, для построения прогнозной оценки с помощью нейронной сети использовались в качестве обучающих выборок 3 вида временных рядов ежедневных изменений денежного притока (*PR*) и оттока предприятия (*OT*) [3]:

1 – временные ряды денежного притока (*PR1*) и денежного оттока (*OT1*) предприятия без предварительной обработки;

2 – временные ряды денежного притока (*PR2*) и денежного оттока (*OT2*) предприятия, очищенные от высокочастотного шума с помо-

щью вейвлет-преобразования с использованием короткого вейвлета 3-го порядка, 2 уровня разложения для сохранения полезной информации, распределения которых не подчиняются нормальному закону. Для подавления шумов применялась система *Matlab7* с мягким режимом обработки сигнала для сохранения исходных данных. После серий экспериментов были выбраны параметры, влияющие на качество шумоподавления сигнала. Для шумоподавления оценивался спектральный состав шумовой компоненты, на основе которого выбирался тип пороговой обработки детализирующих коэффициентов и критерий расчета самого порога;

3 – временные ряды денежного притока (*PR3*) и денежного оттока (*OT3*) предприятия, сглаженные (очищенные от шума и случайных выбросов) при помощи короткого вейвлета 3-го порядка, 2 уровня разложения, распределения которых являются гауссовскими. Для подавления шумов фактических данных применялась программа *Deductor ver.5.3*. При удалении шума и сглаживания данных в программе автоматически установлены параметры выбора порогового уровня фона (оценки дисперсии шума) и критерий его расчета.

С учетом предпрогнозных характеристик этих временных рядов, полученных с помощью фрактального и вейвлет-анализа принимаются в качестве параметров временных окон: $n=5$ – количество значений по которым строится прогноз (учитывая доминирующую размерность квазициклов); $m=5$ – горизонт прогноза (ежедневный прогноз выполняется непрерывно на пять дней в соответствии с принципами, заложенными в разработанную нами финансовую модель); $s=1$ – шаг сдвига окон.

В теории НМ известно множество способов обучения модели [1, 2] и обоснования взвешивающих коэффициентов. Их анализ не является целью настоящей работы, поскольку в работе решается не задача развития НМ, а задача их использования при обосновании управленческих решений по управлению ОДС.

В ходе исследования были рассмотрены различные модификации архитектуры многослойных персептронов НМ, основанные на применении спрягающих градиентов, разные эвристические алгоритмы, такие как: быстрое распространение (quick propagation), метод Левенберга–Маркара и др., типологии сети (числа слоев, количество элементов в скрытых слоях) и варианты настройки обучения (скорость обучения, инерция, количество эпох, значение моментов). В качестве алгоритма обучения сетей применялся метод обратного распространения ошибки (back propagation). Для улучшения качества экстраполяции логистическая функция в выходном слое была заменена на линейную,

которая не меняет уровня активации и при этом в отличие от логистической не насыщается, поэтому способна лучше прогнозировать.

Использованная в работе модель основана на использовании алгоритма, применяемого в пакете прикладных программ *Statistica Neural Networks*. В основе алгоритма обучения сетей применялся метод обратного распространения ошибки. В ходе исследований найдены типы нейронных структур для 3 видов данных исследуемых временных рядов денежных потоков. Для временного ряда денежного притока (*PR1*) наилучшим оказался следующий вариант: трехслойная MPL-сеть с пятью входными ячейками, одним скрытым слоем, состоящий из 15-ти нейронов, что привело к архитектуре сети *5-15-1* и для временного ряда денежного оттока (*OT1*) архитектура трехслойной MPL-сети - *5-9-1*. Для временных рядов денежного притока (*PR2*) и оттока (*OT2*) архитектуры четырехслойных MPL-сетей - *5-25-12-1*. и *5-21-11-1*. Для временных рядов денежного притока (*PR3*) и оттока (*OT3*) архитектуры трехслойной MPL-сетей - *4-9-1* и *5-10-1*.

Анализируя полученные результаты можно сделать вывод, что наилучшие прогнозные оценки можно получить после прогнозирования обработанных исходных данных временных рядов притока (*PR3*) и оттока (*OT3*) с применением вейвлет-преобразования в программе *Deductor 5.3*. Коэффициенты корреляции *Pearson-R = 0,9997* для временного ряда притока денежных средств (*ПР3*) и *Pearson-R = 0,9998* для временного ряда оттока денежных средств (*OT3*) очень близки к значению *1,0*, что означает получение более точной прогнозной оценки. Величины *S.D.Ration =0,032* по притоку ДС и *S.D.Ration =0,0195* по оттоку ДС значительно меньше 0,1, что соответствует прекрасному качеству регрессии и являются отличным результатом обучения нейронной сети.

Результаты исследований показали, что нейронные сети способны находить скрытые динамические закономерности в данных, на которых они обучаются, и (на этой основе) прогнозировать динамику, статистически оценивая результаты прогноза.

По сути, единственным требованием к сети было достаточное число нейронов в скрытом слое, поскольку их число должно возрастать пропорционально сложности анализируемых данных. Для рассматриваемой задачи сложность данных оказалась такой, что удалось ограничиться 25 нейронами в скрытом слое.

Наиболее трудным в использовании нейросетей является выбор момента остановки обучения. Если сеть обучать недолго, то она не выучит выборку обучающих примеров. Если сеть обучать слишком долго, то она выучит примеры с шумами со сколь угодно высокой точ-

ностью, но окажется неспособной обобщать примеры (т. е. будет действовать схожим образом на данных, не входивших в обучающее множество). Для решения задачи выбора момента остановки обучения использована процедуру калибровки с тем, чтобы оптимизировать сеть, применяя ее к независимому тестовому множеству примеров в процессе обучения. Калибровка позволяет найти оптимум нейросети на тестовом множестве, означая способность сети к обобщению, т. е. получению хороших результатов на новых данных. Это достигается вычислением среднеквадратичного отклонения реальных и предсказанных выходов, а также ошибки аппроксимации. Эффективность предсказаний нейросети проверялась традиционным способом: сравнением фактического значения и предсказанного нейросетью.

На рисунке 1 представлены графики временных рядов притоков (*PR3*) денежных средств предприятия (предварительно очищенные от выбросов) и их проекция как результат нейросетевого прогноза. Визуализации этих графиков показывают, что прогнозируемые временные ряды, построенные с помощью нейронных сетей, имеют очень близкие значения к исходному ряду (практически полностью совпадают) и имитируют его незначительные изменения на протяжении временной оси. Это подтверждает и малая ошибка аппроксимации *MAPE=0,628%* по притоку денежных средств (*PR3*), *MAPE=0,547%* по оттоку денежных средств (*OT3*).

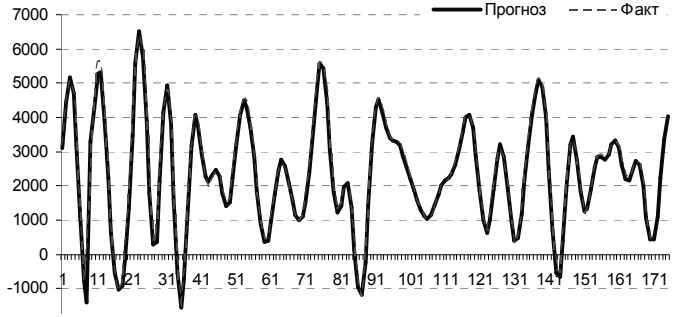


Рис. 1 – График временного ряда ежедневного денежного притока (PR3) и его проекция как результат нейросетевого прогноза, тыс. руб.

Результаты построения прогноза на последующие 5 дней в соответствии с разработанной финансовой моделью для этих временных рядов представлены в таблице 1.

Выполненные расчёты позволяют сделать выводы, что прогнозируемые ряды денежных потоков (*PR3, OT3*) с высокой точностью

аппроксимируют исходную последовательность и как следствие этого дают надежную оценку прогноза.

Большие трудности при обучении нейросетей создала сильная зашумленность данных. Такие помехи скрыли характерные особенности данных или выдали себя за них, что сильно ухудшили результаты обучения.

Таблица 1. Ретроспективные (5 дней) и прогнозные (5 дней) значения денежного притока (ОТЗ) (предварительно очищенные от шума с помощью вейвлет-преобразовния с использованием системы Deductor) предприятия, тыс. руб.

		Дни	*Приток ДС, тыс. руб*	*PR3*		*Отток притока ДС, тыс. руб.*	*OT3*	
				Прогноз притока ДС, тыс.руб.	*Относит. ошибка, ε*		*Прогноз оттока ДС, тыс. руб.*	*Относит. ошибка, E*
1	Ретроспектива	170	2736,652	2734,161	0,00091	2404,208	2420,873	0,006932
2		171	2638,016	2605,76	0,012227	3187,769	3204,372	0,005208
3		172	2008,326	2003,219	0,002543	3566,475	3560,538	0,001665
4		173	1102,179	1089,449	0,01155	3352,475	3311,5	0,012222
5		174	441,4118	439,572	*0,004168*	2773,836	2777,6	0,001357
				MAPE	***0,00628***		*MAPE*	***0,005477***
1	Прогноз	175		429,5769			2267,465	
2		176		1128,548			2182,225	
3		177		2254,709			2570,199	
4		178		3354,841			3175,666	
5		179		4041,594			3609,653	

Прогнозирование денежных потоков может осуществляться различными методами. Выбор наиболее эффективных методов предполагает совокупность оценочных расчетов. Можно предположить отсутствие универсально лучшего метода.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Галушкин, А.И. Теория нейронных сетей. Кн. 1: Учеб. Пособие для вузов / А.И. Галушкин. – М.: ИПРЖР, 2001. – 385 с.

2. Оссовский, С. Нейронные сети для обработки информации / С. Оссовский. – М.: Финансы и статистика, 2002. – 344 с.

3. Терехин, В.И. Подгорнова, Н.А. Моделирование и управление остатками денежных средств предприятия, Монография / В.И. Терехин, Н.А. Подгорнова. — М.: Креативная экономика, 2012. 163 с.

4. Тихонов, Э.Е. Методы прогнозирования в условиях рынка: Учебное пособие / Тихонов Э.Е. – Невинномысск, 2006. – 221 с.

В.Ю. ПОТАПОВА
Рязанский государственный радиотехнический университет

ПРИМЕНЕНИЕ ВЕЙВЛЕТ-ПРЕОБРАЗОВАНИЯ ХААРА

Рассматривается история развития вейвлетов, в частности вейвлет Хаара и его практическое применение.

Применение вейвлет-преобразования начало активно развиваться в начале двадцатого века. Их внедрение, связано с несколькими отдельными рассуждениями, начавшимися с работ Альфреда Хаара, который в 1910 году опубликовал полную ортонормальную систему базисных функций с локальной областью определения [1]. Эти функции получили название вейвлеты Хаара. Они локальны по времени, обладают компактным носителем, но не имеют достаточной гладкости, присущей другим вейвлетам. Несмотря на это, возможность их быстрого использования позволяет применять их в распознавании образов, речи, обработке кардиодиосигнала, сжатия изображений и видео, изучения свойств торбулентных полей и др [2].

Один из способов применение вейвлет-преобразованя при распознавании является метод запроса по содержанию. Он позволяет «отыскать» заданное изображение из базы данных [3]. Вейвлет Хаара разделяет исходный образ на подобласти, которые отражают геометрические свойства исследуемого объекта [4]. В результате преобразования получают информацию обо всем изображении, хранящуюся в шести переменных: три из которых – это среднее значение пикселей для каждого из цветового канала (обычно применяется телевизионный формат YIQ) и три массива вейвлет-коэффициентов, также для каждого из каналов. Далее значения вейвлет-коэффициентов сортируют по модулю в порядке убывания, и производится усечение «не значимых». Сформированный ряд называются массивами поиска [5]. Сопоставление исходного изображения с изображениями из базы данных осуществляется на основе среднего значения для каждого из цветового канала, массивов поиска и первоначального расположения пикселей. Высчитывается количественный показатель сходства с применением коэффициентов в кратномасштабной метрике формирования запроса [6]. Он может быть как положительным, так и отрицательным. Наиболее похожим, считается изображение с наименьшим показателем сходства [7, 8].

Применение вейвлет-преобразования в медицине связно с нестационарностью сигнала и его сложными частотно-временными характеристиками. Они способны анализировать не только его частотный

спектр, но и в какой момент времени появилась та или иная гармоника [9]. Данное свойство позволяет осуществить моделирование электрокардиосигнала (ЭКС). Входными параметрами являются массивы времени и частоты, к которым впоследствии применяются растяжение по горизонтали и сжатие вертикали, сдвиг и усреднение в окрестности точки масштаба. Результатом данных преобразований является некий дискретный однородный входной сигнал, каждой паре чисел которого ставиться в соответствие полусумма и полуразности [3, 8]. Данный процесс называется прореживанием входного сигнала. Далее осуществляется сглаживание и по получившимся точкам откладывается электрокардиограмма (ЭКГ).

Вейвлет Хаара локален во времени, что делает его перспективными для анализа ЭКС [7,9]. Суть метода заключается в поиске резких скачков ЭКС, их анализе и удалении шумов. В обработке ЭКС используется анализирующие функции, которые раскладывают сигнал на некоторую последовательность коэффициентов, соответствующих конкретным компонентам ЭКС [10]. По их форме можно судить о физическом состоянии человека.

Вышеизложенные примеры вейвлет-преобразования являются малой частью применения вейвлета Хаара, которые используются в разных областях.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Воробьев В.И., Гри В.Г. Теория и практика вейвлет-преобразования Спб 1999

2. Столниц Э., ДеРоузб Т., Салезин Д. Вейвлеты в компьютерной графике: теория и приложения. – И: Регулярная и хаотическая динамика. : М-И 2002. – стр 59-75.

3. Гринченко Н.Н., Потапова В.Ю. Алгоритм поиска изображений в базе данных. Проблемы передачи и обработки информации в сетях и системах телекоммуникации материалы конф. Рязань. Горячая линия Телком 2015 С. 161-164.

4. Костров, Б.В., Гринченко Н.Н., Геращенко Е.С., Потапова В.Ю. Выборка изображений из базы данных методом запроса по содержанию. Сборник Интеллектуальные и информационные системы (Интеллект 2015) : материалы всерос. науч.-тех конф. Тула ТулГу 2015 С. 13-18.

5. Теория вейвлетов и ее применение к поиску изображений [электронный ресурс] http://autayeu.com/projects/oko/autayeu-master-thesis-rus.pdf

6. Charles E. Jacobs Adam Finkelstein David H. Salesin Fast Multiresolution Image Querying. : Department of Computer Science and

Engineering University of Washington Seattle. [электронный ресурс] http://grail.cs.washington.edu/ projects/query/mrquery.pdf

7. Потапова В.Ю. Разработка программных средств для поиска изображений по методу подобия. Новые информационные технологии в научных исследованиях: материалы конф. Рязань 2015 С. 207-209

8. Гринченко Н.Н., Тарасов А.С. Разработка алгоритма распознавания автомобильных номеров. Проблемы передачи и обработки информации в сетях и системах телекоммуни-кации материалы конф. Рязань. Горячая линия Телком 2015 С. 216 – 217.

9. Забейворота О.Ю., Саитгалина А.Д. , Губайдуллин И.М. [электронный ресурс] http://agora.guru.ru/abrau2012- /pdf/111.pdf

10. Гринченко Н.Н., Геращенко Е.С., Потапова В.Ю. Использование вейвлет-преобразования для анализа формы электрокардиосигнала. Сборник Метематика: фундаментальные исследования и вопросы образования. материалы межд.. науч.-тех конф. РГУ им. С.А.Есенина.

А.А. РЯБОВ
Национальный исследовательский университет «МИЭТ»

ИСПОЛЬЗОВАНИЕ ДЕРЕВА РЕШЕНИЙ ДЛЯ АНАЛИЗА ДАННЫХ

Рассматриваются принцип работы и построение дерева решений.

Быстрое развитие информационных технологий привело к тому, что организации могут собирать большое количество информации, которое необходимо анализировать. В связи с этим вырастает потребность в методах автоматического анализа данных. Одним из таких методов является деревья принятия решений, которые широко используются на данный момент.

Дерево решений – схематическое представление процесса принятия решения по определённой проблеме, изображаемое в виде древовидной структуре. Структура дерева представляет собой «листья» и «ветки». На «ветках» дерева записаны атрибуты, от которых зависит целевая функция, в «листьях» записаны сами значения целевой функции, а в остальных узлах записаны атрибуты, по которым можно различить случаи.

Задачи, решаемые этим методом объединены в три класса:

- Описание данных: деревья решений позволяют хранить информацию в компактной форме.

- Классификация: решаются задачи классификации, определению к какому классу отнести объект.
- Регрессия: решаются задачи прогнозирования, с помощью установления зависимости целевой и входной переменных.

Пусть дано множество объектов, которые характеризуются некоторыми атрибутами, один из них указывает на принадлежность объекта к определённому классу. Возможны три ситуации:

- Множество содержит объекты, принадлежащие одному классу, следовательно, дерево решения для этого множества – это лист, определяющий этот класс.
- Множество пустое, тогда это лист, и класс выбирается родительского множества.
- Множество содержит объекты, принадлежащие разным классам. Разбиваем множество на некоторые подмножества. Для этого выбирается один из атрибутов, имеющий два или более отличных друг от друга значений. Теперь каждое подмножество содержит все объекты, имеющие определённое значение для выбранного атрибута. Рекурсивно продолжаем этот процесс до тех пор, пока конечное множество не будет состоять из объектов, относящихся к одному классу.

Построение дерева будет происходить сверху вниз, при использовании данной процедуры.

Приведём простой и наглядный пример дерева решений:

Рис. 1 – Пример дерева решений.

Основной вопрос в том, как выбирать очередной атрибут. На сегодняшний день имеется множество способов выбора атрибута:

- Алгоритм ID3, где выбор атрибута происходит на основании прироста информации, либо на основании индекса Гини.

- Алгоритм C4.5 (улучшенная версия ID3), где выбор атрибута происходит на основании нормализованного прироста информации.
- Алгоритм CART.
- MARS и другие.

Деревья решений имеют ряд преимуществ:

- Просты в понимании и очень наглядны.
- Высокая точность прогноза, сопоставимая с другими методами (статистика, нейронные сети).
- Работа с большим количеством информации, не требуя специального оборудования.
- Построение непараметрических моделей.
- Не требует подготовки данных.

Деревья решений являются отличным инструментом в системах поддержки принятия решений, интеллектуального анализа данных. Они помогают аналитикам находить ошибки, в сложных ситуациях. Применяются в следующих областях:

- Промышленность (поиск дефектов, проверки готовых продуктов и т.д.).
- Банковская система (оценка кредитоспособности клиента).
- Медицина и т.д.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Материалы взяты с сайта datareview.info/ http://datareview.info/article/derevya-resheniy-i-algoritmyi-ih-postroeniya/
2. Материалы взяты с сайта basegroup.ru https://basegroup.ru/community/articles/description
3. С.А. Айвазян, В.С. Мхитарян Прикладная статистика и основы эконометрики, 1998.

А.О. САПРЫКИНА, А.Н. САПРЫКИН
Рязанский государственный университет имени С. А. Есенина
Рязанский государственный радиотехнический университет

ЭЛЕКТРОННОЕ ПОРТФОЛИО КАК ИНСТРУМЕНТ ОЦЕНИВАНИЯ КАЧЕСТВА ОБРАЗОВАНИЯ

Данная статья рассматривает понятие качества образования, проблемы, связанные с его оцениванием и возможности электронного портфолио по их решению.

Системы электронного портфолио, активно развивающиеся в наши дни, являются многофункциональными образовательными инструментами. Они доказали свою эффективность как средства хранения, демонстрации и обсуждения работ обучаемых, но теперь очевидным становится их потенциал и в качестве средства оценивания этих работ. Правильная организация процесса оценивания, вне зависимости от выбранных средств оценивания, в первую очередь требует однозначного толкования характеристик оцениваемых работ, что неизбежно приводит к проблеме определения понятия качества образования.

Качество образования является многоаспектным явлением и поэтому проблема его определения всегда стоит особенно жестко. Выделение элементов качества образования всегда приводит к появлению у каждого из них своих определений и, соответственно, критериев оценивания. Например, согласно ЮНИСЕФ [1], структура качества образования состоит из следующих элементов (рис. 1):

– обучаемые и педагоги;
– образовательная среда;
– содержание образования;
– процесс обучения;
– результаты обучения.

Рассмотрим подробнее перечисленные элементы качества образования.

