Министерство науки и высшего образования РФ

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ ЛЕСОТЕХНИЧЕСКИЙ
УНИВЕРСИТЕТ имени С. М. Кирова»

Кафедра информационных систем и технологий

ИНФОРМАЦИОННЫЕ СИСТЕМЫ И ТЕХНОЛОГИИ: ТЕОРИЯ И ПРАКТИКА

Сборник научных трудов

Выпуск 12

Санкт-Петербург
2020

Рассмотрено и рекомендовано к изданию
учебно-методической комиссией Ученого совета
Санкт-Петербургского государственного лесотехнического
университета имени С. М. Кирова
23 марта 2020 г.

Р е д а к ц и о н н а я к о л л е г и я :

кандидат технических наук, профессор (отв. редактор) **А. М. Заяц,**
кандидат технических наук, доцент (отв. секретарь) **М. А. Шубина,**
кандидат технических наук, доцент **С. П. Хабаров**

С о с т а в и т е л ь
кандидат технических наук, доцент (отв. секретарь) **М. А. Шубина**

Р е ц е н з е н т ы :

доктор технических наук, профессор **И. В. Иванова**
(Национальный минерально-сырьевой университет «Горный»),
кандидат технических наук, доцент **А. Г. Карманов**
(Университет ИТМО, факультет инфокоммуникационных технологий)

Информационные системы и технологии: теория и практика: сб. науч. тр. Вып. 12 / отв. ред. А. М. Заяц. – Санкт-Петербург: СПбГЛТУ, 2020. – 138 с.

ISBN 978-5-9239-1159-6

Представлен кафедрой информационных систем и технологий.

Сборник подготовлен по материалам кафедры вуза, представленным на научно-технической конференции Института леса и природопользования СПбГЛТУ в феврале 2020 г., и практических работ, выполненных ее сотрудниками.

Темплан 2020 г. Изд. № 209.
ISBN 978-5-9239-1159-6 © СПбГЛТУ, 2020

А.К. Бойцов, магистрант
СПбГЛТУ им.С.М.Кирова
A.K.Boitsov@yandex.ru
Н.В. Лушкин, кандидат технических наук, доцент
Кафедра информационных систем и технологий
СПбГЛТУ им.С.М.Кирова
Lushkin52@mail.ru
Н.Б. Смелова, старший преподаватель
Кафедра информационных систем и технологий
СПбГЛТУ им.С.М.Кирова
smelova.nb@gmail.com

ОПРЕДЕЛЕНИЕ МИКРОБИОЛОГИЧЕСКИХ ПАРАМЕТРОВ ЛЕСНЫХ ОБЪЕКТОВ ПО ИХ ГРАФИЧЕСКИМ ФАЙЛАМ НА ПРИМЕРЕ КЛЕТОЧНОГО СТРОЕНИЯ СОСНЫ СИБИРСКОЙ (*PÍNUS SIBÍRICA*) И ИВЫ ЛОМКОЙ (*SÁLIX FRAGÍLIS*)

Оценка анатомических признаков — это основа для освоения ряда биологических дисциплин, включающих физиологию растений, общую экологию, фитопатологию, древесиноведение, и многих других наук.

На сегодняшний день развитие анатомии растений идёт очень быстро. Это связано с большим прогрессом во всех биологических науках, который был обусловлен созданием новейших и универсальных методов исследований за счёт быстрого развития информационных технологий.

Определение микробиологических параметров лесных объектов по их графическим файлам — это современный подход, который позволяет выявить актуальные направления и обратить внимание на развитие компьютерного анализа объектов лесной отрасли.

Программная реализация алгоритма позволяет создать систему оперативной обработки микрофотографий древесины и оценки их анатомических признаков. Обработку снимка можно проводить и вручную, но это сильно сказывается на времени проведения исследований. А иногда ручной подсчет практически невозможен. Также необходимо учитывать погрешность, связанную с человеческим фактором.

Определение микробиологических параметров лесных объектов по их графическим файлам рассмотрим на примере клеточного строения сосны сибирской (Pínus sibírica) и ивы ломкой (Sálix fragílis).

Микроскопическое строение древесины сосны состоит из выстилающих клеток, мертвых клеток, клеток сопровождающей паренхимы, каналов хода, трахеид и сердцевинных лучей, а микроскопическое строение древесины ивы состоит из сосудов, сердцевинных лучей и волокон либриформа. Мы будем исследовать сосуды, сердцевинные лучи, волокна либриформа и трахеиды.

Трахеиды — это прозенхимные клетки, не имеющие отверстий в месте соединения друг с другом или, по-другому, это сильно вытянутые волокна с одревесневшими стенками. У хвойных пород трахеиды выполняют не только свойственные им проводящие функции, но и механические, что придаёт этим породам прочность. Либриформ — это механическая ткань, которая придаёт прочность древесине у лиственных пород. Сосуды представляют собой трубки длиной около 2 см, а в отдельных случаях до 10 см и более [1]. Сердцевинные лучи — группы из крупных прямоугольных паренхимных живых клеток, вытянутых в поперечном к оси стебля направлении [2].

Рассмотрим алгоритм определения микробиологических параметров лесных объектов по их графическим файлам на примере клеточного строения сосны сибирской (Pínus sibírica) и ивы ломкой (Sálix fragílis).

Ставится задача оценить и сравнить площади клеток сосны сибирской (Pínus sibírica) (рис.1) и ивы ломкой (Sálix fragílis) (рис.2).

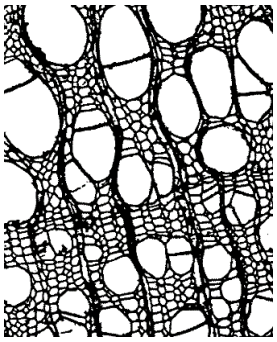


Рис. 1 Микроскопическое строение ивы ломкой (*Sálix fragílis*)

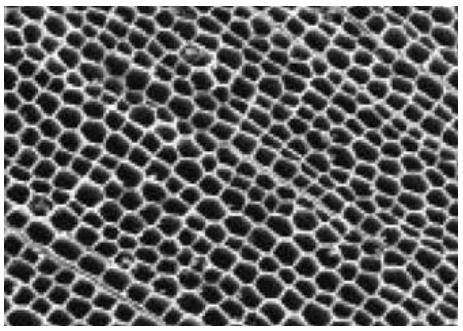


Рис. 2. Микроскопическое строение древесины сосны сибирской (*Pínus sibírica*)

После запуска программы открываются два окна. В левом окне отметим цвет нужных нам клеток ивы ломкой (Sálix fragílis) (рис.3) и сосны сибирской (Pínus sibírica) (рис.4). Затем запускаем программу кнопкой "пуск". В правом окне отображаются с помощью различных цветов связанные объекты, в нашем случае — это нужные клетки.

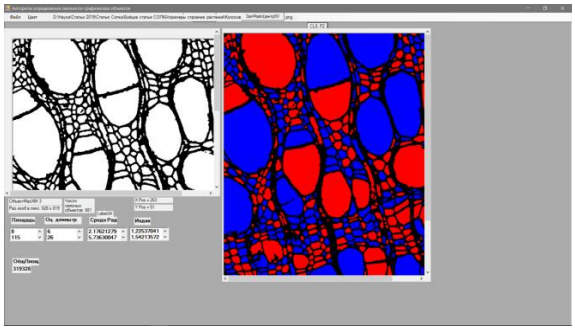


Рис. 3. Связанные клетки ивы ломкой (*Sálix fragílis*)

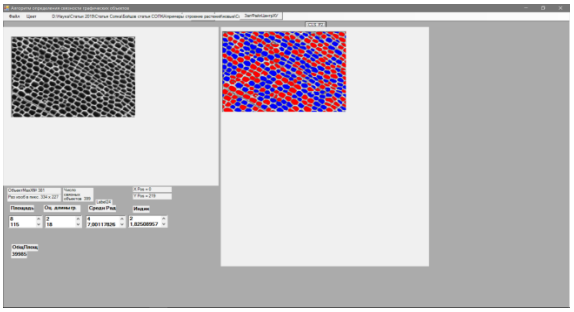


Рис. 4. Связанные клетки сосны сибирской (*Pínus sibírica*)

Последовательно отметив клетки, получим расчёт площадей клеток ивы ломкой (*Sálix fragílis*); площадь клеток сосудов представлена на рис.5;

площадь клеток сердцевинных лучей — на рис. 6; площадь клеток волокон либриформа — на рис 7. У сосны сибирской (*Pínus sibírica*) площадь клеток трахеид представлена на рис. 8.

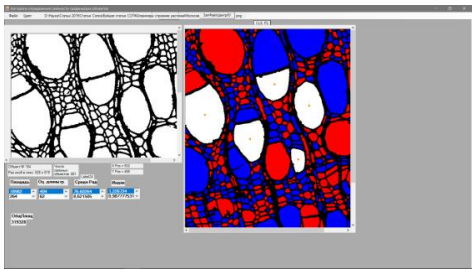


Рис.5. Расчёт площади клеток сосудов

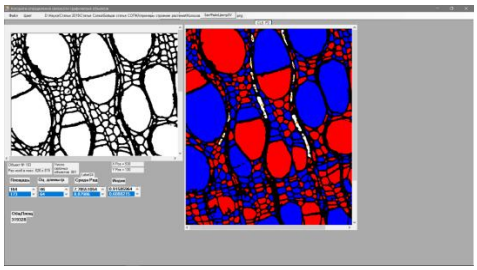


Рис. 6. Расчёт площади клеток сердцевинных лучей

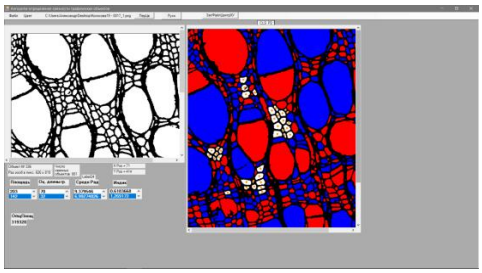


Рис. 7. Расчёт площади клеток волокон либриформа

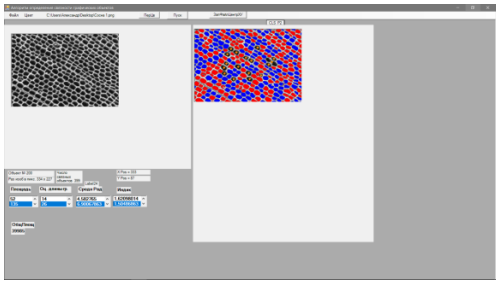


Рис. 8. Расчёт площади клеток трахеид

Полученные в результате расчета площади клеток ивы ломкой (*Sálix fragílis*) и сосны сибирской (*Pínus sibírica*) представлены в табл. 1.

Т а б л и ц а 1

Ива ломкая (*Sálix fragílis*)			Сосна сибирская (*Pínus sibírica*)
Сосуды S (у.е)	Серцевинные лучи S (у.е)	Волокна либриформа S (у.е)	Трахеиды S (у.е)
7653	543	200	182
10005	582	194	174
10997	698	429	173
2758	94	382	214
18982	86	398	149
	142	331	124
	141	170	118
	144	170	151
		488	197
		424	172
		359	169
Scp = 6826,8			Scp = 165,7

Средняя площадь клеток ивы ломкой (Sálix fragílis) Scp = 6826,8 значительно превосходит среднюю площадь клеток сосны сибирской (*Pínus sibírica*) Scp = 165,7. Так как эти клетки выполняют не только проводящие, но и механические функции, то это говорит о свойствах древесины, а именно – о её плотности, упругости и прочности. Так как ива ломкая (Sálix fragílis) имеет большие площади клеток, то она не является физически крепкой и плотной древесиной, она скорее более водянистая. У сосны сибирской (Pínus sibírica) площади клеток небольшие — такая древесина значительно более крепкая и плотная.

Данная компьютерная обработка клеточного строения сосны сибирской (*Pínus sibírica*) и ивы ломкой (*Sálix fragílis*) несопоставима по скорости с ручным подсчетом, а также при использовании этой методики исключаются погрешности, связанные с человеческим фактором. Данный подход при обработке графических файлов микробиологических объектов позволяет сделать вывод об актуальности замены ручного метода подсчета на компьютерный. Программа позволяет создать метод оперативной и точной оценки микроскопических характеристик строения древесины.

Библиографический список

1. Н.В. Рыжова, В.В. Шутов. Физика древесины: учебное пособие – Кострома: Изд-во КГТУ, 2009. – 75 с.
2. Сайт Gufo.me. – Режим доступа: https://gufo.me /dict/botany_terms / сердцевинные лучи.
3. М.И. Колосова, Н.Г. Соловьева. Основные анатомические признаки древесины лиственных деревьев и кустарников: учебное пособие – Санкт-Петербург, 2013 - 104 с.
4. Н.В. Лушкин. Алгоритм определения связности графических объектов. –Труды Санкт-Петербургской Государственной лесотехнической академии Актуальные проблемы развития высшей школы Санкт-Петербург. 2010. Стр. 308-309.
5. Н.В. Лушкин. Реализация алгоритма определения чисел Бетти в графических объектах. //Сборник научных трудов. Вып. 8 Информационные системы и технологии: Теория и практика – СПб.: СПбГЛТУ, 2016. – 61-70 с.

Н.П. Васильев, кандидат технических наук, доцент
Кафедра информационных систем и технологий
СПбГЛТУ им.С.М.Кирова
nikpv@mail.ru

БЕСПЛАТНЫЙ SSL-СЕРТИФИКАТ ДЛЯ AD HOC УСТАНОВКИ IOS-ПРИЛОЖЕНИЙ

Введение

Процесс развертывания, то есть загрузка и установка (deploing) на мобильное устройство iOS-приложении, тем более, их централизованное распространение (distributing) — довольно сложный и не бесплатный процесс. Следует, однако, подчеркнуть, что с появлением Xcode 7-ой версии стало возможна установка приложении без участия в одной из платных программ Apple (Apple Developer Program или Apple Developer Enterprise Program), то есть бесплатно, правда, с рядом существенных ограничении. Эту возможность иногда называют Free Provisioning. Пожалуй, самое существенное ограничение состоит в том, что приложение будет запускаться на мобильном устройстве только в течение недели. Однако, если пересобрать приложение и вновь развернуть на мобильном устройстве, то оно снова просуществует очередную неделю.

В остальных случаях потребуется приобщиться (зачислиться — Enroll) к Apple Developer Program или Apple Developer Enterprise Program на портале developer.apple.com. Различие этих двух программ в основном состоит в том, что участие в первой дает право на централизованное распространение приложении через App Store. Участие во второй ориентировано на распространение приложении только внутри фирмы (организации). Это так называемое In-House распространение (за пределами App Store). Участие в первой программе обойдётся в 99$, участие во второй — 299$ в год. В рамках любой программы допускается так называемое Ad Hoc распространение — можно устанавливать приложения на 100 устройствах в год [1-5].

Когда приложение готово для демонстрации и тестирования в узком кругу лиц, то последний способ является наиболее удобным и приемлемым, поскольку для его реализации достаточно разместить специально подготовленный архив на сайте и организовать ссылку, использующую специальный протокол itms-services:

```
<html>
<head>
<title>My Ad-Hoc Distribution Site</title>
</head>
<body>
<a href=
"itms-services://?action=download-
```

```
manifest&url=https://mysite_dns/manifest.plist">
AudioPlayer
</a>
</body>
</html>
```

Отметим, что архив и файл манифеста (manifest.plist) xCode готовит автоматически, и от разработчика требуется только указание точного месторасположения архива и картинок, которые появятся при загрузке. Однако, Apple допускает загрузку и установку приложений на устройство только по защищённому каналу ssl (Secure Socket Level), то есть в ссылках должен использоваться https-протокол. Если на сайте не установлен ssl-сертификат, то это — очередная "головная боль" для разработчика. Конечно, можно воспользоваться многочисленными предложениями в сети, либо непосредственно от хостера, стоимость которых будет зависеть от статуса сертификата, точнее, организации его выдавшей.

Существуют и бесплатные варианты, один из которых подробно обсуждается в данной статье, и для реализации которого потребуется понимание основ работы защищённого канала.

Принцип работы защищённого канала

Главная идея состоит в том, что данные, которыми обмениваются клиент и сервер шифруются и дешифрируются с помощью, так называемого, сеансового ключа, о котором знает только клиент и только сервер. Ключ симметричный — то есть он используется как для шифрования, так и дешифрования данных, поэтому если ключ будет перехвачен, то, очевидно, соединение будет взломано. Оказывается, что этот ключ никогда не передаётся между клиентом и сервером в незашифрованном виде, и вот как это реализуется.

Предположим, что клиент и сервер уже договорились об алгоритмах шифрования (это происходит на начальной стадии формирования соединения). Затем сервер посылает клиенту ssl-сертификат, который содержит публичный ключ, данные о типе сертификата, его владельце и третьей стороне, которая подтверждает его подлинность. Клиент после подтверждения подлинности сертификата третьей стороной создаёт случайную последовательность — сеансовый ключ и шифрует его с помощью полученного публичного ключа. Напомним, что при асимметричном шифровании расшифровать эту последовательность можно только с помощью приватного ключа. Приватный ключ хранится только на сервере, поэтому вполне безопасно можно передать зашифрованный сеансовый ключ серверу (даже если его перехватят, расшифровать его не представляется возможным, поскольку нет приватного ключа, который в обмене не участвует). Сервер дешифрует сеансовый ключ с помощью приватного ключа. Таким образом, и клиент, и сервер обладают единым ключом для симметричного шифрования. Далее обмен осуществля-

ется сообщениями, зашифрованными с помощью этого уникального сеансового ключа. Ещё раз подчеркнём, что при передаче сеансового ключа от клиента к серверу используется ассиметричное шифрование, а затем этот сеансовый ключ уже используется для симметричного шифрования данных как клиентом, так и сервером. Наконец, отметим, что шифруются и дешифрируются данные запросов и ответов, которые полностью соответствуют http-протоколу, который в таком варианте принято называть https (http secure).

Исходя из представленного механизма, очевидно, требуется создать два ключа — приватный и публичный. Затем публичный ключ упаковать в сертификат ssl, который вместе с приватным ключом должны быть установлены на сервере. В процессе генерации ключей нет ничего сложного, однако, сложность составляет именно упаковка публичного ключа в ssl-сертификат, которая выполняется третьей стороной — центром сертификации (Certificate Authority — CA) после выполнения ряда проверок. В зависимости от типа проверок и солидности центра сертификации определяется и стоимость получения сертификата. В самом простом варианте проверяется только принадлежность домена заявителю. В принципе не исключается возможность формирования так называемых самоподписных ssl-сертификатов без привлечения третьих лиц, но в большинстве случаев такие сертификаты не признаются браузерами — как минимум будут выданы пугающие предупреждения.

Бесплатный сертификат от Let’s Encrypt

В настоящее время появилась возможность получения бесплатного сертификата от центра сертификации Let’s Encrypt (https://letsencrypt.org). Этот центр функционирует несколько необычно — через специальный сервис, о котором можно подробнее узнать на официальном сайте. Однако, делать это не обязательно, поскольку некоторые хостеры уже обеспечивают своим клиентам автоматическое взаимодействие с этим сервисом. Кроме этого, хорошим посредником является сайт https://www.sslforfree.com/, который обеспечивает стандартный способ получения сертификата, принятый у большинства центров CA. Недостатком такого сертификата является слишком малый срок его действия — всего три месяца, после чего его потребуется переиздать.

Итак, в любом случае, прежде всего, потребуется сгенерировать два ключа приватный и публичный. При этом специальные утилиты, предназначенные для этих целей, вместо публичного ключа создают так называемый запрос в центр сертификации на создание сертификата — Certificate Secure Request (CSR), который содержит публичный ключ.

Наиболее известной утилитой является утилита openssl, которая входит в состав unix подобных операционных систем, в том числе mac os. Чтобы

сгенерировать с её помощью приватный ключ достаточно в терминальном окне выполнить команду:

```
openssl genrsa -out private.key
```

В результате, в текущем каталоге появится файл private.key, содержащий приватный ключ (конечно, имя файла можно придумать и другое).

Далее для этого приватного ключа создаём публичный ключ — точнее CSR:

```
openssl req -new -key private.key -out request.csr
```

В ответ утилита запросит дополнительную информацию, которая будет встроена в сертификат. Далее приводятся примеры ответов (выделены серым шрифтом), среди которых важно правильно указать доменное имя сервера DNS, для которого создаётся сертификат:

```
You are about to be asked to enter information that will be in-
corporated
into your certificate request.
What you are about to enter is what is called a Distinguished
Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:RU
State or Province Name (full name) [Some-State]:Saint-Petersburg
Locality Name (eg, city) []:Saint-Petersburg
Organization Name (eg, company) [Internet Widgits Pty
Ltd]:Peterhost.ru
Organizational Unit Name (eg, section) []:Virtual Hosting
Common Name (e.g. server FQDN or YOUR name) []:my.sitedns
Email Address []:support@peterhost.ru

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
```

В результате, в текущем каталоге появится файл request.csr (имя файла можно выбрать и другое, чаще всего используется доменное имя сервера, для которого создаётся сертификат).

После создания запроса CSR пора его использовать — переходим на сайт sslforfree.com. Здесь, прежде всего, необходимо указать доменное имя сервера, для которого создаётся сертификат (рис.1):

Рис. 1. Вводим доменное имя сервера, для которого создаётся сертификат

Далее потребуется подтвердить владение данным доменным именем одним из предложенных способов, например, "Manual Verification" (рис. 2):

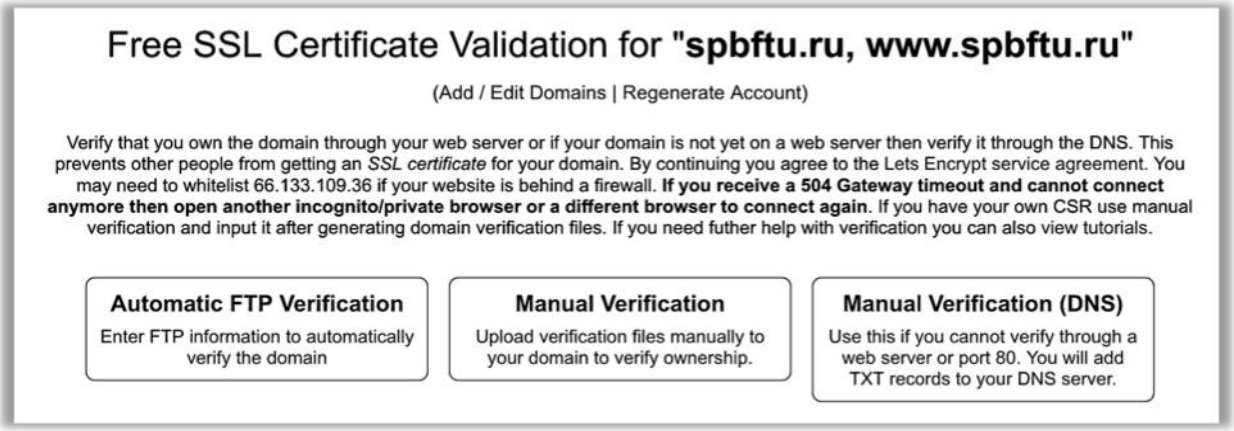


Рис. 2. Выбираем ручной (Manual Verification) способ проверки принадлежности сайта

Для ручной верификации потребуется разместить на сервере в заданном каталоге специальный файл (или файлы), содержащий случайную последовательность символов как в названии, так и в имени и перейти по соответствующей ссылке (или ссылкам), как показано на рис. 3.

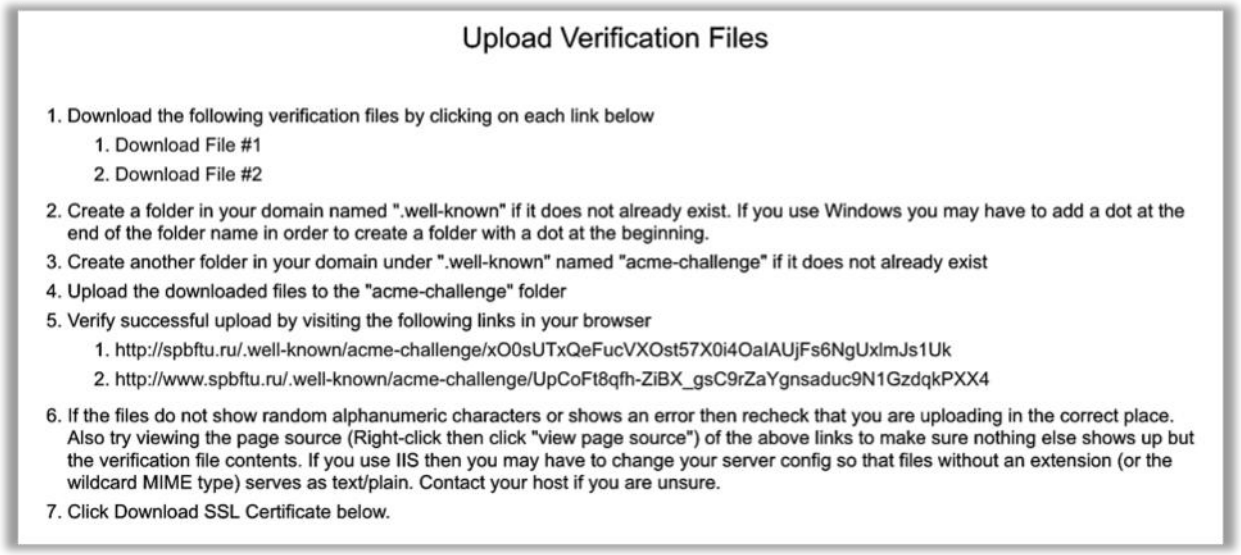


Рис. 3. Для двух указанных выше доменов потребуется загрузить два файла (Download File #1 и #2) и переместить их в указанное место на сервере, так чтобы отработали без ошибок указанные в п.5 ссылки

Если всё сделано правильно, то останется только сгенерировать сертификат. При этом генерацию приватного ключа и CSR сервис может взять на себя. В нашем случае, когда и ключ и CSR уже созданы следует не забыть установить соответствующую галочку (I have my own CSR) с последующим вводом CSR, как представлено на рис. 4.

Рис. 4. Достаточно скопировать CSR в предназначенное для этого поле и нажать "Download SSL Certificate"

После перехода по ссылке "Download SSL Certificate" сервис предоставит для скачивания сгенерированные сертификаты, готовые к последующей установке на целевом сервере с заданным доменным именем. Скачиваем сертификат — результат скачивания в файле certificate.crt. Заметим, что кроме обычного сертификата сервис также подготовит так называемый bundle-сертификат (ca_bundle.crt), предназначенный для поддоменов и приватный ключ в файле — private.key. В нашем случае в этом файле будет содержаться сообщение: You provided your own CSR which means the private key was probably generated when you got the CSR. We do not have access to the private key in this case to show it. Действительно, приватный ключ мы сгенерировали самостоятельно.

Осталось только правильно установить приватный ключ и сертификат на сервере. С этой целью требуется скорректировать конфигурационный файл сервера. Обычно панель управления сайтом, которая в том или ином виде предоставляется хостерами своим клиентам, позволяет выполнить эту задачу проще. Например, peterhost (peterhost.ru) преподносит эту возможность в следующем виде — интерфейс представлен на рис.5.

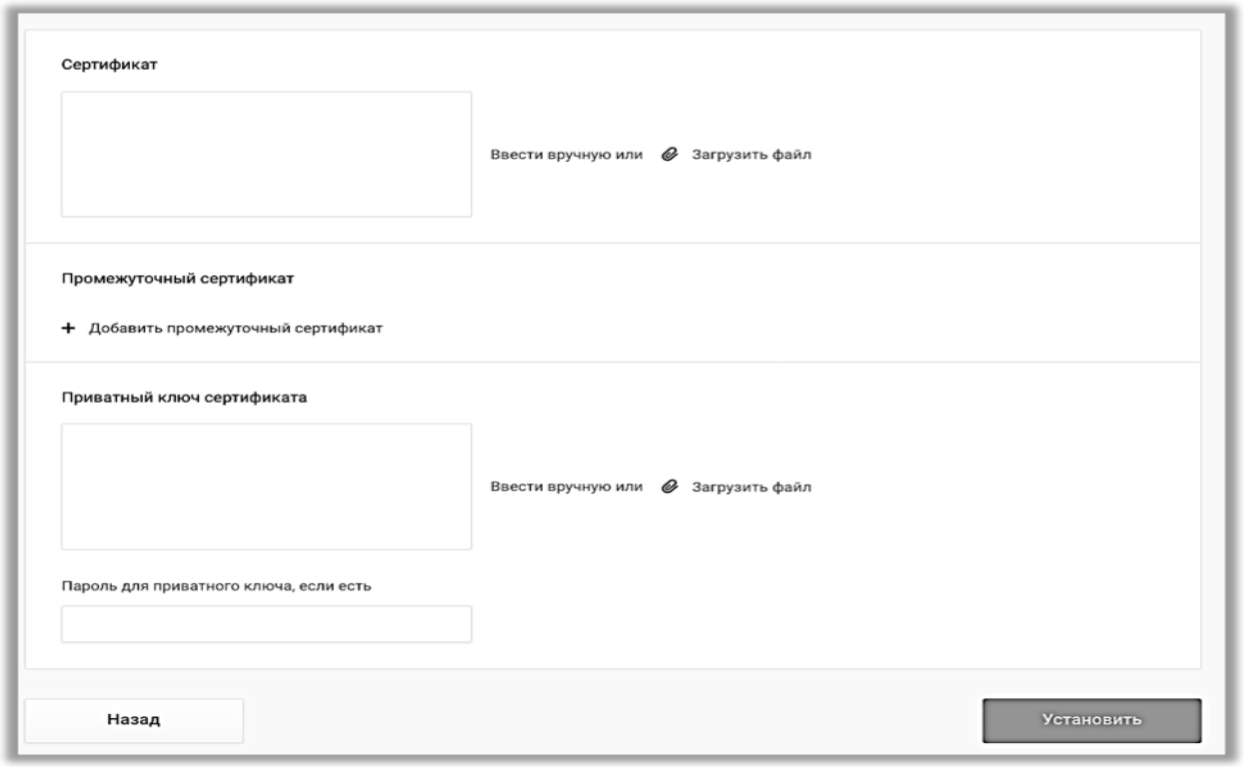


Рис. 5. Для установки SSL-сертификата, достаточно заполнить соответствующие поля

Таким образом, достаточно скопировать сертификат и приватный ключ в соответствующие поля (или загрузить файлы с соответствующим содержимым) и нажать кнопку "Установить" — изменения в конфигурационный файл сервера будут внесены автоматически.

Заключение

Одним из легальных вариантов обойти AppStore при распространении iOS-приложений является Ad Hoc загрузка. Этот вариант позволяет устанавливать приложение на 100 мобильных устройствах через Internet и для его реализации вполне достаточно штатного браузера устройства (Safari).

Очевидно, это удобный вариант распространения iOS-приложений в узком кругу лиц, или на стадии тестирования и демонстрации демоверсий заказчику.

Однако, на сайте, где будет расположен архив, должен быть установлен ssl-сертификат.

В статье подробно изложена технология получения бесплатного сертификата, от генерирования приватного и публичного ключей, запроса сертификата в центре сертификации и до установки его на сервере с описанием сути работы шифрованных ssl-каналов, а также технологии размещения Ad Hoc приложения на сайте.

Библиографический список

1. Заяц А.М., Васильев Н.П. Проектирование и разработка web- приложении. Введение в frontend и backend разработку на JavaScript и node.js: Учебное пособие. — СПб.: Издательство "Лань", 2019. — 120с.
2. Васильев Н.П. Мобильные Cordova-приложения сбора данных о состоянии лесных территории с привязкои к геопозиции. // Известия СПбЛТА — СПб.:СПбГЛТУ, 2019. No 229.
3. Васильев Н.П., Лушкин Н.В. Интеграция гибридных приложений Cordova с web - серверной обработкой графики. // Информационные системы и технологии: теория и практика: сб. научн. тр. Вып. 11. — СПб.: СПбГЛТУ, 2019. — с.10-24
4. Васильев Н.П. Универсальные технологии разработки мобильных приложении. // Информационные системы и технологии: теория и практика: сб. научн. тр. Вып. 10 Ч.1. — СПб.:СПбГЛТУ, 2018. — с.23-30.
5. Васильев Н.П. Гибридные технологии разработки приложений для мобильных платформ. // Информационные системы и технологии: теория и практика: сб. научн. тр. Вып. 9. — СПб.:СПбГЛТУ, 2017. — с.12-21.

Н.П. Васильев, кандидат технических наук, доцент
Кафедра информационных систем и технологий
СПбГЛТУ им.С.М.Кирова
nikpv@mail.ru
Н.В. Лушкин, кандидат технических наук, доцент
Кафедра информационных систем и технологий
СПбГЛТУ им.С.М.Кирова
Lushkin52@mail.ru
М.А. Шубина, кандидат технических наук, доцент
Кафедра информационных систем и технологий
СПбГЛТУ им.С.М.Кирова
nemsha@mail.ru

МОДЕЛИРОВАНИЕ ЛЕСОПОКРЫТЫХ ТЕРРИТОРИЙ ПРИ КЛАССИФИКАЦИИ ИХ ИЗОБРАЖЕНИЙ МНОГОУГОЛЬНИКАМИ ВОРОНОГО

В статье обсуждается задача моделирования структуры изображений наземных лесных угодий многоугольниками Вороного, что расширяет возможности анализа параметров территорий. Многоугольники Вороного получаются при разбиении плоскости, на которой заданы произвольные точки, на соответствующее количество областей. Каждая такая область - это множество точек плоскости наиболее близких к соответствующей точке рис.1.

Постановка задачи

Задача формулируется следующим образом. Пусть имеется плановое изображение лесного массива (например, рис. 2). Построить с помощью многоугольников Вороного модель, наиболее точно отображающую размещение лесного покрова на земной поверхности рис.3.

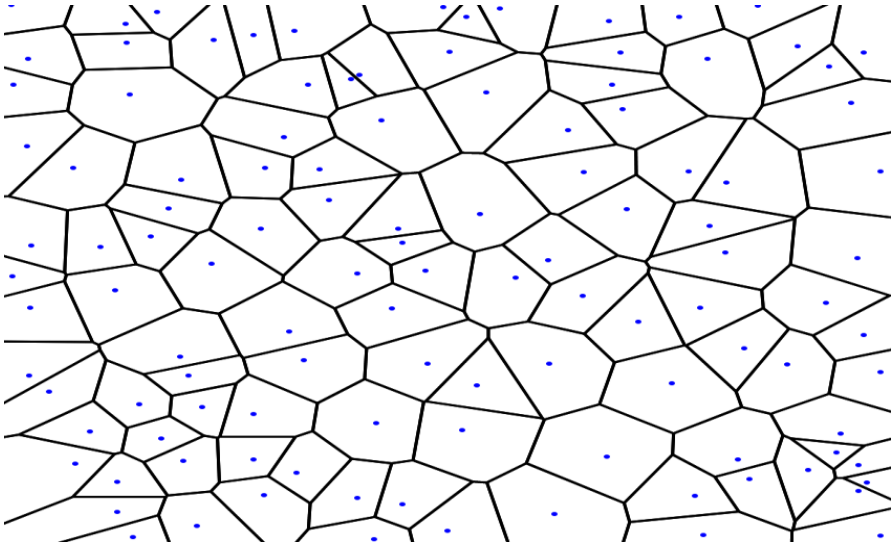


Рис.1. Ячейки Вороного для множества точек на плоскости

Для решения поставленной задачи моделирования древесных структур многоугольниками Вороного, кратко опишем последовательность действий. На изображении лесного массива определяется множество связанных объектов и с помощью программы Sopka [1] фиксируются координаты центров каждого объекта. Задавая эти координаты, как множество точек на плоскости, используя алгоритм Форчуна [2], строятся многоугольники Вороного. Пример расчета координат (x , y) центров связных объектов приведен на рис. 4, а сопоставление модели Вороного с изображением объекта – на рис. 5.