Первый элемент включает в себя самих *обучаемых* и *педагогов* не только в обычной роли преподавателя, но и в роли таких же обучаемых. Данная концепция учитывает необходимость педагога в самосовершенствовании и самообразовании, а также повышении квалификации, подтверждению компетентности и прочее. В таких ситуациях педагог и сам является обучаемым, и, следовательно, должен быть оценен по схожим критериям. Для обеспечения качества образования важно обеспечить физическое здоровье обучаемых, посещаемость занятий, поддержку семьи и общества. Данные требования тесно связаны со вторым элементом – *образовательной средой*. В это понятие входит обеспечение обучаемых физической возможностью посещать учебное заведение и техническая обеспеченность данного учебного заведения, то есть обеспечение пригодной среды для получения знаний. Требования, предъявляемые к данному элементу, отнюдь не заканчиваются только технической стороной – образовательная среда должна удовлетворять и психологические потребности обучаемых, способствуя процессу обучения, а также быть тесно связанной с другими общественными структурами вне системы образования.

Содержание образования является одной из наиболее часто рассматриваемых и оцениваемых в значении общего качества образования характеристикой. В него входят знания, умения и навыки (ЗУН), которые должны приобрести обучаемые после завершения программы обучения, причем они должны быть не только теоретическими, но и практическими, применимыми на практике навыками. Следующие из этого элемента системы качества образования *результаты обучения* могут быть представлены как раз сформированностью этих ЗУНов, но не ограничиваться ими. *Образовательные достижения* (конкретные работы обучаемых: эссе, презентации, иллюстрации, тесты, грамоты и т.д.) представляют собой подтверждение полученных навыков и более наглядно демонстрируют их заинтересованным сторонам (например, работодателям).

Под *процессом обучения* с точки зрения качества образования подразумевается единство методов и средств обучения и их ориентированность на поставленные цели каждого конкретного курса.

Рис. 1 – Структура качества образования по ЮНИСЕФ.

Электронное портфолио в качестве универсального образовательного инструмента обладает рядом сильных сторон, в том числе и возможностью использования его как инструмент оценивания.

Применимо к структуре качества образования, предложенной ЮНИСЕФ, электронное портфолио может помочь в оценивании или полностью провести его для всех элементов. Состояние обучаемых и образовательной среды может быть оценено с помощью анкетирования или опроса, результаты которых размещены в портфолио или включены в презентацию. Процесс обучения может быть оценен с по-

мощью механизма обратной связи, предлагаемого системой электронного портфолио (например, проводить занятия по разным методикам в течении одной недели, а затем организовать обсуждение с целью выяснить, в какой день информация субъективно усваивалась лучше) или сбора статистических данных (использовать основную и контрольную группы обучаемых, организуя для них учебный процесс по-разному), что также применимо и для оценивания содержания образования и результатов обучения. Помимо этого, образовательные достижения могут и должны быть оценены неотрывно от процесса их обсуждения как самими обучаемыми (не обязательно из одной группы), так и другими педагогами.

Таким образом, очевидными становятся возможности систем электронного портфолио для оценивания качества образования в целом, с последующим сохранением оценки и гибкими инструментами обработки данных, обратной связи и рефлексии. Однако для использования полного потенциала данного образовательного инструмента требуется тщательно разработанная методика его использования как инструмента оценивания, ориентированная в соответствии с предъявляемыми требованиями и поставленными задачами.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Defining Quality in Education. UNICEF, Italy, 2000. – 44 p. [Электронный ресурс]. URL: http://www.unicef.org/education/files-/QualityEducation.PDF (дата обращения 01.02.2016)

А.Л. САФОНОВ
Рязанский государственный радиотехнический университет

ОБЪЕКТНО-ОРИЕНТИРОВАННЫЙ ПОДХОД ПРИ РАБОТЕ СО СТРУКТУРНЫМИ ЯЗЫКАМИ

Рассматривается принципиальная возможность реализации элементов объектно-ориентированной парадигмы для языков, которые не имеют родной поддержки. Исследуются принципы работы объектно-ориентированных механизмов с точки зрения нижележащих уровней.

Как известно, прежде чем ООП стало повсеместно применяться, большинство программистов «исповедовали» структурную парадигму. Хотя объектные и структурные языки программирования появлялись и в ранние времена, тогда они не смогли завоевать широкую нишу, так как были заметно медленнее более низкоуровневых языков. Однако в

последние 20-25 лет появилось множество объектно-ориентированных языков, выросших на почве мощных структурных языков. Не считая C++, который явно вырос из Си, существуют также такие языки, как, например, Python и Rust, комбинирующие в себе гибкость объектного подхода и простоту структурных языков.

В данной статье рассматривается организация объектного подхода на основе языка Си, известного как самый низкоуровневый структурный язык. Известно, что язык C++ начинался как обертка над Си, поэтому этот язык особенно интересен в сравнении и изучении.

Некоторые исследователи различают объектные и объектно-ориентированные языки. Пр этом до сих пор не существует точного понятия, по которому можно определить, является ли язык объектным или нет. Можно сказать, что объектный язык поддерживает синтаксис объявления классов, вызов методов (либо другой способ передачи сообщений) и другие особенности ООП, в то время как объектно-ориентированный язык предполагает, что именно ООП и будет использовано. Другими словами — объектный язык *разрешает* использование ООП, в то время как объектно-ориентированный *принуждает* к ООП.

Язык программирования — это прослойка между человеческим языком и машинным. Все, что человек описывает с помощью языков программирования, в конечном итоге транслируется в машинный код. При этом допускается промежуточная трансляция одного языка в другой. Например, сейчас любой код транслируется сначала на язык ассемблера, а только затем — на машинный код.

Каждый язык оперирует некоторой моделью, которую он в состоянии транслировать на язык нижележащего уровня. Все абстракции языка имеют детерминированное определение. Это позволяет использовать в качестве формата выходных данных не машинный код или ассемблер, а другие языки, например Си, обработкой которого будут заниматься другие трансляторы. Так работал язык C++ до 1993 года.

История языка C++

Разработка C++ началась в 1979 году с проекта Бьёрна Страуструпа под названием «C with Classes (Си с классами)». Изначально он представлял собой препроцессор над языком Си, который поддерживал классы, наследование, статическую типизацию и аргументы по умолчанию. Препроцессор носил название Cpre.

К 1983 году Страуструп улучшил свой транслятор, добавив в него виртуальные функции, перегрузку методов и операторов, ссылки, типобезопасное управление памятью через new и delete, улучшенную

типизацию. Тогда же язык был переименован в C++ (инкремент Си), а транслятор переименовали в Cfront.

В 1985 году была выпущена книга «*The C++ Programming Language*», являвшаяся на тот момент единственной полноценной документацией по языку (так как стандарт на язык тогда еще не был принят). Тогда же выпустилась первая коммерческая реализация языка.

До 1993 года Cfront определял рамки языка, и многое из известных не самых лучших моментов языка связано с особенностями трансляции в Си. К тому времени удалось реализовать множественное наследование, абстрактные классы, статические методы классов, константные методы, защищенные члены классов, шаблоны и наработки по обработке исключений. Неудачная попытка реализации исключений в трансляторе Cfront привела к тому, что с того момента язык C++ начал компилироваться в обход Си.

Однако, учитывая, какой большой объем синтаксиса языка был реализован в трансляторе, можно сказать, что на Си можно писать в объектно-ориентированном стиле, правда, с некоторыми особенностями.

Особенности реализации ООП на Си.

1. **Инкапсуляция полей**

Поля в классах могут быть приватным, публичными и защищенными. Эти модификаторы указывают препроцессору, кому разрешить доступ к этим полям, а кому — нет. При трансляции в Си эти модификаторы приводят к тому, что описание класса присутствует одновременно в трех местах:

1) реализация класса, содержит полное описание всего класса, так методы класса имеют доступ ко всем полям этого класса;

2) заголовочный файл для потомков, содержит защищенные и публичные секции класса, используется при описании классов-потомков;

3) публичный заголовочный файл, содержит только публичные поля и методы.

Разграничение доступа основано на нескольких идеях.

1) В Си аналогом класса является структура. В общем виде структура — это простой класс с публичными полями, ни от кого не наследующийся.

2) Данные в структурах Си располагаются в памяти точно так же, как и при описании, если не используются модификаторы упаковок структур. Эта особенность языка позволяет безопасно линковать модули, использующие общие структуры.

3) При размещении данных компилятор учитывает их размер и старается расположить поля в структуре так, чтобы доступ к любому полю был выровненным.

4) В стандарте C11 появилась поддержка доступа к полям анонимных структур напрямую по их имени (на самом деле эта поддержка есть еще в GNU C99).

Используя перечисленные свойства, легко описать «класс» в Си. Для этого будет использовано несколько дефайнов:

```
#define PRIVATE      char _[sizeof (struct
#define PRIVATE_END  )];
#define private      struct

#define PROTECTED    char __[sizeof (struct
#define PROTECTED_END )];
#define protected    struct

#define class            struct
```

Вместо ключевого слова class будет использоваться struct, а приватные и защищенные поля будут заключены в секции private {} и protected {}, которые сами по себе являются структурами. При этом в тех файлах, где доступ к секции запрещен, должны использоваться аналогичные дефайны PRIVATE / PRIVATE_END и PROTECTED / PROTECTED_END. Эти дефайны, вместо того, чтобы объявлять структуру секции, объявляют массив байт размером со структуру этой секции. Пример использования в исходном файле:

```
/*  Class private definition */
class _Point
{
        private {
                const Object _; /* must be first */
                char a;
                int x;
        };
        protected {
                char b;
        };
        int y;
};
```

2. **Наследование**

Наследование — это расширение функциональности базового класса новыми полями и методами. Важной особенностью наследова-

ния является то, что объекты класса-потомка остаются совместимыми и могут использоваться вместо объектов базового класса.

Для написания реализации такого свойства достаточно знать, как работают объединения (unions). Объединение — не что иное, как синтаксический сахар над ручным приведением типа. То есть при обращении по полю объединения фактически происходит обращение к памяти по смещению этого поля.

Еще одна особенность заключается в том, что адрес на начало структуры совпадает с адресом на ее первый элемент (так как у него нулевое смещение). То есть можно в начало класса-потомка положить объект базового класса, и приведение этого объекта к базовому классу будет валидным.

3. **Конструкторы и деструкторы**

Создание и удаление объектов в C++ делится на 2 части — работа с памятью и собственно инициализация/уничтожение данных объекта. Чтобы можно было создавать объекты единообразно, можно написать функции malloc и new, которые будут выделять и освобождать память и вызывать специальные функции — конструкторы и деструкторы. В общем случае приходится использовать для этого дескрипторы классов (метаклассы), так как создание и удаление объектов происходит в реальном времени, и нет возможности получить никакой информации о типах.

4. **Метаклассы**

Метаклассы реализуют модель интроспекции, позволяя во время работы программы узнавать информацию о классе объекта. Это необходимо для правильной работы new/delete (нужна хотя бы информация о размере объекта), определения отношений между объектами. Для хранения информации о классах можно создать специальный класс, представляющий собой описание класса. Объекты этого класса — это дескрипторы классов. При этом в системе должны явно существовать два класса — Object и Class. Первый используется как базовый класс для наследования объектов, а второй — как базовый класс всех дескрипторов классов. При этом сам Class является тоже объектом, наследуемым от Object, класс которого — Class.

С помощью метаклассов удается реализовать не только базовые возможности ООП. Так как каждый объект знает, к какому классу он принадлежит, следовательно, он хранит внутри себя ссылку на дескриптор этого класса. И так как все объекты некоторого класса хранят ссылку на один и тот же дескриптор класса, то с помощью него можно реализовать статические поля и методы класса. К тому же, если хранить методы не в объектах, а в дескрипторах классов, то такие методы

можно переопределять в классах-потомках, получая виртуальные методы. Для выбора нужного метода используются селекторы.

5. **Селекторы**

Селекторы используются при использовании виртуальных методов. Они необходимы для того, чтобы можно было вызывать методы базового класса, так как они хранятся в дескрипторах класса, используемых по ссылкам. Они «раскручивают» ссылки, пока не найдут первый определенный метод класса. Раскручивание ссылок происходит с помощью дерева классов (аналог таблицы символов).

6. **Дерево классов**

Для интроспекции классов и определения связей требуется хранить историю наследований. Дерево классов представляет собой обычное дерево, корнем которого является Object, от которого затем наследуются все другие классы. Во-первых, появляется возможность отследить, что объект является валидным (т. е. наследуется от класса Object, или Class в случае дескриптора класса), во-вторых — что он содержит нужный метод, является предком/потомком другого класса. Так как большая часть ООП-обертки обрабатывается во время работы программы, то такое дерево необходимо для правильной работы классовой системы. Оно инициализируется в начале программы и заполняется классами по мере их использования.

7. **Исключения**

Модель исключений не является обязательной для ООП, однако применяется вместе с ним повсеместно. Для реализации исключений приходится использовать низкоуровневые функции для работы с регистрами процессора. В момент входа в блок try с помощью функции setjmp регистры процессора сохраняются в стеке исключений, затем идет выполнение как обычно. Как только вызывается метод throw(), происходит поиск соответствующего контекста в стеке и прыжок на этот контекст функцией longjmp. Дополнительно происходит сохранение информации о месте произошедшего исключения, трассировка вызовов до этого места, и эта информация сохраняется в объекте исключения. Для исключений существует специальный класс Exception, от которого можно наследоваться, чтобы получать новые виды исключений.

8. **Работа со стеком**

При создании объектов на куче обычно нет проблем. Однако при создании локальных объектов возникает вопрос — как проинициализировать и удалить объект прозрачно для пользователя? Первый вопрос решается достаточно просто — вместо динамического выделения памяти через new достаточно выделить память на стеке. Для этого

можно использовать нестандартные функции вроде alloc, но можно сделать проще — создать локальный массив байт с динамически генерируемым именем, размером с объект, создать указатель на объект, указывающий на этот массив, проинициализировать его конструктором объекта и оставить пометку, что это локальный объект (чтобы вызов delete не пытался освободить память по указателю).

С удалением сложнее — правильный ООП язык должен уметь вызывать деструктор при выходе локального объекта из области видимости. Здесь остается лишь использовать нестандартные расширения компиляторов. К примеру, GCC (и возможно, другие) поддерживает __attribute__((cleanup (cleanup_cb))). При указании такого атрибута при объявлении переменной функция cleanup_cb вызовется в момент выхода ее из области видимости. Таким образом, остается в макрос создания локального объекта при создании указателя указать ему такой атрибут, с функцией delete.

```
#define COMBINE1(X,Y) X##Y
#define COMBINE(X,Y) COMBINE1(X,Y)

void * _static_new (void *object, const void * _class, ...);

#define static_new(_class, var, ...) \
        char COMBINE(__static_, __LINE__) [sizeOfClass(_class())]; \
        __attribute__((cleanup (static_delete))) _class *var = (_class *)
&COMBINE(__static_, __LINE__); \
        _static_new(var, _class(), ## __VA_ARGS__)\
```

Автоматизация разработки

При использовании подхода, описанного в данной статье, приходится писать большой объем сопровождающего кода для того, чтобы объектная модель работала. Причем большая часть этого кода шаблонная, то есть можно автоматизировать генерацию шаблонов, дав программисту возможность удобного использования объектной модели. Сейчас существует язык Vala, который, по аналогии с ранним C++ и Cfront, транслируется прямо в Си, его используют разработчики GTK. Можно описать грамматику конструкций, упрощающих кодирование, тем самым описав новый язык программирования, и разработать под него транслятор, переводящий новые конструкции языка в вышеописанные шаблоны.

Заключение

В данной статье был описан способ организации кода на языке Си, позволяющий оперировать объектной моделью. Такая модель может быть закреплена грамматикой, таким образом может быть описан

новый язык программирования, который будет транслироваться в Си, используя все вышеперечисленные методы.

Если не использовать генераторы шаблонов или языки-обертки, то сильно вырастает сложность написания и поддержки кода, при этом иногда вместо статической типизации используется динамическая типизация через интроспекцию, что замедляет код.

Однако основное преимущество такого подхода — компиляция в сравнительно простой язык, под который имеется огромное количество продвинутых компиляторов и анализаторов. При использовании в производстве он не будет требовать особое окружение для сборки.

С исходным кодом можно ознакомиться на git@rpi.hldns.ru:c-oop.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Керниган Б., Ритчи Д.. The C Programming Language [Текст] / Б. Керниган, Д. Ритчи — 2-е изд. — Prentice Hall, 1988. — 272 с.

2. Керниган Б., Ритчи Д.. Язык программирования С [Текст] / Б. Керниган, Д. Ритчи — 2-е изд. — Москва, Вильямс, 2015. — 288 с.

3. Шрайнер А. Object Orientated Programming in ANSI-C [Текст] / А. Шрайнер — 221с.

4. Object oriented C [Электронный ресурс]. — Режим доступа: http://habrahabr.ru/post/205570/ — (Дата обращения: 20.03.2016).

5. Страуструп Б.. A History of C++: 1979−1991 [Электронный ресурс] / Б. Страуструп — AT&T Bell Laboratories — Режим доступа: http://www.stroustrup.com/hopl2.pdf/ — (Дата обращения: 20.03.2016)

С.В. СКВОРЦОВ, В.И. ХРЮКИН
Рязанский государственный радиотехнический университет

УЧЕБНАЯ КРОСС-СИСТЕМА ПРОГРАММИРОВАНИЯ ОДНОКРИСТАЛЬНОГО МИКРОПРОЦЕССОРА

Рассматривается учебная кросс-система программирования вычислительных устройств с архитектурой x85, обеспечивающая моделирование исполнения машинных команд в пошаговом и непрерывном режимах с отображением состояний программно-доступных элементов, а также трансляцию исходных программ с языка ассемблера.

В современных условиях при проектировании микропроцессорных устройств и их программировании возникает необходимость использования машинно-ориентированных языков, непосредственно взаимодействующих с аппаратными средствами. Для разработки и отладки программных модулей общепринятым является использование

эмуляторов, которые обеспечивают формирование машинного кода для микропроцессоров различных архитектур на базе инструментальных персональных ЭВМ [1].

Целью работы является создание кросс-системы программирования, обеспечивающей разработку машинных программ и эмуляцию процессов их исполнения на модели однокристального микропроцессора с архитектурой x85. Для достижения поставленной цели требуется решить две задачи:

- разработать эмулятор вычислительного устройства на базе однокристального микропроцессора с архитектурой x85, обеспечивающий имитацию выполнения машинных команд на уровне программно-доступных элементов архитектуры, таких как регистры, ячейки оперативной памяти и порты ввода-вывода;

- разработать кросс-ассемблер для входного машинно-ориентированного языка программирования, позволяющий выполнить анализ корректности текста исходной программы и сформировать машинный код, размещаемый в памяти и готовый к исполнению.

Укрупненная схема программы представлена на рисунке 1.

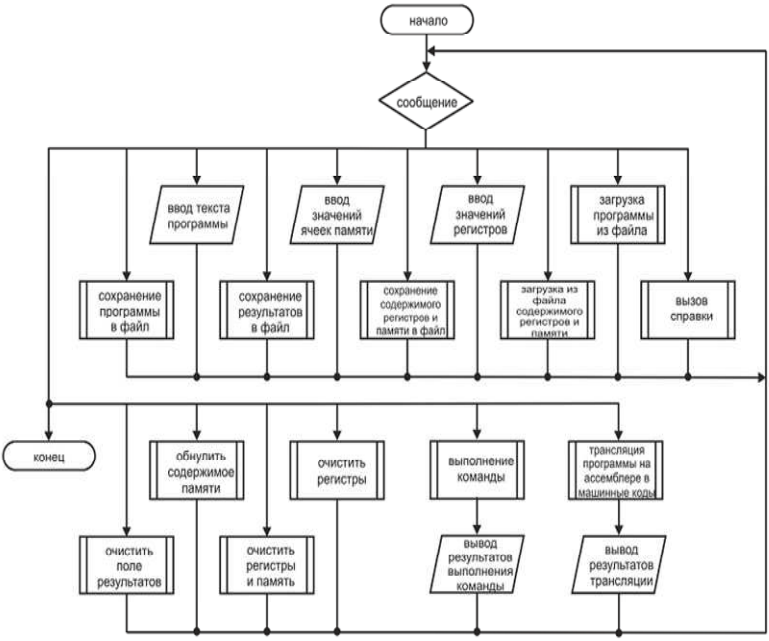


Рис. 1 – Укрупненная схема программы.

Эмулятор вычислительной системы моделирует исполнение заданной последовательности машинных команд, представленных в 16-ричных кодах, и позволяет работать с оперативной памятью объемом 64 Кбайт и 264 портами ввода-вывода. При разработке машинных программ может использоваться система команд МП с архитектурой x85: команды передачи данных между памятью и регистрами или между регистрами; команды арифметических и логических операций; команды передачи управления; команды управления микропроцессором.

Эмулятор моделирует выполнение машинных программ в непрерывном или пошаговом режиме с возможностью одновременного контроля за состоянием регистров, ячеек памяти и портов ввода-вывода. Результаты выполнения команд отображаются в соответствующих окнах программы [2]. Главное окно программы представлено на рисунке 2.

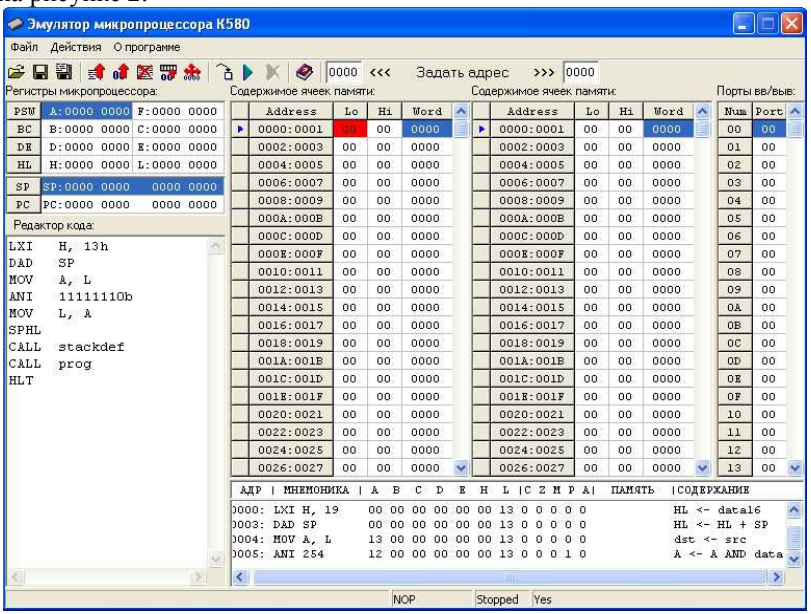


Рис. 2 – Главное окно программы.

В главном окне эмулятора расположены следующие элементы: системное меню; панель инструментов; таблица регистров микропроцессора; таблица оперативной памяти; таблица портов ввода-вывода; поле редактирования кода; поле вывода результатов трассировки; строка состояния.

Входными данными для эмулятора является прикладной программный код, написанный на языке ассемблера, или в машинных кодах, записываемых в ячейки памяти. При использовании языка ассемблера можно заранее подготовить программу в любом текстовом редакторе и загрузить этот код в эмулятор. Также можно написать программу на языке ассемблера, используя для этого поле «Редактор кода» в главном окне программы.

После запуска эмулятора появляется главное окно программы и ожидается ввод входных данных при помощи клавиатуры или с машинного носителя информации (из файла). Возможно сохранение текста написанной программы, результатов выполнения программы и отдельных команд, содержимого памяти и регистров. Если для написания программы используется язык ассемблера, то сначала выполняется трансляция программы в машинные коды, а затем осуществляется моделирование выполнения команд в пошаговом или непрерывном режиме.

Для проверки корректности ввода машинных кодов в ячейки памяти в кросс-системе предусмотрена возможность дизассемблирование кода с выводом мнемонического обозначения команды в строке состояния.

В программе предусмотрена возможность обращения к справочной системе, содержащей разделы: структуры микропроцессора с архитектурой x85 и вычислительного устройства на его основе, организация регистров, пространства памяти и пространства ввода-вывода; форматы команд и данных, способы адресации данных; правила записи программ на языке ассемблера; описание системы команд микропроцессора; описание работы с эмулятором.

Применение разработанной кросс-системы программирования позволяет повысить эффективность учебного процесса за счет повышения наглядности процесса исполнения машинных программ. Кроме того, представленная система может использоваться для решения научно-практических задач, связанных с разработкой и исследованием алгоритмов и программ, критичных ко времени исполнения [3-6] и поэтому реализуемых в виде низкоуровневого специализированного программного обеспечения.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Бакулев А.В., Бакулева М.А., Козлов М.А., Скворцов С.В. Технологии разработки параллельных программ для современных многоядерных процессоров // Экономика, статистика и информатика. Вестник УМО. 2014. № 6. С. 211-215.

2. Скворцов С.В., Хрюкин В.И. Программа «Эмулятор учебной вычислительной системы на базе однокристального микропроцессора». Свидетельство о регистрации в объединенном фонде электронных ресурсов «Наука и образование» (ОФЭРНиО) № 20430, выдано 23.10.2014.

3. Скворцов Н.В., Скворцов С.В., Хрюкин В.И. Дешифрация диагностического синдрома многопроцессорной системы в реальном времени // Системы управления и информационные технологии. 2010. Т. 39. № 1. С. 49-53.

4. Скворцов С.В. Применение симметричной диагностической модели при организации активной отказоустойчивости многопроцессорных систем // Вестник Рязанского государственного радиотехнического университета. 1998. № 4. С. 57-64.

5. Скворцов Н.В., Скворцов С.В., Хрюкин В.И. Синтез диагностических графов для многопроцессорных систем с активной отказоустойчивостью // Вестник Рязанского государственного радиотехнического университета. 2012. № 39-2. С. 83-89.

6. Скворцов Н.В., Скворцов С.В. Автоматизация проектирования процессов самодиагностики для многопроцессорных систем с активной отказоустойчивостью // Вестник Рязанского государственного радиотехнического университета. 2013. № 4-2 (46). С. 71-77.

И.В. СОЛОТЕНКОВ, А.В. БАКУЛЕВ
Рязанский государственный радиотехнический университет

ИССЛЕДОВАНИЕ ТЕХНОЛОГИЙ WEB-СЕРВИСОВ ДЛЯ ОРГАНИЗАЦИИ АВТОМАТИЗИРОВАННОГО ДОСТУПА К РАСПРЕДЕЛЕННЫМ ИНФОРМАЦИОННЫМ РЕСУРСАМ

Рассматриваются теоретические и практические вопросы технологий web-сервисов для организации автоматизированного доступа к распределенным информационным ресурсам.

Развитие современного общества всё больше основывается на использовании различных информационных технологий. В настоящее время идёт бурный процесс их перехода в цифровой формат, что затрагивает всё новые аспекты жизни человека. Это даёт возможность вывести на принципиально иной уровень качество и доступность информации.

Актуальной проблемой является организация эффективного информационного сопровождения подобного рода Всероссийских науч-

ных мероприятий. Публикация и обмен материалами в сети Интернет является обязательным требованием при проведении научных конференций. Конференция НИТ имеет свою веб-страницу в сети Интернет, однако информационные и функциональные возможности этого ресурса на сегодняшний день остаются весьма ограниченными.

Целью работы является исследование технологий web-сервисов для организации автоматизированного доступа электронного портала Всероссийской научно-технической конференции «Новые информационные технологии в научных исследованиях» (НИТ) к распределенным информационным ресурсам.

Стремительное развитие ИТ-среды, ее направленность на решение бизнес-задач — эти характеристики приобретают важное значение при проектировании ИТ-инфраструктур. В этих условиях отдельные решения по процессу объединения некоторых частей в единое целое настолько усложняют и саму инфраструктуру, и процесс управления ею, что становятся абсолютно неприемлемыми. Еще одна серьезная проблема — избыточность программных компонентов и сложность их многократного использования.

Все эти проблемы нашли выход в идее сервисно-ориентированной архитектуры (service-oriented architecture, SOA). Здесь используется общий архитектурный подход, концепция архитектуры программной среды предприятия, в которой возможна адекватная потребностям бизнеса динамика разработки, интеграции и эксплуатации приложений. Он ориентируется на сервисы, характеризуется распределенной архитектурой и слабосвязанными интерфейсами. Сервис в данном случае - это не что иное, как единица работ, выполняемая сервис-провайдером для обеспечения желаемого результата потребителю сервиса [1, 2]. Покажем на рисунке 1 взаимодействие между компонентами сервисно-ориентированной архитектуры.

Рис. 1 – Взаимодействие между компонентами SOA.

Для того, чтобы применить веб-сервисы для сайта НИТ будет использоваться каркас программной системы Yii Framework [2].

Веб-сервис — программная система, разработанная для обеспечения взаимодействия между несколькими компьютерами через сеть. В веб-приложении это обычно набор API, который можно использовать через интернет для выполнения действий на удалённом сервере, обслуживающем веб-сервис [2].

В Yii включены CWebService и CWebServiceAction, чтобы упростить задачу разработки веб-сервиса. API сгруппированы по классам, которые называются *провайдерами*. WSDL(Web Services Description Language), описывает функционал предоставляемого API и правила его использования пользователем. При обработке вызова клиента, Yii создаёт соответствующий WSDL экземпляр провайдера, вызывает метод API и отвечает на запрос.

Сервис, отдающий информацию о регистрации пользователя, имеет следующий вид.

```
class StockController extends CController {
  public function getStatus ($symbol) {
  $status =array('User1'=>Да, 'User2'=>Нет);
  return isset($status [$symbol])?$status[$symbol]:0;}
}
```

Метод *getStatus* является частью API веб-сервиса Дополнительные методы API могут быть описаны точно таким же образом.

После создания провайдера необходимо сделать его доступным для клиентов. Для этого необходимо описать действие контроллера CWebServiceAction:

```
class StockController extends CController {
  public function actions() {
  return array(
  'quote'=>array('class'=>'CWebServiceAction',),);}
  public function getStatus ($symbol) {}
}
```

Это всё, что требуется для создания веб-сервиса.

Также можно использовать массивы. Для этого необходимо дописать [] в конец примитивного или составного типа. Таким образом получим массив с элементами заданного типа.

Ниже приведён пример определения метода API getPosts, возвращающего массив объектов класса Post:

```
class PostController extends CController {
   public function getPosts() {
      return Post::model()->findAll(); }
```

```
}
class Post extends CActiveRecord {
   public $id;
   public $title;
   public static function model($className=__CLASS__) {
      return parent::model($className); }
}
```

В дальнейшем, для полноты исследования, будет создан клиент, использующий веб-сервис, и реализованы другие веб-сервисы.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. В. Филиппов, Информационные взаимодействия и Web-сервисы, Ленанд, 2009г.

2. Статьи и материалы по теме SOA [Электронный ресурс]: URL: http://www.ibm.com/developerworks/ru/webservices/ (дата обращения 01.04.16).

3. Полное руководство по Yii // Веб-сервисы [Электронный ресурс]: URL: http://yiiframework.ru/doc/guide/ru/topics.webservice/ (дата обращения 01.04.16).

Ю.К. СТОЛЯРОВА
Московский государственный технический университет им. Баумана

СРЕДСТВА РАЗРАБОТКИ АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ «ВЕДЕНИЯ ЕДИНОГО РЕЕСТРА ПОЛЬЗОВАТЕЛЕЙ ПОЛИКЛИНИКИ»

Рассматриваются и анализируются средства разработки, сервер приложений, архитектура сети и структура программного изделиядля применения в разработке в автоматизированной системе «Ведения единого реестра пользователей поликлиники».

Предметной областью данной информационной системы является совокупность объектов, объединенных общим понятием (более того, событием реального мира) – процессом поликлиники. Данная предметная область обладает большим количеством сущностей и сложными взаимосвязями между ними. В этот процесс (в процесс поликлиники) вовлечено огромное количество людей, имеющих разные функциональные возможности и обязанности. Если разделить их по группам, то мы можем выявить: медицинских сестер, докторов. Что же касается неживых объектов (сущностей иного типа), то тут могут быть представлены: номера кабинетов докторов. Количество и набор сущ-

ностей данного типа зависит только от желания разработчиков информационной системы охватить как можно больший объем информации.

В этом процессе можно выделить две основные группы пользователей, имеющих различные функциональные возможности:

- Сотрудники;
- Администраторы.

Для каждой группы и подгруппы пользователей должны быть определены права и установлены ограничения на доступ к базе данных.

Для проектируемого программного комплекса приоритетными являются следующие критерии качества:

- скорость работы;
- выявление ошибочной информации;
- удобство работы;
- отказоустойчивость системы;
- документация.

Скорость работы включает требование к временным характеристикам:

- время ввода информации не должно превышать 1 минуту;
- время поиска необходимой информации (ее расположение) не должно превышать 2 минут;
- время выдачи информации по запросу не должно превышать 1 минуты.

Выявление ошибочной информации включает обнаружение ошибок при вводе данных с несоответствующим необходимому типу, а также выявление зависимостей, препятствующих корректной обработке данных алгоритмов. Обеспечивает простоту эксплуатации и достоверность итоговых данных.

Удобство работы означает простоту и понятность работы с подсистемой. Обеспечение удобства интерфейса позволяет минимизировать время, затрачиваемое при работе с системой и получать максимальную эффективность работы. Интерфейс, основанный на web-страницах, является наиболее доступным, так как нет необходимости в установке программного продукта – необходимо только наличие браузера.

Разработка программного изделия. Для системы будет использоваться:

- база данных, поскольку информация о пациентах, докторах будет обновляться;
- web-сервер, чтобы принимать запросы от пользователей;

- сервер приложений, поскольку web-сервер самостоятельно не может самостоятельно формировать запросы для базы данных.

Описание СУБД. В качестве системы управления базами данных (СУБД) была выбрана PostgreSQL. Данный выбор обусловлен тем, что PostgreSQL включает API для большого количества языков программирования (Python, C++, C,), а также является кроссплатформенной системой (UNIX системы и среда MicrosoftWindows).

PostgreSQL – свободная объектно-реляционная система управления базами данных (СУБД). PostgreSQL используется как полигон для исследований нового типа баз данных, ориентированных на работу с потоками данных.

PostgreSQL предоставляет командный интерфейс для работы с системным каталогом, с помощью которого можно не только получать информацию об объектах системы, но и создавать новые.

Система безопасности обеспечивается 4 уровнями:

- PostgreSQL нельзя запустить под привилегированным пользователем – системный контекст;
- сложная система аутентификации на уровне хоста или IP адреса/подсети. Система аутентификации поддерживает пароли, шифрованные пароли и прочие системы, которые могут подключаться, используя механизм подключаемых аутентификационных модулей;
- детализированная система прав доступа ко все объектам базы данных, которая совместно со схемой, обеспечивающая изоляцию названий объектов для каждого пользователя, PostgreSQL предоставляет богатую и гибкую инфраструктуру.

Пределы PostgreSQL:

- максимальный размер таблицы – 32 ТБ
- максимальная длина записи – 4000 Гб
- максимальная длина атрибута – 1Гб

Средства разработки. В качестве средства разработки был выбран Python. Python – интерпретируемый язык программирования. С одной стороны, это позволяет значительно упростить отладку программ, с другой это обуславливает сравнительно низкую скорость выполнения. Однако для решения данной задачи это не является критичным, поскольку в ней не производится ресурсоемких вычислений. Простой и понятный синтаксис языка позволяет значительно сократить время разработки сайта. В python предусмотрена динамическая типизация: не надо заранее объявлять тип переменной, что очень удобно при разработке. Программа, написанная на Python, будет

функционировать совершенно одинаково вне зависимости от того, в какой операционной системе она запущена.

Сервер приложения. Наиболее полным и распространенным фреймворком для Python является Django. Именно ее будет использовать для разработки автоматизированной системы «Ведение единого реестра пользователей поликлиники».

Django – свободный фреймворк для веб-приложений на языке Python, использующий шаблон проектирования MVC. Проект поддерживается организацией DjangoSoftwareFoundation.

Архитектура Django похожа на «Модель-Представление-Контроллер» (MVC). Контроллер классической модели MVC примерно соответствует уровню, который в Django называется представлением (view), а презентационная логика представления реализуется в Django уровнем шаблонов (template). Из-за этого уровневую архитектуру Django часто называют «Модель-Шаблон-Представление» (MTV).

Django проектировался для работы под управлением Apache с модулем modpython и с использованием PostgreSQL в качестве базы данных.

В настоящее время, помимо базы данных PostgreSQL, Django может работать с другими СУБД: MySQL, SQLite, MicrosoftSQLServer, DB2, Firebird, SQLAnywhere, Oracle.

В составе Django присутствует собственный веб-сервер для разработки. Сервер автоматически определяет изменения в файлах исходного кода проекта и перезапускается, что ускоряет процесс разработки на Python, но пригоден только для процесса разработки и отладки приложения.

Некоторые возможности Django:

- API доступа к БД с поддержкой транзакций;
- встроенный интерфейс администратора, с уже имеющимися переводами на многие языки;
- диспетчер URL на основе регулярных выражений;
- расширяемая система шаблонов с тегами и наследованиями;
- система кеширования;
- библиотека для работы с формами (наследование, построение форм по существующей модели БД).
- встроенная автоматическая документация по тегам шаблонов и моделям данных, доступная через административное приложение.

Архитектура системы. Для визуального представления была выбрана технология HTML. Язык гипертекстовой разметки страниц –

HTML (HyperTextMarkupLanguage) используется для создания статистических веб-страниц, состоящих из форматированного текста с комбинацией графики, гипертекстовыми ссылками на другие документы, находящиеся в любом месте интернет-сети. Этот язык не зависит от аппаратно-программной платформы. HTML-документ состоит из тэгов, которые являются указателями на операции, выполняемые приложением (браузером) на стороне сотрудника.

В результате использования вышеописанных технологий архитектура системы имеет следующий вид:

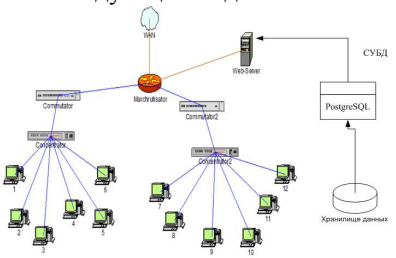


Рис. 1 – Архитектура сети.

Структура программного изделия. В рамках проекта по созданию системы были созданы сайт поликлиники и система администрирования сайта поликлиника.

Сайт поликлиники состоит из следующих модулей:

- модуль регистрации клиента: медицинская сестра регистрирует клиента, вводя персональные данные в базу данных;
- модуль записи клиента на прием: медицинская сестра записывает клиента на прием к доктору;
- модуль просмотра очереди доктором: просматривается очередь из клиентов, записавшихся к доктору на прием;
- модуль просмотра доктором персональной информации клиента: просматривается персональная информация о клиенте;
- модуль просмотра истории болезни клиента: доктором просматривается информация о предыдущих посещениях и его результатах посещения к докторам;
- модуль записи/сохранения о текущем визите клиента: заполняется и сохраняется результаты приема пациента у доктора.

Система администрирования сайта поликлиники позволяет сотрудникам добавлять, информацию состоит из следующих модулей:

- модуль администрирования информации о пользователях: ввод, редактирование и удаление информации о пользователях;

- модуль администрирования информации о клиентах: ввод, редактирование и удаление информации о клиентах;
- модуль администрирования информации о приеме: ввод, редактирование и удаление информации о визитах пациента;
- модуль администрирования информации о талонах: ввод, редактирование и удаление информации о талонах;
- модуль администрирования информации о медицинских сестрах: ввод, редактирование и удаление информации о медицинских сестрах;
- модуль администрирования информации о формах рецептов, справок: ввод, редактирование и удаление информации о формах рецептов и справок;
- модуль администрирования информации о режиме работы: ввод, редактирование и удаление информации о режиме работы.

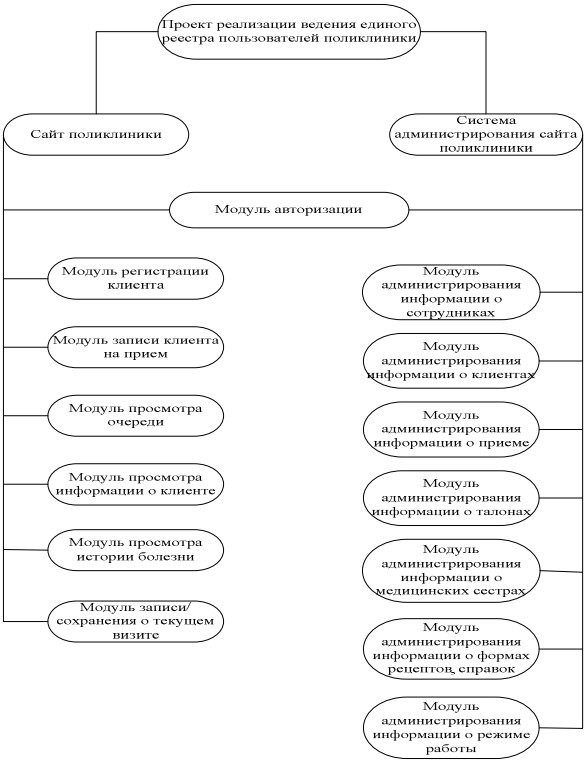


Рис. 2 – Структура программного изделия.