Рис. 2. Пример изображения участка лесного массива, полученного БПЛА

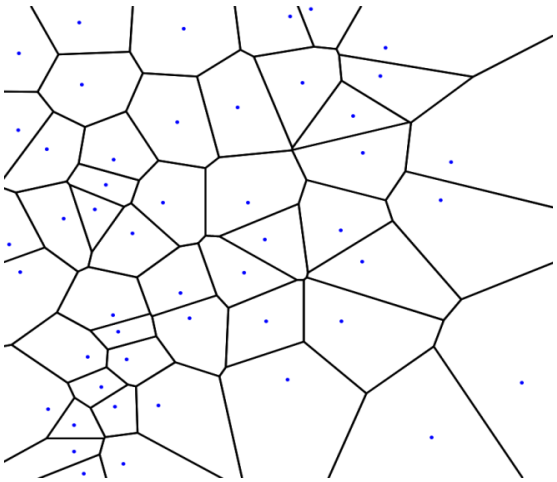


Рис. 3. Многоугольники Вороного

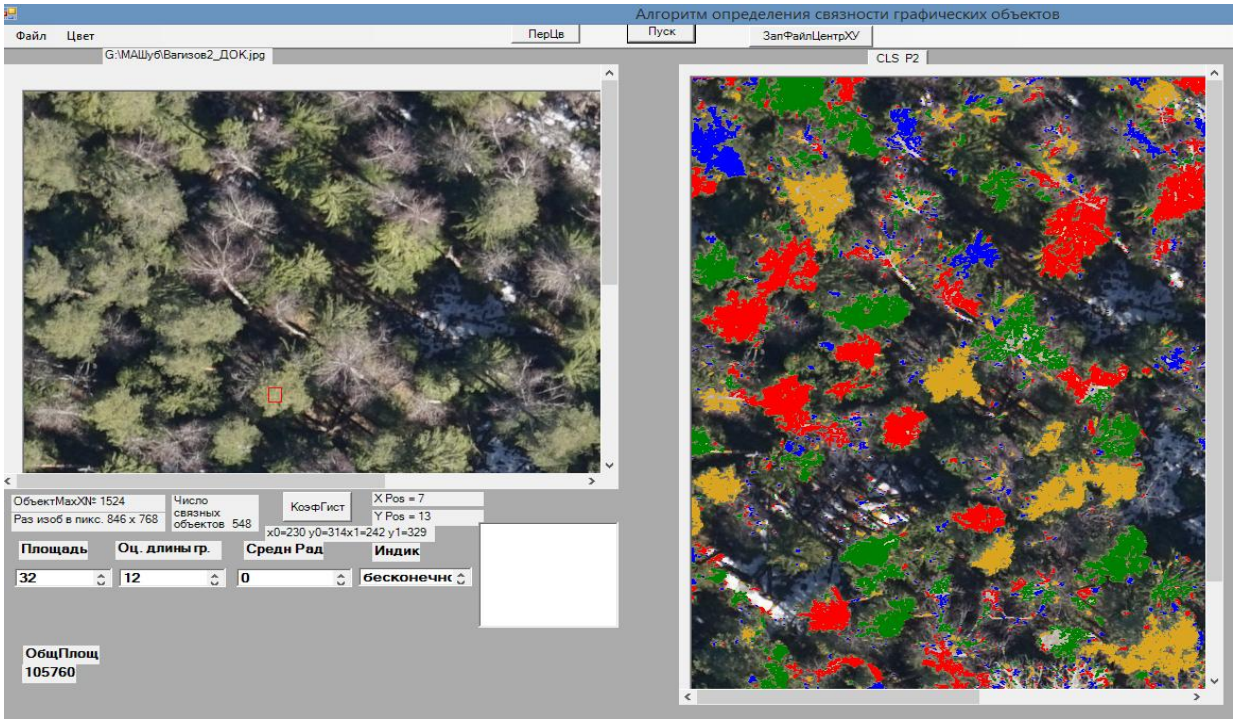


Рис. 4. Расчет программой Sopka координат (x , y) центров связных объектов

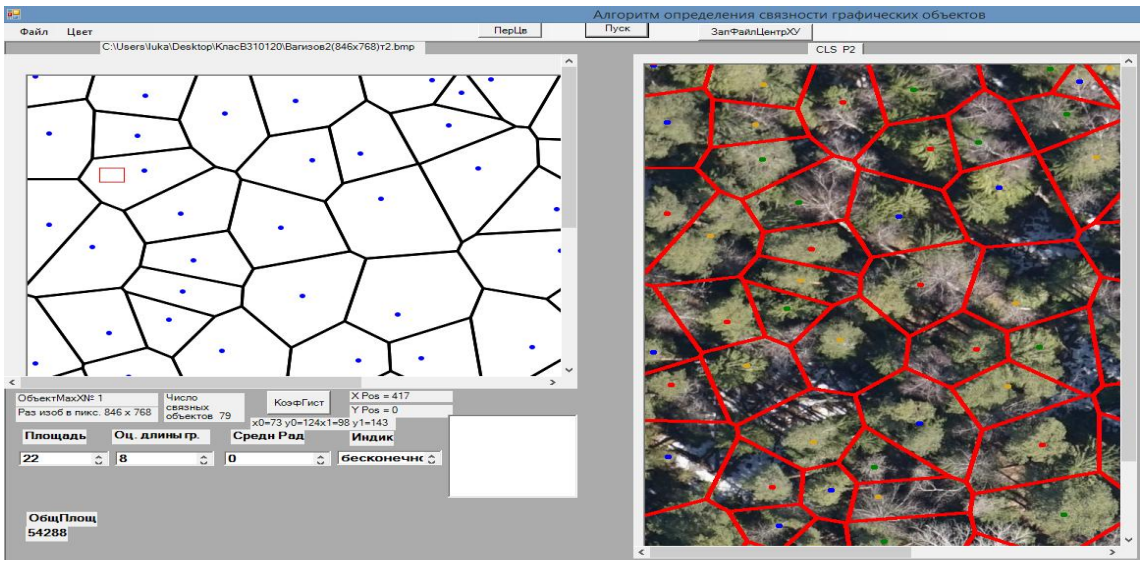


Рис.5.Сопоставление графического объекта, полученного по координатам (x ,y) центров, с моделью Вороного

Пример классификации изображения деревьев с помощью программы Sopka приведен на рис. 6, а пример моделирования изображения крон деревьев многоугольниками Вороного при минимальном размере связного объекта по координате X=10 пикселей на рис. 7.

Примеры моделирования того же изображения при минимальных размерах связного объекта по координате X=15 и X=20 приведены на рис. 8 и 9.

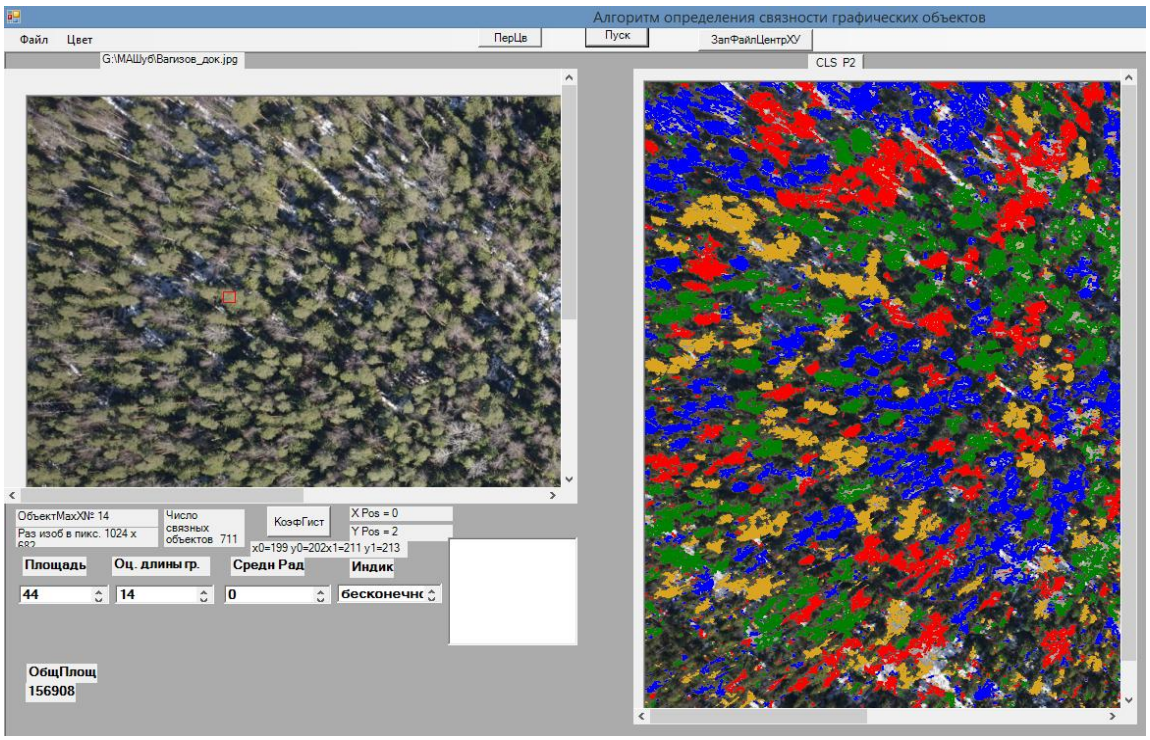


Рис.6. Пример классификации изображения крон деревьев

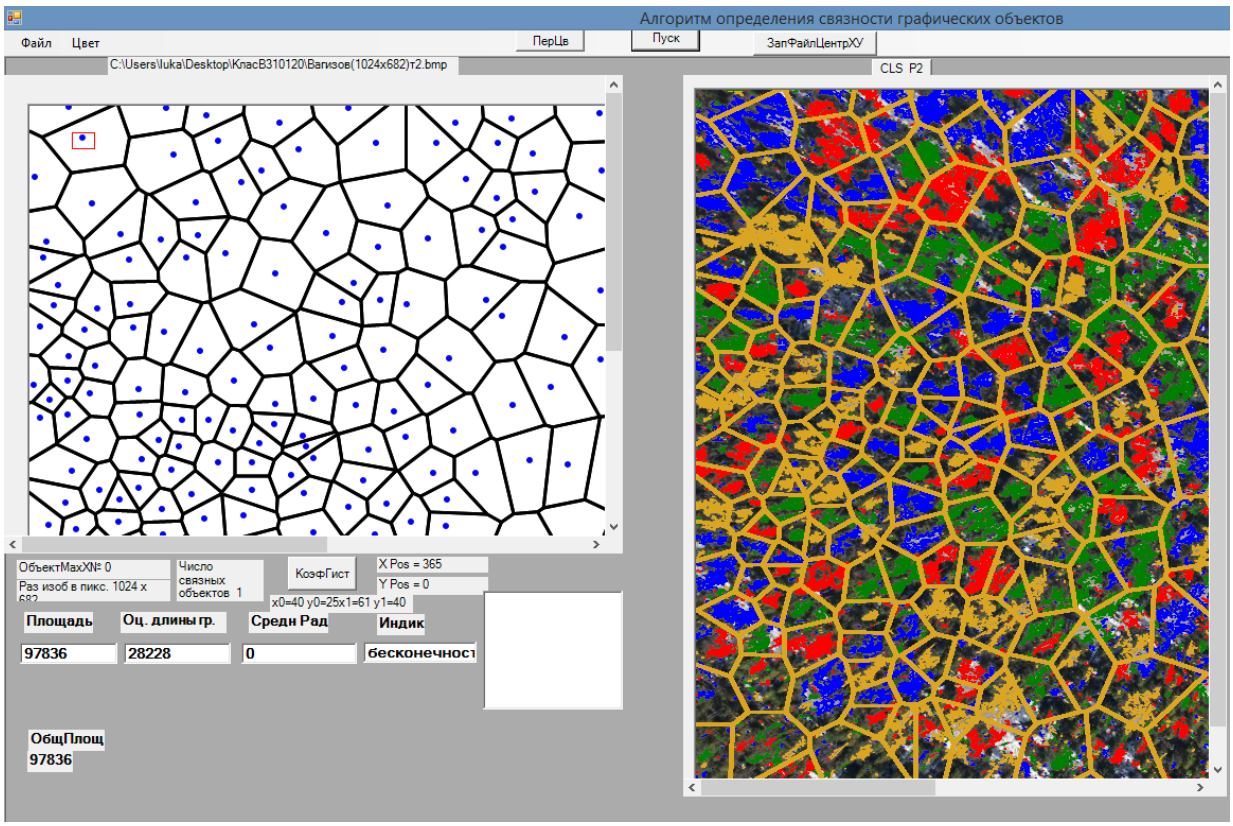


Рис.7. Пример моделирования крон деревьев многоугольниками Вороного при минимальном размере связного объекта по координате X=10 пикселей

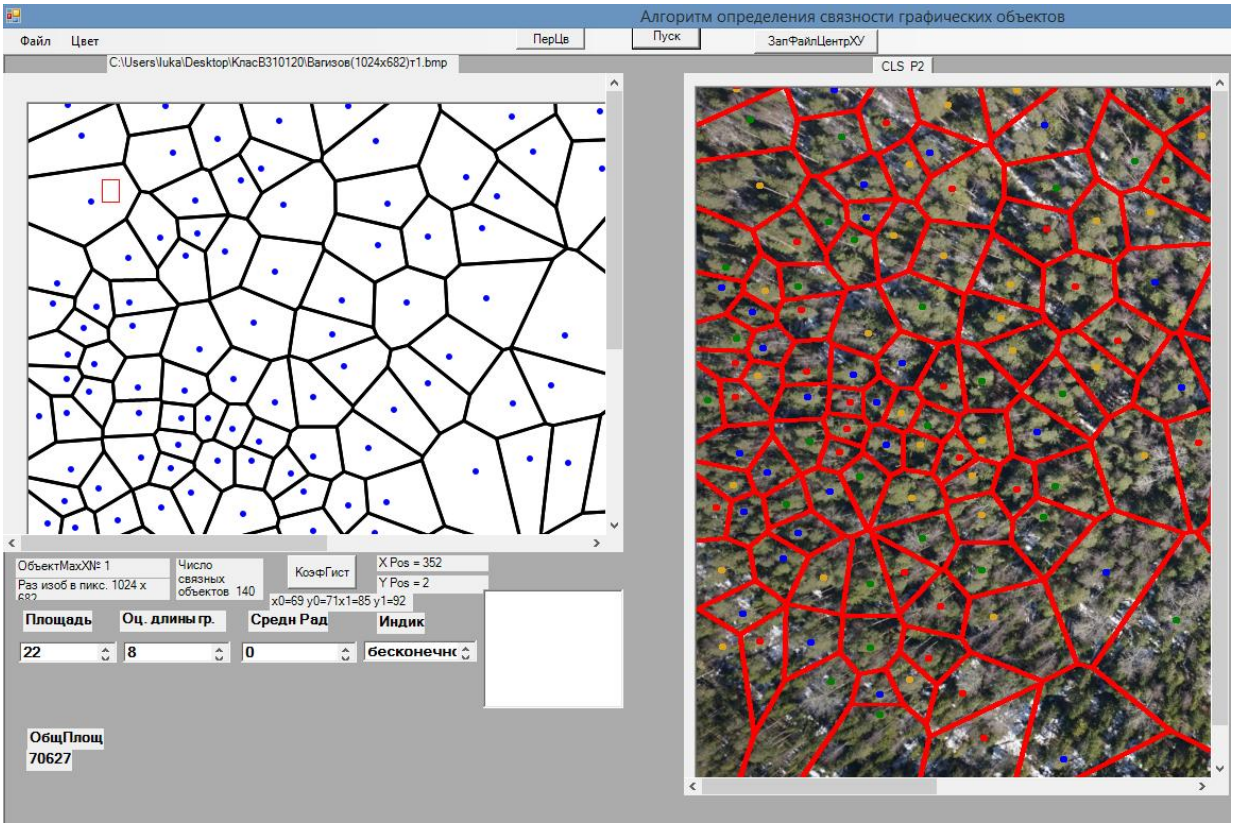


Рис. 8. Пример наложения модели на изображение крон деревьев (Xmin=15)

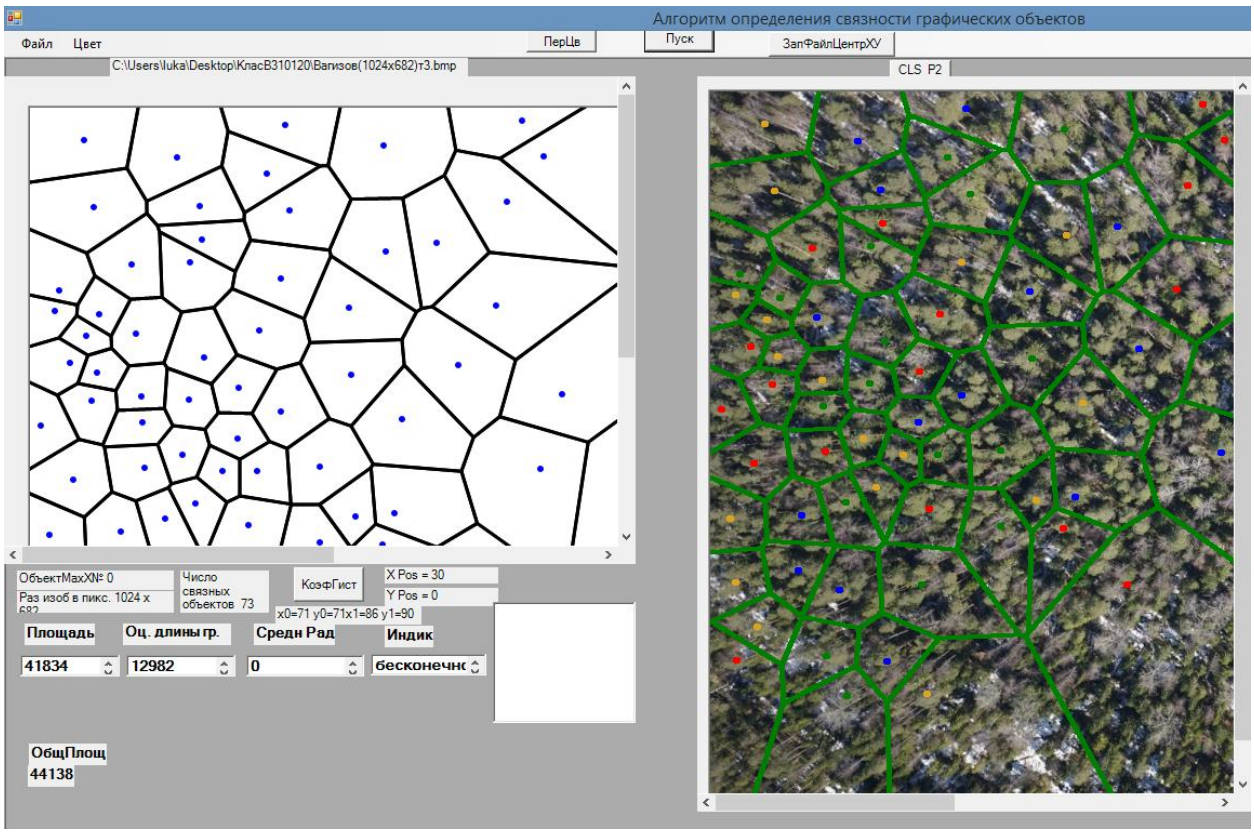


Рис. 9. Пример наложения модели на изображение крон деревьев (Xmin=20)

Применение модели к увеличенным фрагментам изображения приведено на рис. 10.

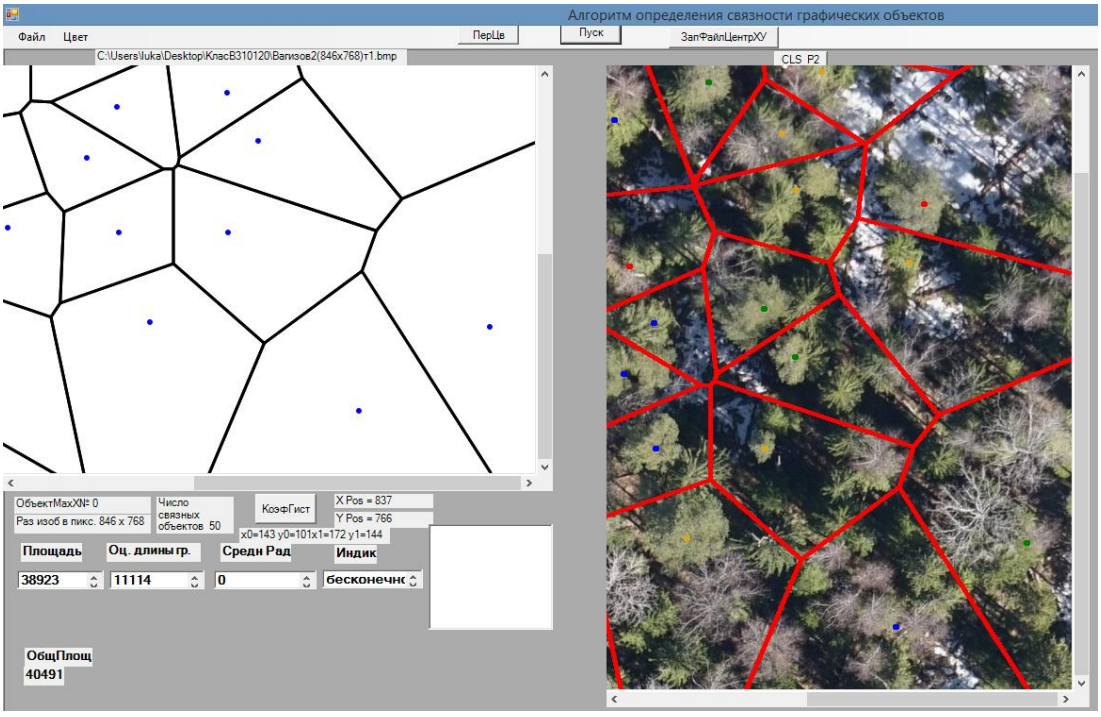


Рис. 10. Пример наложения модели на нижний угол изображения крон деревьев (Xmin=10)

Заключение.

Таким образом, приведенные примеры подтверждают возможность использования алгоритмов Форчуна на основе многоугольников Вороного для моделирования наземных объектов, в том числе для анализа лесопокрытых территорий.

Библиографический список

1. Лушкин Н.В. Алгоритм определения связности графических объектов. - Труды Санкт-Петербургской Государственной лесотехнической академии Актуальные проблемы развития высшей школы Санкт-Петербург. 2010. Стр. 308-309.

2. Васильев Н.П. Реализация алгоритма Форчуна расчета диаграммы Вороного на PHP. В сб. "Информационные системы и технологии: теория и практика" Сб. науч. Трудов. Вып.7 СПб, СПбГЛУ, 2015. с. 10-16.

3. Шубина М.А.Использование методов классификации космических изображений для мониторинга особо охраняемых территорий. В сб. "Информационные системы и технологии: теория и практика" Сб. науч. Трудов. Вып.5 СПб, СПбГЛУ, 2013. с. 51-60.

Н.П.Васильев, кандидат технических наук, доцент
Кафедра информационных систем и технологий
СПбГЛТУ им. С.М.Кирова
nikpv@mail.ru
Н.В.Лушкин, кандидат технических наук, доцент
Кафедра информационных систем и технологий
С ПбГЛТУ им. С.М.Кирова
Lushkin52@mail.ru

МОДЕЛИРОВАНИЕ МИКРОБИОЛОГИЧЕСКИХ СТРУКТУР (КОЛОНИЙ БАКТЕРИЙ) МНОГОУГОЛЬНИКАМИ ВОРОНОГО

В статье обсуждается программная реализация моделирования графической структуры многоугольниками Вороного биологических объектов на примере колоний бактерий, что расширяет представление о зонах их развития. В данной работе микробиологическим объектом исследования является бактерии рода (Actinomyces).

Актиномицеты — микроорганизмы, имеющие признаки и бактерий, и грибов. По строению и биохимическим свойствам актиномицеты аналогичны

бактериям, а по характеру размножения, способности образовывать гифы и мицелий похожи на грибы (рис.1).

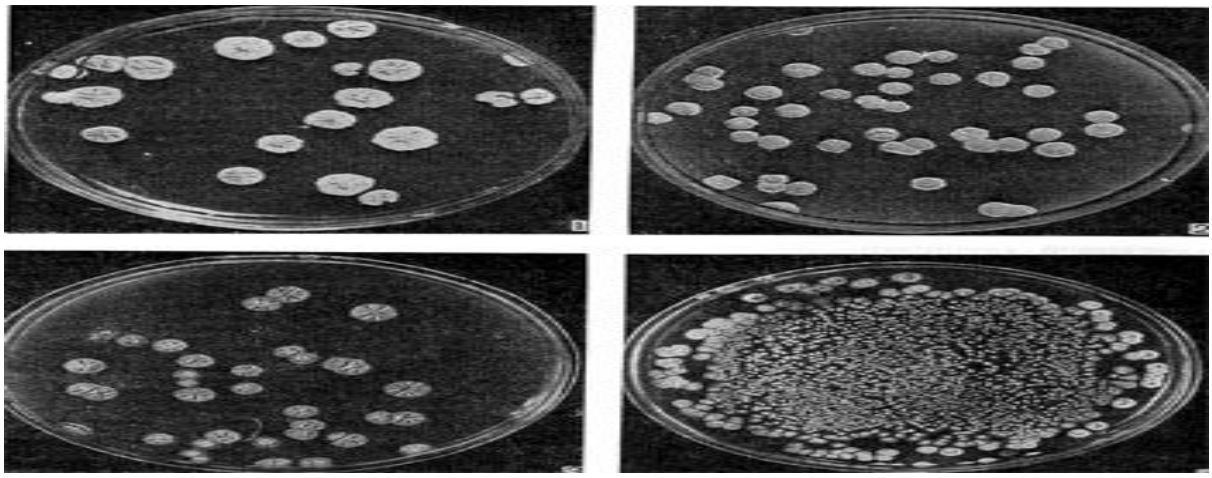


Рис.1. Колонии актиномицетов

Многоугольники Вороного получаются при разбиении плоскости, на которой заданы произвольные точки, на соответствующее количество областей. Каждая такая область - это множество точек плоскости наиболее близких к соответствующей точке (рис.2*)*.

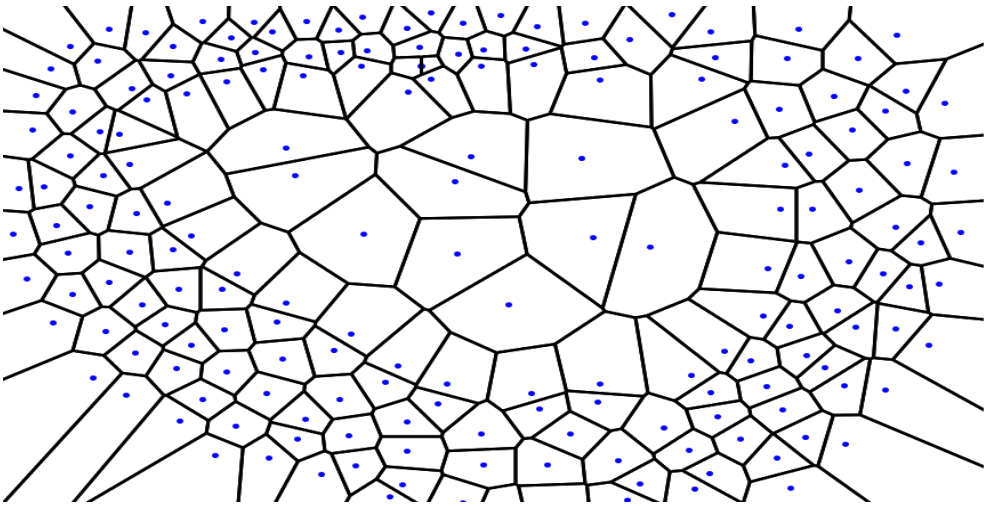


Рис. 2. Многоугольники Вороного

Необходимо построить наиболее близкую модель Вороного, для графических изображений колоний (рис.3), при этом необходимо определить: количество бактерий в каждом многоугольнике и их площади, т.е. область их развития.

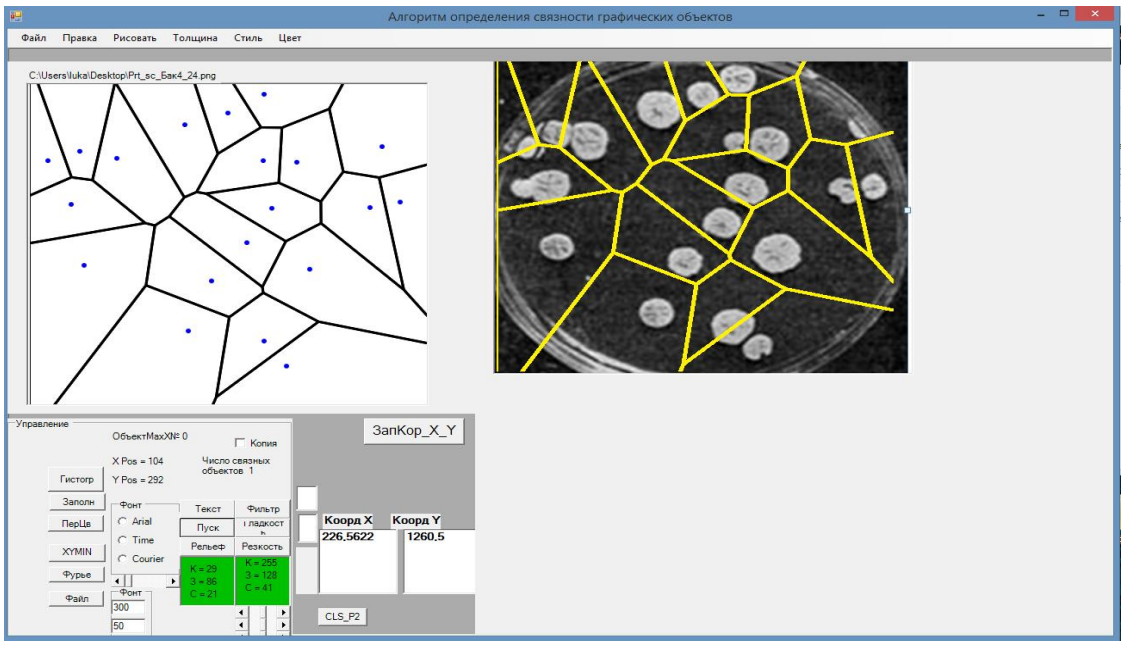


Рис. 3. Моделирование микробиологических структур (колоний бактерий) многоугольниками Вороного

Используя алгоритм определения связности графических объектов [3], определяем координаты центров связных объектов (рис.4.).

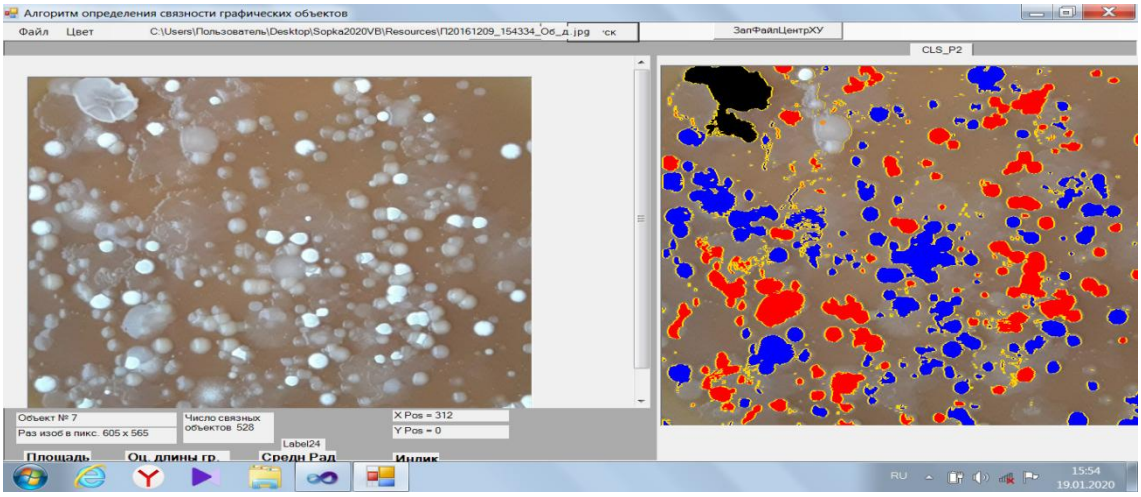


Рис. 4. Определение центров связных объектов

Получение координат связных объектов и построение многоугольников Вороного. Текстовый файл координат связных областей по координатам *x* и *y* графического файла:

{38 33 88 24 ..... 529 104 387 100 201 108 21 100 726 104 344 124}

Используя программу на языке PHP, реализующую алгоритм Форчуна строим многоугольники Вороного, которые связаны с нахождением областей близости [4] (рис. 5).

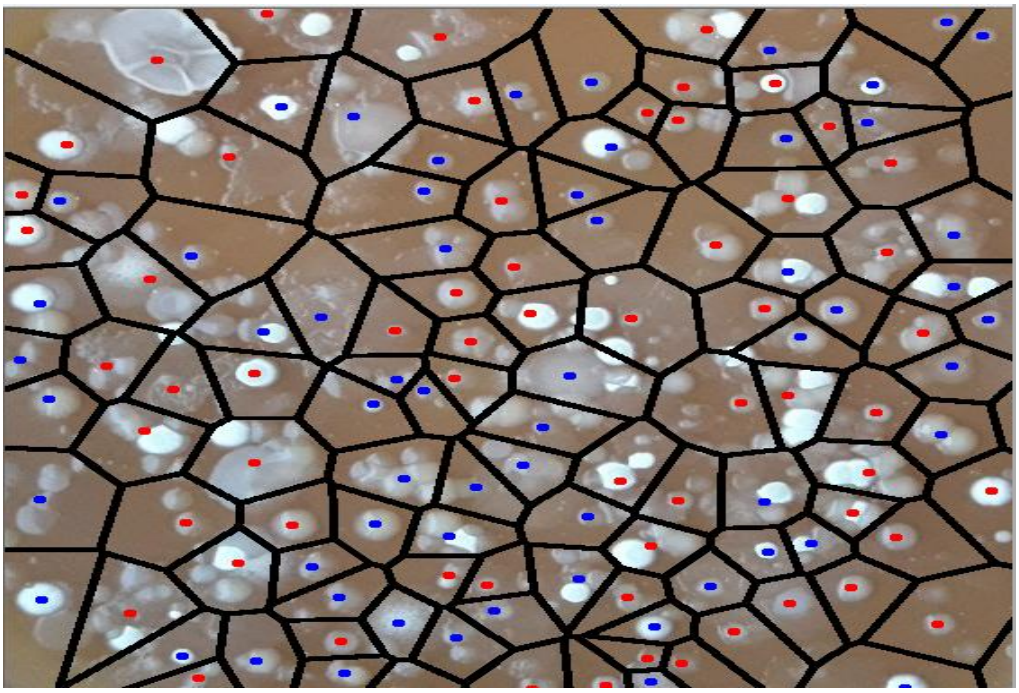


Рис. 5. Построение многоугольников Вороного

Необходимость перехода к компьютерным вычислениям

Несмотря на то, что методы серийных разведений являются наиболее точными и информативными, их постановка в практических лабораториях сопряжена со значительными трудностями.