Универсальный модуль:

Модуль авторизации: регистрация, проверка авторизации зарегистрированных сотрудников, ввод и редактирование персональной информации, назначение прав доступа.

Структура схемы представлена на рисунке 2.

Таким образом, приведенный сравнительный анализ средств разработки, возможных платформ и механизмов реализации показал, что наиболее подходящими и отвечающими поставленным задачам средствами являются PostgreSQL, Python с фреймворком Django. Информационная система имеет модульную структуру, то есть отвечает требованиям расширяемости и масштабируемости и готова к дальнейшему внедрению для автоматизации процесса работы поликлиники.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Григорьев Ю.А., Ревунков Г.И. Банки данных: учеб.для вузов. М.: Изд-во МГТУ им. Н.Э. Баумана, 2002. 320 с.

2. Т. Коннолли, К. Бегг, А. Страчан Базы данных: проектирование, реализация и сопровождение. Теория и практика. М.: Издательский дом «Вильямс», 2000. 1120 с.

3. Ведение журнала и восстановление в SQLServer: http://www.cyberguru.ru/database/sqlserver/log-recovery-page3.html (дата обращения 15.02.2016).

4. Восстановление базы данных: https://msdn.microsoft.com/ru-ru/library/ms177429(v=sql.120).aspx (дата обращения 29.01.2016).

Ю.М. ТОБРАТОВ

Рязанский государственный радиотехнический университет

ИСПОЛЬЗОВАНИЕ МОБИЛЬНЫХ УСТРОЙСТВ В ЛВС УЧЕБНОГО ПОДРАЗДЕЛЕНИЯ ВУЗА

Рассмотрены вопросы использования мобильных устройств в локальной вычислительной сети (ЛВС) учебного подразделения вуза при организации беспроводного доступа к ней на примере кафедры САПР ВС РГРТУ в учебных и научных целях студентами, преподавателями и сотрудниками кафедры. Приведены способы организации зон беспроводного доступа мобильных устройств в корпоративных сетях учебных заведений, а так же возлагаемые на них задачи и категории пользователей активно их использующие, предложены проекты решений построения беспроводного сегмента ЛВС кафедры для работы с мобильными устройствами.

В настоящее время учебный процесс в высших учебных заведениях невозможно представить без электронных обучающих средств и

информационных сервисов, а так же без современных телекоммуникаций, позволяющих осуществлять надежный и высокоскоростной доступ к ним. Использование в этих целях мобильных устройств с функцией работы в беспроводных сетях, число которых у потенциальных пользователей этих средств быстро и постоянно увеличивается, является оправданным и необходимым.

Возможность использования мобильных устройств в локальной вычислительной сети учебного подразделения вуза, за счет организации беспроводного доступа к ней, и сети Интернет рассмотрена на примере кафедры САПР ВС РГРТУ. Метод использует существующую сетевую структуру и предназначен для применения в учебных и научных целях студентами, преподавателями и сотрудниками кафедры.

Зоны Wi-Fi-доступа в корпоративных сетях учебных заведений можно организовать двумя способами [1]:

- публичная зона беспроводного доступа (Wi-Fi «хот-спот»), охватывающая всю территорию учебного заведения с централизованным управлением и доступом к общим сервисам вуза и сети Интернет с определенными ограничениями.
- локальная зона беспроводного доступа, организованная на территории кафедр с доступом к их локальным сетевым ресурсам.

Создание локальных зон беспроводного доступа кафедр при существующей публичной зоне обусловлено необходимостью наличия надежной и устойчивой связи Wi-Fi с учетом сложной планировки учебных корпусов вуза и ограничении контингента пользователей с целью снижения нагрузки на каналы и точки беспроводного доступа [2]. Особенно это важно при проведении учебных занятий с применением электронных обучающих средств во избежание их срыва.

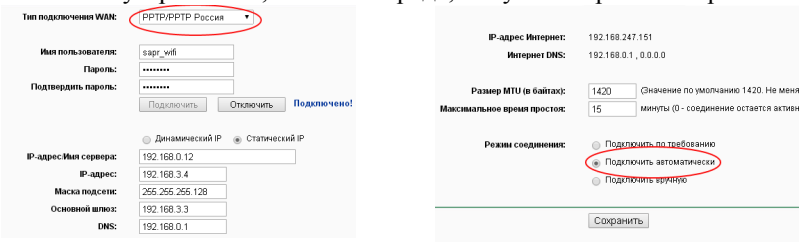
Wi-Fi сегмент ЛВС кафедры при организации и проведении учебного процесса выполняет следующие задачи [3]:

- доступ к средствам электронного обучения, методической литературе и учебным материалам, представленным в электронном виде и хранимым на файловом сервере кафедры;
- доступ к размещенным на файловом сервере кафедры «материалам для скачивания»;
- работа с «диском общего доступа» (хранение и копирование пользовательских файлов);
- доступ к информационным сервисам, предоставляемым локальным WEB-сервером кафедры;
- высокоскоростной, нелимитированный доступ к сети Интернет в учебных и научных целях.

Категории пользователей, активно использующие ЛВС кафедры [3]:

- контингент обучающихся на кафедре (студенты, аспиранты, слушатели различных курсов проводимых на кафедре);
- профессорско-преподавательский состав кафедры;
- административные клиенты;
- учебно-вспомогательный состав кафедры;
- участники различных конференций и семинаров.

При работе в беспроводной сети пользователями приведенных категорий активно применяются мобильные электронные устройства различных типов: ноутбуки, планшеты, смартфоны, и др., с установленными на них различными операционными системами (ОС): Windows, Android, iOS. Операционные системы, установленные на мобильных устройствах, в свою очередь, могут быть разных версий.



А Б

Рис. 1 - Основные настройки Wi-Fi-роутера.

На основании анализа всего выше перечисленного автором сформированы проекты решений построения беспроводного сегмента ЛВС кафедры с добавлением Wi-Fi-роутеров (беспроводных точек доступа) при использовании существующей топологии сети, сетевого и кабельного оборудования. Были протестированы несколько Wi-Fi-роутеров разных изготовителей и ценовых групп на предмет устойчивой работы при подключении через них пользователей [4], как к локальным ресурсам кафедры, так и к сети Интернет, определены рекомендации по их выбору и настройке. Для данной цели необходим Wi-Fi-роутер с возможностью работы с PPTP-протоколом (туннельный протокол типа «точка-точка») для организации VPN-соединения, осуществляющего подключение к сети Интернет через центральный сервер университета, а так же с параметром автоподключения к сети при разрыве соединения. Эти функции являются минимально необходимыми для выполнения роутером рассматриваемых в работе задач. На рис. 1 а, б представлены примеры установки значений этих параметров в программе управления роутером.

Предлагается использовать бюджетный Wi-Fi-роутер TP-Link модели TL-WR841N, удовлетворяющий вышеперечисленным требованиям и имеющий ряд других преимуществ, таких как: высокая скорость беспроводного доступа – 300 Мбит/с, и мощный радиосигнал – усиление 5 дБ (две антенны). При недостаточном радиусе охвата беспроводной сети нужно использовать аналогичный роутер с большей передающей мощностью.

Для подключения к сети и работе с представленным в ней контентом на мобильные устройства необходимо установить и настроить различные программные приложения, в зависимости от их типа и установленной на них ОС. Протестированы некоторые свободно распространяемые программные приложения и даны рекомендации пользователям по их установке, настройке и применению.

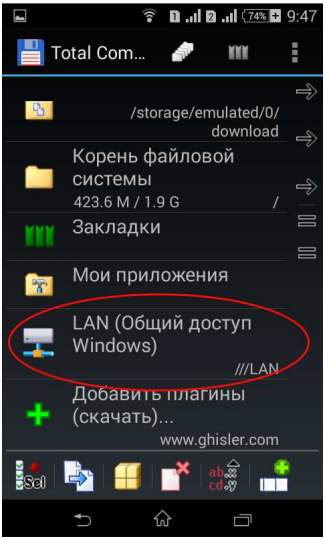
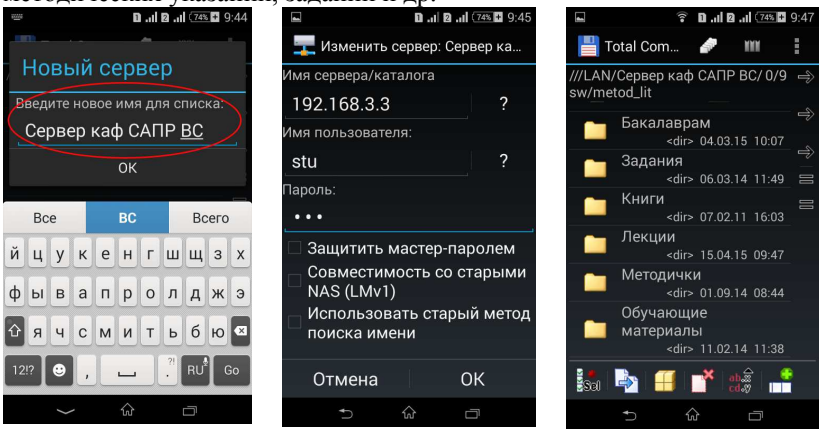


Рис. 2 - Окно приложения Total Commander с установленным сетевым плагином.

Например, для ОС Android версий 4.x.x [5] необходимо скачать и установить бесплатные приложения: файловый менеджер Total Commander и плагин к Total Commander - Android LAN (Windows network) Plugin 2.09. На рис. 2 показано окно приложения Total Commander с установленным сетевым плагином LAN на примере смартфона Sony D2105 с установленной ОС Android 4.4.2. Настройка установленных приложений в данном случае сводится к добавлению

«нового сервера» (сервера содержащего необходимый контент), в качестве которого может выступать любой компьютер подключенный к локальной сети (рис. 3 а), задания его имени (используется в дальнейшем при его идентификации пользователем), ввода его IP-адреса в локальной сети, задания имени пользователя и его пароля для подключения к этому серверу (пользователь должен быть предварительно зарегистрирован на этом сервере) (рис. 3 б). В данном случае подключение осуществляется к файловому серверу кафедры, где и хранится весь необходимый контент и установлены все необходимые сервисы, через него также осуществляется выход в сеть Интернет. На рис. 3 в представлен экран смартфона подключенного к папке файлового сервера содержащей учебно-методический контент, такой как файлы лекций, методических указаний, заданий и др.



А Б В

Рис. 3 - Настройка Total Commander с установленным сетевым плагином.

Для просмотра учебно-методического контента необходимо применять соответствующие приложения, установленные на мобильном устройстве и позволяющие просматривать файлы различных форматов используемых для этих целей. Если таковые приложения на мобильном устройстве отсутствуют, то рекомендуется установить бесплатную (или платную с расширенными возможностями) версию офисного приложения OfficeSuite v.8.4.[5], которое позволяет просматривать и редактировать текстовые файлы, файлы презентаций, электронных таблиц, а так же читать файлы формата PDF. Этого в

большинстве случаев достаточно для выполнения рассматриваемых задач.

Приложение Total Commander с сетевым плагином LAN так же позволяет работать с «материалами для скачивания», предоставляя возможность их сохранения на свое устройство и пользовательскими файлами расположенными как на «дисках общего доступа» сетевых ресурсов, так и на карте памяти самого мобильного устройства: операции копирования, переноса, удаления, создания папок и т.д.

Аналогичные программные приложения используются и для ОС iOS компании Apple, например рекомендуется использовать сетевое приложение для подключения к локальным серверам – RManager (бесплатно при работе с двумя серверами). В ОС Windows все необходимые приложения встроены в саму операционную систему и требуется только некоторая несложная настройка сетевых подключений.

Применение мобильных устройств и современных электронных обучающих средств при организации и проведении учебного процесса и при самостоятельной подготовке студентов значительно сокращает расходы на приобретение и издание бумажных носителей информации и на их хранение. Рассмотренный в работе метод упрощает поиск необходимой при обучении информации и время доступа к ней, позволяет своевременно изменять и корректировать информационный учебный контент поддерживая его всегда в актуальном виде.

Проекты и решения, представленные в данной работе, повысили степень применения электронных средств обучения и учебно-методического контента ЛВС в учебном процессе, за счет добавления значительного количества пользователей мобильных устройств при его проведении и организации на кафедре. В конечном итоге представленное решение привело к повышению эффективности и качества обучения студентов.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Корячко В.П., Перепелкин Д.А. Анализ и проектирование маршрута передачи данных в корпоративных сетях. М.: Горячая линия – Телеком, 2012. 238 с.

2. Перепелкин Д.А., Цыганов И.Ю. Усовершенствованный алгоритм сегментации структур корпоративных сетей по критерию минимальной стоимости // Вестник Рязанского государственного радиотехнического университета. 2015.№53 С.48-58.

3. Тобратов Ю.М., Осин И.Г. Использование технологии Wi-Fi в ЛВС учебного подразделения вуза / Материалы XX Всероссийской

научно-технической конференции студентов, молодых ученых и специалистов НИТ-2015. РГРТУ. 2015. С.183-184.

4. Тобратов Ю.М. Применений технологий Cisco при модернизации ЛВС учебного подразделения вуза / Информационные технологии: межвуз. сб. науч. тр. Рязань. РГРТУ. 2015. С. 115-117.

5. Тобратов Ю.М. Метод расширения функциональных возможностей учебной ЛВС / Материалы III Всероссийской молодежной научно-технической конференции ИТ в науке и производстве -2016. ОмГТУ. 2016. С.192-197.

П.А. УЛЬКИН, В.П. КОРЯЧКО
Рязанский государственный радиотехнический университет

ИССЛЕДОВАНИЕ ОБЛАЧНЫХ ВЫЧИСЛЕНИЙ ПРИ РАБОТЕ С УДАЛЕННЫМ СЕРВЕРОМ

Рассматриваются теоретические и практические вопросы реализации облачных вычислений.

Методика включения в учебный процесс глубокого и детального исследования реальной или имитированной ситуации, выполняемого для того, чтобы выявить ее частные и/или общие характерные свойства. Это развивает аналитическое мышление учащихся, помогает осуществить системный подход к решению проблемы, позволяет определить варианты правильных и ошибочных решений, выбрать критерии нахождения оптимального решения. Методика была разработана в 80-е годы 20 в. в Гарварде, применялась в бизнес-школах, а сейчас широко используется в системе профессионального образования.

Каждая сцена причинно-следственных связей иллюстрирует типичные ошибки.

Опыт и навыки полученные в результате прохождения обучения с использованием ситуационных обучающих сцен позволяют предвидеть различные варианты развития ситуаций и как следствие значительно снизить возникновение внештатных ситуаций

Облачные вычисления (от англ. Cloud Computing) – это распределённая обработка информации, где компьютерные ресурсы конечному пользователю предоставляются в качестве сервиса (услуги).

Национальным институтом стандартов и технологий США зафиксированы следующие обязательные характеристики облачных вычислений:

- Самообслуживание по требованию (англ. self service on demand) – потребитель самостоятельно определяет и изменяет свои вычислительные потребности (серверное время, скорость доступа, объём хранимых данных и т.д);
- Универсальный доступ – отсутствие зависимости от устройства (доступ к сервису должен осуществляться с любого устройства, подключенного к сети Интернет)
- Объединение ресурсов (англ. resource pooling) - поставщик услуг объединяет ресурсы для обслуживания большого числа потребителей в единый пул для динамического перераспределения мощностей между потребителями в условиях постоянного изменения спроса на мощности; при этом потребители контролируют только основные параметры услуги (например, объём данных, скорость доступа), но фактическое распределение ресурсов, предоставляемых потребителю, осуществляет поставщик (в некоторых случаях потребители всё-таки могут управлять некоторыми физическими параметрами перераспределения, например, указывать желаемый центр обработки данных из соображений географической близости);
- Эластичность – предоставление, расширение, а также сужение услуг (сервисов) в любой момент времени в автоматическом режиме;
- Учёт потребления – на основе потреблённых ресурсов (объём данных, пропускная способность канала, количество пользователей и др.) поставщик оценивает объём предоставленных услуг [1, 5, 9].

Вычислительные «облака» могут состоять из нескольких десятков, а то и тысяч серверов, размещенных удалённо, способных обеспечивать работу очень большого количества приложений, с одновременным их использованием сотен тысяч пользователей. Максимально полная автоматизация является обязательным условием эффективного управления такой крупномасштабной системой. Аналитики называют облачные вычисления наиболее перспективной и многообещающей технологией будущего, ссылаясь на то, что большая часть информационных технологий будет перенесена в облачные сервисы в течение нескольких следующих лет [2-4].

Основными моментами, которые поспособствовали развитию облачных технологий являются рост производительности компьютеров, появление многоядерных и многопроцессорных вычислительных систем, развитие блейд-систем, появление систем и сетей хранения данных, а также консолидация инфраструктуры.

Согласно мнению специалистов – консолидация ИТ-инфраструктуры – это первый шаг на пути к «облаку», так без неё невозможно единое предоставление сервисов [1].

Консолидация представляет собой объединение вычислительных ресурсов в едином центре.

В настоящее время прослеживается четкая тенденция к консолидации ИТ-ресурсов корпораций. Именно она позволяет существенно снизить расходы на ИТ. Кроме того улучшается управляемость компании благодаря наиболее полной информации о функционировании ИТ-ресурсов. Существуют следующие разновидности консолидаций:

- консолидация серверов – размещение приложений в одном кластере серверов;
- консолидация систем хранения – совместное использование централизованной системы хранения информации несколькими узлами;
- консолидация приложений – расположение на одном хосте нескольких приложений.

Различают два типа консолидации – логическую и физическую. Физическая консолидация – это единая площадка, логическая - централизация управления.

Облачные вычисления представляют собой доступ к выделенным ресурсам в виде сервиса, предоставляемого посредством Интернета, при этом у пользователя отсутствует необходимость в особых знаниях того, как устроена работа в этом облачном сервисе. При этом, компьютер пользователя выступает в качестве терминала, подключенного к сети.

Ярким примером служат поисковые системы. Работа с которыми по своей сути очень проста, но и очень плодотворна, так как поисковые системы предоставляют поистине огромные ресурсы для поиска. Сегодня крупные вычислительные системы помимо хранения и обработки информации позволяют пользователю самому создавать свои собственные виртуальные вычислительные центры.

Элементами концепции облачных вычислений являются: инфраструктура, платформа и программное обеспечение как сервисы, а также бизнес-приложения, доступные через Интернет [5-7].

Облачные вычисления имеют следующие уровни архитектуры:

1) уровень клиента – это клиентское программное обеспечение, обеспечивающее доступ к «облаку»;

2) уровень сервисов – это сами услуги (сервисы), используемые через облачную модель;

3) уровень приложений – это программное обеспечение, доступное из «облака», которое не требует установки на компьютере;

4) уровень платформы – это программная платформа, включающая полный набор инструментов для использования облачных вычислений на компьютере пользователя;

5) уровень памяти – хранение информации на «облаке», а также доступ к ней;

6) уровень инфраструктуры – предоставление полной виртуальной платформы через облачный сервис [4, 5, 8].

«Облако» представляет собой не что иное как сеть серверов, связанных между собой.

Облачные системы являются сервис-ориентированными, основная задача которых заключается в предоставлении пользователю качественных услуг.

Принято выделять следующие модели сервисов (услуг):

- IaaS – инфраструктура как услуга – предоставление сетевой инфраструктуры, хранилища и виртуального сервера;
- PaaS – платформа как услуга – предоставление доступа платформе. Пользователи имеют возможность в создании и размещении своих собственных приложений на базе платформы, а также имеют доступ к таким ресурсам как операционная система, хранилище данных и т.д.;
- SaaS – ПО как услуга – предоставление программного обеспечения.

Так как облачные технологии появились совсем недавно, их внедрение вызывает ряд противоречий в организации единого информационного пространства. Принятие эффективного решения по внедрению вычислений – это довольно сложная задача, требующая полного анализа предприятия, среды, для которой предполагается, разворачивание удалённого сервера (рабочего стола) [3, 4, 8].

Среди достоинств «облаков» выделяют: низкие первоначальные инвестиции в IT, оптимизация расходов, доступность с любого устройства, подключенного к сети Интернет, обеспечение непрерывности работы и возможность быстрого восстановления сервисов после «падения».

Основными недостатками облачных вычислений являются контроль, безопасность, нестабильность системы затрат, вероятное ухудшение гибкости бизнеса, а также претензии (часто не обоснованные), связанные с безопасностью защиты информации в «облаке».