Прежде всего, речь идет о необходимости ручного подсчета КОЕ (колониеобразующая единица) каждой чашки Петри с исследуемым образцом, что сильно сказывается на времени проведении исследования.

Помимо этого при изобилии контаминаций или КОЕ (колониеобразующая единица), ручной подсчет практически невозможен, что в дальнейшем приводит к повторному проведению опыта. Также необходимо учитывать погрешность, связанную с человеческим фактором.

Испытав теорию подсчета КОЕ, при помощи программы, нами были получены результаты сопоставимые с ручным подсчетом.

Описание алгоритма определения количества бактерий

Изображения бактерий могут быть как изолированными, так, и объединены в группы связных областей по несколько бактерий. Ставится задача оценить количество бактерий находящихся на изображении графического файла. Предлагается следующие этапы решения поставленной задачи:

1) отмечаем в левом окне цвет в исходном изображении (цвет бактерий);

2) включаем кнопку "ПУСК";

3) программа на правом окне изображает связные объекты(колонии бактерий) различными цветами;

4) последовательно для каждого связанного объекта программа определяет площадь и отображает на информационном окне;

5) последовательно отмечаем небольшие объекты(колонии бактерий), где возможно малое количество бактерий (до 10);

6) программа восстанавливает исходный цвет отмечаемого объекта и просит ввести количество наблюдаемых бактерий (рис.6.);

7) программа определяет среднюю площадь бактерии;

8) в зависимости от исходного графического файла и опыта пользователя, пункты 5, 6, 7 выполняем несколько раз;

9) отказываемся от ввода количества бактерий;

10) программа показывает количество отмеченных бактерий и рассчитывает среднюю площадь бактерии для вычисления общего количества бактерий;

11) включаем кнопку "РасчБак", чтобы на информационном окне отобразить общее количество бактерий в исходном графическом файле (рис.7).

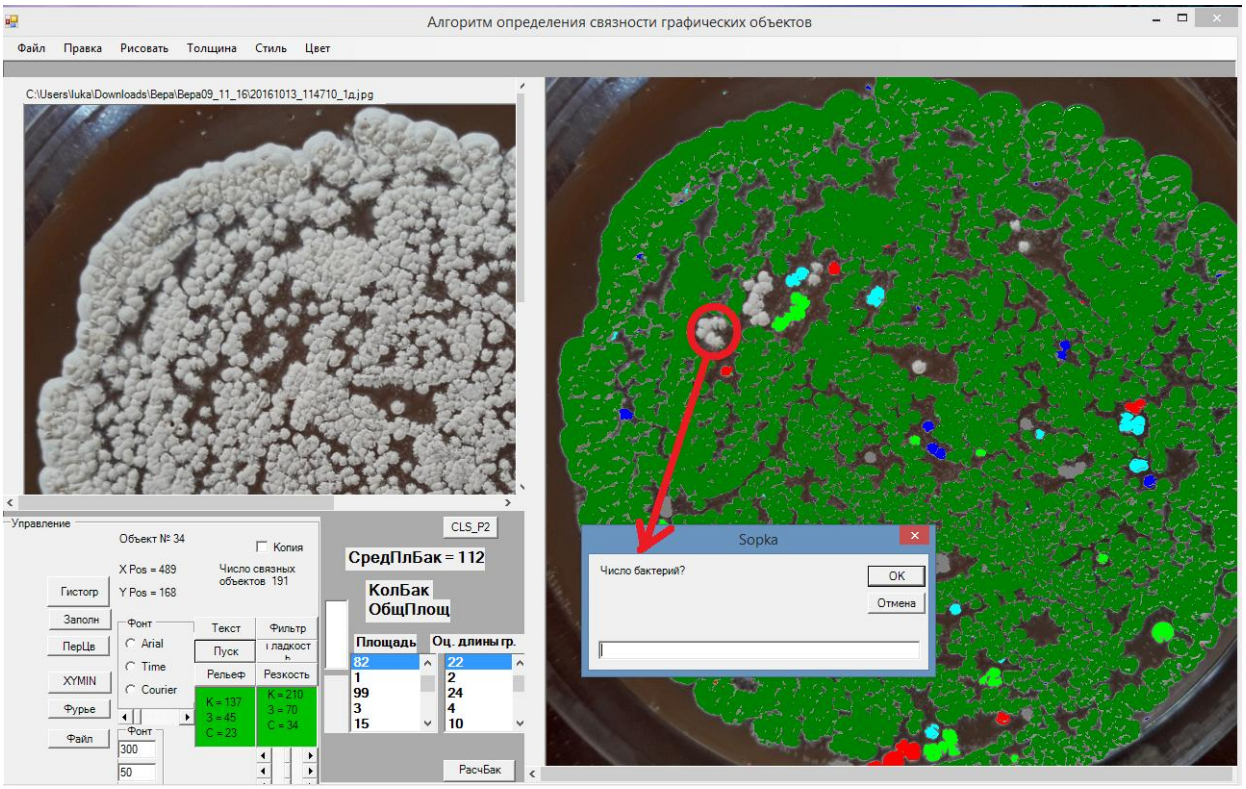


Рис. 6. Интерфейс программы (Демонстрация ручного ввода количества бактерий)

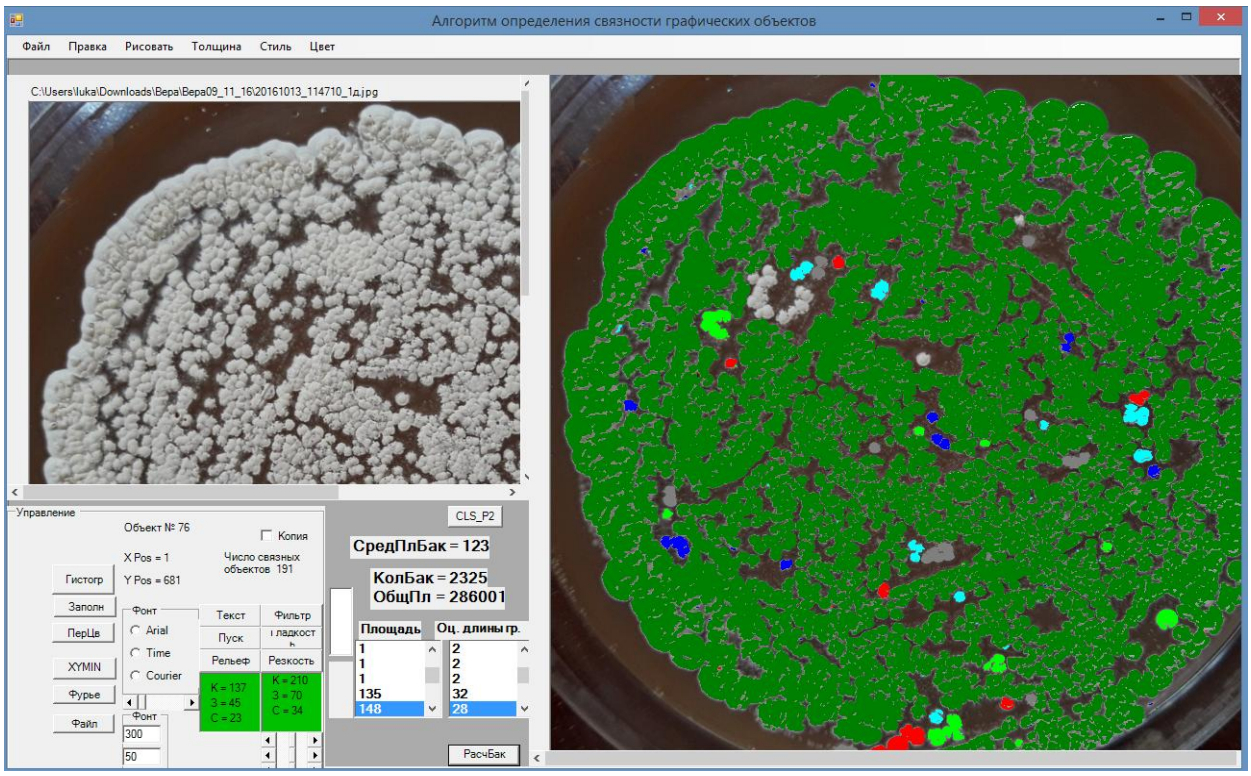


Рис. 7. Демонстрация расчета общего количества бактерий

Моделирование колоний бактерий многоугольниками Вороного позволяет значительно упростить обработку количества колоний бактерий (рис.8-рис.10).

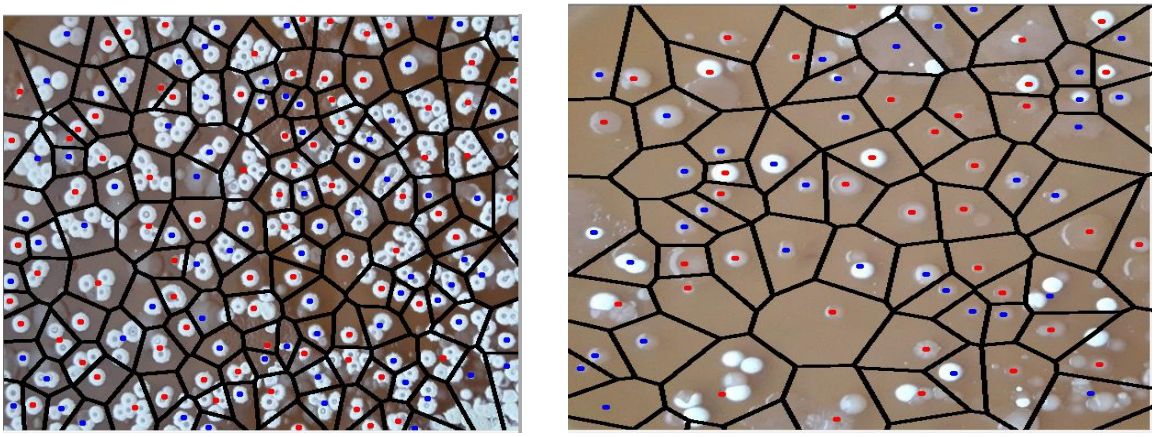


Рис. 8.Демонстрация работы программы моделирования колоний бактерий многоугольниками Вороного

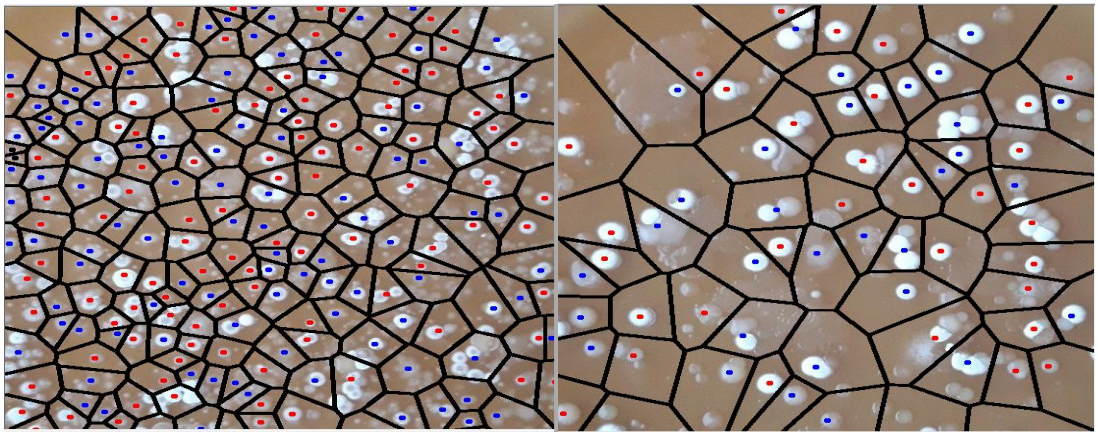


Рис .9.Демонстрация работы программы моделирования колоний бактерий многоугольниками Вороного

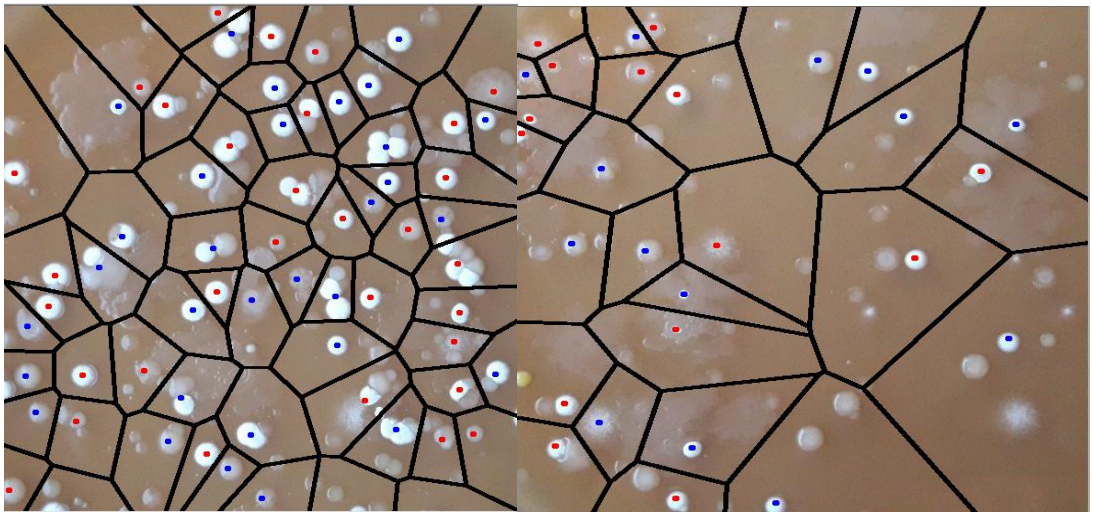


Рис. 10.Демонстрация работы программы моделирования колоний бактерий многоугольниками Вороного

Выводы

Данная компьютерная обработка колоний Actinomyces хорошо согласуется с результатами, полученными ручным подсчетом.

Заявленный подход при обработке графических файлов микробиологических объектов, позволяет выявить актуальные направления и обратить внимание на развитие компьютерного анализа характеристик микробиологических объектов.

Программа позволяет создать систему оперативной обработки и оценки характеристик развития колоний Actinomyces.

Библиографический список

1. Жизнь растений. Том 1. Введение. Бактерии и актиномицеты' \\Под редакцией члена-корреспондента АН СССР профессора Н. А. Красильникова и профессора А. А. Уранова - Москва: Просвещение, 1974 - с.487.

2. Дикий И.Л., Сидорчук И.И., Холупяк И.Ю. и др. Микробилогия: Руководство к лабораторным занятиям: Учебное пособие для студентов фармацевтических ВУЗов и фармацевтических факультетов медицинских институтов. // К.: ИД “Профессионал”, 2004. – С.

3. Лушкин Н.В. Алгоритм определения связности графических объектов. - Труды Санкт-Петербургской Государственной лесотехнической академии Актуальные проблемы развития высшей школы Санкт-Петербург. 2010. Стр. 308-309.

4. Васильев Н.П. Реализация алгоритма Форчуна расчета диаграммы Вороного на PHP. В сб. "Информационные системы и технологии: теория и практика" Сб. науч. Трудов. Вып.7 СПб, СПбГЛУ, 2015. с. 10-16.

5. Лушкин Н.В., Васильев Н.П. Моделирование древесных структур многоугольниками Вороного (ячейками Дирихле). В сб. "Информационные системы и технологии: теория и практика" Сб. науч. Трудов. Вып.9 СПб, СПбГЛУ, 2017. с. 43-53.

В.А. Горбачев кандидат экономических наук, доцент
Кафедра информационных систем и технологий
СПбГЛТУ им. С.М. Кирова
ist@spbftu.ru

ПРОЕКТИРОВАНИЕ ПРИЛОЖЕНИЙ В СРЕДЕ VISUAI STUDIO НА ПРИМЕРЕ РАЗРАБОТКИ ПРОЕКТА «УПРАВЛЕНИЕ КАФЕДРОЙ»

Visual Studio, в дальнейшем VS, является профессиональным инструментом для разработки приложений, и позволяет использовать объектно-ориентированное программирование для разработки как приложений в виде Windows Form, так и WEB приложений. Общие характеристики VS приведены в приложении 1. Задача, которую поставил автор, показать практические приемы разработки Windows Form приложения с использованием языка VB.Net. В данной работе отражается личный опыт программирования автора. В принципе, если сравнивать процедурно ориентированный подход к разработке программ, когда каждый программист (команда), нарабатывал свой комплекс процедур, реализующих некоторый функционал, и использование объектно-ориентированного подхода с применением библиотеки классов FCL, то богатство методов имеющихся в Framework .Net позволяет более эффективно разрабатывать приложения. Однако, главным условием успешной разработки приложения, в последнем случае, является хорошее знание классов и методов Framework .Net. Поскольку библиотека обладает большой избыточностью методов, то есть один и тот же алгоритм можно реализовать различными методами, то на этапе освоения такого инструмента, как VS, по-

лезно использовать некий целостный пример разработки проекта. Кроме того, такой подход полезен еще и потому, что попытки поиска в интернете поиска некоторого функционала часто бывают долгими и мало успешными. Поэтому автор делится своим опытом в разработке приложения на VB.Net**.** В принципе, для программирования на C# необходимо привыкнуть к его синтаксису, поэтому может быть полезно использование материала данной публикации для программирования на C# .

Для изложения поставленных задач удобно, в качестве примера, использовать разработку какого-либо проекта. Рассмотрим возможности Visual Studio для создания приложения «Управления кафедрой» с использованием Windows Form. Для первого варианта приложения определим функциональность как печать списков студентов, учет посещаемости занятий и подготовка графиков сессий.

Разработка проекта

В соответствии с указанной функциональностью можно предложить следующую схему проекта (рис. 1).

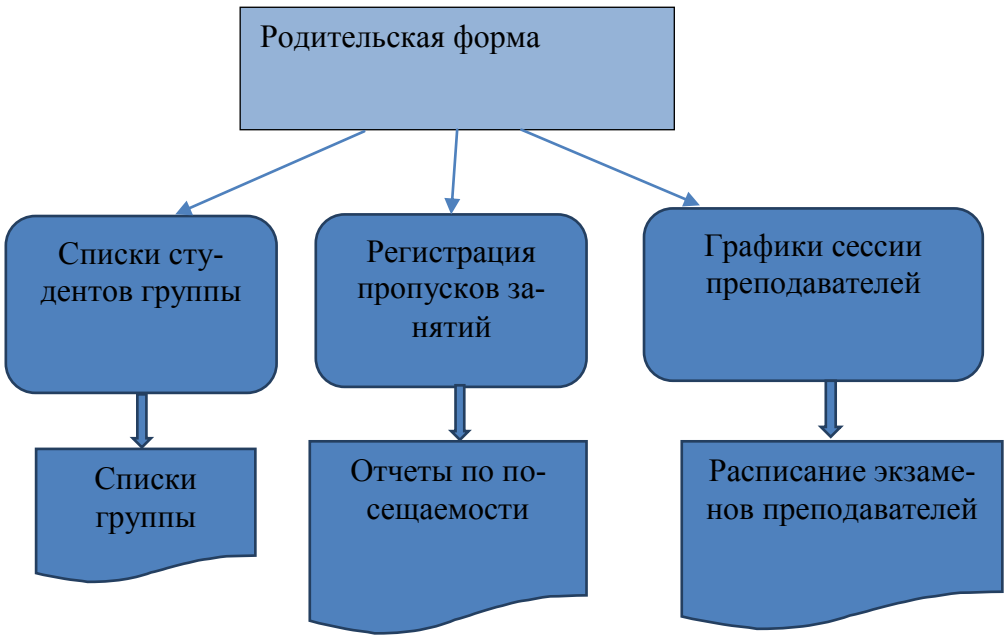


Рис.1. Схема проекта

В родительской форме должны выбираться учебный год, институт и кафедра. Причем, институт и кафедра должны настраиваться с помощью параметра, который указывается во внешнем для приложения файле. Это необходимо при установке приложения на компьютеры кафедр. Кроме того, в родительской форме должно быть обращение к форме настройки привязки к базе данных.

Рассмотрим некоторые особенности разработки программной реализации проекта.

Проект назван как *UprKafedra* . Для создания родительской формы проекта StartKaf в VS применяется MDI форма (замена параметра IsMdiContainer на True). Затем на форму добавляется элемент MenuStrip, ко-

торый позволяет организовать меню вызова форм реализовывающих указанный выше функционал. После добавления в MenuStrip пунктов меню, необходимо разместить на форме поля, позволяющие выбрать учебный год, институт и кафедру. Причем институт и кафедру нужно привязать к настройкам, которые хранятся в CSV файле. В итоге родительская форма имеет следующий вид (рис.2).

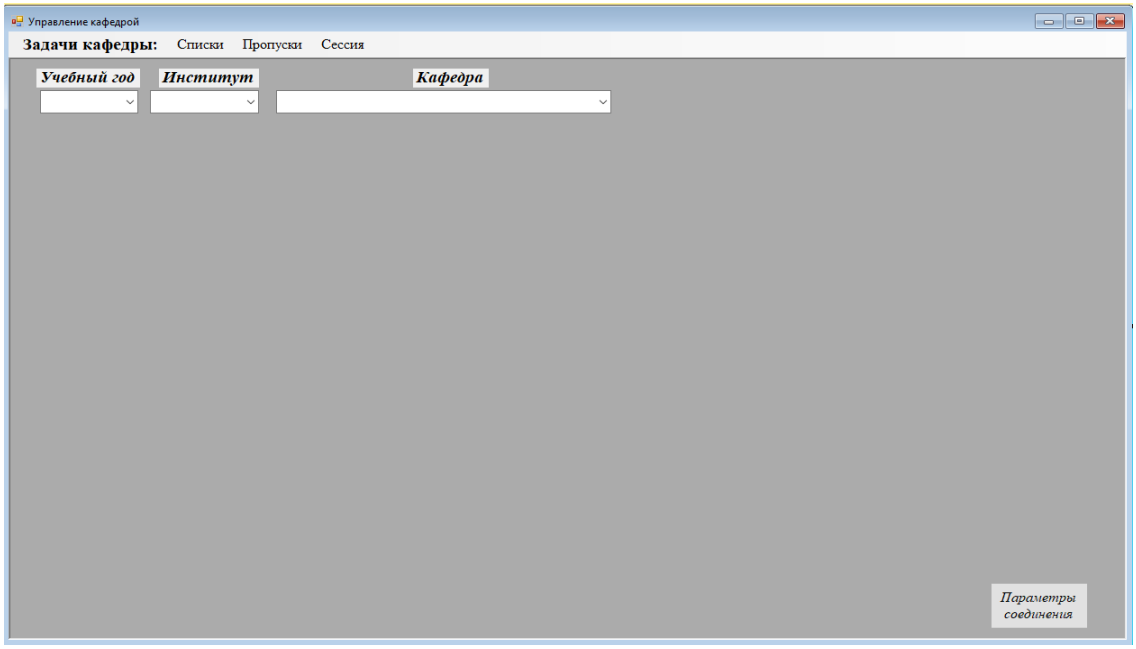


Рис. 2. Вид родительской формы

Поскольку основным источником данных приложения будет являться база данных SQL Microsft «Деканат», при разработке проекта необходимо сделать привязку к БД, что делается с помощью кнопки «*Параметры соединения*».

В Visual Studio имеются два подхода организации работы с данными таблиц БД: *Организация и использование таблиц связанных с внешней средой в DATASET* и с помощью *Language-Integrated Query (LINQ),* которые описаны в приложениях 1 и 2. В проекте используются оба подхода, чтобы продемонстрировать их возможности.

Для организации использования таблиц БД в с применением *LINQ* используется компонент dbml. В проекте он называется LINQKaf.dbml. Кроме того создан компонент PosZanDataSet.xsd, который организует подключение к БД через технологию *DATASET*.

Ниже представлены таблицы БД подключенные через LINQKaf.dbml (рис. 3).

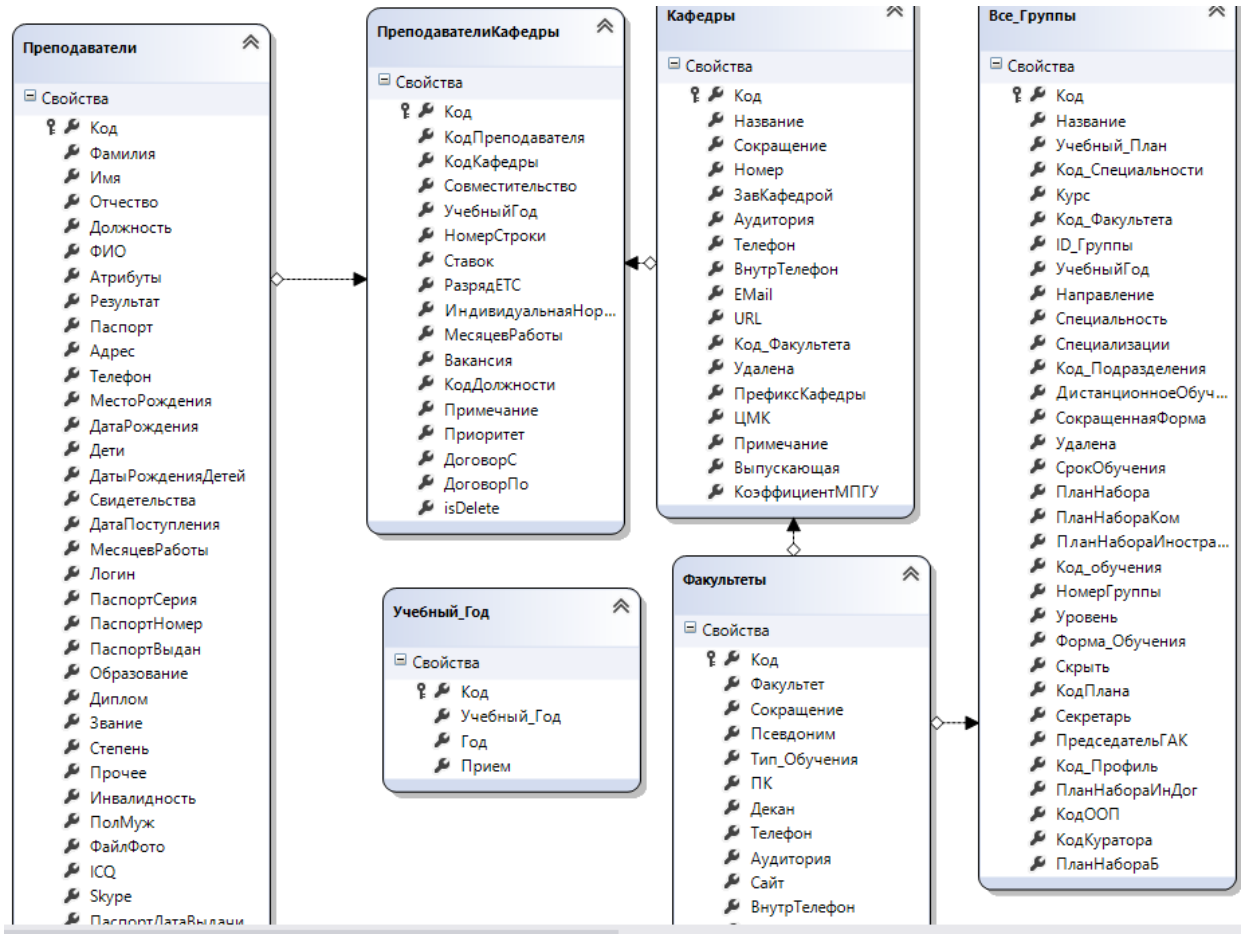


Рис. 3. Таблицы БД

- Рассмотрим использование данных этих таблиц для решения задачи заполнения родительской формы. Последовательно решим следующие задачи:
- Загрузим из csv файла коды факультета и кафедры, для настройки приложения на кафедру, где будет установлено приложение.
- Заполним список учебных годов в элемент управления UchGodComboBox.
- Заполним название института (факультета) в элемент управления InsComboBox.
- Заполним название кафедры в элемент управления KafComboBox.

Коды института (факультета) и кафедры хранятся в CSV файле с именем inkaf.csv в папке проекта и имеют примерно такой вид (рис.4).

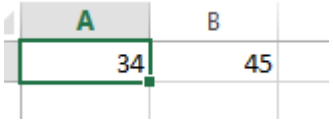


Рис. 4. Коды института и кафедры

Используемый ниже фрагмент кода позволяет решить задачу заполнения исходными данными элементы управления родительской формы.

```
    ' объявим пременную компонента DataContext для использоания
    ' таблиц LINQToSQL
    Dim LinkDB As New LINQKafDataContext

' присвоим строковой переменной TekPath путь к текущей папке
        Dim TekPath As String =
My.Computer.FileSystem.CurrentDirectory
        ' теперь уберем из полученного пути последние 31 символ
        ' внимание в VB.Net функция Left не работает, поэтому
        ‘ используем TekPath.Substring
        TekPath = TekPath.Substring(0, TekPath.Length - 31)
        ' этот оператор позволяет считать данные из файла inkaf.csv в
массив строк CSV()
        Dim CSV() As String = IO.File.ReadAllLines(TekPath +
"\inkaf.csv",
        System.Text.Encoding.Default)
        ' теперь объявим массив для двух целых чисел
        ' для хранения кодов института и кафедры
        Dim Cody(1) As Integer
        ' счетчик индексов массива
        Dim NumCod As Integer = 0
        ' и цикл выборки из 0 строки массива CSV(0) кодов, разделенных
;
        For Each c In CSV(0).Split(";")
            ' с занесением в элементы массива
            Cody(NumCod) = c
            NumCod = NumCod + 1
        Next
        ' сформируем индекс года для поиска текущего учебного года в
таблице Учебный_Год
        Dim NachGod As Integer = Year(Date.Today) -
IIf(Month(Date.Now) < 8, 2001, 2000)
        ' а вот теперь запрос LINQToSQL
        ' который формирует список учебных годов из таблицы Учеб-
ный_Год
        Dim listyears = From god In LinkDB.Учебный_Год
                        Where god.Код <= NachGod Order By god.Код De-
scending
                        Select UchGod = god.Учебный_Год, CodUchG =
god.Учебный_Год
        ' и делает его источником данных для UchGodComboBox
        UchGodComboBox.DataSource = listyears
        ' соотетственно задаются переменные сформированные в запросе
UchGod и CodUchG
        ' в качестве отображаемого и значимого полей - здесь они оди-
наковы
        UchGodComboBox.DisplayMember = "UchGod"
        UchGodComboBox.ValueMember = "CodUchG"
```

```
        ' теперь запрос на выборку сокращенного названия института и
кода из таблицы Факультеты
        ' здесь в фильтре участвует нулвой элемент массива Cody -код
института
        Dim listins = From institut In LinkDB.Факультеты
                      Where institut.Код = Cody(0) And institut.ПК =
True
                      Select Sokr = institut.Сокращение, CodIns =
institut.Код
        ' и формирование исходных данных для InsComboBox
        InsComboBox.DataSource = listins
        InsComboBox.DisplayMember = "Sokr"
        InsComboBox.ValueMember = "CodIns"
        ' и наконец запрос на выборку  названия  и кода кафедры из
таблицы Кафедры
        ' здесь в фильтре участвует первый элемент массива Cody -код
кафедры
        Dim listkaf = From kafedra In LinkDB.Кафедры
                      Where kafedra.Код = Cody(1)
                      Select namekaf = kafedra.Название, CodKaf =
kafedra.Код
        ' и формирование исходных данных для KafComboBox
        KafComboBox.DataSource = listkaf
        KafComboBox.DisplayMember = "namekaf"
 KafComboBox.ValueMember = "CodKaf"
```

Представленный выше фрагмент обслуживает родительскую форму. В нем представлены операторы обеспечивающие считывание данных из файла настройки на кафедру, где устанавливается приложение, и инициализацию полей родительской формы. Семантика действий операторов отражена в комментариях.

Родительская форма содержит меню отражающее ссылки на формы реализующие перечисленные выше задачи. В каждой из них есть свои программные решения, которые отражают те или иные аспекты программирования. Первая из форм "Списки" позволяет преподавателю получить печатную форму списка группы для дальнейшей регистрации посещаемости (аналог журнала). Для данного проекта она имеет следующий вид.

В данной форме представляет интерес организация программного решения дерева групп, вывод списка студентов на экран и организация вывода печатной формы с использованием FastReport.

Для организации дерева групп используются два стандартных элемента управления: TreeView и ImageList. Первый позволяет организовать в нем отображение дерева элементов, а второй содержит пиктограммы, которые используются для визуализации элементов дерева. Рассмотрим процесс формирования дерева групп на указанной форме. Вначале в форму добавим компонент ImageList и назовем его для программных нужд ImageTree, а элемент управления TreeView в программе назовем GruppyTreeView. Для формирова-

ния дерева групп создадим процедуру FormTree , которая имеет следующий вид (рис. 5).