В наше время использование облачных технологий в любых областях деятельности, от хранения документов простыми пользователями, до систем крупного бизнеса – это своего рода компромиссное решение между по-настоящему широкими возможностями с одной стороны, и определёнными (порой не малыми) рисками с другой. С экономической точки зрения ситуация аналогична: баланс между очевидной инвестиционной привлекательностью облачных вычислений и финансовыми рисками подобных вложений пока не найден [9-10].

Однако, несмотря на это – облачные сервисы активно развиваются, а рынок неуклонно растёт. К 2020 году «Forrester Research» прогнозирует увеличение объёма облачных вычислений до 240 миллиардов долларов. Очевидно, что будущее – за облачными технологиями [1, 3].

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Cloud Computing Market: Global Forecast [Электронный ресурс]. URL: http://marketsandmarkets.com (дата обращения: 29.03.2016).

2. Власов В.К., Королев Л.Н. Элементы информатики./ Под. Ред. Л.Н. Королева.- М.: Наука, 2008 г.

3. Облачные вычисления: обзор и рекомендации. Общая среда облачных вычислений - Рекомендации Национального Института Стандартов и Технологий (США), NIST, USA, 2007.

4. Эталонная архитектура облачных вычислений - Рекомендации Национального Института Стандартов и Технологий (США), NIST, USA, 2007

5. Определение Облачных Вычислений - Рекомендации Национального Института Стандартов и Технологий (США), NIST, USA, 2007

6. Грейс Уокер, "Основы облачных вычислений", Справочник IBM.

7. Gillam, Lee Cloud Computing: Principles, Systems and Applications/ Nick Antonopoulos, Lee Gillam. - L.: Springer, 2010.

8. SoCC '10: Proceedings of the 1st ACM symposium on Cloud computing / Hellerstein, Joseph M. - N. Y.: ACM, 2010.

9. Mell, Peter and Grance, Timothy The NIST Definition of Cloud Computing (англ.). Recommendations of the National Institute of Standards and Technology. NIST (20 October 2011).

10. Андрей Крупин, "Cloud Computing: высокая облачность". Компьютерра, 2009 г.

Х.Л. ФАМ, А.П. ШИБАНОВ
Рязанский государственный радиотехнический университет

ИМИТАЦИОННОЕ МОДЕЛИРОВАНИЕ АГРЕГИРОВАННОГО КАНАЛА ПЕРЕДАЧИ ИЗМЕРИТЕЛЬНОЙ ИНФОРМАЦИИ

Рассматривается задача проверки адекватности аналитических моделей агрегированного канала путем сравнения расчетов, проведенных с их использованием, с результатами имитационного моделирования.

Введение. Канал сети передачи данных полигонного измерительного комплекса выполняет прием, группирование и передачу информации от нескольких измерительных систем, таких как антенные комплексы передачи телеметрической информации, внешнетраекторные радиотехнические и оптико-электронные стации, лазерные дальномеры и т.п. Информация от различных систем поступает на устройство предварительной обработки, в котором реализуется операции слияния потоков данных, проводится сжатие информации перед передачей в канал и т.п. Далее кадр измерительной информации с определенной длиной поступает в канал, состоящий из двух подканалов. Кадр информации делится на две равные части (пакеты), передаваемые одновременно в параллельном режиме. На выходе канала пакеты снова преобразуются в кадр для передачи в центр обработки информации. Время передачи пакета по агрегированному каналу представляется случайной величиной. Исходная статистическая информация о работе подканала передачи измерительной информации представляется нормированным распределением Эрланга.

Целью работы является сравнение результатов имитационного моделирования агрегированного канала передачи измерительной информации с аналитическими выкладками. Результаты аналитических расчетов получены при условии, что случайное время передачи кадра по подканалу распределено по нормированным законам Эрланга второго, третьего и четвертого порядков.

Теоретическая часть. В работе [1] определены первый $\overline{X}$ и второй $\overline{X^2}$ моменты распределения времени передачи кадра по агрегированному каналу передачи измерительной информации.

I. При параметре вероятностного распределения $k = 2$

$$\overline{X} = \frac{T}{2} + \frac{11}{16\alpha},\ \overline{X^2} = \frac{T^2}{4} + \frac{11T}{16\alpha} + \frac{39}{64\alpha^2}.$$

II. При параметре вероятностного распределения $k = 3$

$$\overline{X} = \frac{T}{2} + 0{,}656\,\alpha^{-1},\ \overline{X^2} = \frac{T^2}{4} + 0{,}656\,T\alpha^{-1} + 0{,}516\,\alpha^{-2}.$$

III. При параметре вероятностного распределения $k = 4$

$$\overline{X} = \frac{T}{2} + 0{,}637\,\alpha^{-1},\ \overline{X^2} = \frac{T^2}{4} + 0{,}637\,T\alpha^{-1} + 0{,}466\,\alpha^{-2}.$$

В этих формулах T – постоянная составляющая задержки в канале, α – параметр интенсивности распределения Эрланга.

Выберем следующие значения входных параметров:

− входной поток информации считаем простейшим с интенсивностью $\Lambda = 0{,}04\,мс^{-1}$;

− параметр интенсивности распределения Эрланга, характеризующего вариации времени передачи канала $\alpha = 1$;

− постоянную составляющую задержки в обычном канале, равную $T = 10\,мс$.

− число заявок, вырабатываемых блоком-генератором, равно 20000, а число заявок, уничтожаемых блоком-терминатором равно 40000.

Согласно формулам Поллачека-Хинчина [2,3] для системы массового обслуживания $M/G/1$ определяются основные вероятностно-временные характеристики агрегированного канала в стационарном режиме при указанных значениях параметра распределения Эрланга k. Результаты аналитических расчетов приведены в таблице 1.

Таблица.1. Теоретические значения основных характеристик агрегированного канала.

№ п/п	Наименование Параметра	Обозначение параметра	Значение параметра		
			$k = 2$	$k = 3$	$k = 4$
1	Среднее число кадров, находящихся в очереди	$\overline{N_q}$	0,034	0,033	0,033

2	Среднее число кадров, проходящих через канал	$\overline{N}$	0,261	0,259	0,258
3	Среднее время нахождения кадра в очереди, мс	$\overline{W}$	0,841	0,829	0,822
4	Среднее время передачи кадра через канал , мс	$\overline{T}$	6,529	6,485	6,459

Имитационное моделирование.

Экранная форма модели агрегированного канала, полученная с использованием имитационной программы SIMULATION, изображена на рисунке 1.

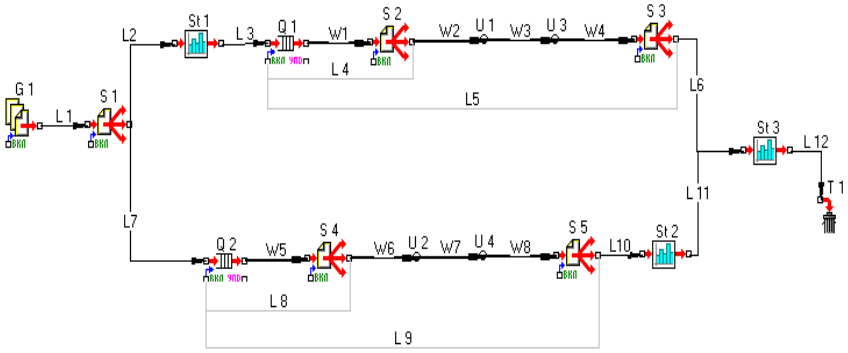


Рис.1 – Модель агрегированного канала передачи измерительной информации.

Имитационный блок-генератор заявок G1 вырабатывает заявки (кадры) с интервалом, задаваемым экспоненциальным распределением с интенсивностью $\Lambda = 0{,}04\,мс^{-1}$. С выхода блока-генератора входные кадры поступают на блок-селектор, который имитирует операции разделения и ветвления поступающих на его вход кадров. Селектор S1 делит кадр на две равные части и направляет их по дугам L2, L7 на два одинаковых по быстродействию подканала. Постоянная составляющая задержки имитируется блоком W1 (для первого подканала) и W5 (для второго подканала). Переменная составляющая задержки описывается нормированным распределением Эрланга k порядка и имитируется последовательным выполнением k раз композиции законов распреде-

ления экспоненциальных случайных величин. В модели, приведенной на рис.1, используется распределение Эрланга 3 порядка ($k = 3$). На выходе селекторов S2, S4 заявка одновременно поступает в следующий блок задержки и по дугам L4, L8 выдается сигнал, блокирующий выходы очередей Q1, Q2. Эти выходы не разблокируются до тех пор, пока заявка не пройдет через все блоки подканала. С выходов блоков-селекторов S3, S5 заявки направляется в блок-терминатор, и одновременно выдаются сигналы разблокирования выходов очередей Q1, Q2 через дуги L5, L9.

В модель включены имитационные блоки-сборщики статистики St1, St2, St3, которые строят диаграммы частот попадания случайных событий в интервалы времени по оси ординат.

Проанализирована работа трех разновидностей модели канала, соответствующих трём значениям параметра нормированного распределения Эрланга, которым описывается задержка времени передачи информации.

Таблица 2. Сравнение результатов имитационного моделирования с теоретическими значениями.

№ п/п	Наименование Параметра	Обозначение параметра	Значение параметра					
			$k = 2$		$k = 3$		$k = 4$	
			теорет	модел	теорет	модел	теорет	модел
1	Среднее число кадров, находящихся в очереди	$\overline{N_q}$	0,034	0,036	0,033	0,033	0,033	0,035
2	Среднее время нахождения кадра в очереди, мс	$\overline{W}$	0,841	0,891	0,829	0,828	0,822	0,859

На рисунках 2-4 представлены результаты имитационного моделирования, полученные с использованием программы Simulation.

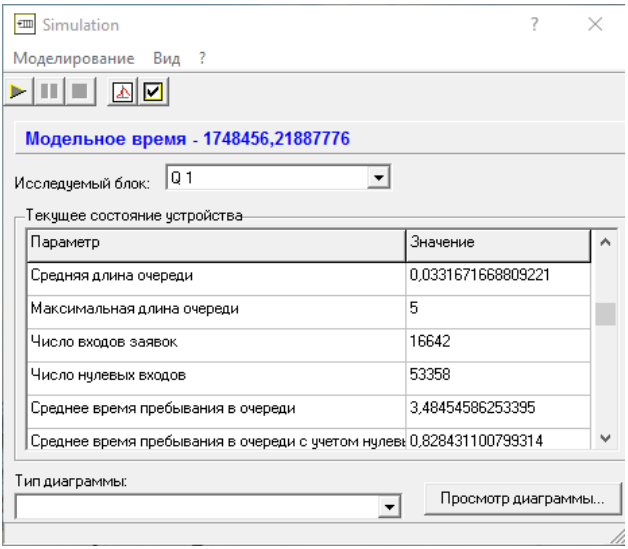


Рис.2 – Характеристики канала передачи, полученные при имитации.

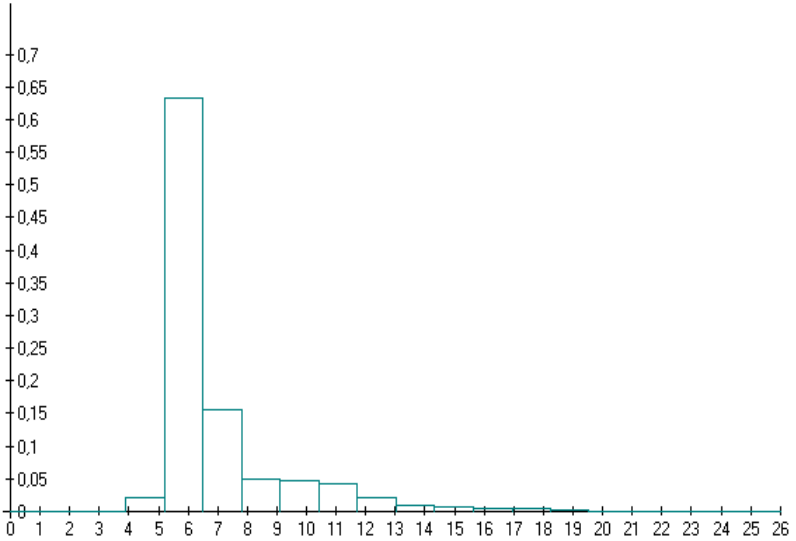


Рис.3 – Гистограмма времени нахождения заявок в подканале.

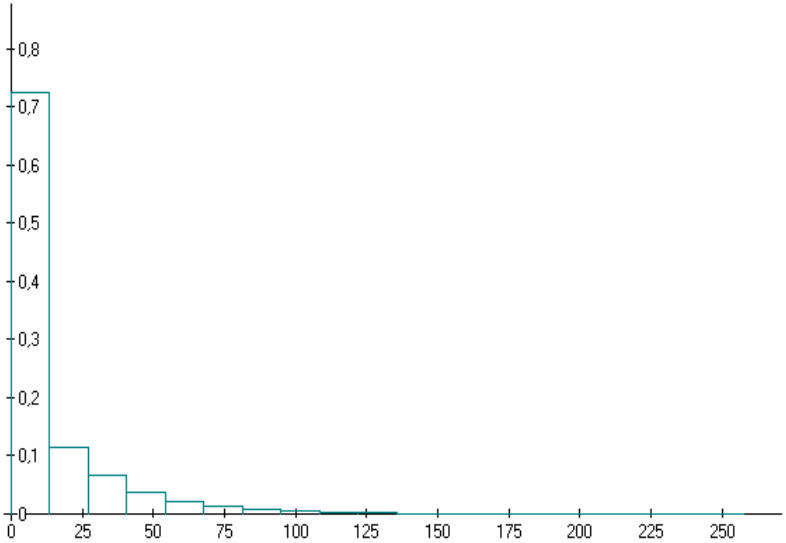


Рис.4 – Гистограмма интервалов времени между заявками на выходе системы.

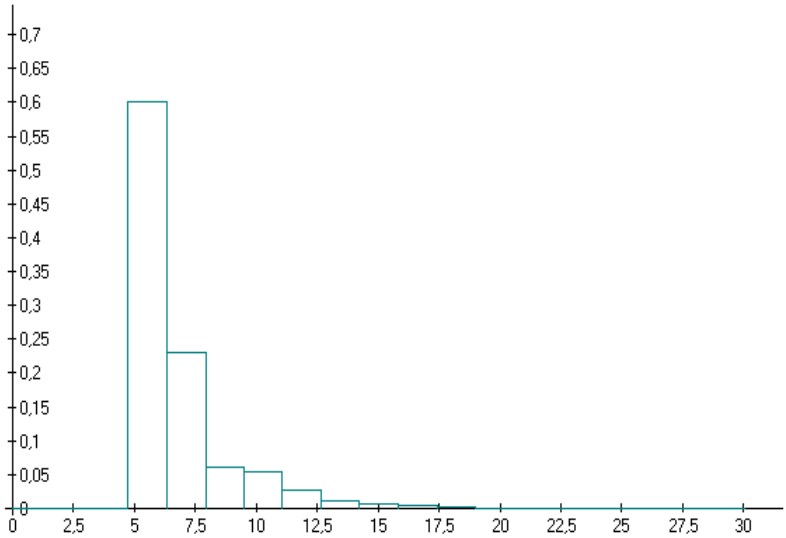


Рис.5 - Гистограмма времени достижения заявок выходного блока.

Можно отметить хорошее совпадение теоретических значений моментов распределения $\overline{X}$ и $\overline{X^2}$ времени передачи кадра по агрегированному каналу, полученных в работе [1], и результатов имитации (см. таблицу 2).

Работа поддержана Российским фондом фундаментальных исследований, грант 14-07-00106-а.

Работа проведена в рамках выполнения госзадания № 9-14Г (№ госрегистрации НИР: 115011560084).

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Корячко В.П., Шибанов А.П., Сапрыкин А.Н., Фам Х.Л. Нахождение характеристик агрегированного канала передачи измерительной информации // Вестник Рязанского радиотехнического университета. 2015. № 4. С. 72-77.

2. Клейнрок Л. Теория массового обслуживания. Пер. с англ. М.: Машиностроение. 1979. 432 с.

3. Бертсекас Д., Галлагер Р. Сети передачи данных. М.: Мир. 1989. 544 с.

А.С. ФРОЛОВ

Рязанский государственный радиотехнический университет

ИСПОЛЬЗОВАНИЕ ПОДКЛЮЧАЕМОГО ОБОРУДОВАНИЯ НА ПЛАТФОРМЕ «1С:ПРЕДПРИЯТИЕ 8.3»

В данной статье рассматриваются виды подключаемого оборудования, а также доступные возможности использования.

1С:Предприятие – программный продукт разработанный отечественной компанией «1С», предназначенный для автоматизации финансово-хозяйственной деятельности на предприятие. Фактически, 1С:Предприятие является системой автоматизированного проектирования баз данных и систем управления БД. 1С:Предприятие позволяет разрабатывать гибкие клиент-серверные приложения, ориентируясь на бизнес-процессы предприятия.

В работе реального предприятия для автоматизации всех процессов одной лишь учётной системы чаще всего оказывается недостаточно. Для комфортного и эффективного взаимодействия пользователя с системой необходимо применять различное подключаемое оборудования, обеспечивающее ввод и вывод информации.

Для использования возможностей подключаемого оборудования в разрабатываемых настольных и серверных решениях используется специализированная компонента - «1С: Библиотека подключаемого оборудования» (БПО). БПО – это набор механизмов для унифицированной работы с подключаемым оборудованием. Применение данной библиотеки позволяет ускорить реализацию поддержки широкого спектра моделей подключаемого оборудования в разрабатываемых конфигурациях, а также добавить готовые функциональные блоки в уже работающие прикладные решения.

Библиотека уже включает в себя сертифицированные драйверы для широкого спектра оборудования, а также фрагменты кода в качестве примера работы с оборудованием.

В настоящее время, БПО поддерживает работу со следующими видами оборудования:

- Сканеры штрихкода (подключаемые преимущественно посредством USB и COM-портов)
- Считыватель магнитных карт (как правило подключаются в разрыв клавиатуры)
- Терминал сбора данных (ТСД) (классическим вариантом подключения является док-станция с интрефейсом USB и дополнительно управляемым питанием для контроля заряда мобильного терминала сбора данных, существуют так же модификации позволяющие организовывать обмены данными посредством протоколов TCP\IP и методами синхронизации данных через посреднические процедуры облачных хранилищ)
- Фискальный регистратор (ФР) (данный класс устройств применяется для контроля финансовых сделок с контрагентами)
- Дисплей покупателя (в соответствии с действующим Законодательство Российской Федерации – дисплеем покупателя обязана быть оборудована каждая розничная торговая точка)
- Эквайринговый терминал (в последние годы данные устройства получили широкое примерение в практике розничной торговли и услуг – использование безналичных расчетов с физическими и юридическими лицами посредством использования систем эквайринга позволяет минимизировать или вовсе отказаться от наличного оборота платежных средств)
- Электронные весы (особенностью подключения данного типа оборудования является возможность так называемого «обтаривания» в процессе проведения измерения веса – применяется для учета неэффективного веса тары)

- Весы с печатью этикеток (основным способом подключения является Ethernet –как правило такие весы являются интеллектуальным устройством со встроенной специализированной операционной системой - позволяющей хранить в ПЗУ таблицы с информацией о видах продукции и ее характеристиках)
- Принтер чеков (отличается от фискального регистратора отсутствием фискальной памяти и для отдельных случаев применения особых режимов налогообложения так называемой электронной кассовой ленты защищенной ЭКЛЗ)

Рассмотрим каждый вид оборудования и работу с ним более подробно.

Сканер штрихкода.

Сканер штрихкода – это устройство, которое считывает штриховой код (как правило с использованием лазера или фотоэлемента со счетчиком светлых и затемненных областей изображения) и передает его значение в конфигурацию. Обычно в прикладных решениях сканеры штрихкода используются либо непосредственно в формах списка справочников для поиска элемента справочника по штриховому коду и позиционирования на найденном элементе, либо в документах при подборе элементов справочника в табличную часть.

Удачными примерами использования являются следующие варианты:

- поиск номенклатуры в справочнике;
- подбор номенклатуры в документы товародвижения;
- заполнение поля серия по считанному штриховому коду серии;
- поиск документа по штриховому коду.

Штриховые коды на плоскости могут быть одномерными и двухмерными. Существует ряд различных стандартизованных форматов (таких как EAN одномерный, QR и PDF-417 двухмерные и прочие). Различные модели сканеров поддерживают определенные форматы штриховых кодов.

Для работы со сканером ШК оборудование прежде всего необходимо инициализировать (обычно это делается при открытии формы) при помощи процедуры *ПодключитьОборудованиеПоТипу()* общего модуля *МенеджерОборудованияКлиент*. Входным параметром данной процедуры является тип подключаемого оборудования (в данном случае *«СканерШтрихкода»*).

Для обработки полученных от сканера данных используется стандартный механизм формы – обработка оповещения. Обработка оповещения при возникновении события вызывает определённую про-

цедуру из модуля формы. Входными параметрами данной процедуры являются *ИмяСобытия, Параметр* и *Источник.* В случае сканера ШК источником события будет являться *«ПодключаемоеОборудование»,* имя события – *«ScanData»,* а в качестве параметра будет передаваться массив полученных значений.

При закрытии формы, оборудование необходимо отключить при помощи процедуры *ОтключитьОборудованиеПоТипу()* общего модуля *МенеджерОборудованияКлиент.*

Считыватель магнитных карт

Считыватель магнитных карт – это устройство, которое считывает магнитный код с карты и передает его значение в конфигурацию посредством драйвера (для случая подключения способами отличными от разрыва клавиатуры). Карты с магнитным кодом обычно используются в качестве карт лояльности покупателей или служебных карт сотрудников. На магнитной карте обычно используется две или три дорожки. Используемая часть кода ограничивается префиксом и постфиксом, которые предварительно вносятся в систему пользователем. Состав параметров настройки может изменяться в зависимости от драйвера. Работа с оборудованием на форме, также как и работа со сканером штрихкодов, производится в три этапа:

- Инициализация оборудования (*ПодключитьОборудованиеПоТипу()* с типом *«СчитывательМагнитныхКарт»*)
- Получение и обработка данных (процедура *ОбработкаОповещения()* для события *«TracksData»*)
- Отключение оборудования (*ОтключитьОборудованиеПоТипу()*)

Терминал сбора данных

Терминал сбора данных – это мобильный компьютер с процессором и встроенной памятью, оснащенный сканером штрихкодов. Успешным примером использования ТСД является оперативный сбор информации о товарах, при формировании отгрузки товаров и для решения множества других учётных задач. Данные считываются с помощью встроенного в терминал сканера штрихкодов, обрабатываются и сохраняются в памяти терминала, а затем выгружаются в компьютер через USB-соединение, ИК-порт, Bluetooth или другой интерфейс передачи данных. Часто данные передаются без буфера терминала – например, непосредственно в табличную часть документа.

Поддерживается выгрузка в терминал данных о товаре и загрузка данных о товаре и его количестве для формирования документов.

Перед запуском процедуры выгрузки в конфигурации подготавливается массив данных определенного типа. Полученная таблица с

помощью функции библиотеки подключаемого оборудования (*НачатьВыгрузкуДанныеВТСД()* модуля *МенеджерОборудованияКлиент*) преобразуется в формат .xml и методом драйвера передается в терминал сбора данных.

Загрузка данных из терминала производится в обратном порядке: метод драйвера оборудования (*НачатьЗагрузкуДанныеВТСД()* модуля *МенеджерОборудованияКлиент*) передает данные в формате xml, которые преобразуются функцией библиотеки в соответствующий данным тип. Полученный массив данных обрабатывается в модуле формы в соответствии с запрограммированной логикой прикладного решения.

После загрузки, данные из терминала можно удалить при помощи процедуры *НачатьОчисткуДанныеВТСД()* общего модуля *МенеджерОборудованияКлиент()*

Дисплей покупателя

Дисплей покупателя представляет собой электронное табло, на котором, как правило, отображается информация о покупке (промежуточный итог, сумма чека, сумма сдачи, наименование товара и т. д.)

При использовании дисплея покупателя используются следующие команды:

- Подключение устройств
- Очистка экрана
- Вывод текста
- Отключение устройств

При выводе текста на дисплей покупателя в параметрах команды можно указать, в какой области экрана будет отображаться данная информация.

Подключение и отключение устройства происходит аналогично описанным выше.

Для очистки дисплея используется процедура *НачатьОчисткуДисплеяПокупателя()* общего модуля *МенеджерОборудованияКлиент,* которой в качестве параметра передаётся уникальный идентификатор оборудования.

Для вывода текста на экран используется процедура *НачатьВыводИнформацииНаДисплейПокупателя()* общего модуля *МенеджерОборудованияКлиент.* В качестве параметра данная процедура принимает строковое значение, которое необходимо вывести, а также идентификатор оборудования.

При завершении работы с дисплеем покупателя, его необходимо отключить при помощи процедуры *НачатьОтключениеОборудованияПоТипу*общего модуля *Менедже-*

рОборудованияКлиент, принимающей в качестве параметра уникальный идентификатор данного экземпляра оборудования.

Фискальный регистратор

Фискальный регистратор (ФР) – это устройство для регистрации торговых операций и печати чеков с фискальным признаком (зачастую отпечатком ЭКЛЗ на специальных налоговых режимах). Данные для печати чеков и записи в фискальную память программно передаются в устройство посредством соединения по USB, COM-портом. Кроме того, в фискальной памяти ФР фиксируются операции внесения/выемки денежный средств и выводится на печать соответствующие чеки, так же поддерживаются отчеты смены – открытия и закрытия, отчеты фискальной памяти и отчеты ЭКЛЗ в оборудованных ей моделях.

Отчет с гашением (Z-отчет) формируется при закрытии кассовой смены как правило старшим кассиром облеченным материальной ответственностью. Отчет без гашения (X-отчет) распечатывается по команде кассира-операциониста и не влечет изменения состояния фискальной памяти регистратора. Возможность печати произвольного текста используется при печати дополнительных отчетов или слип-чеков.

Для печати чека устройству необходимо передать массив, содержащий три таблицы: таблица товаров, таблица оплат и таблица общих параметров. Если необходимо оформить чек на продажу, то в параметре "Тип чека" указывается значение "0", а если возврат - "1". Так же можно осуществить настройку распределения оборотов по секциям фискального регистратора: к примеру, обороты денежных средств, поступивших/выданных по документам различного типа могут быть отнесены к разным отделам. Выручка по отдельным видам номенклатуры, таким как подарочные сертификаты также может быть отнесена к определенному отделу.

Настройка секции фискального регистратора осуществляется для каждой строки табличной части формируемого документа.

После подготовки входных параметров можно вызвать процедуры печати чека. В процедуре оповещения о завершении выполнения процедуры возвращается результат выполнения. Для печати чека используется процедура *НачатьПечатьЧека()*, принимающая в качестве входных параметров уникальный идентификатор оборудования, а также подготовленный массив данных.

Разработчику прикладного решения необходимо интерпретировать выходные данные в зависимости от того, удалась ли печать чека, или нет.

Для печати фискальных отчетов используется процедура *НачатьПечатьФискальногоОтчета()*, в качестве входных параметров принимающая идентификатор оборудования, а также тип операции (*Истина* – если отчёт с гашением и *ложь* в обратном случае).

Для регистрации фактов инкассации (внесения/выемки денежных средств) в фискальной памяти регистратора в библиотеке предусмотрена процедура *НачатьИнкассациюНаФискальномУстройстве(),* принимающая в качестве входных параметров идентификатор устройства, а также тип и сумму операции.

Также среди доступного функционала следует выделить печать произвольного текста (*НачатьПечатьТекста()*) и открытие денежного ящика используемого для хранения наличных денежных средств (*НачатьОткрытиеДенежногоЯщика()*)

Принтер чеков

Принтер чеков (ПЧ) – это устройство для печати чеков и информационных документов произвольного содержания (применяется широко на специальных налоговых режимах – например на режиме «единый налог на вмененный доход» с определенными видами деятельности – устанавливаемыми муниципальными властями). Важно отметить: документ распечатанный на ПЧ не является фискальным документом.

Для работы с принтером чеков в библиотеке подключаемого оборудования предусмотрены следующие процедуры, аналогичные процедурам для работы с ФР:

- *НачатьПечатьЧека()*
- *НачатьПечатьТекста()*
- *ОткрытиеДенежногоЯщика()*

Эквайринговый терминал

Эквайринговый терминал (ЭТ) – электронное устройство, позволяющее считывать информацию с магнитной полосы или чипа карты, подключенное по каналам связи к процессинговому центру и предназначенное для автоматизированного совершения безналичных денежных операций. В зависимости от аппаратных возможностей эквайрингового терминала печать подтверждающего оплату слип-чека производится на самом терминале или с помощью печатающего устройства (фискального регистратора, или принтера чеков).

Выполнение операций оплате по карте, отмена оплаты по карте, возврат оплаты по карте осуществляется одной процедурой - *НачатьВыполнениеОперацииНаЭквайринговомТерминале()*.

На вход процедуры *НачатьВыполнениеОперацииНаЭквайринговомТерминале()* подаются следующие параметры:

- *ИдентификаторУстройстваЭТ* - Идентификатор эквайрингового терминала на котором будет производится операция.
- *ИдентификаторУстройстваПУ* - Идентификатор эквайрингового печатающего устройства на котором предполагается печать слип-чека.
- *ПараметрыОперации* - Параметры проводимой операции.
  - *ТипОперации* - Тип выполняемой операции входящие и исходящие параметры;
  - *СуммаОперации* - Сумма операции;
  - *НомерЧека* - Номер чека по которому происходит оплата;
  - *ССылочныйНомер* - Ссылочный номер операции (код RRN (RefeRenceNumber) транзакции);
  - *КодАвторизации* -Код авторизации выполненной операции;
  - *НомерКарты* - Маскированный номер карты, по которой происходила операция;
  - *ТекстСлипЧека* - Текст слип-чека.

Электронные весы

Электронные весы предназначены для взвешивания товара и передачи в компьютер полученного веса.

Для получение веса необходимо вызвать *процедуру НачатьПолученияВесаСЭлектронныхВесов()* подсистемы. Вес будет возвращен в процедуру оповещения о завершении взвешивания.

Весы с печатью этикеток

Весы с печатью этикеток широко применяются на производствах, в лабораторной практике и в сфере торговли для взвешивания и этикетирования различной продукции. Информация, необходимая для печати на этикетках, передается в весы из базы данных 1С:Предприятия. Помимо методов *Подключение* и *Отключение* весов предусмотрено еще два метода для работы с данными: *Очистить базу* и *Выгрузить товары*.

При выполнении команды *Очистить базу* все переданные ранее данные удаляются из памяти весов.

Поддерживается режимы полной и частичной выгрузки товаров. Синхронизация товаров в весах при частичной выгрузке производится по PLU (сокращение от англ. Price-LookUp) товара.

Выгрузка товаров в весы выполняется при помощи функции *ВыполнитьКомандуОборудования()* с параметром *ВыгрузитьТовары*. В качестве входного параметра передаётся идентификатор клиента (формы), а также массив, элементами которого являются структуры определённого формата:

- *PLU*, число;

- *Код*, число;
- *Номенклатура*, строка;
- *НаименованиеПолное*, строка;
- *Цена*, число;
- *ОписаниеТовара*, строка;
- *СрокХранения*, число;
- *ВесовойТовар*, булево.

Признак частичной выгрузки устанавливается, если необходимо обновить существующие номенклатурные позиции в памяти весов и добавить новые. Поле *PLU* является номером товара на весах, на которые происходит выгрузка. Поле *Код* является идентификатором товарной позиции, единицей учёта запасов, используемый в торговле для отслеживания статистики по реализованным товарам. Поле *Код* является числовым кодом товара, предназначенным для формирования штрихового кода для печати на этикетки. Данный код используется на кассе ККМ для идентификации товара при считывании сформированного на весах штрихкода. Поле *ВесовойТовар* - определяет выгружается весовой товар или штучный товар фасуемый на весах. Если установить значение признака частичной выгрузки "*Ложь*", то перед загрузкой данных все товары будут удалены из памяти весов.

Заключение

В данной статье описаны возможности взаимодействия платформы 1С:Предприятие 8.3 с подключаемым оборудованием, виды такого оборудования и особенности реализации взаимодействия с каждым из них в прикладных решениях для целей применения в различных сферах производства, торговли и делопроизводства.

Подводя итоги, можно сказать, что данная отечественная платформа является современным и динамично развивающимся средством разработки, реализующим комфортное взаимодействие пользователя и системы. Важным преимуществом платформы 1С является то что она разрабатывается и поддерживается нашими соотечественниками и коллегами – благодаря чему новые функции и потребности не только конечных пользователей, но и программистов работающих с данным решением – удовлетворяются в сжатые сроки и с максимальным уровнем качества – чему обязана в первую очередь исторически сильной школой программирования в нашей стране.

Платформа 1С поддерживает все современные виды подключаемого оборудования, необходимые для ведения финансово-хозяйственной деятельности предприятий любого профиля начиная сложнейшими наукоемкими производствами и заканчивая услугами и торговлей.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. «Применение функционала подсистемы БПО в конфигурациях» - входит в состав БПО
2. «Методическая поддержка для разработчиков и администраторов 1С:Предприятия 8» - Инженерно-технологическое сопровождение пользователей 1С:Предприятие (http://its.1c.ru)

П.В. ШЕЛЕХИН, П.Ю. ЧЕСАЛИН
Рязанский государственный радиотехнический университет

ПЕРСПЕКТИВЫ РАЗВИТИЯ СОВРЕМЕННОГО ИСКУССТВЕННОГО ИНТЕЛЛЕКТА И ИНТЕЛЛЕКТУАЛЬНЫХ СИСТЕМ

Рассматривается история, темпы развития современного искусственного интеллекта и интеллектуальных систем, их практическое использование в ближайшем будущем.

Искусственный интеллект (ИИ) — наука и технология создания интеллектуальных машин, особенно интеллектуальных компьютерных программ.

В последнее время ИИ делает серьёзные шаги развития как в аппаратном, так и в программном плане. Исторически, зарождение ИИ началось уже в середине XX века.

Список широко известных систем на ИИ:

1. IBM Watson – суперкомпьютер IBM, который может понимать вопросы на естественном языке. При его проектировании применялся нисходящий подход (Top-Down AI). Широко используется в медицинских целях для диагностирования болезней. Также создатели IBM Watson рассматривают возможность его работы в других сферах.

2. Deep Blue – самый известный шахматный суперкомпьютер, в 1997 году обыгравший Гарри Каспарова. Стоит заметить, что это не первая система такого плана, однако именно она добилась в этом плане существенных успехов.

Отдельного внимания заслуживает AlphaGo – программа, разработанная Google DeepMind для игры в ГО. Ранее, в 2015 году ею был обыгран чемпион Европы – Фань Хуэй. Однако, «историческим» матчем можно назвать матч против чемпиона Кореи – Ли Седоля – Матч закончился со счётом 4:1 в пользу AlphaGo. Команда DeepMind собирается в ближайшем будущем также обучить свою систему игре в покер и StarCraft. (StarCraft отличается от остальных современных игр, где нужно принимать мгновенные решения основываясь на ситуации

тем, что в нём практически отсутствует влияние случайностей. Единственным таким проявлением является выбор первоначальной позиции игрока на карте и расы, если игрок не выбрал её при старте игры)

Обучение AlphaGo производилось в несколько этапов – на первом в нейросеть была загружена большая база игр. На втором нейросеть играла против людей по сети. На следующем AlphaGo должна была проводить партии против себя.

ГО – логическая настольная игра, входящая в число пятибазовых дисциплин всемирных интеллектуальных игр. Традиционно считалась сложной для компьютера по ряду причин:

1. Большая доска – 19x19.

2. Большое количество допустимых ходов. Число возможных комбинаций примерно равно $9.8*10^{169}$ По этой же причине невозможно заранее просчитать все варианты развития событий на доске.

3. Отсутствие структурированной дебютной теории. В отличие от шахмат, в ГО «новинки» могут встречаться не на 20 ходу, а уже на четвёртом или пятом. Грамотная игра не возможна без понимания стратегических конструкций.

4. По мере увеличения числа ходов количество фигур на доске не уменьшается, а увеличивается, что затрудняет просчёт ходов.

Поэтому написание такой программы стало бы серьёзным толчком в развитии искусственного интеллекта. Первые серьёзные попытки написать программу, способную победить чемпионов были уже в 2005, но из-за несовершенства как программной, так и аппаратной части попытки были безуспешными.

Также, свидетельством бурного развития ИИ является факт того, что Тест Тьюринга уже является пройденным «Женей Густсманом» в 2014 году в университете Рединга (Великобритания). Но сам по себе прохождение теста Тьюринга не обязательно свидетельствует о достижении первоначальной задумки Тьюринга из-за несовершенства критериев. Многие уверены, что пройдя тест Тьюринга, мы создадим не ИИ, способную размышлять, а машину для обмана судей. Стоит отметить, что системы общения с интеллектуальной системой уже давно существовали, например, Cleverbot.

Другим же направлением развития ИИ является восходящий подход – моделирование нейронных сетей. С развитием вычислительных мощностей удаётся имитировать мозг всё более сложных существ. В 2010 году в рамках проекта «Blue Brain Project» был смоделирована в реальном времени вся нервная система дождевого червя. В предыдущем году был полностью смоделирован мозг крысы на одном суперкомпьютере, состоящем из 8192 процессоров, которые имитирова-

ли функционирование 10 тыс. нейронов. В этом году исследователям также удалось смоделировать работу кусочка неокортекса крысы, объем ткани которого составляет треть кубического миллиметра, число нейронов достигает 30 тысяч, а количество синапсов — мест контакта между двумя нейронами — превышает 40 млн.

Существуют также проекты, исходные коды которых находятся в открытом доступе. Например, OpenWorm – аналогичен «Blue Brain Project», занимается моделированием червя, его поведения, работы нервной системы.

Сочетанием интеллектуальной системы распознавания речи и интеллектуальной информационной системы являются персональные помощники – Siri – от корпорации Apple, Microsoft Cortana, Google Now. Все они достаточно точно могут определить голос пользователя, приспособится к его акценту или даже научится распознавать почерк. Их недостатком же является то, что все они проприетарные и очень сильно зависят от аппаратной части. В часть из них также интегрирована интеллектуальная система распознавания изображений.

Аппаратная часть также развивается достаточно быстро. Недавно корпорация NVidia на конференции GPU Technology Conference представила целый комплекс, направленный на Искусственный Интеллект. На ней был продемонстрирован модуль Tesla P100 на совершенно новой архитектуре Pascal. Также был представлен суперкомпьютер DGX-1, доступный для предзаказа уже сейчас. Суперкомпьютер состоит из восьми модулей P100 и имеет производительность 170 терафлопс. DGX-1 будет специально спроектирован для глубокого обучения нейронных сетей и работы с ними. У DGX-1 будет расширенная поддержка виртуальной реальности.

Как можно заметить, темпы развития интеллектуальных систем сильно возросли за последнее десятилетие. В ближайшем будущем нас ожидают DGX-1, улучшенные системы распознавания речи и изображений. В исследовательских целях уже происходит развёртывание белков, основанное на генетических алгоритмах. Беспилотные системы интеллектуального управления автомобилями уже проходят тестирование на дорогах и показали свою состоятельность.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. The AI Revolution: The Road to Superintelligence - Tim Urban

2. Artificial_intelligence [электронный ресурс] https://en.wikipedia.org/wiki/Artificial_intelligence

3. Neuroscience [электронный ресурс] http://bluebrain.epfl.ch-/page-125407-en.html

4. DEEP LEARNING [электронный ресурс] http://www.nvidia.com/object/deep-learning-system.html

5. Google - AlphaGO [электронный ресурс] https://deepmind.com-/alpha-go.html

А.П. ШИБАНОВ, М.В. ГОЛЬД

Рязанский государственный радиотехнический университет

АНАЛИЗ ХАРАКТЕРИСТИК СЕТЕВЫХ МОДЕЛЕЙ ПРИ СРЕДНИХ И БОЛЬШИХ НАГРУЗКАХ В СИСТЕМЕ

Рассматривается задача анализа характеристик сетевых моделей, построенных в среде имитационного моделирования SIMULATION, при средних и больших нагрузках в системе.

Рассмотрим сетевую модель, состоящую из трех генераторов заявок (G1, G2 и G3), очереди (Q1), обслуживающего устройства (дуги W1 и W2), а также переключателей для выполнения функций коммутации входов блокировки (S1 и S2). Схема модели представлена на рисунке 1.

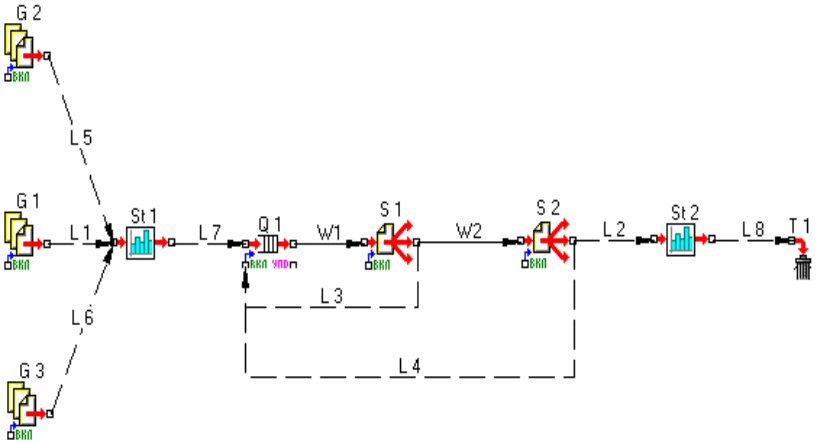


Рис. 1 – Схема сетевой модели.