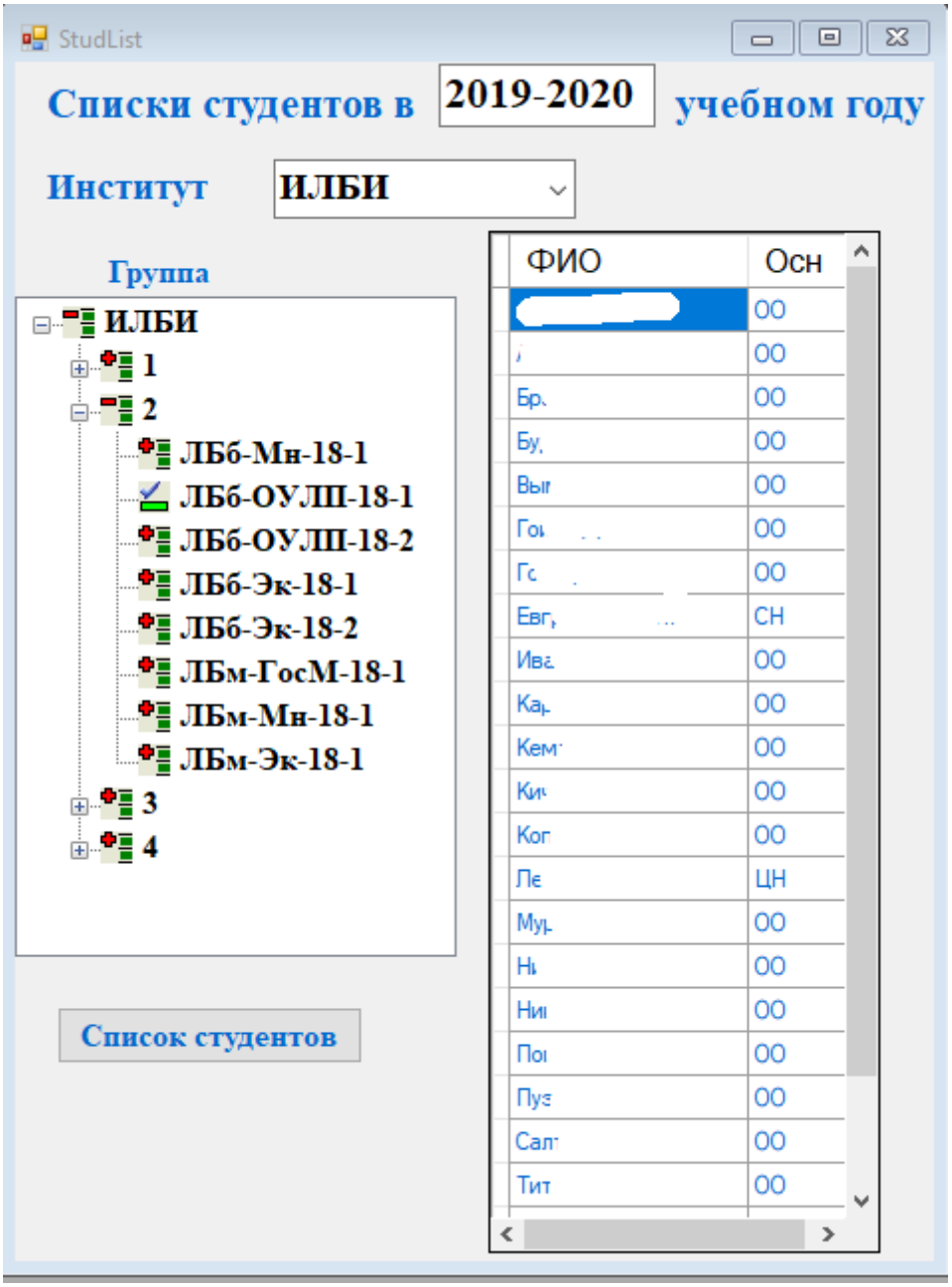


Рис. 5. Дерево групп

```
Private Sub FormTree()
        ' процедура создания дерева групп
        'TekPath хранит путь к папке PicTree , в которой хранятся три
bmp файла
        ‘ с изображениями элементов дерева. Эту папку лучше разместить
на диске C
        TekPath = "C:\DB\Kafedra"
        ‘ а теперь добавим эти файлы к компоненту ImageTree с именами
Im1, Im2 , Im3
```

```
        ImageTree.Images.Add("Im1", Image.FromFile(TekPath &
        "\PicTree\BlockHiddenFalse.bmp"))
        ImageTree.Images.Add("Im2", Image.FromFile(TekPath &
        "\PicTree\BlockHiddenTrue.bmp"))
        ImageTree.Images.Add("Im3", Image.FromFile(TekPath &
"\PicTree\Cheker.bmp"))
        ‘ добавим компонент ImageTree  в GruppyTreeView
        GruppyTreeView.ImageList = ImageTree
        ‘ инициализация GruppyTreeView
        ‘ индекс первой иконки = 0
        GruppyTreeView.ImageIndex = 0
        ‘ а ее номер среди загруженных иконок - второй
        GruppyTreeView.SelectedImageIndex = 2
        ‘ очистка списка элементов дерева
        GruppyTreeView.Nodes.Clear()
        ‘ создаем новое деревво имен с именем "Список групп"
        Dim FakNodes As TreeNode = New TreeNode("Список групп")
        ‘ объявим переменные KursNodes и GrupNodes для хранения имен
курсов и групп
        Dim KursNodes As TreeNode
        Dim GrupNodes As TreeNode
        ‘ в переменную CodF занесем выбранное значение кода факультета
(института)
        Dim CodF As Integer = InsComboBox.SelectedValue
        ‘ и если она больше 0
        If CodF > 0 Then
            ‘ сформируем LINKDB запрос для выборки списка групп фа-
культета
            Dim listgroup = From gruppa In LinkDB.Все_Группы
                        Where gruppa.УчебныйГод = UGod And
gruppa.Код_Факультета = CodF
                        Select КодГр = gruppa.Код, Группа =
gruppa.Название,
                               Курс=gruppa.Курс, FKT =
gruppa.Код_Факультета
            ' переменная с сокращением факультета
            Dim InstSokr = InsComboBox.Text
           ' добавление родительского узла - название факультета - в
           GruppyTreeView и    FakNodes
            FakNodes = GruppyTreeView.Nodes.Add(InstSokr)
            ' добавление в узел кода семейства для анализа при выборе
            FakNodes.Tag = CodF
           ‘ сформируем LINKDB запрос для выборки списка курсов фа-
культета
            Dim listkurs = From kursy In listgroup Where kursy.FKT =
CodF
                           Select НомерКурса = kursy.Курс Distinct Or-
der By НомерКурса
            ' цикл добавления в элементы дерева по курсам факультета
            For Each spKurs In listkurs
```

```
                ' добавления узла - создание дерева дочерних узлов пе-
ред добавлением
                ' создать новый элемент дерева для курса
                KursNodes = New TreeNode
                ' переменная с номером курса
                Dim nKurs = spKurs
                ' установить свойство Text
                KursNodes.Text = nKurs
                ' добавить курс в элемент дочернего узла
                FakNodes.Nodes.Add(KursNodes)
                ‘ сформируем LINKDB запрос для выборки списка групп на
курсе факультета
                Dim listgroupkurs = From gruppy In listgroup
                                    Where gruppy.FKT = CodF   AndAlso
                                    gruppy.Курс = nKurs Order By
gruppy.Группа
                ' цикл по добавлению в дерево групп курса
                For Each spGrupp In listgroupkurs
                    ‘ добавим в дерево новый элемент для группы
                    GrupNodes = New TreeNode
                    ‘ сформируем текс и тэг элемента
                    GrupNodes.Text = spGrupp.Группа
                    GrupNodes.Tag = spGrupp.КодГр
                    ' и добавим дочернего узла для групп
                    KursNodes.Nodes.Add(GrupNodes)
                Next
            Next
        End If
    End Sub
```

Все, процедура готова. Теперь для формирования дерева списка групп нужно написать обращение к процедуре и организовать отклики на действия с элементом управления GruppyTreeView. Чаще всего для обслуживания TreeView используются три события: AfterExpand (после разворачивания узла), AfterCollapse (после сворачивания узла) и AfterSelect (после выбора узла). Процедуры обслуживания событий выглядят следующим образом.

Реакция на события с деревом:

При разворачивании узла (клик мыши на значке + рядом с элементом дерева или двойной клик на самом элементе), если выбранный уровень дерева 0 или 1 (институт или кус) на дереве соответствующего узла отображается и помещается в фокус иконка с именем Im2 .

```
    Private Sub GruppyTreeView_AfterExpand(sender As Object, e As
TreeViewEventArgs) Handles GruppyTreeView.AfterExpand
        Select Case e.Node.Level
            Case 0, 1
                e.Node.ImageKey = "Im2"
                e.Node.SelectedImageKey = "Im2"
                Return
        End Select
```

```
    End Sub
```

При сворачивании узла ему назначается иконка с именем Im1.

```
Private Sub GruppyTreeView_AfterCollapse(sender As Object, e As
TreeViewEventArgs) Handles GruppyTreeView.AfterCollapse
        Select Case e.Node.Level
            Case 0, 1
                e.Node.ImageKey = "Im1"
                e.Node.SelectedImageKey = "Im1"
                Return
        End Select
End Sub
```

При выборе группы выполняются следующие действия

```
    Private Sub GruppyTreeView_AfterSelect(sender As Object, e As
TreeViewEventArgs) Handles GruppyTreeView.AfterSelect
        ‘ если выбран элемент уровня курса, в переменной TKurs запоми-
наем курс
        ‘ для дальнейшего использования
        If e.Node.Level = 1 Then
            TKurs = e.Node.Text
        End If
        ‘ если выбран элемент уровня группы. запоминаем факультет (ин-
ститут), курс
        ‘ и код группы для дальнейшего использования
        If e.Node.Level = 2 Then
            CodF = InsComboBox.SelectedValue
            TKurs = e.Node.Parent.Text
            NGR = e.Node.Tag
            ‘ обращаемся к процедуре заполнения списка группы
            FillListStud()
        End If
     End Sub
```

Для вывода списка студентов на экран используется элемент управления DataGridView с именем ListGroupDataGridView. Процедура заполнения списка студентов выглядит следующим образом.

```
    Private Sub FillListStud()
       ‘ особенностью LINKSQL запроса является использование оператора
LET, с помощью
        ‘ которого формируется сокращенное ФИО студента
      Dim liststud = From stud In LinkDB.Все_Студенты Let ФИО =
      stud.Фамилия & " " &     stud.Имя.Substring(0, 1) & "." &
      stud.Отчество.Substring(0, 1) & "." _
        Order By ФИО Where stud.Код_Группы = NGR Select ФИО, Осн =
stud.Основания
        ‘ делаем запрос источником данных для ListGroupDataGridView
        ListGroupDataGridView.DataSource = liststud
        ListGroupDataGridView.Columns(0).Width = 120
        ListGroupDataGridView.Columns(1).Width = 50
     End Sub
```

И, наконец, обращение к печати списка студентов с помощью шаблона FastReport при нажатии кнопки Button1.

```
    Private Sub Button1_Click(sender As Object, e As EventArgs) Handles
Button1.Click
        TekPath = "C:\DB\Kafedra"
        ' загрузка шаблона с именем СписокСтудентовГруппы.frx
        ListStReport.Load(TekPath & "\СписокСтудентовГруппы.frx")
        ' передача строки соединения с БД шаблону в качестве параметра
        ListStReport.SetParameterValue("ConString",
АУПСпискиПосещаемостиTableAdapter.Connection.ConnectionString)
        ' передача кода группы шаблону в качестве параметра
        ListStReport.SetParameterValue("CodGr", NGR)
        ' визуализация шаблона
        ListStReport.Show()
    End Sub
```

Следующая форма предназначена для регистрации посещаемости занятий (рис. 6). Она является дополнением к предыдущей и позволяет регистрировать посещаемость занятий в базе для дальнейшего анализа посещаемости.

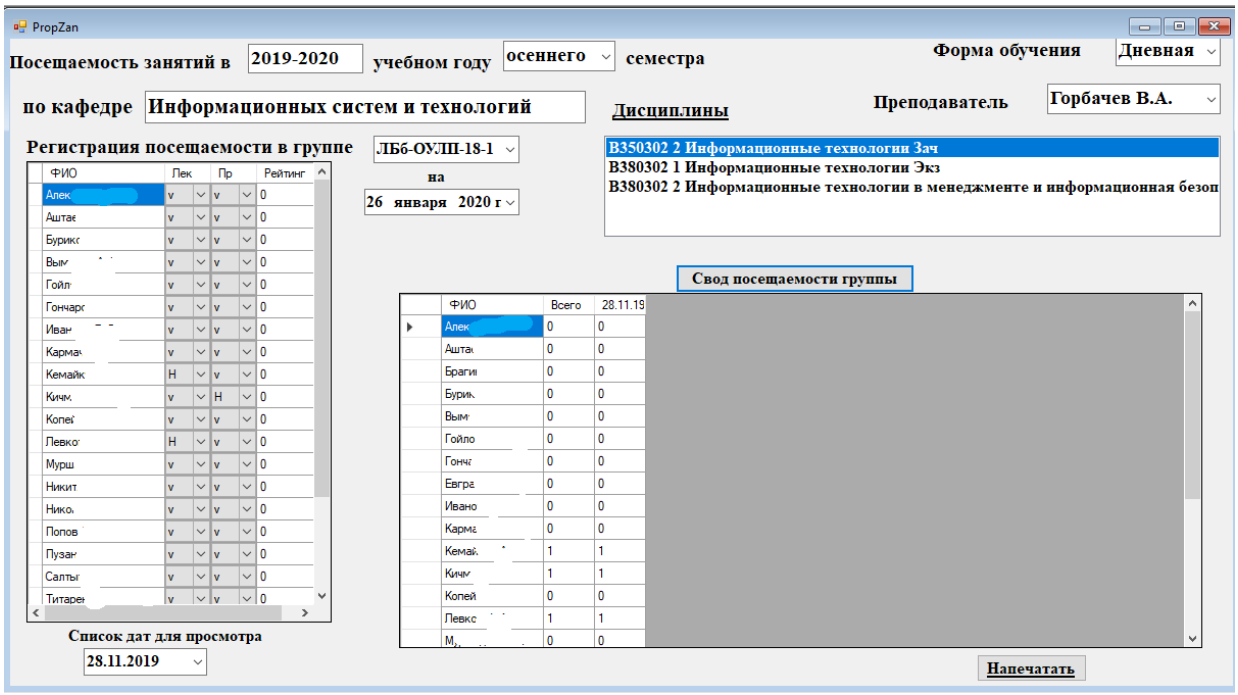


Рис. 6. Форма для регистрации посещаемости занятий

Отметим особенности программной реализации формы. Основных две.

1. Источником данных для элемента данных DataGridView при регистрации посещаемости является представление, а не таблица.

2. При отображении свода посещаемости используются сводные таблицы, для получения которых требуется специальная хранимая процедура.

Замечания по первой особенности. Для отображения списка регистрации посещаемости используется элемент DataGridView с именем RegPosDataGridView, источником данных которого является таблица из DataSet, а не одна из таблиц *LINQ* SQL. Это обусловлено тем, что таблицы из DataSet удобнее привязывать и настраивать к DataGridView через объект DataAdapter посредством компонента BindingSource. DataAdapter таблицы при его создании включает SQL запросы, позволяющие выбирать, обновлять и добавлять данные таблицы в БД. Однако это работает в отношении таблиц базы. С представлениями сложнее. Не для всех представлений допускаются операции обновления и добавления.

Регистрация посещаемости ведется в таблице АУППосещаемость базы данных. В ней студент представлен кодом студента, а в RegPosDataGridView нужно отобразить ФИО. Есть два способа сделать это. Создать в DataGridView вычисляемое поле или создать представление в БД с решением этой проблемы. В данном проекте был выбран второй вариант, так как хотелось разобраться в использовании связанных таблиц в качестве источников редактируемых данных. Поэтому в БД было создано представление с именем АУПСпискиПосещаемости и добавлено в DataSet с именем PosZanDataSet, содержимое которого имеет следующий вид (рис. 7).

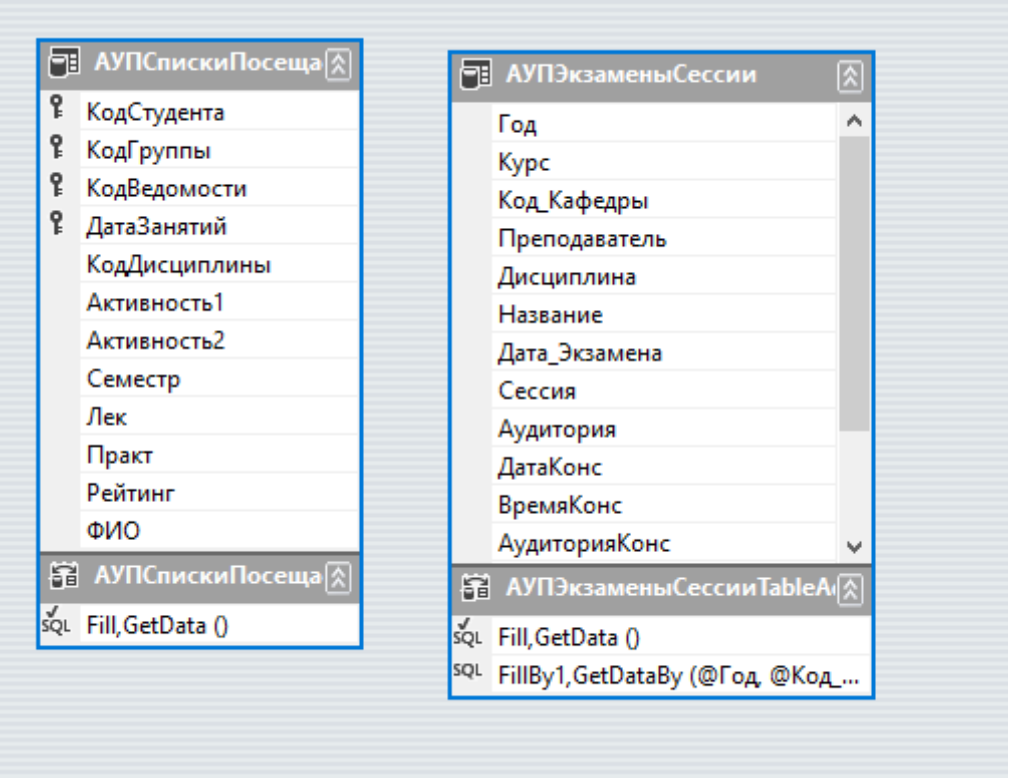


Рис. 7. Таблицы АУПСпискиПосещаемости и АУПЭкзаменыСессии

Вторая таблица потребуется в экранной форме «Сессия».

При использовании представления в качестве редактируемого источника данных настройка RegPosDataGridView производится стандартным спо-

собом. Для регистрации посещаемости на выбранную дату в таблице АУП-Посещаемость при задании новой даты необходимо создать записи с этой датой. То есть если при выборе даты для дисциплины группы отсутствуют соответствующие записи, их надо добавить в таблицу, а если они есть – отразить в RegPosDataGridView.

Для добавления записей используется следующий код. Здесь показаны возможности выполнения SQL запросов в VisualStudio. Код выполняется при выборе даты в элементе управления TimePicker.

```
    Private Sub RegPosDateTimePicker_CloseUp(sender As Object, e As
EventArgs) Handles RegPosDateTimePicker.CloseUp
        ‘ переменная strSQK для хранения текста запроса
        Dim strSQL As String
        ‘ содержимое поля RegPosDateTimePicker с выбранной датой от-
форматировать
        ‘ и строковую переменную для фильтра
        Dim DateTXTFltr As String = Format(RegPosDateTimePicker.Value,
"MM\/dd\/yyyy")
        ‘ содержимое поля RegPosDateTimePicker с выбранной датой от-
форматировать
        ‘ и строковую переменную добавления в таблицу
        Dim DateTXTInsert As String = For-
mat(RegPosDateTimePicker.Value, "dd\/MM\/yyyy")
        ‘ сформируем текст запроса для поиска записей с выбранной да-
той
        ‘ для ведомости группы и дисциплины
        Dim SQLStr As String = "SELECT КодГруппы, КодВедомости, Дата-
Занятий, КодДисциплины, Семестр FROM АУПСпискиПосещаемости " _
 & "GROUP BY КодГруппы, КодВедомости, ДатаЗанятий, КодДисциплины, Се-
местр HAVING  (КодВедомости = " & CodVed & " And ДатаЗанятий = '" &
DateTXTInsert & "')"
‘ если сформированный запрос возвращает 0 записей (функция MeSQLTable
показана ниже)
If MeSQLTable(SQLStr,
АУПСпискиПосещаемостиTableAdapter.Connection.ConnectionString).Rows.Co
unt = 0 Then
  ‘ то спрашиваем разрешение на добавление записей
  If MsgBox("Сейчас будет создан список посещаемости группы по выбран-
ному предмету и дате. Вы действительно хотите сделать это?",
MsgBoxStyle.YesNo) = vbYes Then
‘ формирование запроса на добавление записей в таблицу АУПСпискиПосе-
щаемости
strSQL = "INSERT INTO АУПСпискиПосещаемости ( КодСтудента, КодГруппы,
КодДисциплины, ДатаЗанятий, Семестр, КодВедомости )" _
& " SELECT Код,  " & NGR & " , 1 , '" & DateTXTInsert & "',  " &
CodSem & "," & CodVed _
& " FROM Все_Студенты  WHERE (((Все_Студенты.Код_Группы)=" & NGR &
"));"
‘ выполнение запроса на добавление записей с помощью процедуры
MeExecuteNonQuery
```

```
 MeExecuteNonQuery(strSQL,
АУПСпискиПосещаемостиTableAdapter.Connection.ConnectionString)
 ‘ загрузить в TableAdapter из БД полученные записи
Me.АУПСпискиПосещаемостиTableAdapter.Fill(Me.PosZanDataSet.АУПСпискиПо
сещаемости)
 End If
 End If
‘ и, наконец, установим филтр для АУПСпискиПосещаемостиBindingSource
‘ который является источником данных для RegPosDataGridView
 АУПСпискиПосещаемостиBindingSource.Filter = "КодВедомости =" & CodVed
& " And ДатаЗанятий = #" & DateTXTFltr & "#"
  SaveButton.Visible = True
End Sub
```

Для данной процедуры необходимы вспомогательные функции, на которые указаны ссылки в тексте выше. Они являются широко распространенными стандартными процедурами. Приведем их текст.

```
    Public Shared Function MeSQLTable(ByVal query As String, ByVal
ConStr As String) As DataTable
        Dim dTable As DataTable = New DataTable()
        If query = "" Then
            Return dTable
        End If
        Dim dbConnection As SqlConnection = New SqlConnection(ConStr)

        Dim dAdapter As SqlDataAdapter = New SqlDataAdapter(query,
dbConnection)
        dbConnection.Open()
        dAdapter.Fill(dTable)
        dbConnection.Close()
        Return dTable
    End Function

    Public Shared Function MeExecuteNonQuery(ByVal query As String,
ConStr As String) As Boolean
        Dim dbConnection As SqlConnection = New SqlConnection(ConStr)
        Dim cmd As SqlCommand = dbConnection.CreateCommand()
        cmd.CommandText = query
        dbConnection.Open()
        If cmd.ExecuteNonQuery() < 1 Then
            dbConnection.Close()
            Return False
        End If
        dbConnection.Close()
        Return True
    End Function
```

Замечания по второй особенности. Для получения сводной таблицы, отражающей итоги посещаемости по дисциплине в выбранной группе необходимо транспонировать данные таблицы АУППосещаемость. В SQL версии СУБД Access есть замечательный оператор TRANSFORM …. PIVOT, который решает эту задачу универсальным способом. В Transact SQL есть похо-

жий оператор PIVOT, но для его использования необходимо ряд ухищрений. Одним из них является создание хранимой процедуры в БД и ее использование в программе. В интернете была найдена подходящая процедура «SP_Dynamic_Pivot». Ниже показано ее использование в программе.

```
   Private Sub SvodGrButton_Click(sender As Object, e As EventArgs)
Handles SvodGrButton.Click
‘ согласно описанию обращения к хранимой процедуре SP_Dynamic_Pivot
‘ требуется подготовить следующие параметры
‘ @TableSRC --Таблица источник (Представление)
 '@ColumnName --Столбец, содержащий значения, которые станут именами
столбцов
 '@Field --Столбец, над которым проводить агрегацию
 '@FieldRows --Столбец (столбцы) для группировки по строкам (Column1,
Column2)
 '@FunctionType,--Агрегатная функция (SUM, COUNT, MAX, MIN, AVG), по
умолчанию SUM
 '@Condition --Условие (WHERE и т.д.). По умолчанию без условия
  Dim ParamPivot() As Object = {"@TableSRC",
"[dbo].[АУПСводПосещаемости]", "@ColumnName", "ДатаПосещаемости",
"@Field", "Пропуски", "@FieldRows", "ФИО, Всего", "@FunctionType",
"SUM", "@Condition", "WHERE КодВедомости = " & CodVed}
‘ сделать источником данных для SvodPosDataGridView результат выполне-
ния
‘ хранимой процедуры
SvodPosDataGridView.DataSource =
MeGetStoredProcedure("SP_Dynamic_Pivot",
АУПСпискиПосещаемостиTableAdapter.Connection.ConnectionString,
ParamPivot)
‘ установить ширину столбцов SvodPosDataGridView равной 50
        For i As Integer = 1 To SvodPosDataGridView.ColumnCount - 1
            SvodPosDataGridView.Columns.Item(i).Width = 50
        Next
  End Sub
```

Для выполнения хранимой процедуры используется следующая функция

```
  Public Shared Function MeGetStoredProcedure(ByVal procedureName As
String, ByVal ConStr As String, ByVal ParamArray Parametrs() As Ob-
ject) As DataTable
        Dim dbConnection As SqlConnection = New SqlConnection(ConStr)
        Dim dTable As DataTable = New DataTable()

        Dim cmd As SqlCommand = New SqlCommand(procedureName,
dbConnection)
        cmd.CommandType = CommandType.StoredProcedure
        Dim i As Integer
        For i = 0 To UBound(Parametrs) - 1 Step 2
            If Parametrs(i + 1) Is Nothing Then
                cmd.Parameters.AddWithValue(Parametrs(i).ToString(),
DBNull.Value)
            Else
```

```
                cmd.Parameters.AddWithValue(Parametrs(i).ToString(),
Parametrs(i + 1))
            End If
        Next
        Dim dAdapter As SqlDataAdapter = New SqlDataAdapter(cmd)
        dbConnection.Open()
        dAdapter.Fill(dTable)
        dbConnection.Close()

        Return dTable
    End Function
```

Следующей экранной формой проекта является форма планирования сессии на кафедре – «Сессия». В ней отражается информация о графике сессии, который формируется на уровне деканатов и служит для контроля прохождения сессии. Форма имеет следующий вид (рис. 8).

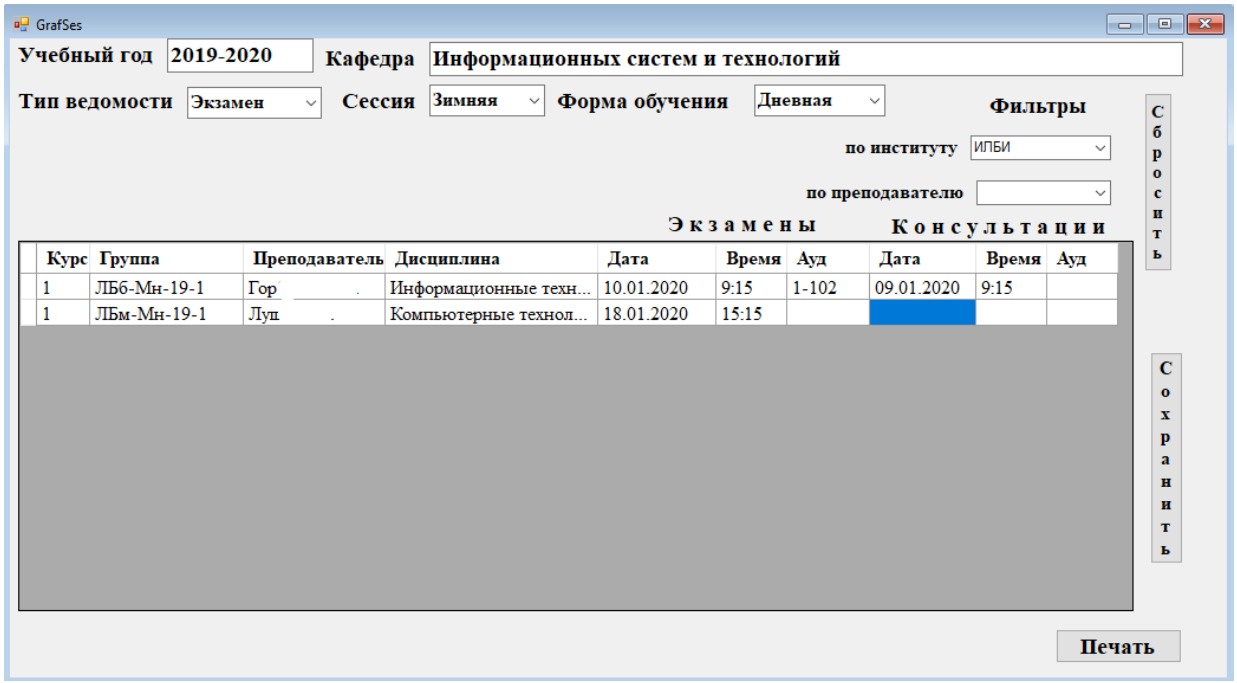


Рис. 8. График сессии

В ней отражены сведения об экзаменах (и других видах контроля) на кафедре, на которые может быть наложены фильтры по институтам и преподавателям. Дата и время экзамена устанавливаются в деканате. На кафедре определяются дата и время консультаций. С точки зрения программных решений занесение дат и времени в эту таблицу, а так же сохранение изменений в БД представляет интерес. Источником данных для таблицы экзаменов является представление АУПЭкзаменыСессии хранящееся в БД и реализованное через объект DataGridView с именем ExListDataGridView. Представление реализуется следующим SQL выражением, из которого видно, что оно основано на трех таблицах Все_Ведомости, Все_Группы и АУПРазмещение-Экзаменов:

```
SELECT dbo.Все_Ведомости.Год, dbo.Все_Ведомости.Курс,
dbo.Все_Ведомости.Код_Кафедры, dbo.Все_Ведомости.Преподаватель,
dbo.Все_Ведомости.Дисциплина, dbo.Все_Группы.Название,
dbo.Все_Ведомости.Дата_Экзамена, dbo.Все_Ведомости.Сессия,
dbo.АУПРазмещениеЭкзаменов.Аудитория,
dbo.АУПРазмещениеЭкзаменов.ДатаКонсультации AS ДатаКонс,
dbo.АУПРазмещениеЭкзаменов.ДатаКонсультации AS ВремяКонс,
dbo.АУПРазмещениеЭкзаменов.АудиторияКонс,
dbo.Все_Ведомости.Код_Факультета,
dbo.Все_Ведомости.Тип_Ведомости, dbo.Все_Группы.Форма_Обучения,
dbo.АУПРазмещениеЭкзаменов.[Код ведомости] AS КодВедомости,
dbo.Все_Ведомости.Код
FROM dbo.Все_Ведомости INNER JOIN
dbo.Все_Группы ON dbo.Все_Ведомости.Код_Группы = dbo.Все_Группы.Код
INNER JOIN
dbo.АУПРазмещениеЭкзаменов ON dbo.Все_Ведомости.Код =
dbo.АУПРазмещениеЭкзаменов.[Код ведомости]
```

В последней из таблиц хранятся данные об аудиториях экзаменов и консультаций, а так же дата консультаций. Такое выделение данных в отдельную таблицу обусловлено тем, что структура таблицы Все_Ведомости создана и используется разработчиком лаборатории ММИС и корректировки не подлежит. При добавлении этого представления в DataSet создается TableAdaptor представления (таблицы) с именем АУПЭкзаменыСессии-TableAdapter, в который входит строка ConnectionString, обеспечивающая подключение к БД; команды Select, Update и Delete обеспечивающие обмен данными с БД. Однако для этого представления автоматически создается только выражение Select, так как одновременное обновление данных во всех трех таблицах не допускается. Поскольку в контексте действий с данной таблицей необходимо менять данные только таблицы АУПРазмещениеЭкзаменов, в TableAdaptor добавляем SQL запрос.

```
UPDATE      АУПЭкзаменыСессии
SET         Аудитория = @Аудитория, ДатаКонс = @ДатаКонс, АудиторияКонс =
@АудиторияКонс, КодВедомости = @Код
WHERE       (КодВедомости = @Original_Код) AND (Год = @Original_Год) AND
(Код_Кафедры = @OriginaКод_Кафедры)
```

Как видно, он обновляет только поля, относящиеся к таблице АУПРазмещениеЭкзаменов.

После создания этого запроса, можно приступать к обсуждению настроек ExListDataGridView. Основные настройки этого объекта соответствуют стандартным. Однако при вводе даты и времени консультаций стандартных настроек не хватает. В частности, при вводе данных хотелось бы использовать класс выбора даты из календаря. Да, в списке элементов управления есть DateTimePicker, который поддерживает класс с использо-ванием календаря. Но в DataGridView он не поддерживается. Поэтому к нему был добавлен класс GridDateControl, решающий эту задачу. Другой задачей настроек являлось использование двух разных полей для отображения даты и времени консультаций. Поскольку и дата и время хранятся в одном поле таблицы, то

отобразить их в разных полях достаточно просто, задав различные форматы представления. При вводе даты проблем нет, она отображается как в календаре. При этом часть поля, относящаяся ко времени имеет значение 0:00. Если после выбора даты попытаться изменить время, то с изменением времени дата заменится на текущую системную. Решение этой проблемы было выполнено следующим образом.

```
' это событие возникает при входе в любое из полей Grid
Private Sub ExListDataGridView_CellEnter(sender As Object, e As
DataGridViewCellEventArgs) Handles ExListDataGridView.CellEnter
        ' если индекс столбца равен 11 и индекс строки больше -1
        ‘ то есть это столбец для ввода времени и строка имеет реаль-
ное значение
        If e.ColumnIndex = 11 And e.RowIndex > -1 Then
            ' запомнить дату из 10 столбца в символьной переменной
datestr
            ‘ то есть запомнить дату из столбца, где вводится дата
            datestr = Me.ExListDataGridView.Item(10,
e.RowIndex).Value.ToString
        End If

    End Sub
' это событие возникает при завершении редактирования полей Grid
Private Sub ExListDataGridView_CellEndEdit(sender As Object, e As
DataGridViewCellEventArgs) Handles ExListDataGridView.CellEndEdit
        ' если индекс столбца равен 11 и индекс строки больше -1
        If e.ColumnIndex = 11 And e.RowIndex > -1 Then
            ' после занесения времени в ячейку заносим дату время в
символьную
            ‘ переменную datehour
            Dim datehour As String =
ExListDataGridView.Item(e.ColumnIndex,
            e.RowIndex).Value
            ' из переменной datestr выделяем первые 10 символов отно-
сящихся к дате
            ' из переменной datehour выделяем последние 8 символов от-
носящихся
            ‘ ко времени и объединяем в переменной datestr
            datestr = Mid(datestr, 1, 10) + Mid(datehour, 11, 8)
            'теперь преобразуем символьную переменную datestr в формат
            ‘ даты и заносим в значение поля текущей ячейки
            Me.ExListDataGridView.Item(e.ColumnIndex,
e.RowIndex).Value = CDate(datestr)
        End If
End Sub
```

Материал данной статьи является основой для решения последующих задач. В разработки комплексного проекта приложения по управлению деятельностью кафедры можно добавить анализ посещаемости студентами занятий, анализ распределения и исполнения нагрузки и другое.

Библиографический список

1. В.А. Горбачев Автоматизация решения задач управления кафедрой Информационные системы и технологии: теория и практика. Сборник научных трудов. Выпуск 4. СПб 2012 г. с. 98-109.
2. В.А. Горбачев Проектирование программного модуля «Управление сессией» Лесной сектор России: проблемы и пути решения. Сборник научных трудов факультета экономики и управления, СПбГЛТУ 2014 г.
3. В.А. Горбачев Проектирование программного комплекса «Управление кафедрой» Экономические проблемы лесного сектора. Сборник научных трудов. », СПбГЛТУ, 2012 г.
4. В.А. Горбачев Вопросы проектирования информационных технологий при создании электронной информационно-образовательной среды СПбГЛТУ, 2018 г., 11 с.

Т.К. Екшикеев, кандидат экономических наук, доцент
СПХФУ
tager-ekshikeev@pharminotech.com
И.А. Обухова, кандидат технических наук, доцент
Кафедра информационных систем и технологий
СПбГЛТУ им.С.М.Кирова
lobukhova@inbox.ru

ОЦЕНКА ЭФФЕКТИВНОСТИ СИМУЛЬТАННОСТИ ИНТЕРАКТИВНЫХ ЛАБОРАТОРНЫХ ИССЛЕДОВАНИЙ В ФОРМАЦИИ НА ОСНОВЕ ИНФОРМАЦИОННЫХ МОДЕЛЕЙ СЕТЕВОГО ПЛАНИРОВАНИЯ И УПРАВЛЕНИЯ

Введение – актуальность исследования

Понятие «интерактив» пришло из английского языка и состоит из двух частей, «interact». «Inter» – «взаимный», «act» – действовать. Интерактивное исследование – это исследование на основе взаимодействия, постоянного нахождения в режиме сотрудничества. Востребована организация параллельных процессов исследований. Это возможно при соблюдении основных принципов организации интерактивных процессов:

– совместная работа;

– постоянный учет активного участия в работе;

– деление участников каждого исследования на независимые группы;

– перемещение в пространстве исследователей в процессе групповой работы;

– фиксация регламента и процедур каждого элемента исследования.

В целом, по мнению авторов, развитие интегративного взаимодействия тормозится отсутствием адекватных организационных механизмов, как накопленных, так и формируемых в настоящее время. То есть имеется актуальная необходимость последующей реализации потенциала знаний в области организационных и информационных механизмов интегративных исследований.

Большой потенциал актуальности выше изложенного – у высших учебных заведений. Так к 2016 г. была закончена реконструкция и открыт Центр превосходства по разработке инновационных лекарств и фармацевтических технологий Санкт-Петербургского химико-фармацевтического университета, в подразделениях которого, таких как Центр экспериментальной фармакологии, GMP тренинг-центр и других, осуществляется научно-исследовательская работа, в том числе и по заказу предприятий.

Цитируя ректора СПХФУ И.А. Наркевича [4]: «Задача центра – обучение основным навыкам работы на современном фармацевтическом предприятии студентов всех факультетов – и фармацевтического, и промышленной технологии лекарств. С одной стороны, они обучаются таким простым вещам, как переодевание в рабочую одежду, вход в чистое помещение, обработка помещений и т. д. С другой стороны, они отрабатывают на современном оборудовании ряд технологических навыков работы на производстве. Понятно, что не возможно охватить все технологии, которые существуют сейчас в фармацевтической отрасли, но основные из них GMP тренинг-центр охватывает. Студенты, прошедшие здесь обучение, приходят на производственную практику подготовленными. Будущим руководителям практики, а впоследствии и работодателям, не нужно тратить время на разъяснение им базовых основ – студенты и выпускники превосходно подготовлены к условиям работы на современном производстве». Авторы уверены, что студенты и выпускники вузов, освоившие вопросы оценки эффективности симультанности интерактивных лабораторных исследований в фармации, на основе информационных моделей сетевого планирования и управления будут приходить на производство подготовленными и способными решать поставленные задачи с высоким уровнем производительности труда [5].

Цель исследования

Одним из наиболее эффективных способов и механизмов организации любой деятельности – является универсальное сетевое планирование и управление. Цель исследования авторы сформировали следующим образом: представить оценку и результат эффективности симультанности при организации интегративного исследования в области фармации на основе сетевого планирования и управления.

Материалы и методы

Система сетевого планирования и управления (СПУ) есть комплекс графических и расчетных методов, организационных мероприятий, контрольных приемов, обеспечивающих моделирование, анализ и динамическую

перестройку плана выполнения совокупности процессов, в частности элементов исследований в области фармации [1, 2].

Основным документом в системе СПУ является сетевой график, т. е. информационно-динамическая модель, в которой представлены взаимосвязи и результаты всех работ, необходимые для достижения конечной цели совокупности процессов, в частности элементов исследований в области фармации.

В основе сетевого планирования лежит представление планируемого комплекса работ в виде ориентированного замкнутого графа, называемого сетевым графиком проведения работ. В нем детально или укрупнено показывается, в какой последовательности, когда (за какое время) и для чего необходимо выполнить работу, чтобы обеспечить окончание проектных или исследовательских работ не позже заданного (директивного) срока.

В сетевом графике имеются два основных элемента [1, 2, 3]: ребра, которые обозначают работу, и вершины, обозначающие события.

Работой называется любой процесс, действительно приводящий к достижению определенных результатов (событий).

Работы могут быть изображены сплошной, либо пунктирной линией со стрелкой. Сплошной линией со стрелкой представляется действительная работа, т. е. требующая затрат времени, измеряемая в данном исследовании в днях или неделях. Кроме работ действительных существуют так называемые фиктивные работы. Фиктивной работой (зависимостью) называется связь между какими-то результатами работ (событиями), не требующая затрат времени (пунктир на графике).

Событием называется результат произведенных работ (кружок на графике). Каждое событие является отправным моментом для начала последующих работ. В отличие от работы, которая имеет протяженность во времени, событие представляет только момент свершения работы или нескольких работ, предшествующих данному событию.

Любая последовательность работ в сетевом графике, в которой конечное событие одной работы совпадает с начальным событием следующей за ней работы, называется путем. Путь от начального события до завершающего называется полным путем. Как правило, полных путей в сетевом графике может быть несколько.

Полный путь, имеющий максимальную продолжительность выполнения всех работ, лежащих на этом пути, называется критическим. Любое незначительное увеличение срока выполнения одной из работ, принадлежащих этому пути, повлечет срыв сроков выполнения всей разработки в целом.

Для построения сетевого графика (рис. 1) необходимо знать основные понятия и обозначения, применяемые в сетевых моделях.

Начальное (исходное) событие I – момент начала выполнения задания (проекта), обычно имеется одно начальное событие.

Завершающее событие J – момент достижения конечной цели, обычно имеется одно завершающее событие.

Промежуточные события i и j (рис. 2). Работы характеризуются временем выполнения – t_{ij}, где t – длительность работы (часы, дни, в данной работе – недели); i – номер начального события работы (начало стрелки); j – номер конечного события работы (конец стрелки).

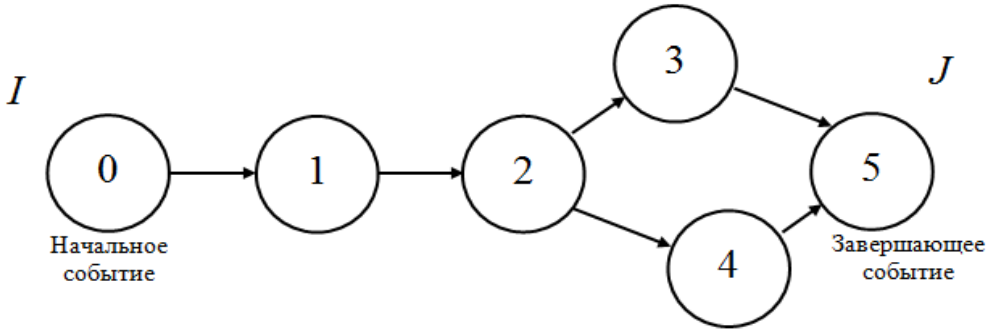


Рис. 1. Сетевой график

Продолжительность критического $L_{кр}$ и других полных путей L (i, j, …, k) равна сумме продолжительности составляющих работ: Σt_{ij} (см. рис. 1), L_1 (0, 1, 2, 3, 5), L_2 (0, 1, 2, 4, 5).

Ранний срок свершения события $t_{р(t)}$ равен максимальной продолжительности путей, предшествующих событию i: $t_{рi} = t_{[L(J,\ldots,i)\max]}$.

Поздний срок свершения события $t_{п(i)}$ равен разности между продолжительностью критического пути $L_{кр}$ и наибольшей из продолжительностей путей, следующих за событием i до завершающего: $t_{пi} = L_{кр} - t_{[L(i,\ldots,C)\min]}$.

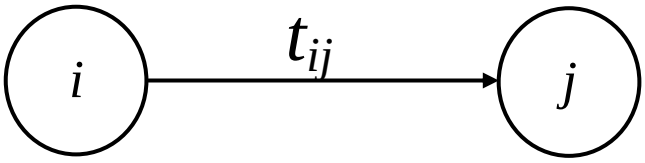


Рис. 2. Событие – работа – событие

Ранний срок начала работы $t_{рн(ij)}$ равен раннему сроку свершения i события: $t_{рн(ij)} = t_{рi}$.

Поздний срок начала работы $t_{пн(ij)}$ равен позднему сроку свершения j события этой работы минус продолжительность самой работы: $t_{пн(ij)} = t_{пj} - t_{ij}$.

Ранний срок окончания работы $t_{ро(ij)}$ равен раннему сроку свершения события i плюс продолжительность самой работы: $t_{ро(ij)} = t_{рj} + t_{ij}$.

Поздний срок окончания работы $t_{по(ij)}$ равен позднему сроку свершения j события: $t_{по(ij)} = t_{пj}$.

Резерв времени событий R_i есть разность между поздним и ранним сроками свершения события: $R_i = t_{пi} - t_{рi}$, $R_j = t_{пj} - t_{рj}$.

Полный резерв времени путей R_{Li} есть разница между длиной критического пути $L_{кр}$ и длиной любого другого полного пути L_i: $R_{Li} = L_{кр} - L_i$.

Полный резерв времени пути показывает, насколько могут быть предельно увеличены в сумме продолжительности всех работ, принадлежащих пути L_i.

Полный резерв времени работы $R_{п(ij)}$ – это максимальное количество времени, на которое можно увеличить продолжительность данной работы, не изменяя при этом продолжительности критического пути:
$R_{п(ij)} = t_{пj} - t_{рi} - t_{ij}$, где $t_{пj}$ – поздний, $t_{рi}$ – ранний сроки свершения событий, t_{ij} – продолжительность работы.

Свободный резерв времени работы $R_{с(ij)}$ – это максимальное количество времени, на которое можно увеличить продолжительность работы или отсрочить ее начало, не изменяя при этом ранних сроков начала последующих работ, при условии, что начальное событие этой работы наступило в свой (ранний) срок. Он равен разности между ранними сроками наступления событий i и j за вычетом продолжительности работы t_{ij}: $R_{с(ij)} = t_{рj} - t_{рi} - t_{ij}$.

Резервы времени позволяют исполнителям маневрировать сроками начала и окончания работ. Процесс оптимизации сетевых графиков в значительной мере связан с их использованием.

Организация условий «запараллеливания» работ – симультанность, осуществляется – с целью достижения эффективности выполнения всего комплекса процессов по времени [6].

Учитывая ранее представленное, применительно к симультанности разработки элементов интегративного исследования, целесообразно учитывать следующие универсальные характеристики совокупной эффективности процесса [1]:

– наглядная результативность процесса, определяющая уровень соответствия характеристик организации исследования требованиям наличия ресурсов;

– четкая результативность процессов исследования, определяющая значение соотношения затраченного в ходе реализации этих процессов времени и закрытых элементов текущего контроля;

– жесткая временная результативность процессов исследования, представляющая способность выполнения задач в поставленные сроки.

Информационная оценка эффективности симультанности разработки при организации интегративного исследования должна базироваться, по мнению авторов, на следующих принципах:

– разработки критериев и показателей эффективности – для анализа проведения исследования – на протяжении всего времени: от краткого представления теории до защиты отчетов по группам;

– обеспечения сопоставимости условий сравнения различных показателей и вариантов по группам исследователей работающих параллельно;

– многоэтапности оценки, предполагающей, что на различных этапах исследования – эффективность должна оцениваться заново с глубиной проработки соответствующих последующих задач той или иной стадии.

К основному критерию эффективности в сетевом планировании и управлении относят коэффициент напряженности работы.

Коэффициент напряженности работы ($К_{н(ij)}$) – это отношение продолжительности не совпадающих, заключенных между одними и теми же событиями, отрезков пути, одним из которых является отрезок проходящего через эти события критического пути, а другим – путь максимальной продолжительности.

Числовое значение этого коэффициента определяется по формуле: $К_{н(ij)} = (t_{max(ij)} – t`_{кр}) / (t_{кр(ij)} – t`_{кр})$, где $t_{max(ij)}$ – максимальная продолжительность пути, проходящего через события *i, j*; $t_{кр(ij)}$ – длительность критического пути между событиями *i, j*; $t`_{кр}$ – *i, j* отрезок на максимальном пути между событиями, совпадающий с критическим путем.

Выводы и результаты

Проверку сетевого графика интегративного исследования следует реализовывать в два такта. Первый – определение правильности построения сети (нумерация, нахождение замкнутых контуров, недопустимых событий). Второй – расчет напряженных комплексов работ на основе коэффициентов напряженности. Коэффициент напряженности работы – это отношение продолжительности не совпадающих (заключенных между одними и теми же событиями, отрезков пути), одним из которых является отрезок проходящего через эти события критического пути, а другим – путь максимальный по времени. Чем больше коэффициент напряженности, тем труднее выполнить рассматриваемую работу в установленное время в лаборатории.

После проверки сетевого графика следует его оптимизация, цель которой – сокращение (увеличение) длительности комплекса элементов исследования в лаборатории до кратности последней в условиях вуза– 45 мин.

Продолжительность критического пути может быть уменьшена путем расчленения работ дополнительными событиями на составляющие части и параллельное их достижение (симультанности) и за счет перераспределения – увеличение количества исследователей в подгруппах на элементах работ, имеющих резерв времени с уменьшением количества исследователей в группах выполняющих родственные элементы работ, лежащие на критическом пути.

Наглядный пример представлен авторами на примере получения микрокапсул методом испарения легколетучего растворителя в лабораторных условиях, основанном на комплексе работ – рис. 3.

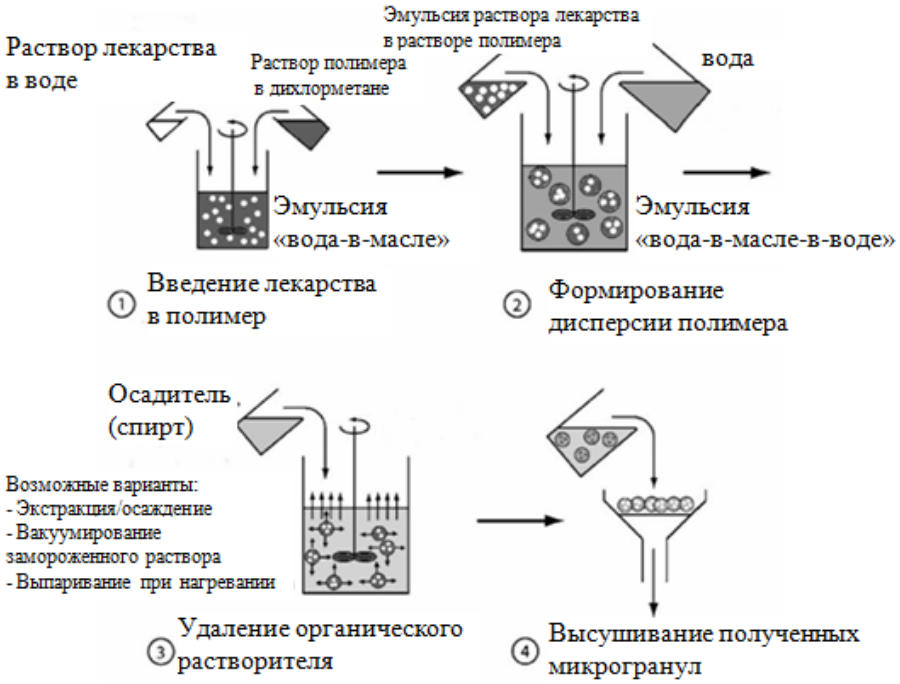


Рис. 3. Укрупненное схематическое представление методики получения микрокапсул методом испарения легколетучего растворителя

1. Для выполнения работы – используют приготовленный заранее 10 %-й раствор этилцеллюлозы (ЭЦ) в ацетоне.

2. На весах отвешивают 3,0 г ацетилсалициловой кислоты (АСК). Если в навеске есть крупные частицы или конгломераты АСК, измельчают их в фарфоровой ступке (но не до пылевидного размера частиц).

3. Раствор ЭЦ в ацетоне перемешивают с помощью электромешалки (n = 400 об/мин) в течение 10…15 мин до образования однородного вязкого раствора.

4. В полученный однородный вязкий раствор при работающей мешалке (n = 400 об/мин) постепенно небольшими порциями вносят АСК и диспергируют в течение 10 мин до образования однородной дисперсии.

5. В фарфоровый стакан вместимостью 500 мл вносят 250 мл вазелинового масла, помещают в стакан мешалку и устанавливают ее скорость 700…800 об/мин.

6. При работающей мешалке, медленно по каплям вливают в вазелиновое масло дисперсию АСК в ацетоновом растворе ЭЦ в течение 30 мин (не менее). Необходимо помнить, что скорость подачи дисперсии в вазелиновое масло влияет на качество и размер микрокапсул.

7. По окончании подачи дисперсии АСК, помещают фарфоровый стакан с дисперсией АСК в вазелиновом масле в водяную баню с температурой воды 35…40 °С и продолжают перемешивание при скорости вращения мешалки 700…800 об/мин.

8. Процесс перемешивания продолжают в течение 2-х часов, при этом поддерживают температуру водяной бани в интервале 35…40 °С, подливая в нее горячую воду.

9. По истечении 2-х часов (необходимо убедиться по отсутствию запаха – в полном испарении ацетона) отфильтровывают через бумажный фильтр на воронке Бюхнера полученные микрокапсулы от вазелинового масла.

10. Отмывают на фильтре микрокапсулы от вазелинового масла тремя порциями по 20 мл гексана.

11. Осторожно вынимают влажный фильтр с микрокапсулами из воронки Бюхнера и высушивают в вытяжном шкафу при комнатной температуре до полного удаления гексана.

12. Взвешивают сухие микрокапсулы.

Продолжительность укрупненного перечня работ по получению микрокапсул методом испарения легколетучего растворителя – в табл. 1.

Т а б л и ц а 1

Укрупненный перечень работ по получению микрокапсул методом испарения легколетучего растворителя

№ п/п	Перечень работ	Время, ч
1	Отмерить 300 мл 10 %-го раствора ЭЦ в ацетоне	0,2
2	На весах отвесить 3,0 г АСК	0,2
3	Проверить навеску АСК на наличие крупных частиц или конгломератов	0,1
4	При наличии крупных частиц или конгломератов в навеске АСК – измельчить их в фарфоровой ступке	0,3
5	Раствор ЭЦ в ацетоне перемешать с помощью электромешалки (n = 400 об/мин) до образования однородного вязкого раствора	0,3
6	В полученный однородный вязкий раствор при работающей мешалке (n = 400 об/мин), постепенно небольшими порциями внести АСК и диспергировать до образования однородной дисперсии	0,2
7	В фарфоровый стакан вместимостью 500 мл внести 250 мл вазелинового масла и поместить в стакан мешалку с ее установкой на скорость 700…800 об/мин	0,2
8	При работающей мешалке, медленно по каплям влить в вазелиновое масло дисперсию АСК в ацетоновом растворе ЭЦ	0,5
9	Поместить фарфоровый стакан с дисперсией АСК в вазелиновом масле в водяную баню с температурой воды 35…40 °С и продолжать перемешивание при скорости вращения мешалки 700…800 об/мин	2
10	Убедиться по отсутствию запаха – в полном испарении	0,1

№ п/п	Перечень работ	Время, ч
	ацетона	
11	Отфильтровать через бумажный фильтр на воронке Бюхнера полученные микрокапсулы от вазелинового масла	0,3
12	Отмыть на фильтре микрокапсулы от вазелинового масла тремя порциями по 20 мл гексана	0,3
13	Влажный фильтр с микрокапсулами высушить в вытяжном шкафу при комнатной температуре до полного удаления гексана	0,5
14	Взвесить сухие микрокапсулы	0,2

Учитывая лабораторные условия получения микрокапсул методом испарения легколетучего растворителя, представляем детальный упорядоченный перечень работ и их продолжительность – табл. 2.

Таблица 2

Детальный перечень работ по получению микрокапсул методом испарения легколетучего растворителя и их продолжительность

№ п/п	Обозначение работ	Перечень работ	Время, мин
1	0-1	Отмерить 300 мл 10 %-го раствора ЭЦ в ацетоне	6
2	1-2	На весах отвесить 3,0 г ацетилсалициловой кислоты (АСК)	5
3	1-3	Раствор ЭЦ в ацетоне перемешать с помощью электромешалки (n = 400 об/мин) до образования однородного вязкого раствора	15
4	1-4	В фарфоровый стакан вместимостью 500 мл внести 250 мл вазелинового масла и поместить в стакан мешалку с ее установкой на скорость 700…800 об/мин	5
5	2-5	Оценить навеску АСК на наличие крупных частиц или конгломератов	3
6	5-6	Измельчить в фарфоровой ступке крупные частицы (конгломераты) в навеске АСК	9
7	6-7	В полученный однородный вязкий раствор при работающей мешалке (n = 400 об/мин), постепенно небольшими порциями внести АСК и диспергировать до образования однородной дисперсии	15
8	7-8	При работающей мешалке, медленно по каплям влить в вазелиновое масло дисперсию АСК в ацетоновом растворе ЭЦ	30

№ п/п	Обозначение работ	Перечень работ	Время, мин
9	8-9	Произвести очистку весов	6
10	8-12	Поместить фарфоровый стакан с дисперсией АСК в вазелиновом масле в водяную баню с температурой воды 35…40 °С и продолжать перемешивание при скорости вращения мешалки 700…800 об/мин	120
11	9-10	Вымыть фарфоровую ступку	3
12	10-11	Вымыть 250 мл емкость	3
13	12-13	Убедиться по отсутствию запаха – в полном испарении ацетона	3
14	13-14	Отфильтровать через бумажный фильтр на воронке Бюхнера полученные микрокапсулы от вазелинового масла	10
15	14-15	Отмыть на фильтре микрокапсулы от вазелинового масла тремя порциями по 20 мл гексана	15
16	15-16	Вымыть 500 мл емкость	3
17	15-18	Влажный фильтр с микрокапсулами высушить в вытяжном шкафу при комнатной температуре до полного удаления гексана	30
18	16-17	Вымыть воронку Бюхнера	3
19	18-19	Взвесить сухие микрокапсулы	5
20	19-20	Произвести очистку весов	6
21	19-21	Осуществить фасовку полученных микрокапсул с учетом требований последующего использования	5

Сформированный сетевой график потенциального интегративного процесса получения микрокапсул методом испарения легколетучего растворителя в программе *NetGraph v*1.0. – представлен на рис. 4.

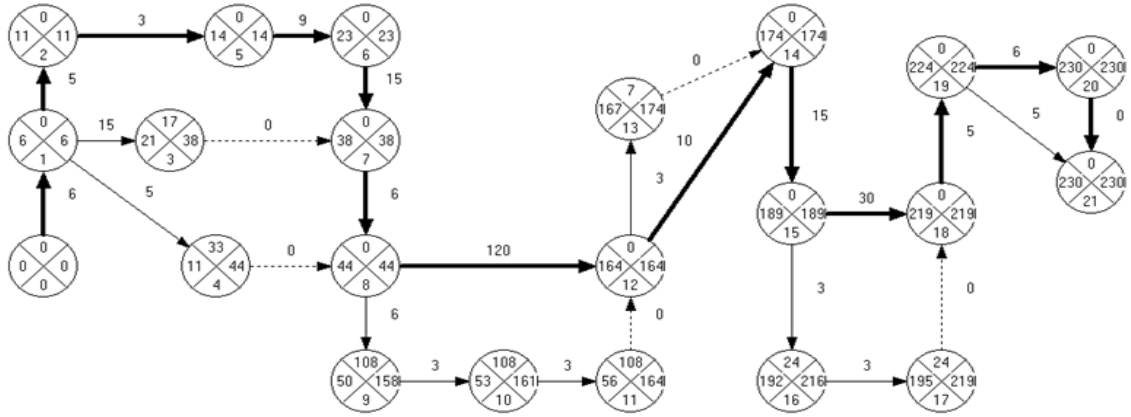


Рис. 4. Сетевой график получения микрокапсул методом испарения легколетучего растворителя

Составлена альтернативная информационная математическая запись модели сетевого планирования и управления потенциальным инновационным процессом получения микрокапсул методом испарения легколетучего растворителя:

$w_{0\text{-}1}(6) \rightarrow w_{1\text{-}2}(5), w_{1\text{-}3}(15), w_{1\text{-}4}(5); w_{1\text{-}2}(5) \rightarrow w_{2\text{-}5}(3); w_{2\text{-}5}(3) \rightarrow w_{5\text{-}6}(9);$
$w_{5\text{-}6}(9) \rightarrow w_{6\text{-}7}(15); w_{1\text{-}3}(15), w_{6\text{-}7}(15) \rightarrow w_{7\text{-}8}(30);$
$w_{1\text{-}4}(5), w_{7\text{-}8}(30) \rightarrow w_{8\text{-}9}(6), w_{8\text{-}12}(120); w_{8\text{-}9}(6) \rightarrow w_{9\text{-}10}(3); w_{9\text{-}10}(3) \rightarrow w_{10\text{-}11}(3);$
$w_{8\text{-}12}(120), w_{10\text{-}11}(3) \rightarrow w_{12\text{-}13}(3), w_{12\text{-}14}(10); w_{12\text{-}13}(3), w_{12\text{-}14}(10) \rightarrow w_{14\text{-}15}(15);$
$w_{14\text{-}15}(15) \rightarrow w_{15\text{-}16}(3), w_{15\text{-}18}(30); w_{15\text{-}16}(3) \rightarrow w_{16\text{-}17}(3);$
$w_{15\text{-}18}(30), w_{16\text{-}17}(3) \rightarrow w_{18\text{-}19}(5); w_{18\text{-}19}(5) \rightarrow w_{19\text{-}20}(6), w_{19\text{-}21}(5).$

Вывод: продолжительность процесса получения микрокапсул методом испарения легколетучего растворителя – 230 минут (3 часа 50 минут). Это время бронирования лаборатории для выполнения заданного исследования. Время – сведено к минимуму за счет симультанности интерактивных процессов организацией параллельных комплексов работ 1-8, 8-12, 12-14, 15-18 и 19-21.

Библиографический список

1. Екшикеев, Т.К. Реализация информационно-аналитических моделей инновационных фармацевтических процессов: сетевое планирование и управление /Т.К. Екшикеев. –М.: КноРус, 2019. -252 с.
2. Екшикеев, Т.К. Фармацевтические процессы: сетевое планирование и управление /Т.К. Екшикеев. –М.: ГЭОТАР-Медиа, 2020. -103 с.
3. Информационные технологии в создании программного обеспечения инновационных разработок: сетевое планирование и управление: методические указания к лабораторным работам для студентов всех форм обучения направления подготовки 38.04.04 «Государственное и муниципальное управление» (магистерская программа «Государственное и муниципальное управление в лесном секторе») / сост.: И. А. Обухова, Т. К. Екшикеев. – Санкт-Петербург: СПбГЛТУ, 2020. – 32 с.
4. Наркевич И.А. К столетию университета: «Мы уверенно смотрим в будущее» //Вестник высшей школы 8(151) октябрь 2019. С. 1-2.
5. Прямая линия с В.В. Путиным – 20 июня 2019: В.В. Путин считает главным способом решения проблемы уровня жизни рост производительности труда. Электронный источник: https://tass.ru/obschestvo/6571104. Дата обращения 05.11.2019.
6. Niininen, P. Innovations and the Success of Firms, VTT, Group for technology studies, Printing office Lars Eriksen Oy, Espoo /P. Niininen, J. Saarinen. – 2000.

А. П. Жернова магистрант
Университета ИТМО
Aly2299
М.Р. Вагизов кандидат технических наук, доцент
Кафедры информационных систем и технологий
СПбГЛТУ им. С. М. Кирова
Bars-tatarin@yandex.ru

РАЗРАБОТКА МЕТОДИКИ АВТОМАТИЗИРОВАННОГО ДЕШИФРИРОВАНИЯ ЕЛИ ЕВРОПЕЙСКОЙ (PICEA ABIES) С ИСПОЛЬЗОВАНИЕМ ГЕОИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И МАШИННОГО ОБУЧЕНИЯ

Класс методов искусственного интеллекта - машинное обучение является одним из наиболее перспективных технологических решений в сфере бизнес-аналитики, статистических исследованиях, в задачах распознавания образов и многих других областей исследований. Достоинства данного метода заключаются в том, что благодаря большой исходной информации (массива данных), алгоритмы, используемые в структуре кода обработки информации, нацелены на поиск зависимостей и интерпретации данных между ними и проведения анализа средствами заранее заложенных программных алгоритмов.

Использование методов машинного обучения решает две основные задачи: упрощает процесс анализа большого объёма информации и минимизирует время на обработку данных. Стоит отметить, что при обработке сверхбольших массивов, содержащих терабайты данных, могут потребоваться значительные вычислительные мощности для проведения обработки данных. На сколько применимы алгоритмы машинного обучения в лесном хозяйстве? И какие конкретные технологические задачи могут быть решены, используя данные методы?

Отвечая на данные вопросы, стоит подразумевать обработку нескольких типов гетерогенных данных, в лесоустроительных предприятиях используются высококачественные изображения, полученные при аэрофотосъёмке высокого пространственного разрешения. Выходными данными будут являться графические файлы, при натурной таксации лесов основными выходными файлами являются файлы базы данных.

Одним из последних усовершенствований при натурной таксации лесов стало использование планшетов, позволяющих оперативно загружать и хранить информацию на устройстве, что снижает трудозатраты камеральной обработки бумажных носителей.

Для технологий анализа машинного обучения могут использоваться не только графические данные, но и данные таксационных баз данных. Таким образом, данные материалы при использовании, как снимков, так и полевых данных на одну изучаемую территорию могут не только повысить информа-

тивность, но так же ещё и дополнить процедуру машинного обучения, что позволит повысить выходную точность при обработке данных.

Для решения задачи формирования признаков распознавания на уровне отдельной породы необходимо определить конкретную породу, с которой можно начать процесс автоматизации дешифрирования. В качестве объекта исследуемой породы будет выбрана Ель европейская (Picea abies), поскольку хозяйственная значимость данной породы имеет важное практическое значение для лесного хозяйства Ленинградской области. Так же Ель обладает особыми резонансно-акустическими свойствами древесины, что делает данную породу ценной для отрасли производства музыкальных инструментов, что говорит об экономической значимости данной породы.[2] Более того, около трети площади лесов в Ленинградской области - ельники.(Рис.1) Именно по этим критериям выбиралась первая порода для проб по разработке полноценного цикла автоматической дешифровке, а также легкостью в определении и выраженности морфологических признаков в каждом классе возраста на материалах дистанционного зондирования Земли.

Общая площадь земель лесного фонда по Ленинградской области составляет 5680,7 тыс.га. Запас по еловой древесине - 236429.6 т. м3 [1] (рис. 1).

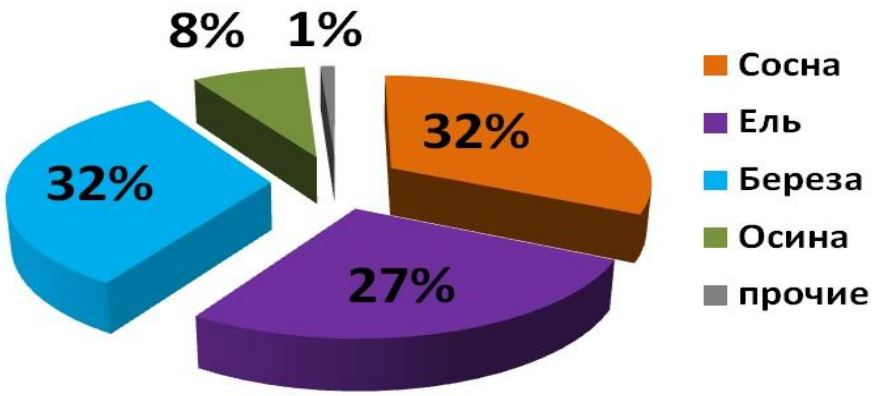


Рис. 1. Распределение по породному составу лесов Ленинградской области

Существуют правила дешифровки лесных насаждений, которые можно применить к формированию базы данных признаков для отдельной породы Ели и определению возраста породы по материалам дистанционного зондирования Земли. Одним из признаков корреляции между морфологическими признаками и дешифровочными на изображениях является возраст породы, таксационный признак – класс возраста, характеризующийся шагом в 20 лет для хвойных пород. Укажем основные признаки насаждения Ели в соответствии с классификацией по возрасту, данные характеристики различимы на

изображениях высокого качества и мелкого масштаба (табл.1).

Таблица 1

Возрастные характеристики ели

Возраст, лет	Возрастные характеристики (признаки дешифровки)
30-50	полог – точечный (скопление крон), характерны отдельные кроны
60-80	полог – точечный, но выделяются отдельные кроны, окончания крон зубчатые, видна собственная тень, нет разновысотности
90-110	начинает увеличиваться крона, появляется конусовидность, разновысотность (можно выделить два яруса), зубчатость сохраняется, но хорошо видны отдельные деревья, кона опускается
От 110 до 140-150	крона становится более конусовидной
От 150 до 180	кроны цилиндрические, расстояние между деревьями очень большое

После формирования основных признаков необходимо выбрать инструмент для формирования связи атрибутивных данных с графическими. Выбор ГИС является важной составляющей работы. Все рассмотренные ниже системы могут быть использованы для данных задач. Проведём краткий анализ основных функциональных особенностей инструментов для анализа (табл. 2).

Учитывая функциональные особенности одних программ перед другими, для разных задач могут использоваться комбинированные подходы обработки материалов. Однако в нашем исследовании на первоначальном этапе будет применяться QGIS, в дальнейшем AutoCad.

Для структурирования данных по классам возраста и последующего создания запросов для выборки необходима достаточно большая база данных с приведенными характеристиками для отдельных деревьев.

Таблица 2

Сравнительные характеристики используемых ГИС

Характеристики	ArcGIS	QGIS	AutoCad Map
Функциональные особенности	Широкая функциональность, многоплатформенность, встроенный язык Python, поддержка разнообразных стандартов, мобильные приложения, серти-	Дружественна к пользователю, с открытым исходным кодом, позволяющая управлять геоданными, отображать редактировать и анализи-	AutoCAD объединяет в себя и САПР и ГИС, это значит, что САПР-ские функции очень удобны для векторизации, составления и редактирования карт по сравнению с другими чисто ГИС програм-

	фикат ФЭСТЭК России	ровать их, а также создавать макеты карт. Работает в различных информационных системах, не ограничена Windows. Работа с ДЗЗ. Постоянно обновляется, включает множество встраиваемых моделей для морфологического анализа. Поддерживает различные СУБД.	мами. AutoCAD Map использует ГИС файлы и базы данных с помощью FDO. Эта отличная технология, тут преобразования или импортирования ГИС-файлов или баз данных не требуется (но такие функции в программе тоже есть). Открытость системы FDO, написанием драйверов можно соединить любую базу данных или ГИС-файлы. В AutoCAD Map есть больше 4000 систем координат, и пользователь может сам создать и добавить для себя геодезические координатные системы (датумы), параметры преобразования датумов, проекции, эллипсоиды и т.д. Есть еще отличная функция, которой нет в не каких программах это - трекинг курсора в разных системах координат одновременно.
Недостатки	Высокая стоимость, необходимость в квалифицированном персонале из-за сложности в эксплуатации	Необходимость поиска нужного модуля. Переведена на русский не полностью. Возможности векторизации слабые.	В AutoCAD Map нет функции для создания рельефа по TIN, только DEM (матрица высот). Трудно работает с большими объемами данных (особенно с большими растрами). Символизация ГИС-слоёв не развита, особенно для линейных объектов. Для топографического картографирования трудно используема. Топология есть только для чертёжных примитивов, а для пространственных объектов нет топологические функции в самом AutoCAD Map.