Блоки модели имеют следующие характеристики:

- G1 генерирует 2000 заявок с равномерным интервалом на отрезке [0; 4];
- G2 генерирует 3000 заявок с равномерным интервалом на отрезке [0; 2];

- G3 генерирует 1000 заявок с равномерным интервалом на отрезке [0; 6];
- Q1 имеет емкость в 500 заявок, а их обслуживание происходит в соответствии с процедурой «первый вошел – первый вышел»;
- S1 и S2 посылают заявки на все выходы одновременно;
- W1 и W2 производят задержку заявки согласно экспоненциальному закону распределения с параметром $\lambda = 5$.

Роль селекторов S1 и S2 заключается в следующем. Когда заявка поступает на селектор S1, он одновременно посылает сигнал на блокирование очереди Q1 и передает заявку по дуге W2, имеющей указанную выше задержку. После того, как передача («обслуживание») заявки заканчивается, она поступает на селектор S2, который снимает блокировку с очереди Q1 и передает заявку дальше по сети.

Блоки St1 и St2 предназначены для сбора статистической информации и никаким образом не влияют на процесс передачи заявок.

В качестве анализируемых характеристик будем рассматривать интервалы между поступающими заявками, а также время нахождения заявки в системе.

Рассмотрим гистограмму интервалов между заявками, полученную на блоке St1 в результате моделирования (рисунок 2).

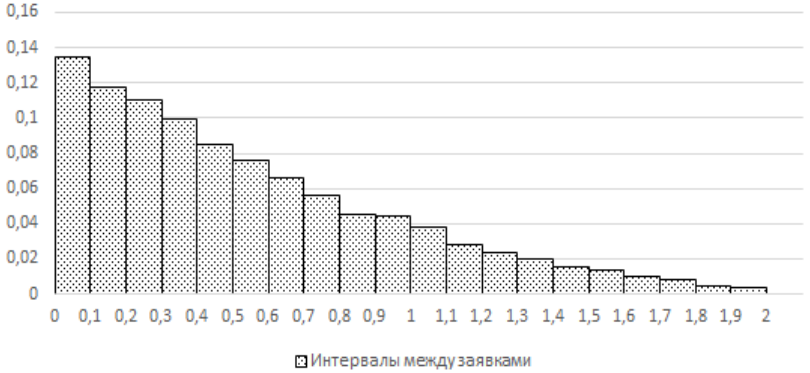


Рис. 2 – Гистограмма интервалов между заявками (блок St1).

Как видно из рисунка 2, характер гистограммы похож на график плотности вероятностей экспоненциального закона распределения:

$$f(x) = \lambda \cdot e^{-\lambda x}. \qquad (1)$$

Для того, чтобы проверить правдоподобность этой гипотезы, воспользуемся критериями согласия. Наиболее часто применяемым

критерием является критерий χ^2 Пирсона. Данный критерий дает возможность оценить расхождение между гипотетическим распределением и статистическим [1]. Расчетная формула критерия равна:

$$\chi^2 = \sum_{i=1}^{k} \frac{(m_i - m_i^*)^2}{m_i}, \quad (2)$$

где m_i и m_i^* – соответственно теоретические и эмпирические частоты рассматриваемого распределения, k – число экспериментальных значений.

Так как интеграл от плотности распределения по всему пространству равен единице, то для корректного сравнения распределений необходимо полученную в результате моделирования гистограмму масштабировать таким образом, чтобы сумма площадей всех ее прямоугольников также равнялась единице. Для этого определим текущую сумму площадей

$$S_{сум} = \sum_{i=1}^{k} \Delta x \cdot f(x_i) = 0{,}1, \quad (3)$$

а затем поделим каждое из экспериментальных значений $f(x_i)$ на эту величину. Сравним полученные значения и значения плотности экспоненциального распределения (1) при $\lambda = 1{,}5$ (рисунок 3).

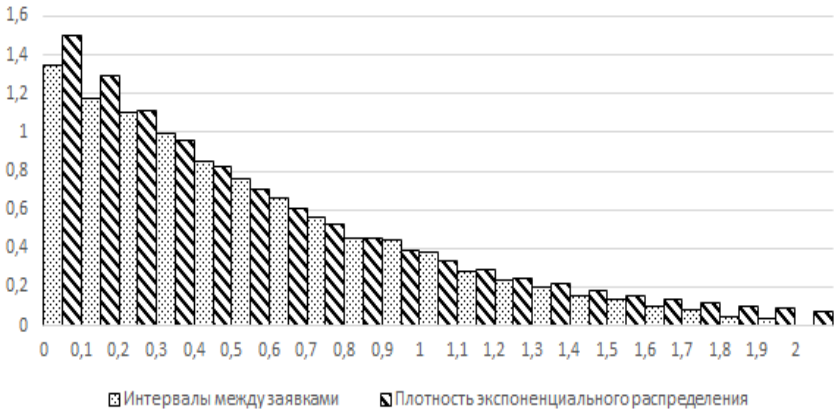


Рис. 3 – Сравнение теоретических и эмпирических значений интервалов между заявками на блоке St1.

Критерий Пирсона будет равен:

$$\chi^2 = \sum_{i=1}^{k} \frac{(f_{теор}(x_i) - f_{эмп}(x_i))^2}{f_{теор}(x_i)} \approx 0{,}22 \cdot \quad (4)$$

Для распределения χ^2 составлены таблицы. Пользуясь ими, можно для каждого значения χ^2 и числа степеней свободы *r* (определяется, как *k* минус число независимых условий) найти вероятность *p* того, что величина, распределенная по закону χ^2, превзойдет это значение [1].

Для $r = k - 0 = 21$ значения *p* и χ^2 представлены в таблице 1.

Таблица 1. Значения χ^2 в зависимости от *r* и *p*.

	P								
r	**0,95**	**0,9**	**0,8**	**0,7**	**0,5**	**0,3**	**0,2**	**0,1**	**0,05**
21	11,59	13,24	15,44	17,18	20,3	23,9	26,2	29,6	32,7

По данной таблице можно судить, что для $\chi^2 = 0{,}22$ вероятность *p* очень близка к 1. Однако это не свидетельствует о большом правдоподобии гипотезы. Это может говорить о том, что число опытов *k* недостаточно велико, чтобы распределение величины стало близко к χ^2 [1]. Тем не менее исходя из этого можно сделать вывод, что гипотеза не противоречит опытным данным.

Данная гипотеза также применима к данным, сформированным на блоке St2. Гистограммы для масштабированных значений интервалов между заявками и значений плотности экспоненциального распределения (1) при $\lambda = 1{,}75$ представлены на рисунке 4.

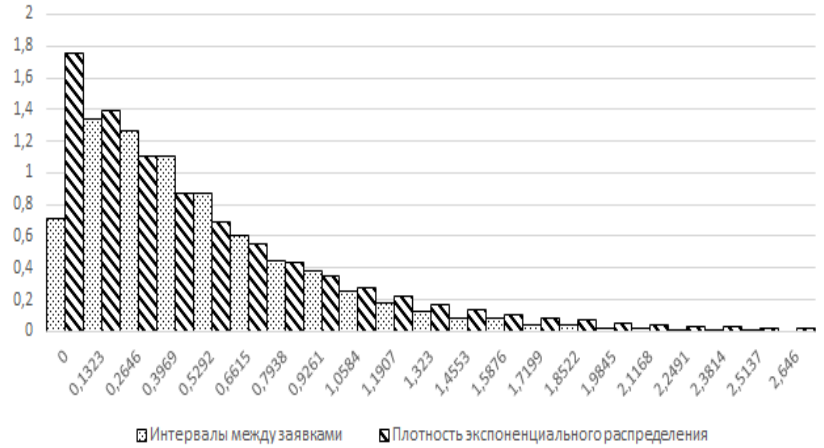


Рис. 4 – Сравнение теоретических и эмпирических значений интервалов между заявками на блоке St2.

Критерий Пирсона будет равен $\chi^2 \approx 0{,}97$.

Гистограммы для масштабированных значений времени нахождения заявок и значений плотности экспоненциального распределения (1) при $\lambda = 1{,}1$ представлены на рисунке 5.

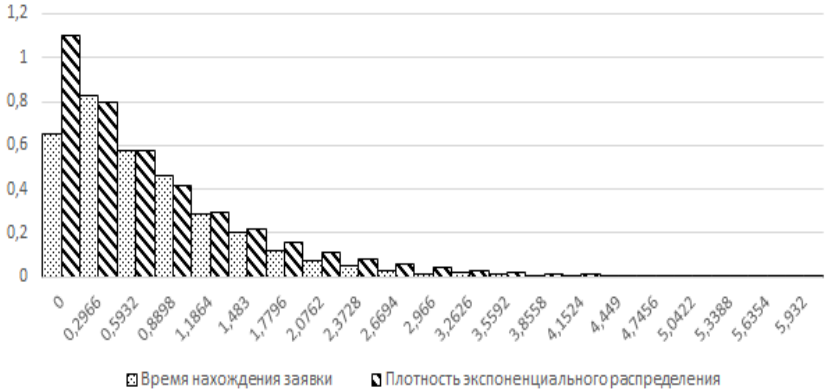


Рис. 5 – Сравнение теоретических и эмпирических значений времени нахождения заявки на блоке St2.

Критерий Пирсона будет равен $\chi^2 \approx 0{,}27$.

Заключение

Значения характеристик рассматриваемой сетевой модели (интервалы между заявками, время нахождения заявки) при средних и больших нагрузках в системе могут рассматриваться, как случайные величины, описанные с помощью законов распределения.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Вентцель Е.С., Овчаров Л.А. Теория случайных процессов и ее инженерные приложения. – Учеб. Пособие для втузов. – 2-е изд., стер. – М.: Высш. шк., 2000. – 383 с.

А.П. ШИБАНОВ, А.Н. САПРЫКИН,
А.С. ТАРАСОВ, А.В. ТОКАРЕВ
Рязанский государственный радиотехнический университет

СИСТЕМА ОТОБРАЖЕНИЯ ТРАФИКА РЕАЛЬНОГО ВРЕМЕНИ ПОЛИГОННОГО ИЗМЕРИТЕЛЬНОГО КОМПЛЕКСА

В статье рассматривается создание визуально ориентированной системы оптимального планирования трафика в сети полигонного измерительного комплекса при проведении летных испытаний сложных технических изделий.

Ключевые слова: балансировка сетевого трафика; моделирование загрузки каналов, визуально-ориентированный интерфейс.

При проведении испытаний множества летательных аппаратов (ЛА) должно осуществляться слежение за параметрами их траекторий и за состоянием бортовых технических систем. Эту функцию выполняют полигонные измерительные комплексы, которые передают траекторную и телеметрическую информацию от своих наземных средств в центры обработки информации и управления ходом полета. На сеть передачи данных полигонного измерительного комплекса (ПИК) возлагается задача передачи трафика реального времени в условиях перемещения нескольких ЛА. При этом одни средства измерений теряют связь с объектами испытаний, а другие подключаются к сопровождению тех ЛА, которые попадают в зоны их «видимости». Процессу сопровождения многих ЛА свойственна значительная динамика. Это приводит к резким изменениям объемов трафика в каналах (звеньях) сети ПИК. Поэтому возникает актуальная задача для системы управления сетью, которая заключается в оперативном планировании полос пропускания каналов и путей из конца в конец при возникновении пульсаций трафика. В тех случаях, когда траектория ЛА может быть рассчитана заранее, либо изменения в матрице трафика могут быть пересчитаны в реальном времени, возможно такое планирование полос пропускания, которое обеспечит передачу по сети ПИК, по крайней мере, наиболее важных кадров измерительной информации. Для составления такого плана необходимо знать: 1) распределение времени передачи кадра «из конца в конец» (в мс) для каждой пары абонентов и 2) распределение совокупной полосы пропускания (в Мбит/с) любого физического канала. Математическое обоснование решения этих вопросов дано в работе [1]. Для построения оптимального плана загрузки каналов сети ПИК используются генетические алгоритмы.

При подготовке и проведении испытаний имеет значение применение современных средств демонстрации. Для этой цели удобно использовать помещения больших объемов с демонстрационными табло, на которые выдаются цветные копии изображений с экранов компьютеров. Инженеры-испытатели, имеющие возможности визуального контроля трафика в каналах могут изучать закономерности изменения трафика в отдельных географических зонах; предсказывать возможность возникновения перегрузок в отдельных местах сети; визуально фиксировать нарушения в логике работы алгоритмов и т. д.

Важна и реализация функции демонстрации общего хода испытаний. Это необходимо как инженерам-исследователям, так и для наблюдения за ходом испытаний представителям заказчиков, исполните-

лям проекта и другому заинтересованному персоналу. Визуальные схемы контроля трафика сети ПИК могут состыковываться со смежными системами, например, с программами рисовки траекторий ЛА, с программами контроля и выдачи на экран остающихся запасов топлива в ЛА и т. п.

Визуальное восприятие процесса испытаний улучшается, если он отображается на фоне географической карты местности.

При отображении состояния каналов целесообразно отражать их состояние, с одной стороны, цветовой гаммой, в которой степень загрузки канала отображается интенсивностью окраски. Например, последовательному возрастанию трафика соответствует изменение цветов: зеленый, светло-желтый, темно-желтый, красный. С другой стороны, к изображению канала должны привязываться минимально необходимые числовые характеристики.

Ниже рассматривается программный комплекс, включающий в себя систему оптимального планирования трафика сети ПИК, с возможностью визуального отражения состояния каналов с привязкой к географической карте местности (в рассматриваемых примерах даются карты центрального региона РФ).

Комплекс предоставляет возможность записать готовую схему сети в файл (рис. 1). Для этого в меню программы имеется пункт «Сохранить как...». Для загрузки готовой схемы имеется пункт меню «Открыть…».

Рис. 1 - Пример спроектированной схемы.

Возможно также создание новой схемы без сохранения изменений. Для этого предусмотрена команда «Создать» (рис. 2).

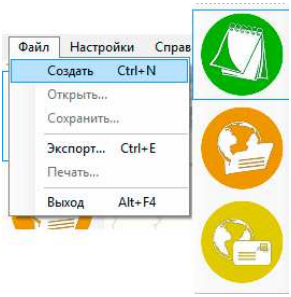


Рис. 2 - Команды для работы со схемой.

При работе со схемой сети имеется возможность сохранять область редактирования. Для этого предусмотрена возможность экспорта в формате PNG, а также вывод на печать.

Для представления сетей на реальной карте местности в программе предусмотрена возможность установки фона.

Выбор карты осуществляется с помощью команды меню «Настройки» - «Выбор карты» - «Открыть…». В том случае, если карта не умещается на форме, имеется возможность выбора режима размещения.

Вместо изображения карты можно установить фоновый цвет с помощью команды меню «Настройки» - «Выбор карты» - «Выбрать цвет…» (рис. 3).

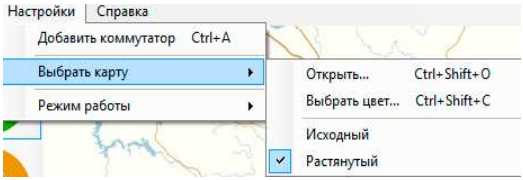


Рис. 3 - Выбор фона.

Следует учитывать, что не все типы каналов обладают одинаковыми характеристиками и потому в системе предусмотрена возможность выбора скорости и вида каналов.

Загруженность канала отражена в цвете тонкой пунктирной линии, рядом с основным отрезком.

В программе предусмотрено три режима редактирования (рис. 4).

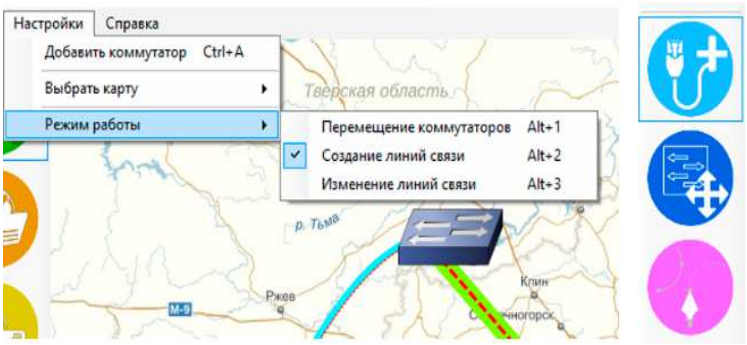


Рис. 4. Режимы редактирования карты.

Режим «Перемещение коммутаторов» позволяет изменять положение ключевых узлов на схеме. При этом возможность редактирования линий заблокирована.

Режим «Создание линий связи» предоставляет возможность редактирования каналов на схеме.

Режим «Изменение линий связи» делает активными контрольные точки кривых Безье у линий связи и предоставляет возможность менять кривизну отрезков.

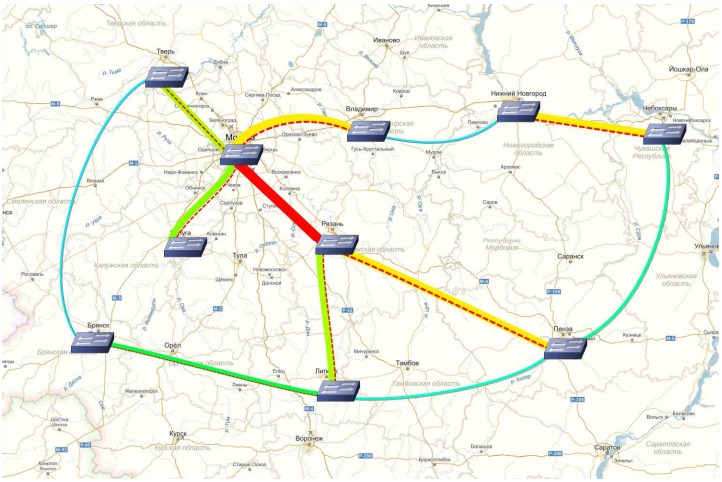


Рис. 5 - Компьютерная сеть на карте ЦФО.

Данный программный комплекс позволяет оперативно перераспределять загрузку каналов в соответствии с изменениями матриц

трафика в процессе испытаний ЛА с целью предотвращения потерь пакетов из-за возможного переполнения буферов в коммуникационных устройствах сети (в коммутаторах и маршрутизаторах). Балансировка сети по матрицам трафика может осуществляться как предварительно (в предположении, что не будет отклонений от запланированных траекторий полета), так и в реальном времени с учетом нештатных ситуаций в процессе испытаний, например, с учетом отклонений ЛА от заданной трассы. Комплекс предназначен для балансировки трафика, как в пределах локальных сетей измерительных пунктов, так и для региональных сетей (рис. 5), расположенных вдоль трасс полета.

Работа поддержана Российским фондом фундаментальных исследований, грант 14-07-00106-а.

Работа проведена в рамках выполнения госзадания № 9-14Г (№ госрегистрации НИР: 115011560084).

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. В.П. Корячко, А.П. Шибанов, О.В. Лукьянов. Характеристики сети передачи данных для проведения летных испытаний изделий // Вестник РГРТУ. 2015. № 4 (выпуск 54). Часть I. С.98-103.

А.И. ШМАКОВА
Рязанский государственный радиотехнический университет

МЕХАНИЗМЫ СОВМЕСТИМОСТИ IPV4 И IPV6

Рассматриваются механизмы совместимости протокола IPv4 и IPv6, а также реализация механизма туннелирования 6to4.

В настоящее время происходит стремительное развитие Глобальной сети. Возникает большая нехватка адресного пространства. В связи с этим и некоторыми другими проблемами протокола IPv4 возникла потребность перехода на протокол следующее поколения IPv6. Так как установленная база программного и аппаратного обеспечения IPv4 достаточно велика, возникает проблема обеспечения обратной совместимости. Сейчас ряд провайдеров предоставляют нативный IPv6-доступ конечным пользователям. Такими провайдерами в настоящее время являются: Дом.ru, CityTelecom, Билайн (моб.), ПГК Телеком и многие другие.

Протокол IPv4 имеет ряд недостатков. Такими являются: ограниченность адресного пространства, проблема перенумерации адресов, слабая расширяемость протокола и многие другие.