Дальнейшее решение в исследовании, является в создании процедуры обработки на основе логического вывода сформированных характеристик Ели и наличия связи между базой данных и анализируемого снимка. В работах [3,4] предлагается решение создания специализированной экспертной системы для классификации растений, в работах авторов [5,6] подход основан на пиксельной обработке и аллометрических зависимостях самих древесных растений. Для получения наиболее точного результата, может использоваться комбинированный метод, основанный на синтезе предлагаемых авторами решений. После проведения процедуры создания эталонов (рис.2), решается две основные задачи (последовательность создания эталонов приведена в табл.3).

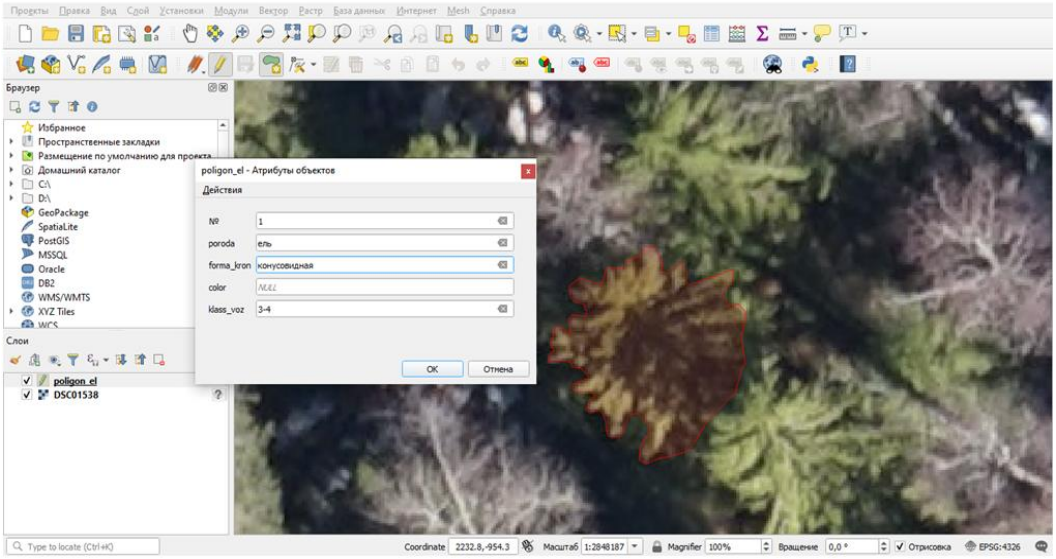


Рис. 2. Пример создания эталона в QGIS

Т а б л и ц а 3

Методика создания эталонов

Этап 1	Подгружаем отредактированный снимок в выбранную нами геоинформационную систему.
Этап 2	Создаем базу данных
Этап 3	Создаем слои для обеспечения быстрой навигации. (Растр; Сосна; Ель; Береза; Осина;)
Этап 4	В каждом из слоев, кроме растра, создаем атрибутивную таблицу с полями: № дерева, порода, форма кроны, цвет, класс возраста.
Этап 5	Переходим непосредственно к самому растру и функцией «Полигон» обрисовываем дерево той породы, в слое которого мы работаем.
Этап 6	Заполняем атрибутивную таблицу согласно полям.

Первая - каждому созданному слою на анализируемом снимке задаются определенные свойства, соответствующие тому классу возраста, которому соответствуют данные в натуре. Вторая - при формировании алгоритма запроса к данным, анализируемого снимка можно сформировать основные последовательные логические запросы, согласно логике предикатов:

1. Что является на снимке породой – Ель.
2. Укажи на снимке породу Ель, в требуемом диапазоне возраста.
3. Укажи на снимке только породу Ели.

Таким образом, формируемый класс (предикатор) Ель функционально содержит в себе определенные предметно-истинные свойства. Следовательно, свойства соответствующие каждому эталону на изображении, присвоенные дешифровщиком при помощи QGIS, будут соответствовать всем схожим объектам на изображении. Следующий этап в исследовании авторов, будет являться описание структурированного запроса на языке высокого уровня (Python), что позволит связать характеристики формального класса с объектами на изображении не входящими в группу эталонов, это позволит дешифрировать большую по площади территорию, на снимке, занимаемой лесом, используя данную методику.

Библиографический список

1. . Официальный сайт Комитета по природным ресурсам Ленинградской области [Электронный ресурс] / Официальный сайт; Лесной план Ленинградской области. – Режим доступа: http://nature.lenobl.ru/

2. Antonov O., et. Al. Acoustic and physico-mechanical properties of spruce timber: influence of differently intensive pruning Antonov O., Dobrovolsky A., Kuznetsov E. В сборнике: IOP Conference Series: Earth and Environmental Science 2019. С. 012025.

3. Хабаров С.П. Модуль логического вывода экспертной системы классификации растений. Хабаров С.П., Шалаев Е.И., Васильев С.П. В книге: Леса России: политика, промышленность, наука, образование материалы научно-технической конференции. Под. ред. В.М. Гедьо. 2016. С. 164-167.

4. Хабаров С.П. Обоснование выбора интерфейсов в системе координированного управления с использованием групповой экспертной оценки. Хабаров С.П., Амбросовский В.М., Коренев А.С. В сборнике: Информационные системы и технологии: теория и практика Сборник научных трудов. отв. ред. А. М. Заяц. 2016. С. 43-49.

5. Вагизов М.Р. Инвентаризация лесов на основе обработки технологий интеллектуального анализа геоданных. // Материалы научно-технической конференции — Леса России: политика, промышленность, наука, образование. / Том 1/Под.ред. В.М.Гедьо -Спб.:СПБГЛТУ, 2018 г.-224с. –С.17-20.

6. Vagizov M.R. Ustyugov V.A., Kvochkin D.O. Determination of the forest inventory indicators according to the photographs of the unmanned aerial vehicles. // Ecology, Environment and Conservation. 2017. Т. 23. № 1. С. 582-586.

7. Михайлова А.А. Вагизов М.Р. Методика обработки данных дистанционного зондирования земли с применением информационных технологий и аллометрических зависимостей для определения лесотаксационных показателей древостоев. // «Успехи современного естествознания» – 2018. № 4-С. 80-85.

В.С. Колыгин, магистрант 1 курса
СПб ГЛТУ им. С.М.Кирова
v.kolygin@yandex.ru
С.П. Хабаров, кандидат технических наук, доцент
Кафедра информационных систем и технологий
СПб ГЛТУ им. С.М.Кирова
serg.habarov@mail.ru

ИССЛЕДОВАНИЕ РАБОТЫ VPN НА БАЗЕ МОДЕЛИ ИЗ НЕСКОЛЬКИХ ВИРТУАЛЬНЫХ МАШИН

Введение. Современное развитие информационных технологий и, в частности, сети Internet, приводит к необходимости защиты информации, передаваемой в рамках распределенной корпоративной сети, использующей сети открытого доступа [1,2]. Виртуальная частная сеть (VPN – Virtual Private Network) – это обобщённое название технологий, позволяющих обеспечить одно или несколько сетевых соединений (логическую сеть) поверх другой сети. В качестве такой сети может выступать любая общедоступная сеть, в том числе и Интернет. VPN состоит из внутренней сети и внешней, по которой проходит инкапсулированное соединение.

В процессе работы VPN проводит шифрование данных и проверку подлинности, что гарантирует конфиденциальность пересылаемых через внешнюю сеть данных, а также позволяет подключаться к сети только тем пользователям, которые имеют соответствующие права. Для обеспечения безопасности передачи VPN использует протоколы туннелирования:

- PPTP (Point-to-Point Tunneling Protocol – протокол туннелирования от точки к точке)
- L2TP (Layer 2 Tunneling Protocol – протокол туннелирования второго уровня)
- GRE (Generic Routing Encapsulation – общая маршрутизирующая инкапсуляция)

В процессе их работы создаются туннели, обеспечивающие высокую защищенность данных при передаче между компьютерами через Интернет. С точки зрения пользователя VPN-подключение через Интернет работает как

выделенный канал глобальной сети. Подключение удалённых пользователей производится посредством VPN-сервера, который подключён как к внутренней, так и к внешней сети.

Первоначально сеть Интернет была создана как безопасная среда передачи данных между военными. С ней работал определенный круг лиц, людей образованных и имеющей представления о политике безопасности. Явной нужды построения защищенных протоколов не было, так как безопасность организовывалась на уровне физической изоляции объектов от посторонних лиц. Однако когда Интернет стал публичным и начал активно развиваться, такая потребность возникла. И пример тому VPN – частное соединение двух узлов, которое шифруется, создавая туннель, проходящий сквозь внешнюю сеть.

Организации физической модели VPN-соединения предполагает наличие нескольких физических компьютеров, а так же дополнительного сетевого оборудования и маршрутизаторов. Это значительно затрудняет возможность настройки и исследования VPN-соединения, так как далеко не каждый имеет в своем распоряжении настроенную сеть с маршрутизаторами.

Постановка задачи. Эта достаточно сложная задача будет рассмотрена на базе упрощенной модели. Существует несколько способов организации модели сети. Одним из них является использование специализированных программных продуктов – эмуляторов сети [3]. Это многофункциональные платформы, которые были разработаны с целью получения практических навыков в области сетевых технологий. Они достаточно информативны, удобны в использовании и позволяют настраивать сеть, следить за ее состоянием и выполнять различные сценарии. Главным недостатком таких платформ является то, что они показывают лишь принципы организации сети, упуская индивидуальные особенности операционных систем (ОС).

Другой способ, который наиболее приближен к реальным условиям, использует стандартные средства ОС Windows и среды виртуализации Oracle VirtualBox. Среда Oracle VirtualBox позволяет создать на одном физическом компьютере несколько виртуальных машин, каждая из которых способна функционировать как отдельный компьютер, с полным функционалом ОС, установленной на эту машину. Упрощенная модель рассматриваемой сети [4] будет содержать по одному узлу в каждой сети, два маршрутизатора, связывающие эти сети и VPN-туннель, организующий виртуальную сеть, объединяющую ws1 и ws2 (рис. 1).

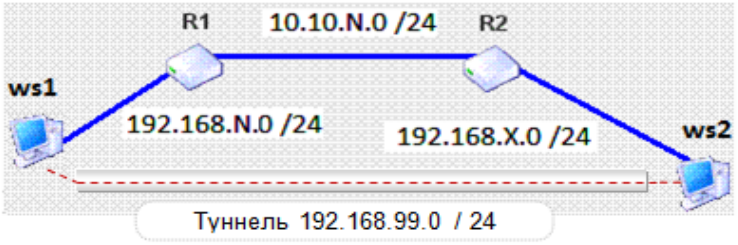


Рис. 1. Модель исследуемой структуры сети

Формирование модели и запуск ее в работу. В состав исследуемой сети входят 4 узла, каждый из которых представлен виртуальной машиной на базе ОС Windows 7. Преимуществом использования виртуальных машин является то, что при желании, гостевая ОС на любом из узлов может быть заменена на другую, нужную для исследования, гостевую ОС [5,6].

Роль VPN-сервера будет выполнять узел сети ws2, а в качестве удалённого компьютера будет использоваться ws1. Именно с узла ws1 будет производиться подключение по VPN-каналу к узлу ws2. Для этого на ws1 необходимо установить и настроить VPN-клиента. Узлы R1 и R2 будут выступать в качестве маршрутизаторов.

VPN-канал – это логическая связь, а реальная сеть, где перемещаются пакеты от ws1 к ws2, включает в себя маршрутизаторы R1 и R2. Организация VPN-соединения предполагает использование протокола PPTP, для нормального функционирования которого необходимо, чтобы на всех маршрутизаторах и персональных брандмауэрах, находящихся между сервером и клиентом VPN, были открыты следующие порты:

- Порты клиента — 1024 ÷ 65535/TCP
- Порт сервера — 1723/TCP

Необходимо также разрешить IP-протокол 47 (GRE). Протокол PPTP (Point-to-point tunneling protocol) - это туннельный протокол типа "точка-точка", который позволяет компьютеру пользователя устанавливать защищённое соединение с сервером за счёт создания специального туннеля в стандартной незащищённой сети [8]. Что касается ОС Windows, то начиная с Windows 95 OSR2, все версии уже включают в свой состав PPTP-клиент.

Для создания подключения к виртуальной частной сети необходимо установить и настроить VPN-сервер. Важно учесть, что при создании сервера, учётная запись должна обладать правами администратора. Для создания нового подключения на виртуальной машине ws2 необходимо выполнить следующую последовательность действий:

- Выбрать Пуск -> Панель управления -> Сеть и интернет -> Центр управления сетями и общим доступом, а затем опцию «Изменение параметров адаптера».
- В появившемся окне следует открыть вкладку файл и выбрать «Новое входящее подключение».
- Затем выбрать пользователя, которому будет разрешено подключаться к этому компьютеру.
- В открывшемся окне «Программы работы с сетью» необходимо открыть свойства протокола Интернета и задать диапазон доступных адресов в формируемом VPN-туннеле, а также разрешить звонящим доступ к локальной сети. Для модели с одним VPN-подключением достаточно диапазона всего в два адреса: 192.168.99.1 – 192.168.99.2, сразу идущих друг за другом. Один будет назначен серверу, а другой клиенту.

После выполнения вышеописанных шагов настройка VPN-сервера завершена. Сервер «слушает» входящие соединения по портам 1701 UDP и 1723 TCP, и готов к установлению удаленного соединения. Следующим этапом является настройка клиента, в роли которого выступает виртуальная машина ws1. Для настройки клиентской виртуальной машины необходимо выполнить следующую последовательность действий:

- В центре управления сетями и общим доступом необходимо выбрать пункт «Настройка нового подключения или сети», затем выбрать вариант подключения, а именно «Подключение к рабочему месту».
- В ответ на запрос системы о том, как выполнить подключение, следует указать «Использовать мое подключение к Интернету (VPN)».
- Далее требуется указать IP-адрес сервера и задать имя подключения.
- В поля пользователь и пароль требуется ввести учетные данные, позволяющие получать доступ к удаленной сети. Если надо, чтобы компьютер запомнил эти учетные данные и использовал их при каждом подключении, то следует установить флажок «Сохранить пароль».

ОС Windows сохраняет конфигурацию сети, в результате чего она становится доступной для использования. Для установки соединения VPN-клиента с VPN-сервером достаточно дважды щелкнуть мышкой на значке с именем соединения. В результате откроется окно «Установка связи». В этом окне вводится имя и пароль пользователя, который зарегистрирован на VPN-сервере. После ввода учетных данных выполняется подключение.

Зайдя в сетевые подключения виртуальной машины ws2 можно убедиться в том, что клиент установил связь с VPN-сервером, используя протокол PPTP. С помощью этого протокола и на базе стандартных средств Windows организован туннель между двумя локальными сетями 192.168.10.0 и 192.168.11.2. Определить состояние подключения можно, выбрав опцию «Свойства» VPN-соединения, в окне сетевых подключений на ws2 (рис. 2).

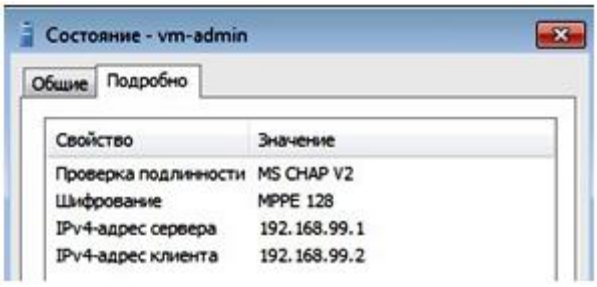


Рис.2. Окно состояние VPN-подключения на узле ws2

Из рис. 2 видно, что компьютеры ws1 (192.168.10.2) и ws2 (192.168.11.2) соединены между собой как VPN-клиент (192.168.99.2) и VPN-сервер (192.168.99.1) по туннелю 192.168.99.0. Откуда такие адреса? Дело в том, что после установки на ws2 VPN-сервера, и включения на ws1 VPN-клиента, в их сетевые конфигурации автоматически добавилось еще по од-

ному виртуальному сетевому адаптеру, подключенному к VPN-туннелю (рис. 3).

```
C:\Users\IEUser>ipconfig

Windows IP Configuration

PPP adapter RAS (Dial In) Interface:

   Connection-specific DNS Suffix  . :
   IPv4 Address. . . . . . . . . . . : 192.168.99.1
   Subnet Mask . . . . . . . . . . . : 255.255.255.255
   Default Gateway . . . . . . . . . :

Ethernet adapter Local Area Connection:

   Connection-specific DNS Suffix  . :
   IPv4 Address. . . . . . . . . . . : 192.168.11.2
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 192.168.11.1
```

Рис. 3. Сетевые интерфейсы узла ws2

Тестирование и работа с VPN-подключением. Тестирование вновь созданного VPN-соединения естественно начать с простейших эхо-запросов (*ping)* с VPN-клиента на VPN-сервер (рис. 4) и наоборот. Если все сделано верно, то они будут успешно выполнены, что докажет работоспособность соединения на сетевом и канальном уровнях.

```
C:\Users\IEUser>ping 192.168.99.1

Pinging 192.168.99.1 with 32 bytes of data:
Reply from 192.168.99.1: bytes=32 time=1ms TTL=128
Reply from 192.168.99.1: bytes=32 time=1ms TTL=128
Reply from 192.168.99.1: bytes=32 time=1ms TTL=128
Reply from 192.168.99.1: bytes=32 time=1ms TTL=128

Ping statistics for 192.168.99.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
```

Рис.4. Эхо-запрос с ws1 на VPN-сервер

Любое сетевое подключение имеет цель получение информации или загрузку программ с удаленного узла. Это предполагает доступ одного приложения к другому, то есть совместимость двух узлов на прикладном уровне. Естественно, если для доступа имеется достаточные права. Для тестирования работоспособности VPN-канала на прикладном уровне можно выполнить доступ из проводника ws1 к ресурсам VPN-сервера ws2 (рис. 5).

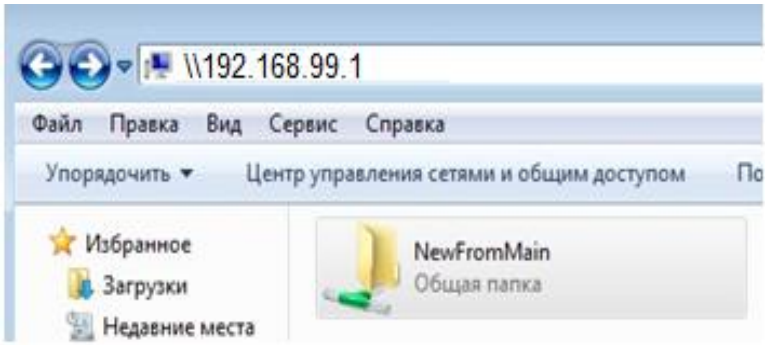


Рис.5. Доступ с узла ws1 к общедоступным ресурсам узла ws2

Так как ранее никаких политик безопасности по удаленному доступу на ws2 определено не было, то клиент получил доступ ко всем общедоступным ресурсам узла ws2. Из этого факта следует вывод о работоспособности VPN-канала на прикладном уровне.

Используя этот канал, удаленный пользователь или его программное обеспечение, как и в случае обычной сети, может получить доступ к любой нужной ему информации на ws2, правами на доступ к которой он владеет. Для этого достаточно указать UNC-имя нужного на ws2 ресурса (рис. 6).

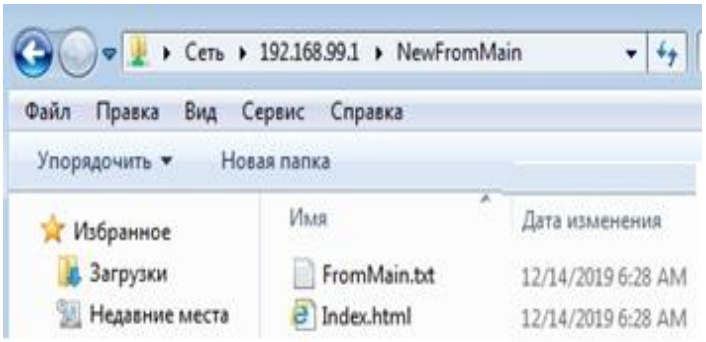


Рис.6. Доступ с ws1 к конкретному ресурсу ws2

Точно также этот пользователь мог бы получить доступ к ресурсу ws2, используя не виртуальный, а обычный канал передачи данных, получая доступ к узлу ws2 через его реальный интерфейс 192.168.11.2. Попробуем выяснить, в чем же состоит отличие и зачем строится VPN-канал.

Логическая структура канала во многом определяется маршрутом и способом передачи данных. Определим маршруты передачи пакетов от узла ws1 к двум разным интерфейсам узла ws2. Для этого в командной строке ws1 введем команду *tracert* с указанием адресов разных интерфейсов ws2 (рис. 7).

```
C:\Users\IEUser>tracert 192.168.99.1

Tracing route to 192.168.99.1 over a maximum of 30 hops

  1     1 ms    <1 ms    <1 ms  192.168.99.1

Trace complete.

C:\Users\IEUser>tracert 192.168.11.2

Tracing route to 192.168.11.2 over a maximum of 30 hops

  1    <1 ms    <1 ms    <1 ms  R1 [192.168.10.1]
  2     1 ms    <1 ms    <1 ms  10.10.10.2
  3     1 ms     1 ms    <1 ms  192.168.11.2

Trace complete.
```

Рис.7. Маршруты передачи пакетов от узла ws1 к узлу ws2

Из этого рисунка видно, что маршрут от ws1 к реальному физическому интерфейсу ws2 (192.168.11.2) прошел по стандартному маршруту через

маршрутизаторы R1 и R2, хотя на этом узле и был установлен VPN-сервер. На этом маршруте пакеты выполняют три перехода. Что касается маршрута от узла ws1 к виртуальному интерфейсу узла ws2 (192.168.99.1), то он не имеет никаких переходов, и пакеты прямо от узла ws1 поступают на ws2, так как ws1 и ws2 находятся в одной виртуальной частной сети (192.168.99.0).

Если выполнить трассировку маршрута передачи пакетов от ws2 к ws1, введя в его командной строке два раза команду *tracert*, то будет получен результат, аналогичный тому, что был получен для ws1.

Естественно, что на физическом уровне пакеты как последовательность электрических или оптических сигналов могут перемещаться только по реальным физическим каналам. Но из полученных результатов следует, что пакеты логически перемещаются, минуя эти каналы, то есть по виртуальному туннелю. В том, что туннель “проходит” мимо узла R1, можно убедиться, выполнив из его командной строки трассировку маршрута (рис. 8).

```
C:\Users\IEUser>tracert 192.168.99.1

Tracing route to 192.168.99.1 over a maximum of 30 hops

  1  Transmit error: code 1231.

Trace complete.

C:\Users\IEUser>tracert 192.168.99.2

Tracing route to 192.168.99.2 over a maximum of 30 hops

  1  Transmit error: code 1231.

Trace complete.
```

Рис.8. Трассировка маршрутов от узла R1 к узлам VPN-сети

Отметим еще одну особенность функционирования сети в которой организован VPN-туннель. С этой целью рассмотрим доступ с узла ws1 к узлам сети 192.168.11.0. В этой сети два узла: ws2 (192.168.11.2) и первый интерфейс маршрутизатора R2 (192.168.11.1), который является шлюзом для этой сети. Для проверки доступа к этим узлам выполним трассировку маршрутов от ws1 к каждому из этих узлов (рис. 9).

```
C:\Users\IEUser>tracert 192.168.11.2
Tracing route to 192.168.11.2 over a maximum of 30 hops
  1    <1 ms    <1 ms    <1 ms  R1 [192.168.10.1]
  2    <1 ms    <1 ms    <1 ms  10.10.10.2
  3    <1 ms    <1 ms    <1 ms  192.168.11.2
Trace complete.

C:\Users\IEUser>tracert 192.168.11.1
Tracing route to 192.168.11.1 over a maximum of 30 hops
  1     1 ms    <1 ms     1 ms  192.168.99.1
  2    <1 ms    <1 ms    <1 ms  192.168.11.1
Trace complete.
```

Рис. 9. Трассировка маршрутов от ws1 к узлам сети 192.168.11.0

Если маршрут от узла ws1 к узлу ws2 очевиден, то маршрут к R2 не столь ожидаем. Маршрут пакетов от ws1 идет к R2 не через R1, а через ws2. Связано это с тем, что при настройке VPN-клиента был выбран параметр «Вход в сеть».

Но если есть доступ с ws1 через VPN-туннель к узлу R2, то должен быть и обратный путь. Это можно проверить, если из командной строки узла R2 выполнить трассировку маршрута до VPN-интерфейса узла ws1 (рис. 10).

```
C:\Users\IEUser>tracert 192.168.99.2
Tracing route to 192.168.99.2 over a maximum of 30 hops
  1    <1 ms    <1 ms    <1 ms  192.168.11.2
  2    <1 ms    <1 ms    <1 ms  192.168.99.2
Trace complete.
```

Рис. 10. Трассировка маршрута от R2 к VPN-интерфейсу ws1

На рис. 10 видно, что маршрут от узла R2 к узлу ws1 существует, и он проходит через узел ws2. При этом узел R2 не имеет никакого физического интерфейса к сети 192.168.99.0, никаких дополнительных шлюзов или маршрутов в узлах сети мы не прописывали, а маршрут существует.

```
C:\Users\IEUser>route print
===========================================================================
Interface List
 13...08 00 27 88 6c f6 ......Intel(R) PRO/1000 MT Desktop Adapter #2
 10...08 00 27 d4 c8 cd ......Intel(R) PRO/1000 MT Desktop Adapter
  1...........................Software Loopback Interface 1
 11...00 00 00 00 00 00 00 e0 Microsoft ISATAP Adapter
 14...00 00 00 00 00 00 00 e0 Microsoft ISATAP Adapter #2
===========================================================================

IPv4 Route Table
===========================================================================
Active Routes:
Network Destination        Netmask          Gateway       Interface  Metric
          0.0.0.0          0.0.0.0     192.168.11.2     192.168.11.1    266
       10.10.10.0    255.255.255.0         On-link        10.10.10.2    266
       10.10.10.2  255.255.255.255         On-link        10.10.10.2    266
     10.10.10.255  255.255.255.255         On-link        10.10.10.2    266
        127.0.0.0        255.0.0.0         On-link         127.0.0.1    306
```

Рис. 11. Таблица маршрутизации узла R2

Если на узле R2 просмотреть таблицу маршрутизации (рис. 11), то можно видеть, что там после установки и запуска на ws2 VPN-сервера в первой строке появился новый маршрут, который объясняет те особенности, что были отмечены выше.

Заключение. Для исследования принципов построения VPN-соединений предложен подход, при котором модель сети, состоящая из двух подсетей, соединенных магистральным каналом передачи данных, была реализована на одном компьютере в среде Oracle VM VirtualBox на базе виртуальных машин с ОС Windows 7.

Результатом явилось настроенное с помощью стандартных средств ОС Windows 7 VPN-соединение, работающее по туннельному протоколу PPTP, что позволило всесторонне исследовать такой способ организации сети, выяснить, как меняются логические маршруты передачи информации в ней.

Показано, что создание на одном из узлов сети VPN-сервера приводит к динамическому изменению таблиц маршрутизации узлов подсети, в которой находится сервер, и к появлению новых логических маршрутов передачи информации с узлов этой подсети в общую сеть.

Библиографический список

1. Хабаров С.П., Шилкина М.Л. Вычислительные машины, системы и сети: Учебное пособие. — СПб.: СПбГЛТА, 2017.— 240 с.
2. Хабаров С.П. Доступ к беспроводным Ad Hoc сетям средствами ОС Windows 10. // В сборнике: Информационные системы и технологии: теория и практика Сборник научных трудов. Ответственный редактор А.М. Заяц. 2018. С. 50-60.
3. Хабаров С.П. Моделирование Ethernet сетей в среде OMNeT++ INET framework // Научно-технический вестник информационных технологий, механики и оптики. 2018. Т. 18. № 3. с. 462–472.
4. Кравченкова И.С., Хабаров С.П. Исследование методов маршрутизации с использованием среды виртуальных машин. // Информационные системы и технологии: теория и практика. Сборник трудов научно-технической конференции. Том 12. – СПб.:СПбГЛТУ, 2020. – С. 71 – 79.
5. Хабаров С.П., Жук Ю.А. Сетевые технологии взаимодействия Ubuntu и Windows платформ: монография. // С. П. Хабаров. — СПб.: Наука и техника, 2013.— 369 с.
6. Хабаров С.П. Использование утилиты WebSocketd в среде ОС Ubuntu Server. // Информационные системы и технологии: теория и практика. Сборник научных трудов научно-технической конференции: Сборник трудов научно-технической конференции. Том 11. – СПб.:СПбГЛТУ, 2019. – С. 95-106.
7. Сусь К.М., Хабаров С.П. Анализ фреймов в процессе использования разных режимов работы маршрутизатора. // Информационные системы и технологии: теория и практика. Сборник научных трудов. Выпуск 10. Часть 2. – СПб.: СПбГЛТУ, 2018. – С. 39-45.
8. Вагизов М.Р. Цифровое лесное хозяйство. Смена парадигм. //Международный научный журнал. // Научные горизонты. Вып. №7, с.132-138.

И.С. Кравченкова, магистрант 1 курса
СПб ГЛТУ им. С.М.Кирова
irakr_02@mail.ru
С.П. Хабаров, кандидат технических наук, доцент
Кафедра информационных систем и технологий
СПб ГЛТУ им. С.М.Кирова
serg.habarov@mail.ru

ИССЛЕДОВАНИЕ МЕТОДОВ МАРШРУТИЗАЦИИ С ИСПОЛЬЗОВАНИЕМ СРЕДЫ ВИРТУАЛЬНЫХ МАШИН

Введение. В процессе эксплуатации компьютерных сетей одной из основных задач сетевого администратора является обеспечение доступа клиентов к общесетевым ресурсам и защита от несанкционированного доступа к ним. Этому в небольших локальных сетях способствует правильное построение маршрутов следования сетевых пакетов, которое выполняется специальными программными или аппаратными средствами [1], получившими название маршрутизаторов. Маршрутизатор – это сетевое устройство, которое, на основании информации о топологии сети (таблицы маршрутизации) и определенных правил, принимает решения о пересылке пакетов сетевого уровня их получателю. При этом функцию маршрутизатора могут выполнять, как специализированные аппаратные средства, так и компьютеры с несколькими сетевыми интерфейсами в случае, когда конфигурация сети не слишком сложная.

Маршрутизация – это важнейший процесс в IP-сетях, и для того чтобы правильно его осуществлять надо хорошо владеть основными подходами к его организации. В этом могут помочь мощные эмуляторы сетей: коммерческие GNS3, PacketTracer или бесплатный OMNeT++ [2], а также достаточно простой javaNetSim. Все эти пакеты позволяют строить работоспособные модели сети, настраивать маршрутизаторы и коммутаторы, моделировать, как проводные, так и беспроводные каналы связей [3-6].

Однако при их использовании остается в стороне формирования навыков настройки маршрутизации на конкретном компьютере в рамках конкретной операционной системы (ОС). Сформировать этот навык позволит настройка маршрутизации в виртуальной сети из нескольких виртуальных машин, реализованной на одном компьютере. При этом на каждой из машин могут быть использованы разные гостевые ОС: Linux, Windows или Ubuntu [7,8], а кроме этого появляется возможность оценки реального сетевого трафика и вложенности протоколов, при использовании для этой цели какого-либо сниффера, типа Wireshark [9].

*Постановка задачи и объект исследования***.** В данной статье рассмотрен подход к исследованию двух методов маршрутизации (по умолчанию и статической), обеспечивающих нормальное функционирование заданной то-

пологии виртуальной сети, реализуемой на хостовом компьютере. При этом настройка будет осуществляться только стандартными средствами гостевых ОС виртуальных машин. Рассмотрим, в качестве примера, межсетевое объединение, состоящее из двух подсетей, связанных между собой магистральным каналом передачи данных (рис.1).

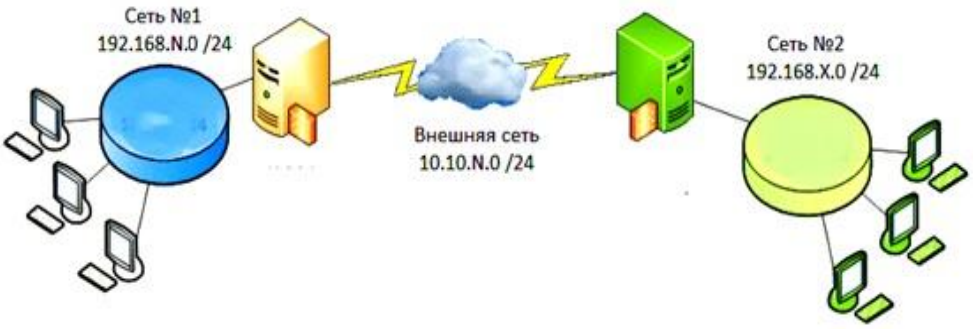


Рис. 1. Общий вид настраиваемого межсетевого объединения

Объединение сетей выполнено на базе маршрутизаторов, в качестве которых используются по одному компьютеру в каждой подсети. Особенность этих компьютеров в том, что они должны иметь по два сетевых интерфейса, один из которых подключен к собственной сети, а второй к магистральной сети. Требуется обеспечить возможность доступа любого узла одной сети к узлам другой сети.

Формирование модели и запуск ее в работу. Для исследования работы сети и маршрутизации в ней используется хостовой компьютер с процессором Intel (R) Core (TM) i5-8400 CPU @ 2.80 GHz, оперативной памятью 16 ГБ и видеопамятью 4 ГБ, на котором установлена ОС Windows 10 версия Pro и среда виртуализации Oracle VM VirtualBox. В дальнейшем исследовании, упрощенная модель рассматриваемой сети будет содержать по одному узлу в каждой сети (ws1 и ws2) и два маршрутизатора (R1 и R2), связывающие эти сети (рис. 2).

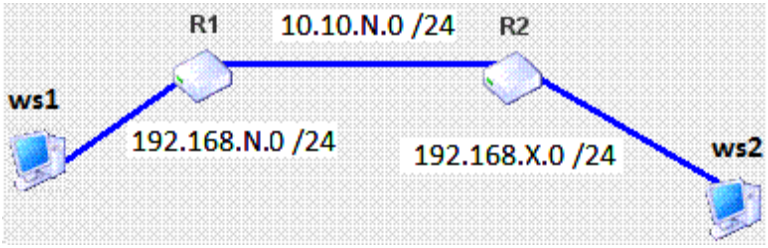


Рис. 2. Упрощенная модель настраиваемой сети

В качестве узлов и маршрутизаторов используются виртуальные машины на базе ОС Windows 7 (32-bit). Однако, работая с виртуальными машинами, в любой момент один из компонентов, будь то маршрутизатор или узел, может быть заменен любой другой ОС, например, Ubuntu, Windows XP или Windows Server. На начальном этапе настройки модели в разделе “Сеть” настроек каждой из виртуальных машин требуется установить требуемое число сетевых адаптеров, и для каждого из них:

- Выбрать тип подключения – «Виртуальный адаптер хоста».

• Указать имя сетевого адаптера – по умолчанию Oracle VM VirtualBox ставит «VirtualBox Host-Only Ethernet Adapter»

• Проверить, что все виртуальные машины имеют разные MAC-адреса. При клонировании машин они могут быть одинаковыми, и тогда их надо изменить.

Для успешной работы, на одном хосте, нескольких виртуальных машин, надо для каждой из них в разделе “Дисплей” режима настроек, указать размер выделяемой ей видеопамяти. Например, 32 Мб при минимально требуемых 27 Мб (рис. 3).

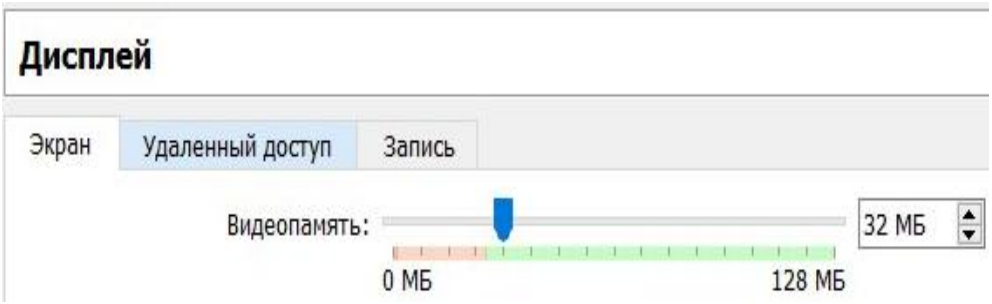


Рис. 3. Настройка видеопамяти виртуальных машин

Использование виртуальной сети для целей проведения в ней всех необходимых настроек и исследований требует одновременного запуска на хостовом компьютере необходимого числа виртуальных машин.

Маршрутизация по умолчанию. Это простейший тип маршрутизации, организуемый посредством шлюзов – узлов подсети, имеющих несколько сетевых интерфейсов, подключенных к разным подсетям. Чаще всего в виде шлюза выступает маршрутизатор. Схема такой маршрутизации:

• На узлах сети задается IP-адрес шлюза по умолчанию.

• При отправке пакета в сеть, каждый узел проверяет совпадение подсети назначения пакета с подсетью узла. Если подсети разные, то пакет отправляется на шлюз.

• Шлюз сравнивает адрес сети назначения с номерами сетей на своих интерфейсах. В случае совпадения, направляет пакет через этот сетевой интерфейс. В противном случае он отправляет пакет узлу, указанному в качестве шлюза по умолчанию на самом шлюзе. Если такового нет, то пакет теряется (рис. 4).

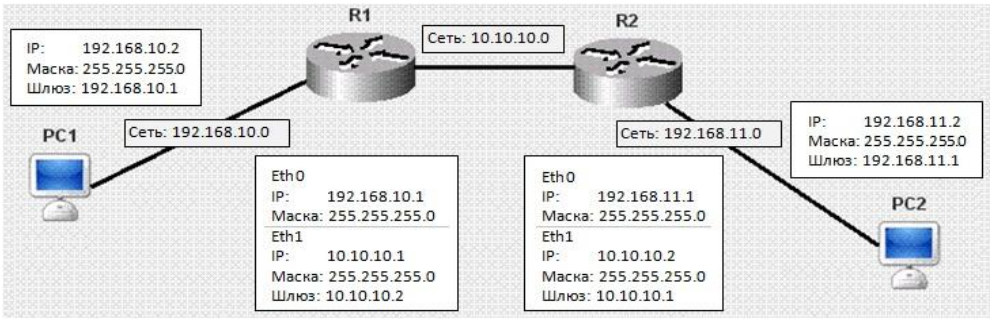


Рис. 4. Настраиваемые параметры маршрутизации по умолчанию

Для настройки и исследования маршрутизации по умолчанию будет использована структуры сети (рис. 2) для случая N=10 и X=11 (рис. 4)

*Настройка и исследование маршрутизация по умолчанию***.** Для организации такой структуры, при первоначальных настройках, необходимо установить на машины R1 и R2 два адаптера с типом подключения «Виртуальный адаптер хоста». Это нужно для того, чтобы обе виртуальные машины могли быть подключены к двум сетям, а именно

- R1: 10.10.10.0 и 192.168.10.0;
- R2: 10.10.10.0 и 192.168.11.0.

В настройках виртуальных машин ws1 и ws2 достаточно подключить всего лишь по одному адаптеру типа «Виртуальный адаптер хоста». После настройки всех виртуальных машин, введя команду ipconfig в командной строке каждой из них, можно получить текущую конфигурация исследуемой виртуальной сети, которая будет иметь вид, аналогичный рис. 5.

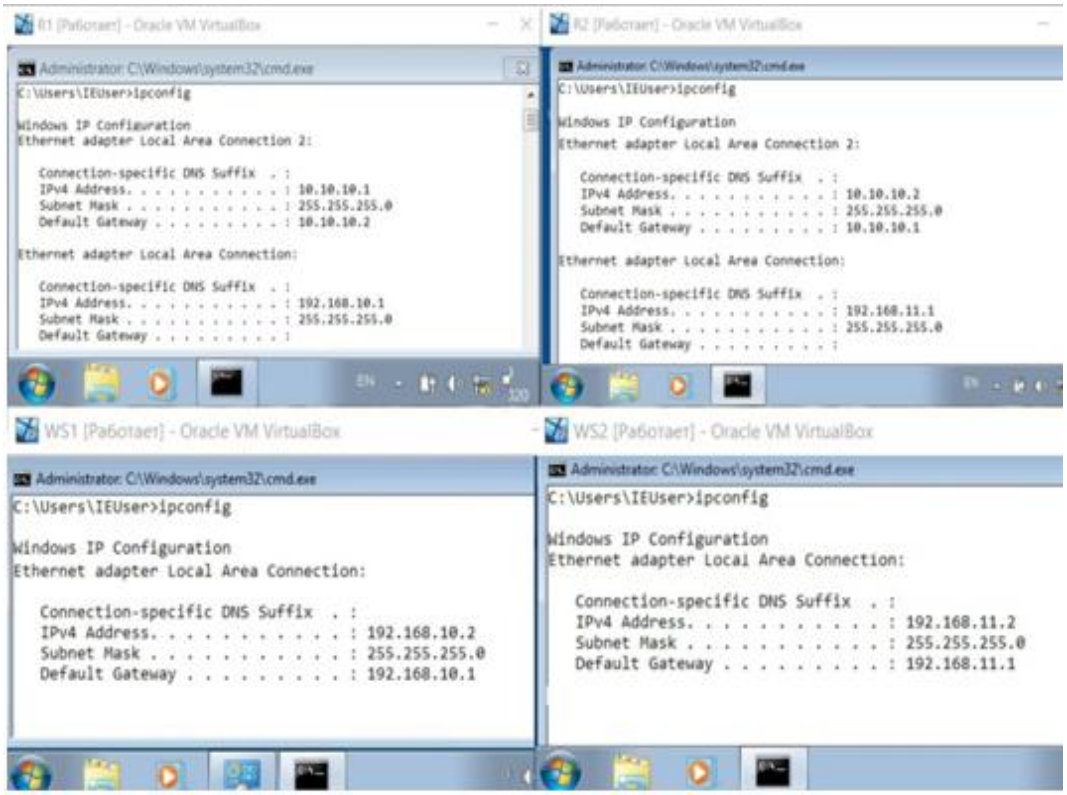


Рис. 5. Конфигурация исследуемой сети на базе Oracle VM VirtualBox

После указания шлюзов на всех узлах сети, можно убедиться в том, что появился доступ с машины R1 на нулевой порт R2 (ping 192.168.11.1) и с машины R2 на нулевой порт R1 (ping 192.168.10.1). Однако, если выполнить эхо-запрос с машины ws2 к нулевому интерфейсу R1, который только что был доступен с R2, то будет получен отрицательный результат:

```
    C:\>ping 10.10.10.1
Обмен пакетами с 10.10.10.1 по 32 байт:
Превышен интервал ожидания для запроса
```

Дело в том, что в исследуемой модели используется не “железный” маршрутизатор, а компьютер на базе ОС Windows 7. При этом, если компьютер с ОС Windows имеет несколько сетевых адаптеров, то для того чтобы узлы сети, подключенные к разным адаптерам и находящиеся в разных сетях, могли "видеть" друг друга, в ОС Windows должна быть включена маршрутизация. При начальной инсталляции она всегда отключена. Для включения маршрутизации на виртуальных машинах R1 и R2 надо выполнить следующую последовательность действий:

- В командной строке каждой из виртуальных машин ввести команду

```
regedit
```

- Откроется окно “Редактор реестра” в котором надо выбрать раздел

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Servic
es \Tcpip\Parameters
```

- В этом разделе найти параметр IPEnableRouter типа REG_DWORD и изменить его значение с 0 на 1 (рис. 6).

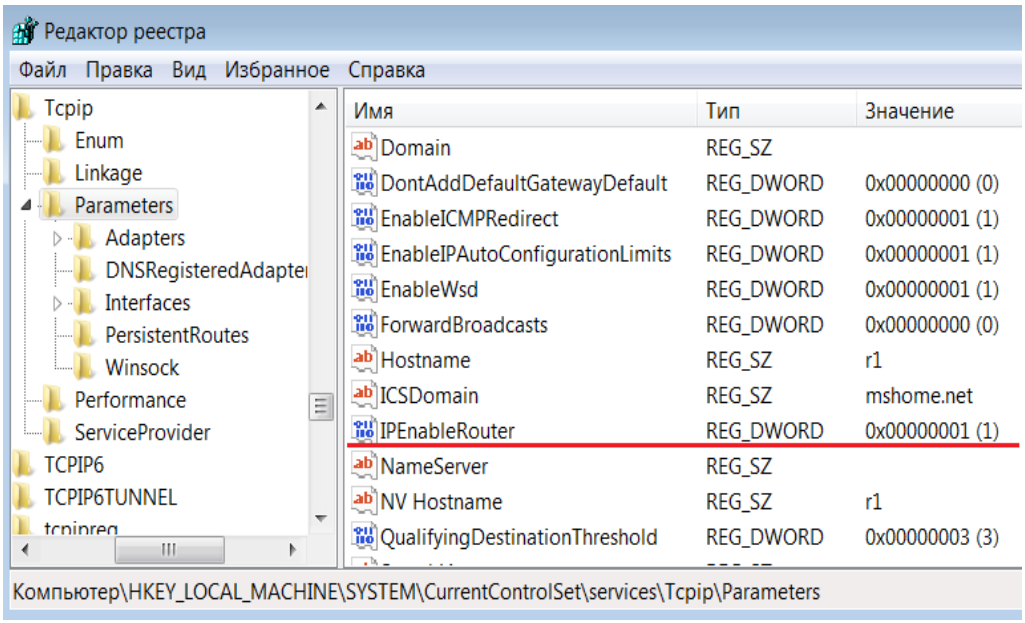


Рис. 6. Изменение параметра реестра IPEnableRouter

- Для того, чтобы новые установки вступили в силу необходимо выполнить перезагрузку виртуальных машин R1 и R2.

Если теперь повторить эхо-запрос с машины ws2 к нулевому интерфейсу R1, то можно убедиться в том, что доступ из сети 192.168.11.0 к маршрутизатору R1 теперь существует

```
C:\>ping 10.10.10.1
Обмен пакетами с 10.10.10.1 по 32 байт:
Ответ от 10.10.10.1: число байт=32 время=4мс TTL=128
```

Как видно из результата выполнения предыдущей команды, доступ из сети №2 существует не только к первому, но и к нулевому порту, а стало быть и ко всей сети 192.168.10.0. То, что маршрутизация из сети 192.168.11.0 в сеть 192.168.10.0 через сеть 10.10.10.0 работает можно убедиться, если на ws2 выполнить команду tracert (рис. 7).

```
C:\Users\IEUser>tracert 192.168.10.1

Tracing route to 192.168.10.1 over a maximum of 30 hops

  1    <1 ms    <1 ms    <1 ms  192.168.10.1

Trace complete.
```

Рис. 7. Выполнение команды tracert на ws2

Такой же результат будет и при проверке доступа к маршрутизатору R2. Чтобы протестировать доступ из сети №2 к узлам сети №1 достаточно на ws2 выполнить две команды:

```
ping 192.168.10.2
tracert 192.168.10.2
```

Нормальное выполнение этих команд позволяет сделать вывод об успешном прохождении пакетов из сети №2 через маршрутизаторы R2 и R1 в сеть №1, а в ней к узлу ws1. Таким образом, мы получили работоспособную модель двух сетей 192.168.10.0 и 192.168.11.0, соединенных магистральной сетью 10.10.10.0, с возможностью обмена информацией между узлами этих сетей.

Но при этом остался неясным вопрос: «Что обеспечило возможность прохождения пакетов через маршрутизаторы и почему данный тип маршрутизации называется прямой или по умолчанию». На первую часть вопроса ответ кажется совершенно очевидным, так как у каждого из маршрутизаторов в качестве шлюза был указан соседний маршрутизатор. Но тут же возникает новый вопрос: «Что изменилось в маршрутизаторе при указании шлюза?».

Для ответа на этот вопрос, надо в консоли R2 выполнить команду *route print*. Добавление шлюза в R2 привело к тому, что в таблице маршрутизации появилась еще одна строка, которая говорит о том, что любой пакет с адресом отличным от других строк таблицы по умолчанию будет отправлен через интерфейс 10.10.10.2 в сеть 10.10.10.0 на узел 10.10.10.1.

Несмотря на очевидные достоинства прямой маршрутизации, она не лишена недостатков. Из-за того, что современные сети имеют сложную структуру, пульсирующий трафик, происходят задержки и потери пакетов данных. При передаче пакетов, не относящихся ни к одному из сегментов в сети, происходит повышенная нагрузка на сеть, количество пересылок паке-

тов может достигать очень больших значений, что неблагоприятно влияет на прямую маршрутизацию в более сложных моделях сети.

Настройка статической маршрутизации. Статическая маршрутизация, не используя каких-либо специальных протоколов, предполагает, что все маршруты указываются в явном виде в таблицах на каждом маршрутизаторе. Эти таблицы обычно составляют вручную и так, чтобы любые два произвольных хоста в сети могли взаимодействовать между собой. Статическая маршрутизация используется обычно в небольших сетях с простой структурой. Ее достоинством является простота настройки, отсутствие затрат трафика на передачу маршрутной информации, низкие требования к ресурсам. Однако если происходят какие-либо изменения в конфигурации сети, то приходится таблицу маршрутизации на всех хостах обновлять вручную.

Для реализации статической маршрутизации, обеспечивающей взаимную доступность узлов сетей №1 и №2, следует прежде всего избавиться от настроек прямой маршрутизации. С этой целью необходимо:

- убрать шлюзы из настроек сетевых интерфейсов обоих маршрутизаторов и перезагрузить виртуальных машины R1 и R2,
- с помощью *ipconfig* проверить текущую конфигурацию R1 и R2,
- убедиться в том, что пакеты с ws2 не доходят до ws1 (рис. 8).

```
C:\Users\IEUser>tracert 192.168.10.2

Tracing route to 192.168.10.2 over a maximum of 30 hops

  1  R2 [192.168.11.1]  reports: Destination net unreachable.

Trace complete.
```

Рис. 8. Эхо-запрос с ws2 на ws1 при ненастроенных маршрутизаторах

Для дальнейшей настройки статической маршрутизации, необходимо создать текстовый файл на рабочем столе виртуальной машины, например, R1. В этот файл требуется ввести всего одну строку (рис. 9) и сохранить его как исполняемый командный файл под именем, например, add_route.bat

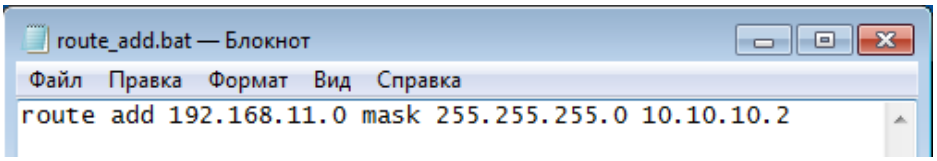
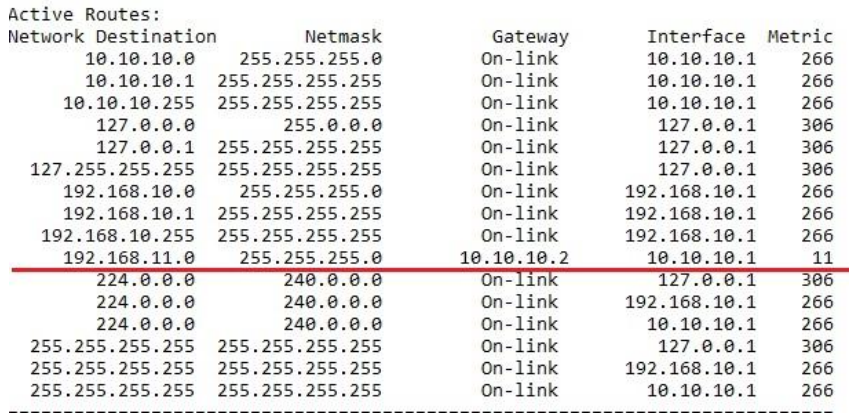


Рис. 9. Маршрут доступа к сети №2 (192.168.11.0) через R2 (10.10.10.2)

Для проверки работоспособности маршрутизации в сети №1 следует запустить на R1 файл add_route.bat на выполнение, а затем, используя *route print*, вывести текущую таблицу маршрутизации R1 (рис. 10).