Все эти недостатки привели к разработке IP протокола следующего поколения, который имеет название протокол IPv6. В этом протоколе учли все недостатки протокола IPv4, а также были добавленные некоторые возможности, которые обеспечиваю удобство работы. Базовый заголовок протокола IPv4 и IPv6 имеет следующий вид.

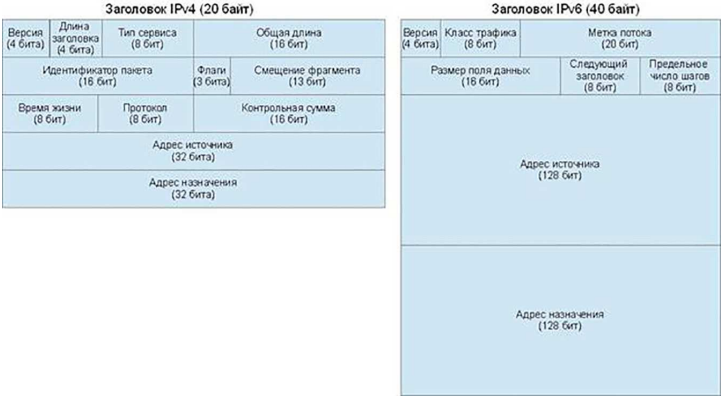


Рис. 1 - Базовый заголовок IPv4 и IPv6.

С самого начала разработки протокола IPv6 разрабатывались и механизмы его совместимости с протоколом IPv4. Среди таких механизмов следует отметить:

1. ALG (application level gateway) - шлюз прикладного уровня. В данном методе предполагается, что для каждого используемого сетевого приложения создается специальное прикладное программное обеспечение, осуществляющее преобразование трафика этого сетевого приложения из трафика IPv4 в трафик IPv6, и наоборот.

2. Бесконтекстный IP/ICMP транслятор - данный метод предполагает установку на границе IPv6 сети специального агента, осуществляющего трансляцию протоколов.

3. Двойной стек - в этом случае на каждом IPv6 хосте, которому требуется взаимодействие с IPv4 хостами, устанавливается стек протокола IPv4 и ему выделяется IPv4 адрес. После этого этот хост может взаимодействовать как с IPv4 хостами, так и с IPv6 хостами. Данный метод является самым простым методом решения проблемы совместимости.

4. Туннелирование - данный метод предназначен для создания IPv6 туннелей сквозь существующие IPv4 сети (в частности INTERNET), не поддерживающие протокол IPv6. Такие туннели создаются вручную либо автоматически различными способами и объединяют отдельные IPv6 сети между собой.

В данной статье более подробно рассматривается механизм тунелированния 6to4.

6to4 — это переходный механизм, позволяющий передавать IPv6-пакеты через IPv4-сети и не требующий создания двусторонних тунелей. Это используется, когда конечный пользователь или сайт хотят получить соединение с IPv6-Интернетом, но не могут получить его от провайдера.

Рассмотрим на практике создание туннеля 6to4, используя симулятор сети передачи данных Cisco Packet Tracer. Для этого возьмем схему, предоставленную на рисунке 2.

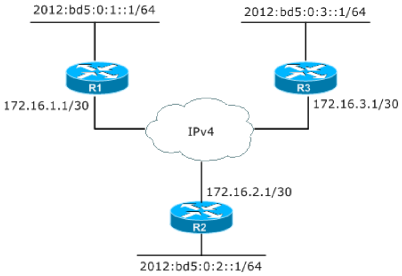


Рис. 2 - Используемая схема.

Три узла, за каждым из которых находится IPv6 LAN, связаны через сеть IPv4. Для того чтобы соединить эти LAN между собой надо настроить индивидуальные туннели IPv6 → IPv4 в стиле "каждый с каждым".

Для организации механизма 6to4 необходимо создать туннельный интерфейс на каждом маршрутизаторе. Выделим три основные момента:

- Режим туннеля (6to4)
- Источник туннеля (интерфейс либо адрес IPv4)
- IPv6-адрес для 6to4 (из сетки 2002::/16)

Для R1:

R1(config)# interface tunnel0

R1(config-if)# tunnel mode ipv6ip 6to4

R1(config-if)# tunnel source 172.16.1.1

Чтобы определить IPv6-адрес туннельного интерфейса, переводим IPv4 в шестнадцатеричную форму. 172.16.1.1 превращается в ac10:0101. После чего добавляем результат к префиксу 6to4 (2002::/16), а оставшееся место заполняем нулями. В примере используется маска /128, но допустимы и другие префиксы.

R1(config-if)# ipv6 address 2002:ac10:101::/128

Для R2:

R2(config)# interface tunnel0
R2(config-if)# tunnel mode ipv6ip 6to4
R2(config-if)# tunnel source 172.16.2.1
R2(config-if)# ipv6 address 2002:ac10:201::/128

Для роутера R3 применяются аналогичные настройки. Теперь, когда известен адрес 6to4 каждого маршрутизатора, можно добавить необходимые статические маршруты для организации связности роутеров между собой.

ipv6 route 2002::/16 tunnel0
ipv6 route 2012:bd5:0:2::/64 2002:ac10:201::
ipv6 route 2012:bd5:0:3::/64 2002:ac10:301::

Для роутера R2 и R3 применяются аналогичные настройки.

Многие эксперты утверждают, что стремительный переход к IPv6 начнется тогда, когда почувствуется ограниченность существующего на сегодняшний день адресного пространства.

Пока международные интернет-организации пытаются донести до интернет-общественности информацию о протоколе IPv6. В частности, корпорация ICANN регулярно проводит форумы и конференции, на которых рассказывается о новой технологии и ее преимуществах.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. RFC 2460 «Internet Protocol, Version 6 (IPv6) Specification», [электронный ресурс] http://www.ietf.org/rfc/rfc2460.txt.
2. RFC 2373 «IP Version 6 Addressing Architecture», [электронный ресурс] http://www.ietf.org/rfc/rfc2373.txt.
3. RFC 791 «Internet Protocol», [электронный ресурс] http://www.ietf.org/rfc/rfc791.txt.
4. RFC 3056 «Connection of IPv6 Domains via IPv4 Clouds «, [электронный ресурс] http://www.ietf.org/rfc/rfc356.txt.

В.А. ЯИЧКИН
Рязанский государственный радиотехнический университет

ПОВЫШЕНИЕ ТОЧНОСТИ ПЕРЕДАЧИ ИНФОРМАЦИИ СОЛИМЕРА, ПРИ ВЛИЯНИИ МЕЖЭЛЕКТРОДНОЙ ЭДС

В данной статье рассматриваются такие вопросы, как: зависимость частоты измерительного прибора от ЭДС (вызванного солесодержанием электролита), математическая модель данной зависимости и способ компенсации паразитного ЭДС.

В определенной промышленности или на производстве уровень солей в воде – очень важный параметр. Жесткость воды приводит к изменению ее определенных качеств, что, в свою очередь, может существенно повлиять на физико-химические процессы. Для определения уровня солесодержания в воде требуется измерительный прибор - солемер, который является составной частью автоматизированных систем контроля и управления различными системами, как первичные средства сбора и обработки информации.

В последнее время всё большее распространение получила частотная кондуктометрия (совокупность электрохимических методов анализа), суть которой состоит в использовании зависимости частоты переменного тока, питающего измерительную ячейку с исследуемым веществом. По сравнению с традиционными амплитудными она имеет повышенную помехозащищённость, простоту взаимодействия с интерфейсом ЭВМ, меньшую погрешность. (Недостатком метода является отсутствие строго линейной зависимости между изменением величины активного сопротивления исследуемого электролита и частотой.). Схема преобразования сопротивления солевого раствора в частоту содержит последовательно соединенные интегратор, триггер Шмитта и буферный усилитель (Рис.1).

Сопротивление R_x преобразуется в постоянное напряжение, а затем с помощью ПНЧ в частоту. При фиксировании емкости C_x, период колебаний определяется межэлектродным сопротивлением R_x. Малая величина тока, протекающего через RC-цепочку при сравнительно высоких частотах, создаёт незначительные поляризационные явления на электродах ячейки и тем самым малые погрешности измерений.

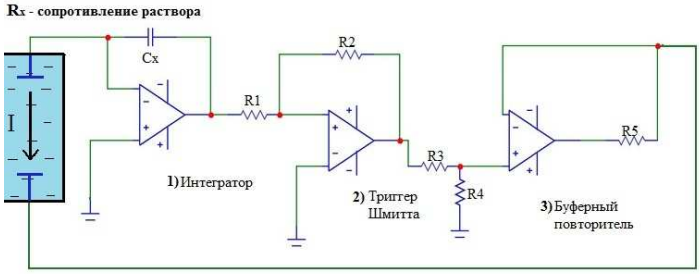


Рис.1- Функциональная схема генератора солемера.

На выходе измерительного прибора должен быть цифровой сигнал с определенной частотой ***f*.** Но практика показала, что на выходе имеем потенциальный цифровой сигнал с различными тактовыми интервалами (периодами), т.е. существует фазовый сдвиг (рис.2).

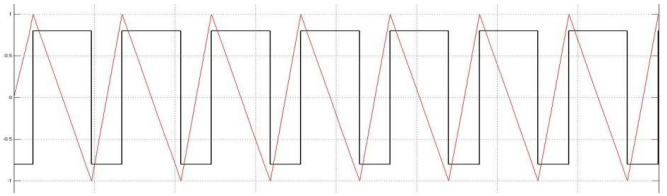


Рис.2 - Выходные сигналы интегратора и триггера Шмитта.

Причина этому то, что между электродами солемера в воде возникает ЭДС, которая является паразитной и зависит от определенных факторов, взаимодействующих с электродами устройства и искажая результаты измерения. Вследствие чего на выходе прибора изменяется частота цифрового сигнала и нарушается точность показания солемера.

Цикл измерения состоит из двух фаз: интегрирования сигнала (ИНТ) и разряда интегрирующего конденсатора (РИ). В течение фазы ИНТ, входной сигнал подается на вход интегратора. Это вызывает на конденсаторе накопление заряда, пропорционального по величине и соответствующего по знаку приложенному входному напряжению. К моменту начала фазы ИНТ заряд на конденсаторе C_x равен нулю. Напряжение на выходе интегратора изменяется с постоянной скоростью по формуле (1), пропорциональной входному сигналу до U_{max} (Рис.3) .

$$U = \frac{1}{C_x}\int_0^T i(t)dt; U_{\max} = k \cdot U \cdot t_0 \qquad (1)$$

При добавлении паразитного напряжения, C_x заряжается быстрей на величину (ΔU) до U_{max} (2).

$$U_{\max} = k \cdot (U + \Delta U) \cdot t_0 \qquad (2)$$

где k - коэффициент пропорциональности, зависящий от C_x.

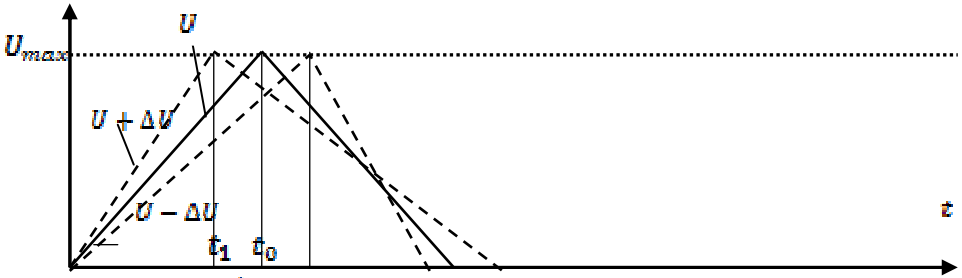


Рис.3- Цикл интегрирования и разрядки на конденсаторе.

$$t_1 = \frac{U_{max}}{k(U+\Delta U)}; t_2 = \frac{U_{max}}{k(U-\Delta U)}; \qquad (3)$$

Тогда полный цикл сигнал будет проходить за период:

$$T = t_1 + t_2 = \frac{2U_{max}}{k}\left(\frac{1}{(U+\Delta U)} + \frac{1}{(U-\Delta U)}\right) =$$

$$= \frac{2U_{max}}{k}\left(\frac{2U}{U^2-\Delta U^2}\right) = \frac{4U_{max}U}{k(U^2-\Delta U^2)} \qquad (4)$$

Соответственно частота сигнала:

$$f = \frac{1}{T} = \frac{k(U^2-\Delta U^2)}{4U_{max}U} \qquad (5)$$

При графическом построении, зависимость изменения частоты (f) от воздействия ΔU будет выглядеть следующим образом (Рис.4).

Из графика видно, что зависимость нелинейная и влияние межэлектродной ЭДС необходимо компенсировать. Одним из способов компенсации - является подключение к выходу буферного усилителя обратной связи (ОС), которая состоит из интегратора и инвертора (Рис.5).

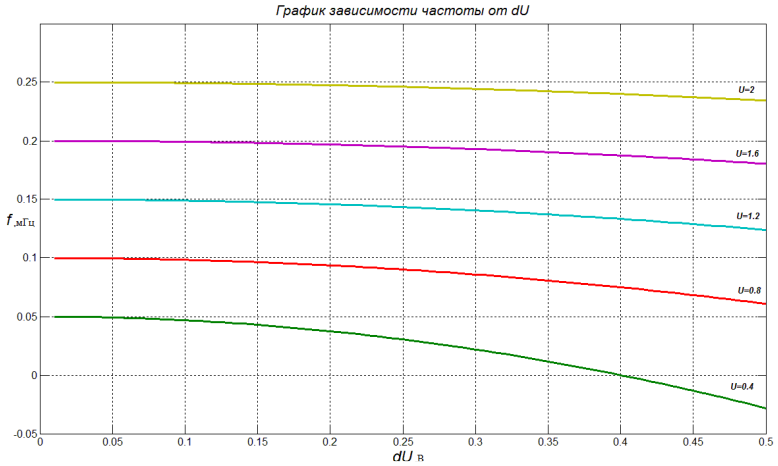


Рис.4 - График зависимости частоты от воздействия.

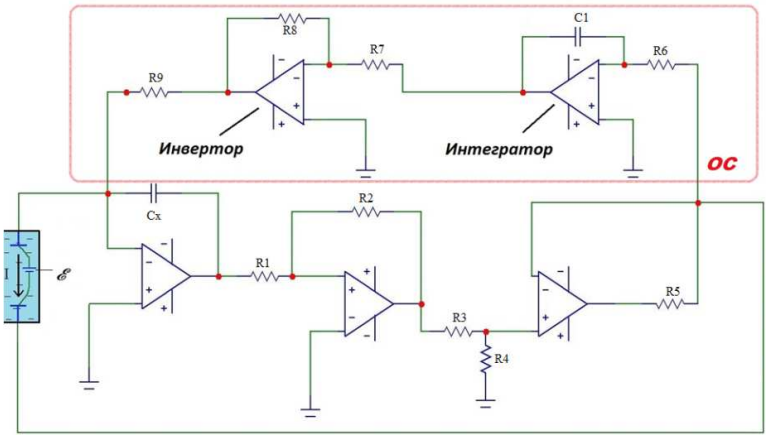


Рис.5 - Функциональная схема генератора солемера с ОС.

В результате на выходе генератора при установившемся режиме наблюдается цифровой сигнал, где отчетливо видно, что уже нет различия между тактовыми интервалами (Рис.6).

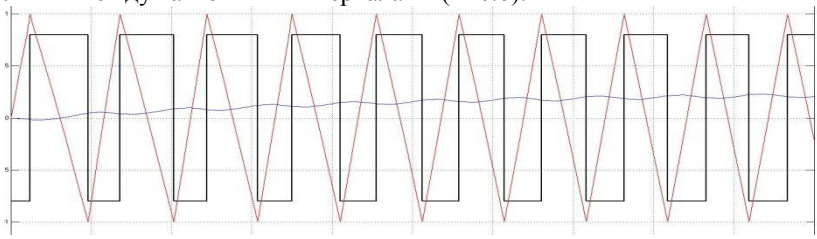


Рис.6 - Выходные сигналы генератора с ОС.

Дальнейшие работы будут посвящены влиянию температурной зависимости на точность измерения (тепловой анализ), защите от электромагнитных помех, определение собственных форм и частот и воздействие других параметров.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Титце У., Шенк К. Полупроводниковая схемотехника. 12 изд.: учебник-справочник. Пер. с нем. – М.: ДМК Пресс, 2008.–832 с
2. Земельман М.А. Автоматическая коррекция погрешностей измерительных устройств.– М.: Издательство стандартов, 1972.–199 с.

Уважаемые коллеги!

Министерство образования и науки Российской Федерации, Российский фонд фундаментальных исследований и Рязанский государственный радиотехнический университет проводят **16-18 ноября 2016 года** XXI Всероссийскую научно-техническую конференцию студентов, молодых ученых и специалистов «Новые информационные технологии в научных исследованиях» (НИТ-2016).

Работа конференции будет проходить по следующим направлениям:

- ***информационные технологии в социальных и экономических системах;***
- ***математические модели в информационных технологиях;***
- ***телекоммуникационные технологии;***
- ***информационные ресурсы и программно-инструментальные средства;***
- ***технологии искусственного интеллекта в проектировании;***
- ***информационные системы и процессы;***
- ***обработка изображений в системах управления;***
- ***геоинформационные технологии;***
- ***информационные технологии в промышленности;***
- ***космические информационные технологии;***
- ***автоматизация проектирования в телекоммуникационных системах.***

Для участия в конференции необходимо **до 5 октября 2016 г.** по эл. почте (**nich@rsreu.ru**, тема НИТ-2016) и **простым** письмом выслать в адрес Оргкомитета конференции (390005, г. Рязань, ул. Гагарина, 59/1, РГРТУ, Периго Н.Б.) следующие документы: тезисы доклада (на бумажном носителе - 2 экз., в эл. виде имя файла по фамилии автора); заявка на участие. Приглашения на конференцию будут высланы по указанным авторами адресам. Сборник тезисов будет опубликован к началу конференции. **ЭЛЕКТРОННАЯ ВЕРСИЯ СБОРНИКА БУДЕТ РАЗМЩЕНА В E-library И ПРОИНДЕКСИРОВАНА В РИНЦ.**

Регистрационный взнос, включающий расходы на издание программы и тезисов докладов, составляет **300 рублей**. Копии платежных поручений по оплате регистрационного взноса также необходимо направить на эл. почту ***nich@rsreu.ru***.

Телефон для справок: (4912) 46-03-09 – Периго Наталья Борисовна.

Подробная информация – на странице в сети Интернет: ***sapr.rsreu.ru***

Требования к оформлению

Тезисы доклада объемом 1-2 страницы должны быть преддставлены в виде документа в формате текстового процессора Microsoft Word. Размер листа – А4, поля со всех сторон 20 мм. Шрифт Verdana. Размер шрифта – 12 кегля, текст печатается через 1 интервал и должен быть выровнен по правой и левой сторонам документа. Красная строка 0.5 см. Автоматическая расстановка переносов. Формулы подготавливаются с применением Microsoft Education 3.0, рисунки в формате BMP или JPEG с разрешением 200x200 dpi, подрисуночная подпись не должна быть частью рисунка. Количество формул и рисунков должно быть минимально возможным.

Тезисы доклада должны быть оформлены в следующем порядке: название доклада прописными буквами по центру, шрифт - полужирный; следующая строка – авторы (инициалы перед фамилией!) по центру; следующая строка – научный руководитель (Ф.И.О., ученая степень, ученое звание) по центру; следующая строка – название организации по центру; далее через одну строку текст доклада. Библиографический список (если имеется) – по ГОСТ 7.1.2003 и отделяется от текста пустой строкой. Ссылки на библиографический список в тексте доклада в квадратных скобках.

Образец оформления

ВОССТАНОВЛЕНИЕ НЕСУЩЕЙ СИСТЕМЫ ПЕРЕДАЧИ ИНФОРМАЦИИ

И.И.Иванов

Научный руководитель – Петров П.П.,
д-р техн. наук, процессор

Рязанский государственный радиотехнический университет

В докладе рассматривается ряд проблемных вопросов по проектированию…

Библиографический список

1. Сосулин Ю.Г. Теоритические основы радиолокачии и радионавигации: учеб. Пособие для вузов. – М.:Радио и связь, 1992. – 304 с.

Информационные технологии

Межвузовский сборник научных трудов

Компьютерная верстка: А.Н. Сапрыкин,
А.С. Буробина, Е.С. Геращенко, В.Ю. Потапова

Подписано в печать 15.05.16. Формат бумаги 60х84 1/16.
Бумага офсетная. Печать струйная. Усл. печ. л. 6,4.
Тираж 500 экз. Заказ № 455.

Отпечатано в типографии BookJet

390046, г. Рязань, Скорбященский проезд, д.20,
тел.:+7 (4912) 466-151, сайт http://bookjet.ru/