```
Active Routes:
Network Destination        Netmask          Gateway       Interface  Metric
       10.10.10.0    255.255.255.0         On-link        10.10.10.1    266
       10.10.10.1  255.255.255.255         On-link        10.10.10.1    266
     10.10.10.255  255.255.255.255         On-link        10.10.10.1    266
        127.0.0.0        255.0.0.0         On-link         127.0.0.1    306
        127.0.0.1  255.255.255.255         On-link         127.0.0.1    306
  127.255.255.255  255.255.255.255         On-link         127.0.0.1    306
     192.168.10.0    255.255.255.0         On-link      192.168.10.1    266
     192.168.10.1  255.255.255.255         On-link      192.168.10.1    266
   192.168.10.255  255.255.255.255         On-link      192.168.10.1    266
     192.168.11.0    255.255.255.0      10.10.10.2        10.10.10.1     11
        224.0.0.0        240.0.0.0         On-link         127.0.0.1    306
        224.0.0.0        240.0.0.0         On-link      192.168.10.1    266
        224.0.0.0        240.0.0.0         On-link        10.10.10.1    266
  255.255.255.255  255.255.255.255         On-link         127.0.0.1    306
  255.255.255.255  255.255.255.255         On-link      192.168.10.1    266
  255.255.255.255  255.255.255.255         On-link        10.10.10.1    266
===========================================================================
```

Рис. 10. Таблица маршрутизации R1 после выполнения add_route.bat

Из рис. 10 видно, что после выполнения на R1 файла add_route.bat, прописанный в нем маршрут, оказывается добавленным в таблицу маршрутизации R1.

Для работоспособности статической маршрутизации по всей сети аналогичные операции необходимо выполнить и на маршрутизаторе R2. В него надо добавить маршрут для прохождения пакетов из сети №2 в сеть №1 через маршрутизатор R1.

Заключение. Для исследования методов маршрутизации предложен подход, при котором модель сети, состоящая из двух подсетей, соединенных магистральным каналом передачи данных, была реализована на одном компьютере в среде Oracle VM VirtualBox на базе виртуальных машин с ОС Windows 7.

Было выяснено, что количество пересылок в таблице маршрутизации позволяет отбросить пакеты, не относящиеся ни к одному сегменту сети через определенное количество пересылок (в нашем случае количество пересылок равно 30). Также выяснен недостаток прямой маршрутизации. Из-за возможности повышенной нагрузки в сети, за счет перемаршрутизации пакетов, не относящихся ни к одному из сегментов сети.

Библиографический список

1. Хабаров С.П., Шилкина М.Л. Вычислительные машины, системы и сети: Учебное пособие. — СПб.: СПбГЛТА, 2017.— 240 с.
2. Хабаров С.П. Основы моделирования беспроводных сетей в среде OMNeT++: Учебное пособие. – СПб: Издательство "Лань", 2019.– 260 с.
3. Думов М.И., Хабаров С.П. Моделирование беспроводных сетей в среде OMNET++ с использованием INET framework // Научно-технический вестник информационных технологий, механики и оптики. 2019. Т. 19. № 6. С. 1151–1161.
4. Хабаров С.П. Моделирование Ethernet сетей в среде OMNeT++ INET framework // Научно-технический вестник информационных технологий, механики и оптики. 2018. Т. 18. № 3. с. 462–472.

5. M.R. Vagizov, A.A. Mihailova, A.A. Fetisova, A.I. Habirova, O.S. Vaisero Study of reforestation after cuttings based on materials of open web-mapping service. // IOP Conf. Series: Earth and Environmental Science 316 (2019)

6. Думов М.И., Хабаров С.П. Моделирование работы узлов Wi-Fi сети с учетом интерференционных помех от соседних узлов. // Информационные системы и технологии: теория и практика. Сборник научных трудов научно-технической конференции. Том 11. – СПб.:СПбГЛТУ, 2019. – С. 57-66.

7. Хабаров С.П., Жук Ю.А. Сетевые технологии взаимодействия Ubuntu и Windows платформ: монография. // С. П. Хабаров. — СПб.: Наука и техника, 2013.— 369 с.

8. Хабаров С.П. Использование утилиты WebSocketd в среде ОС Ubuntu Server. // Информационные системы и технологии: теория и практика. Сборник научных трудов научно-технической конференции: Сборник трудов научно-технической конференции. Том 11. – СПб.:СПбГЛТУ, 2019. – С. 103-113.

9. Сусь К.М., Хабаров С.П. Анализ фреймов в процессе использования разных режимов работы маршрутизатора. // Информационные системы и технологии: теория и практика. Сборник научных трудов. Выпуск 10. Часть 2. – СПб.: СПбГЛТУ, 2018. – С. 33-39.

М. О. Лебедев, кандидат технических наук, доцент
Кафедра информационных систем и технологий
СПбГЛТУ им.С.М.Кирова
lebedevmike@mail.ru

АЛГОРИТМ И РЕАЛИЗАЦИЯ МНОГОШАГОВОГО МЕТОДА РЕШЕНИЯ СИСТЕМ ОДУ

Система обыкновенных дифференциальных уравнений (ОДУ) представляет собой систему, разрешенных относительно производных дифференциальных уравнений. В общем случае система ОДУ может быть представлена в виде:

$$\left\{\frac{dY}{dx}\right\} = \left\{F\left(x, \{Y\}\right)\right\}$$

где $\{Y\}$ - вектор искомых функций, $\{F(x,\{Y\})\}$ - вектор правых частей уравнений, представляющих собой произвольные выражения, включающие искомые функции (но не их производные) и аргумент этих функций. Если заданы начальные условия – значения искомых функций в начальной точке, то получаем классическую задачу Коши.

Аналитическое решение систем ОДУ не всегда возможно, поэтому актуально использование численных методов решения. При численном решении всегда существуют три вида погрешностей:

- ошибки исходных данных;
- ошибки округления;
- ошибки усечения (аппроксимации).

Первые два вида ошибок связаны с невозможностью удерживать число значащих цифр больше некоторого допустимого (например, для типов данных double число значащих цифр равно 15-16, для типов данных extended - 18-19). Третий вид связан с тем, что неизвестный функции (и их производные) чаще всего представляются в виде рядов (чаще всего в виде ряда Тейлора). При этом удерживаются, как правило, только первые 2 - 3 члена ряда.

Все основные методы численного решения дифференциальных уравнений представлены ниже в табл. 1.

Таблица 1.

Методы численного решения дифференциальных уравнений

Метод получения основных разрешающих соотношений	Примеры
Представление производных в виде конечных разностей	Метод Эйлера Методы Рунге-Кутта Метод конечных разностей
Интегрирование правой части системы ДУ	Методы Адамса
Получение функционала и решение методом Ритца	Метод конечных элементов

В данной работе рассматривается развитие метода Адамса.

Основные соотношения рассмотрим на примере ДУ 1-го порядка. Имеем ДУ

$$\frac{dy}{dx} = f(x, y)$$

и начальное условие $y(0) = y_0$. Весь интервал, где ищем решения ДУ, разобьем на равные участки длиной h. Тогда для каждого участка будет справедливо:

$$y_{i+1} = y_i + \int_{x_i}^{x_{i+1}} f(x, y)dx \qquad (*)$$

Последнее выражение справедливо для любого интервала интегрирования и не имеет погрешностей усечения.

Получим основные соотношения для линейного ДУ 1-го порядка

$$\frac{dy}{dx} = a(x)y + b(x)$$

здесь $a(x)$ и $b(x)$ - заданные функции. На каждом i-ом шаге справедливо

$$\boldsymbol{Y_{i+1} = Y_i + \int_{x_i}^{x_{i+1}} (a(x)y + b(x))dx}$$

Допуская линейное изменение искомой функции на участке от x_i до x_{i+1}, получаем

$$\boldsymbol{y(x) = \frac{x_{i+1} - x}{h} Y_i + \frac{x - x_i}{h} Y_{i+1}}, \text{ где } \boldsymbol{h = x_{i+1} - x_i}$$

Тогда для *i*-го шага будем иметь

$$\boldsymbol{Y_{i+1} = Y_i + \int_{x_i}^{x_{i+1}} \left(a(x) \left(\frac{x_{i+1} - x}{h} Y_i + \frac{x - x_i}{h} Y_{i+1} \right) + b(x) \right) dx}$$

После интегрирования получаем

$$\boldsymbol{Y_{i+1} = Y_i + Y_i \int_{x_i}^{x_{i+1}} a(x) \left(\frac{x_{i+1} - x}{h} \right) dx + Y_{i+1} \int_{x_i}^{x_{i+1}} a(x) \left(\frac{x - x_i}{h} \right) dx + \int_{x_i}^{x_{i+1}} b(x) dx}$$

Соответственно, для системы ДУ 1-го порядка будем иметь

$$\boldsymbol{\{Y\}_{i+1} = \{Y\}_i + \{Y\}_i \int_{x_i}^{x_{i+1}} \ddot{e} \left(\frac{x_{i+1} - x}{h} \right) dx + \{Y\}_{i+1} \int_{x_i}^{x_{i+1}} [a(x)] \left(\frac{x - x_i}{h} \right) dx \int_{x_i}^{x_{i+1}} \{b(x)\} dx}$$

Здесь $\boldsymbol{\{Y\}_i}$ - вектор значений функций в *i*-ом узле, $\boldsymbol{[a(x)]}$- матрица коэффициентов исходной системы ДУ, $\boldsymbol{\{b(x)\}}$- вектор свободных членов исходной системы ДУ.

Полученные соотношения используют 2 узла и имеют, соответственно, 2-ой порядок точности. Увеличивая число узлов, увеличиваем порядок точности и, тем самым, уменьшаем ошибку усечения.

В свою очередь выражение (*) можно переписать, используя начальную точку (для рассматриваемого линейного уравнения)

$$\boldsymbol{y_{i+1} = y_0 + \int_{0}^{x_{i+1}} (a(x)y + b(x))dx}$$

или для системы линейных ДУ

$$\boldsymbol{\{Y\}_{i+1} = \{Y\}_0 + \int_{0}^{x_{i+1}} ([a(x)]\{Y\} + \{b(x)\})dx}$$

Здесь $\boldsymbol{\{Y\}_0}$ - вектор начальных значений функций. Интеграл в последнем выражении для верхней границы x_{i+1} будет представлять собой сумму:

$$\boldsymbol{\int_{0}^{x_{i+1}} ([a(x)]\{Y\} + \{b(x)\})dx = \sum_{k=0}^{i} \left(\{Y\}_k \int_{x_k}^{x_{k+1}} [a(x)] \frac{x_{k+1} - x}{h} dx + \{Y\}_{k+1} \int_{x_k}^{x_{k+1}} [a(x)] \frac{x - x_k}{h} dx + \int_{x_k}^{x_{k+1}} \{b(x)\} dx \right)}$$

Последний интеграл (интеграл со свободным членом исходной системы $\boldsymbol{\{b(x)\}}$ ДУ) можно вынести из под знака суммы. К матрице коэффициентов $\boldsymbol{[a(x)]}$ применим линейную интерполяцию на участке от x_k до x_{k+1}, получим

$$\boldsymbol{[a(x)] = [a(x_k)] \frac{x_{k+1} - x}{h} + [a(x_{k+1})] \frac{x - x_k}{h} = [A_k] \frac{x_{k+1} - x}{h} + [A_{k+1}] \frac{x - x_k}{h}}$$

Тогда интеграл при векторе $\boldsymbol{\{Y\}_k}$ в выражении суммы будет:

$$\int_{x_k}^{x_{k+1}}[a(x)]\frac{x_{k+1}-x}{h}dx=\int_{x_k}^{x_{k+1}}\left([A_k]\frac{x_{k+1}-x}{h}+[A_{k+1}]\frac{x-x_k}{h}\right)\frac{x_{k+1}-x}{h}dx$$

Откуда получаем

$$\int_{x_k}^{x_{k+1}}\left([A_k]\frac{x_{k+1}-x}{h}+[A_{k+1}]\frac{x-x_k}{h}\right)\frac{x_{k+1}-x}{h}dx=\frac{[A_k]}{h^2}\int_{x_k}^{x_{k+1}}(x_{k+1}-x)^2dx+\frac{[A_{k+1}]}{h^2}\int_{x_k}^{x_{k+1}}(x-x_k)(x_{k+1}-x)dx$$

Окончательно получим

$$\int_{x_k}^{x_{k+1}}[a(x)]\frac{x_{k+1}-x}{h}dx=\frac{h}{6},$$

В свою очередь, интеграл при векторе $\{Y\}_{k+1}$ в выражении суммы будет:

$$\int_{x_k}^{x_{k+1}}[a(x)]\frac{x-x_k}{h}dx=\frac{h}{6}([A_k]+2[A_{k+1}])$$

Тогда

$$\{Y\}_{i+1}=\{Y\}_0+\sum_{k=0}^{i}\frac{h}{6}((2[A_k]+[A_{k+1}])\{Y\}_k+([A_k]+2[A_{k+1}])\{Y\}_{k+1})+\int_0^{x_{i+1}}\{b(x)\}dx$$

Если использовать линейную интерполяцию свободного члена $\{b(x)\}$ между узлами:

$$b(x)=\{b(x_i)\}\frac{x_{i+1}-x}{h}+\{b(x_{i+1})\}\frac{x-x_i}{h}$$

то интеграл от x_0 до x_{i+1} будет равен

$$\int_0^{x_{i+1}}\{b(x)\}dx=\sum_{k=0}^{i}\frac{h}{2}(\{b_k\}+\{b_{k+1}\})$$

Таким образом, разрешающее соотношение для i+1-го узла будет иметь вид:

$$\left([I]-\frac{h}{6}([A_i]+2[A_{i+1}])\right)\{Y\}_{i+1}=\left([I]+\frac{h}{6}(2[A_0]+[A_1])\right)\{Y\}_0+\sum_{k=0}^{i}\frac{h}{6}([A_{k-1}]+4[A_k]+[A_{k+1}])\{Y\}_k+$$
$$+\sum_{k=0}^{i}\frac{h}{2}(\{b_k\}+\{b_{k+1}\})$$

Видно, что разрешающее выражение i+1-го узла включает в себя суммы по всем предыдущим узлам. Таким образом, порядок метода для i-го узла будет равен i. С другой стороны, входящие в выражения суммы, позволяют получать разрешающие соотношения нарастающим итогом, т.е. сохраняя ранее сделанные вычисления и добавляя к ним значения, получаемые на очередном шаге.

Для i=0 имеем:

$$\left([I]-\frac{h}{6}([A_0]+2[A_1])\right)\{Y\}_1=\left([I]+\frac{h}{6}(2[A_0]+[A_1])\right)\{Y\}_0+\frac{h}{2}(\{b_0\}+\{b_1\})$$

Произведение $\left([I]+\frac{h}{6}(2[A_0]+[A_1])\right)\{Y\}_0$ будет присутствовать при вычислении всех узловых значений. Обозначим это произведение F_0:

$$\boldsymbol{F}_0 = \left([\boldsymbol{I}] + \frac{\boldsymbol{h}}{6}(2[\boldsymbol{A}_0] + [\boldsymbol{A}_1]) \right) \{\boldsymbol{Y}\}_0$$

А всю правую часть обозначим как *F*:

$$\boldsymbol{F} = \left([\boldsymbol{I}] + \frac{\boldsymbol{h}}{6}(2[\boldsymbol{A}_0] + [\boldsymbol{A}_1]) \right) \{\boldsymbol{Y}\}_0 + \frac{\boldsymbol{h}}{2}(\{\boldsymbol{b}_0\} + \{\boldsymbol{b}_{k1}\}) = \boldsymbol{F}_0 + \frac{\boldsymbol{h}}{2}(\{\boldsymbol{b}_0\} + \{\boldsymbol{b}_1\})$$

Таким образом, получаем систему

$$[\boldsymbol{K}_1]\{\boldsymbol{Y}\}_1 = \{\boldsymbol{F}_1\}$$

где

$$[\boldsymbol{K}_1] = \left([\boldsymbol{I}] - \frac{\boldsymbol{h}}{6}([\boldsymbol{A}_0] + 2[\boldsymbol{A}_1]) \right),$$

$$\boldsymbol{F}_1 = \boldsymbol{F}_0 + \frac{\boldsymbol{h}}{2}(\{\boldsymbol{b}_0\} + \{\boldsymbol{b}_1\})$$

Для *i*=1 получим следующее выражение

$$\underbrace{\left([\boldsymbol{I}] - \frac{\boldsymbol{h}}{6}([\boldsymbol{A}_1] + 2[\boldsymbol{A}_2]) \right)}_{[\boldsymbol{K}_2]} \{\boldsymbol{Y}\}_2 = \underbrace{\left([\boldsymbol{I}] + \frac{\boldsymbol{h}}{6}(2[\boldsymbol{A}_0] + [\boldsymbol{A}_1]) \right) \{\boldsymbol{Y}\}_0 + \frac{\boldsymbol{h}}{2}(\{\boldsymbol{b}_0\} + \{\boldsymbol{b}_1\})}_{\boldsymbol{F}_1 = \boldsymbol{F}_0 + \frac{\boldsymbol{h}}{2}(\{\boldsymbol{b}_0\} + \{\boldsymbol{b}_{k1}\})} +$$

$$+ ([\boldsymbol{A}_0] + 4[\boldsymbol{A}_1] + [\boldsymbol{A}_2])\frac{\boldsymbol{h}}{6}\{\boldsymbol{Y}_1\} + \frac{\boldsymbol{h}}{2}(\{\boldsymbol{b}_0\} + \{\boldsymbol{b}_1\})$$

Таким образом, для *i*=1 имеем следующую систему:

$$[\boldsymbol{K}_2]\{\boldsymbol{Y}\}_2 = \{\boldsymbol{F}_2\}$$

где

$$[\boldsymbol{K}_2] = \left([\boldsymbol{I}] - \frac{\boldsymbol{h}}{6}([\boldsymbol{A}_1] + 2[\boldsymbol{A}_2]) \right),$$

$$\boldsymbol{F}_2 = \boldsymbol{F}_1 + ([\boldsymbol{A}_0] + 4[\boldsymbol{A}_1] + [\boldsymbol{A}_2])\frac{\boldsymbol{h}}{6}\{\boldsymbol{Y}_1\} + \frac{\boldsymbol{h}}{2}(\{\boldsymbol{b}_1\} + \{\boldsymbol{b}_2\})$$

Нетрудно видеть, что для *i*-го узла будем иметь:

$$[\boldsymbol{K}_{i+1}]\{\boldsymbol{Y}\}_{i+1} = \{\boldsymbol{F}_{i+1}\}$$

где

$$[\boldsymbol{K}_{i+1}] = \left([\boldsymbol{I}] - \frac{\boldsymbol{h}}{6}([\boldsymbol{A}_i] + 2[\boldsymbol{A}_{i+1}]) \right),$$

$$\boldsymbol{F}_{i+1} = \boldsymbol{F}_i + ([\boldsymbol{A}_{i-1}] + 4[\boldsymbol{A}_i] + [\boldsymbol{A}_{i+1}])\frac{\boldsymbol{h}}{6}\{\boldsymbol{Y}_i\} + \frac{\boldsymbol{h}}{2}(\{\boldsymbol{b}_i\} + \{\boldsymbol{b}_{i+1}\})$$

Программная реализация описанного метода была выполнена с использованием объектно-ориентированного программирования. Ниже показана иерархия классов, реализующая данный алгоритм (рис. 1).

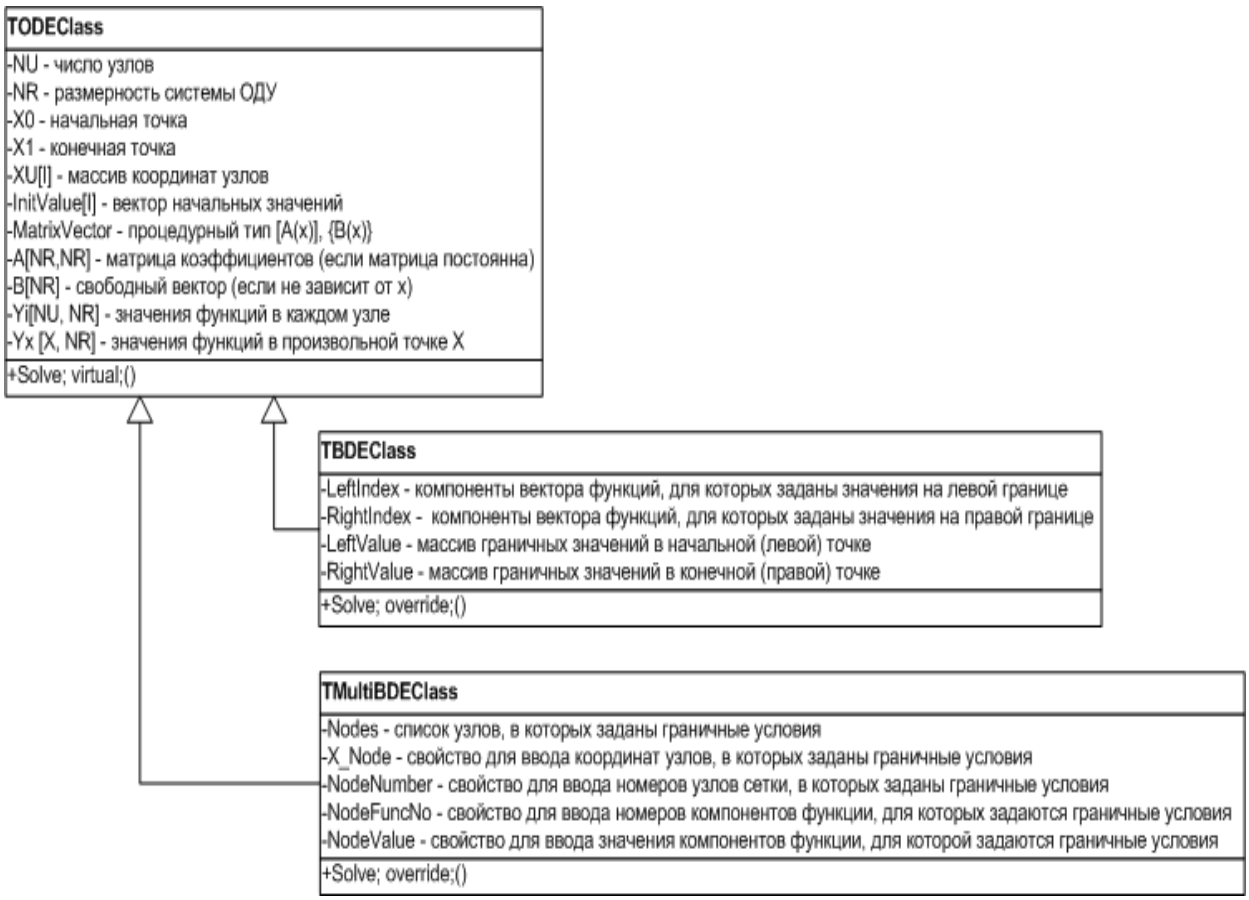


Рис. 1. Иерархия классов

Базовый класс TODEClass содержит основные свойства и методы решения линейной системы ОДУ. Класс содержит свойство-переключатель, определяющее, являются ли матрица коэффициентов [***a(x)***] и вектор {***b(x)***} постоянными (т.е. не зависящими от ***x***), или переменными. В первом случае для этих значений используются свойства-массивы, в которые вводятся значения матрицы [***a(x)***] и вектора {***b(x)***}. Во втором случае используется процедурное свойство - процедура, возвращающее значение матрицы и вектора в точке ***x***. Класс предназначен для решения задачи Коши для системы линейных ДУ.

Дочерние классы TBDEClass и TMultiBDEClass предназначены для решения линейных систем ДУ с граничными (не начальными) условиями.

Класс TBDEClass содержит свойства и методы, необходимые для решения систем ДУ, когда граничные условия заданы в начальной (нулевой) и конечной точках.

Класс TMultiBDEClass - свойства и методы, необходимые для решения систем ДУ, когда дополнительные условия могут быть заданы в любых промежуточных узлах.

Метод Solve - предназначен для поиска значений функций в узлах. Этот метод образует иерархию полиморфных методов. Результаты решения помещаются в свойство-массив Yi, объявленное в родительском классе.

Использование данного алгоритма на тестовых задачах показало хорошую устойчивость метода и хорошее приближение к точным решениям.

Н.В.Лушкин, кандидат технических наук, доцент
Кафедра информационных систем и технологий
СПбГЛТУ им. С.М.Кирова
Lushkin52@mail.ru
М.А.Шубина, кандидат технических наук, доцент
Кафедра информационных систем и технологий
СПбГЛТУ им. С.М.Кирова
nemsha@mail/ru

АЛГОРИТМ КЛАССИФИКАЦИИ ОБЪЕКТОВ НА ОСНОВЕ ПОСТРОЕНИЯ ГРАФОВ СВЯЗНОСТИ ОБЪЕКТОВ

Одной из важнейших задач анализа изображений наземных объектов является их распознавание и классификация на основе изображений поверхности Земли, полученных дистанционными методами. К числу основных стандартных алгоритмов классификации, содержащихся в современных программных средствах геоинформационных систем ERDAS IMAGINE 9.0, IDRISI, ERMapper, ENVI и др. относятся [1]:

- классификация без обучения (unsupervised cclassification),
- классификация с обучением (supervised cclassification),
- пороговая классификация (threshold),
- использование нечетких множеств (fazzy logic),
- использование интеллектуальных алгоритмов, например, нейронных сетей (neural network) и др.

При классификации без обучения изображение автоматически разбивается на классы в зависимости от значений яркостей, от заданного количества классов. При необучаемой классификации задаются ее правила (линейная, квадратичная, непараметрическая и др.) и соответствующие параметры процесса классификации:

- каналы, которые будут участвовать в классификации,
- начальное число классов,
- максимальное число классов,
- минимальное число членов класса,
- максимальное стандартное отклонение, при котором класс будет делиться пополам при последующей итерации,
- минимальное расстояние между классами, выраженное в стандартных отклонениях, при которых классы будут объединяться при последующей итерации,

- процент сходимости итерационного процесса.

При выборе алгоритма важно, исходя из цели задачи, оценить время и точность решения, При классификации с обучением определяются спектральные характеристики эталонов и в соответствии с ними выделяются подобные участки. Здесь возникает задача статистической разделимости классов, выбора оценок, позволяющих однозначно разделить классы. При отнесении фрагмента изображения к тому или иному классу используются различные алгоритмы (максимального правдоподобия, минимального расстояния, ближайшего соседа и др.), и их выбор зависит от анализируемых материалов и заданной точности классификации. Способы классификации зависят от способа выделения и ограничения области значений яркости класса с учетом характера распределения значений яркости между классами и внутри каждого класса. Например, при параметрической классификации по многозональному снимку предполагают, что распределение значений яркости в пределах каждого класса в каждой спектральной зоне нормальное. Интервал $BXm \pm 3\,\sigma x$ содержит все значения яркости для нормально распределенного класса. Чтобы оценить, близость распределения значений спектральной яркости класса к нормальному, строят гистограмму по значениям яркости пикселей в пределах класса и сравнивают ее с графиком кривой нормального распределения, имеющей те же M, и σ.

Далее необходимо проанализировать, все ли выделенные особенности относятся к заданным объектам, и уточнить алгоритм классификации, при необходимости создать и использовать маски, затем повторить классификацию. Этот процесс требует значительного времени.

Предлагаемая программа Sopka по сравнению с вышеперечисленными программами позволяет повысить скорость выбора параметров эталонов, обеспечивающих требуемую для решения конкретных задач точность разделения классов, и оптимального масштаба изображений.

Описание алгоритма программы.

Алгоритм программы основан на классификации графического файла, соответствующего анализируемому изображению, на видимую, интересующую нас, и невидимую части. В видимой части изображения происходит поиск связных областей графического объекта, которые нумеруются и могут быть в дальнейшем проанализированы и сохранены в отдельных файлах. Наиболее трудным шагом является выбор критерия разделения графического файла на видимую и невидимую области.

Следующим шагом алгоритма является составление векторов по выбранной палитре псевдоцветов. В дальнейшем полученные вектора нумеруются как узлы графа. Два узла графа связны, если связны соответствующие этим узлам векторы. Таким образом, определение связности элементов изображения сводится к определению связности графа (рис.1).

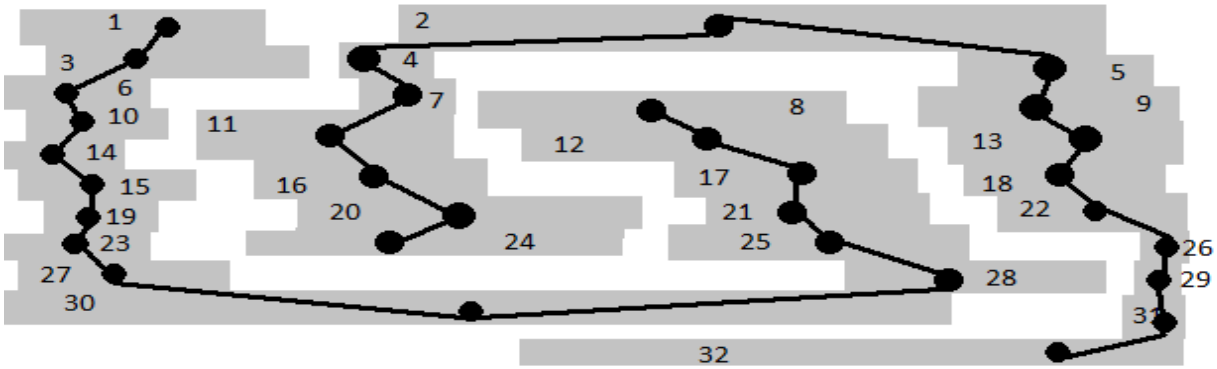


Рис. 1. Пример построения графов связности объектов в процессе построчного сканирования изображения

Определение статистических параметров изображения

Максимальное и минимальное значение псевдоцветов при классификации определяется по формулам:

$$C_{max} = X_{max} + \frac{S_{gis}}{B_m} \cdot \frac{K}{(1 + f(i))}$$

$$C_{min} = X_{max} - \frac{S_{gis}}{B_m} \cdot \frac{K}{(1 + f(i))},$$

где Xmax –значение цвета при максимальном значении гистограммы Bm, Sgis – сумма всех частот в гистограмме. Коэффициент и ступенчатая функция f(i) определяется при обучении по образцам (различные типы растительности, открытые почвы, болота, водные объекты, антропогенные и другие объекты) (рис. 2).

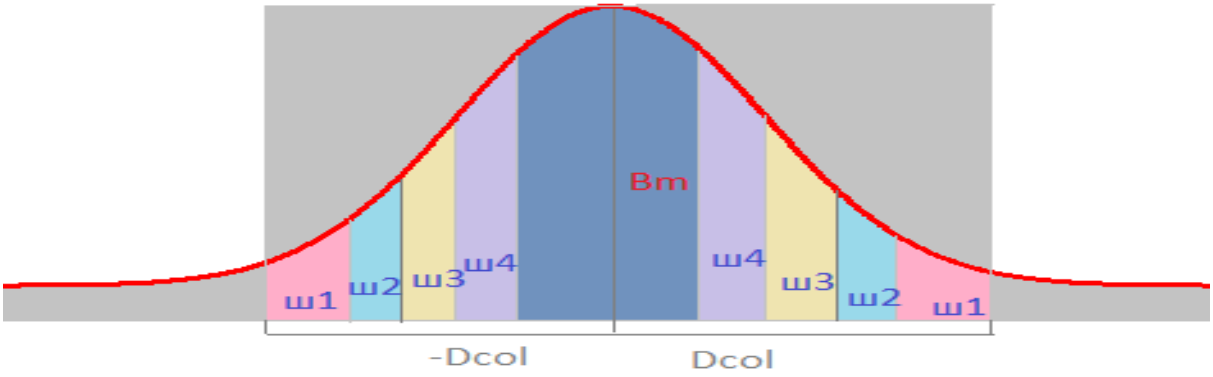


Рис. 2. Статистические параметры изображения

Информация по результатам работы программы:

1) На экране изображаются классифицированные объекты.

2) Номеруются все связные классифицированные объекты, и вычисляется их количество, площади и суммарная их площадь, длины границ, координаты их центров.

3) По необходимости можно вывести эту информацию в соответствующие текстовые файлы, а все объекты в графические файлы.

Пример применения программы к изображению Ленинградской области приведен на рис. 3.

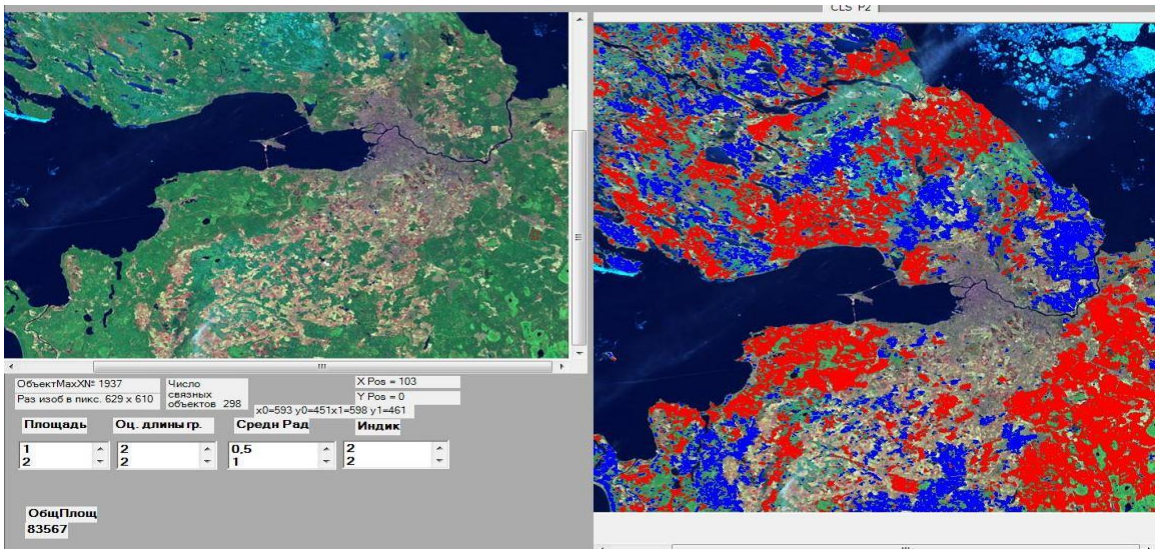


Рис. 3. Фрагмент изображения Ленинградской области, полученное КС Landsat7

При анализе полученного результата очевидна необходимость повторного выполнения процедуры на полученном после первого прохода изображении с целью выделения более мелких объектов.

При выборе в меню программы Sopka пункта «гистограмма полная», получаем изображение рис.4.

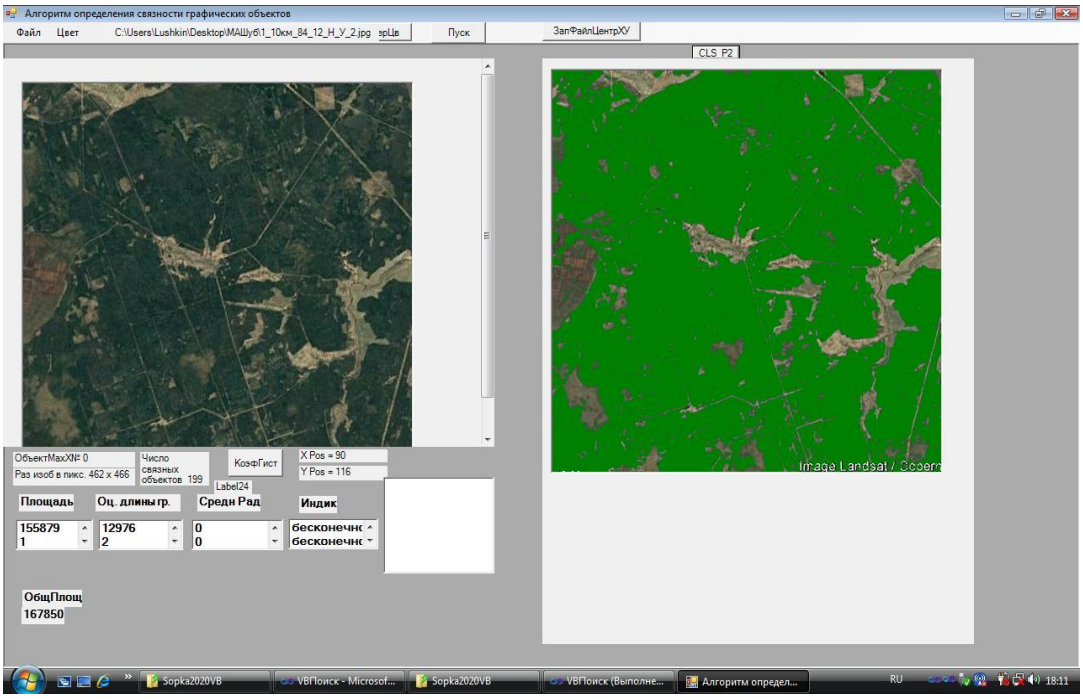


Рис.4. Технология классификации Sopka шаг1

Далее при выборе в меню программы Sopka пункта «гистограмма», получаем изображение рис.5.

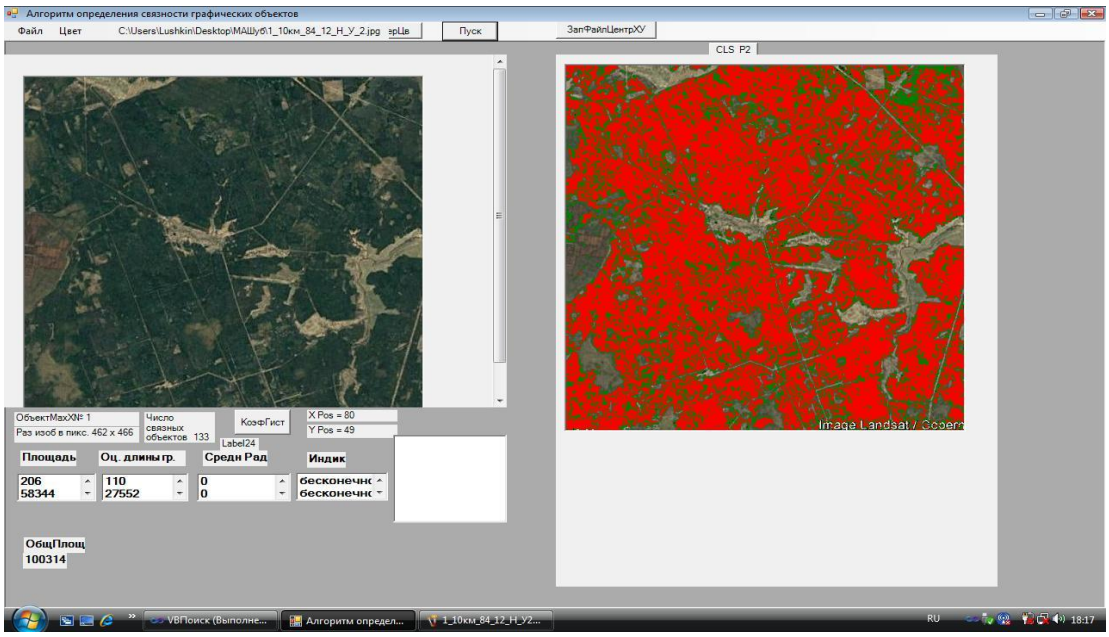


Рис.5. Технология классификации Sopka шаг2

Если результат классификации неудовлетворителен (выделяется 2-3 класса) то повторяем процедуру - выбираем в меню программы Sopka пункта «гистограмма», получаем изображение рис.6, рис. 7 и т.д.

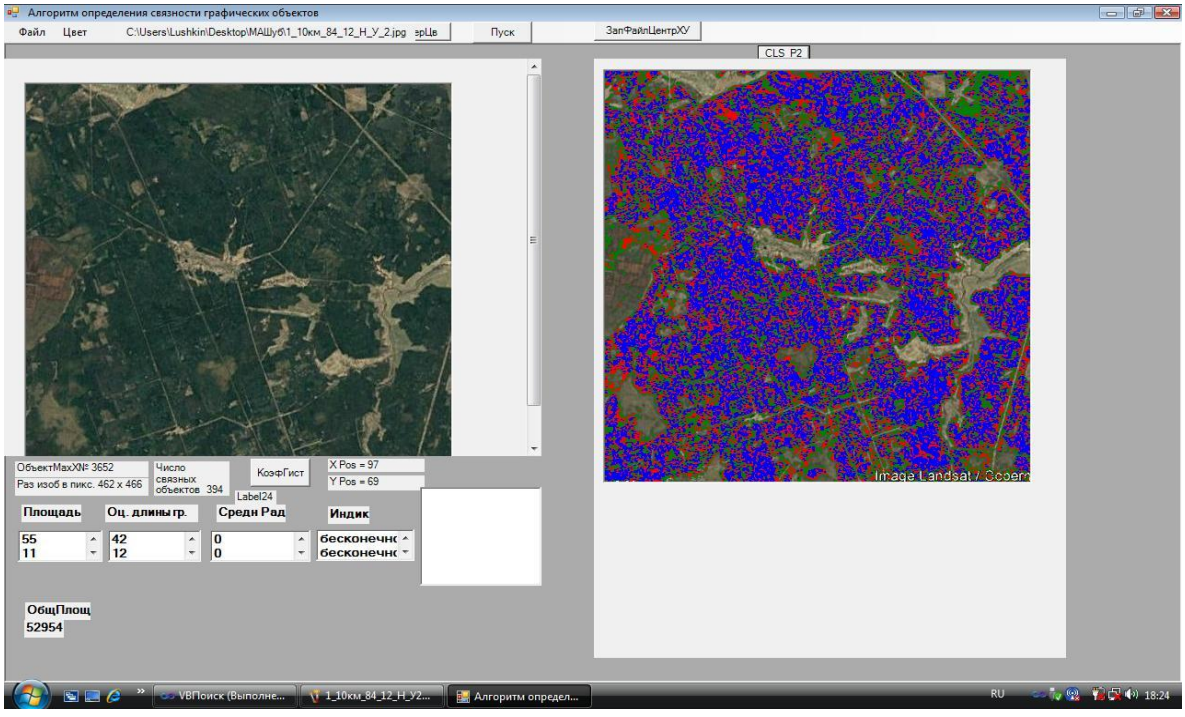


Рис.6. Технология классификации Sopka шаг3

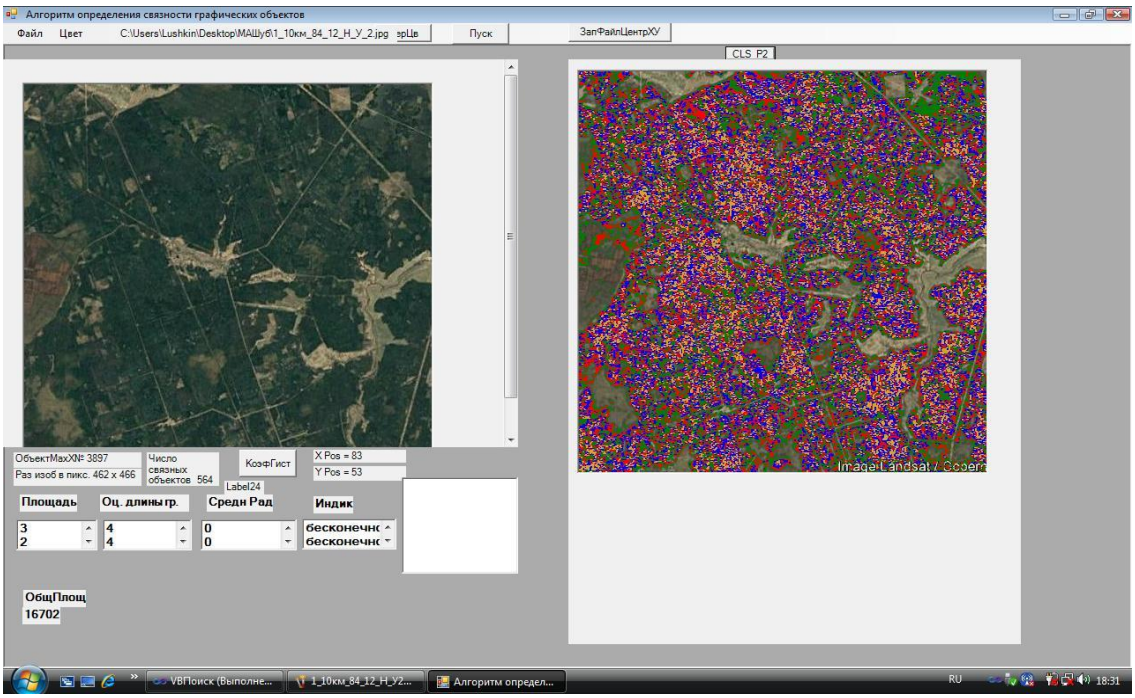


Рис.7. Технология классификации Sopka шаг 4

В результате получаем изображение рис.8 с указанием границ выделенных элементов изображения, принадлежащих разным классам.

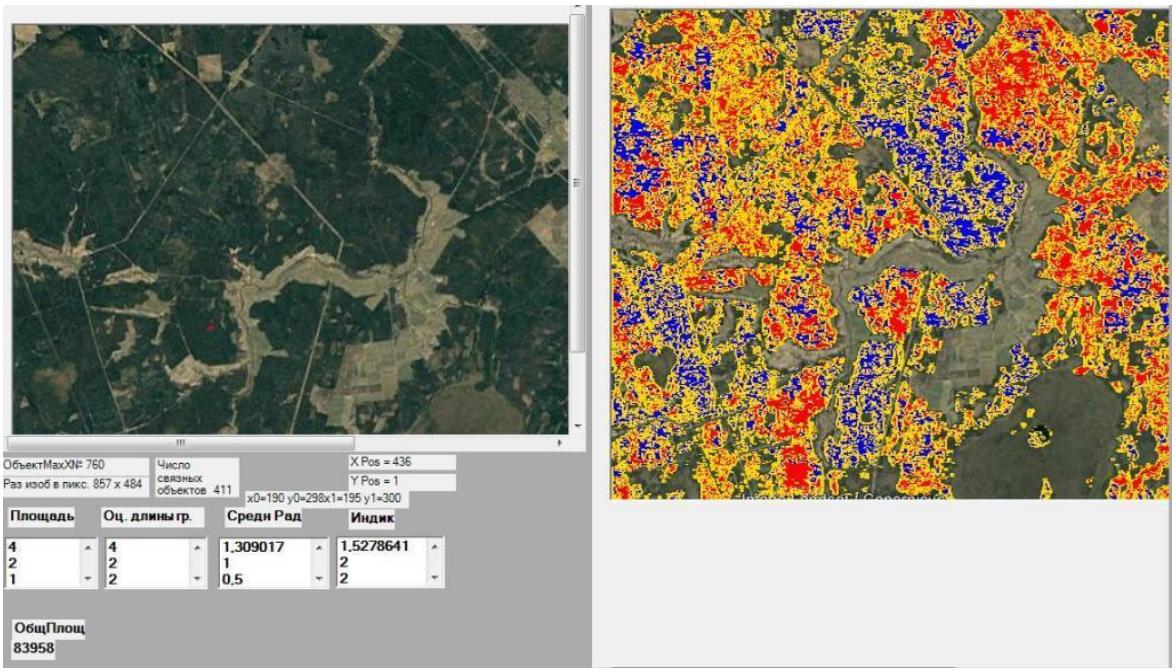
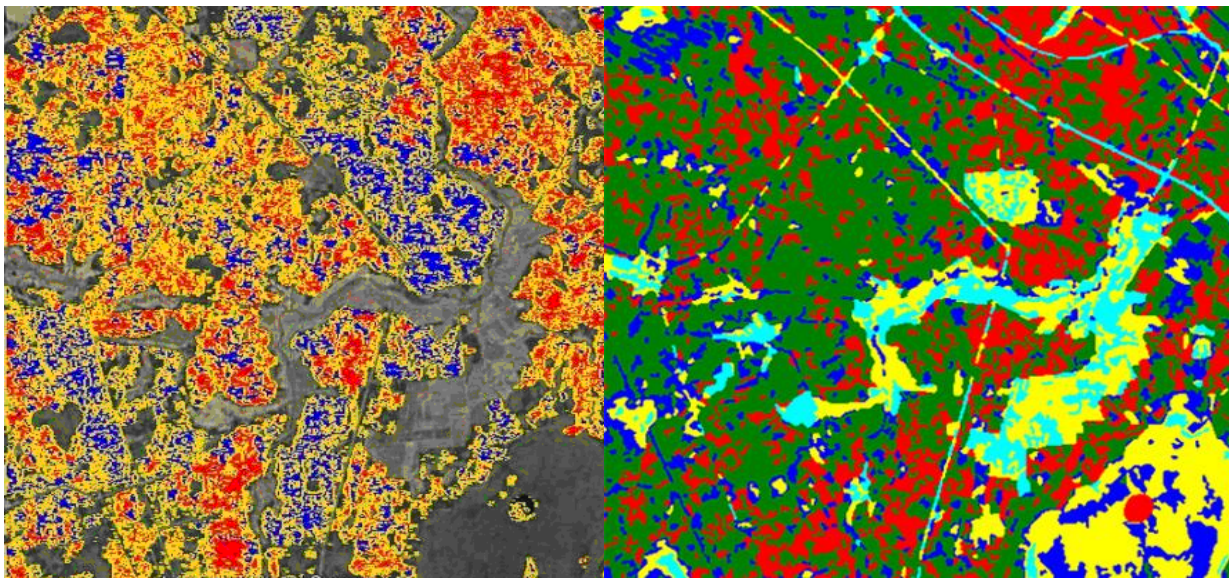


Рис.8. Классификация фрагмента изображения Лисино

Для сравнения на рис. 9 приведен результат классификации того же изображения с помощью программы ENVI. Полученные результаты сопоставимы.



а) б)

Рис. 9. Классификация фрагмента изображения Лисино (Sopka)

Рис. 10. Классификация фрагмента изображения Лисино (5 классов ENVI)

Выводы:

1. Апробация показала работоспособность предложенного алгоритма.

2. Для получения приемлемых результатов (в зависимости от поставленной задачи) достаточно 4-5 проходов.

3. Предлагаемая программа при обработке изображений наземных объектов позволяет оценить имеющиеся материалы дистанционного зондирования и выбрать адекватные алгоритмы классификации, что может быть полезно при решении задач лесопользования.

Библиографический список

1. Шубина М.А.Использование методов классификации космических изображений для мониторинга особо охраняемых территорий. В сб. "Информационные системы и технологии: теория и практика" Сб. науч. Трудов. Вып.5 СПб, СПбГЛУ, 2013. с. 51-60

2. Лушкин Н.В. Алгоритм определения связности графических объектов. - Труды Санкт-Петербургской Государственной лесотехнической академии Актуальные проблемы развития высшей школы Санкт-Петербург. 2010. Стр. 308-309.

3. Лушкин Н.В. Алгоритмы и программы на графах для анализа лесных объектов по графическим изображениям. В сб. "Информационные системы и технологии: теория и практика" Сб. науч. Трудов. Вып.10, часть 1,СПб, СПбГЛУ, 2018. с. 84-89

А.М. Родионов, студент 4 курса
СПбГЛТУ им. С.М.Кирова
alekseyrodionov@mail.ru
М.Д.Кононов, студент 4 курса
СПбГЛТУ им. С.М.Кирова
Е.А.Савченко, студент 4 курса
СПбГЛТУ им. С.М.Кирова

ИНТЕЛЛЕКТУАЛЬНАЯ СИСТЕМА ПО ПРОТИВОДЕЙСТВИЮ ТЕРРОРИЗМУ ПУТЕМ ОБЕСПЕЧЕНИЯ КПП СИСТЕМОЙ РАСПОЗНАВАНИЯ ЛИЦ

Сегодня трудно найти человека, который скажет, что информационные технологии никак не повлияли на его жизнь. И действительно, за последние 30 - 40 лет технологический прогресс совершил скачок. Всё, начиная от автоматизации рутинных процессов человека до высоконаучных промышленных производств, было бы невозможно создать без огромного скачка в этой области. Мы бы хотели вам рассказать о системе, которая значительно упрощает работу и повышает эффективность КПП в любом учреждении, а также помогает предотвратить проход нежелательных лиц и повысить общую безопасность от террористических угроз на объекте.

При разработке проекта, основные цели перед нами стояли следующие: улучшение средств обеспечения безопасности, предотвращение возможных террористических актов, уменьшение вероятности возникновения опасных ситуаций, анализ образов людей и объектов, опасных или потенциально опасных для общества, а также интеграция инноваций и улучшений в уже существующие модели обеспечения безопасности.

Для реализации нашего совместного проекта мы использовали один из передовых языков программирования Python и подключаемые к нему библиотеки: OpenCV, Pyqt5, sqlite3, cv2, os, sys. Подробнее о каждой из них: OpenCV — библиотека алгоритмов компьютерного зрения, обработки изображений и численных алгоритмов общего назначения с открытым кодом. PyQt5 — набор расширений («привязок») графического фреймворка Qt, выполненный в виде расширения, добавлен в проект с целью создания пользовательского интерфейса и удобства использования. Оставшиеся библиотеки были использованы для упрощения пользовательского интерфейса и упрощения работы пользователя. Процесс создания проекта, начиная с создания интерфейса, и заканчивая созданием базы данных и распознаванием лиц, представлен на рис. 1:

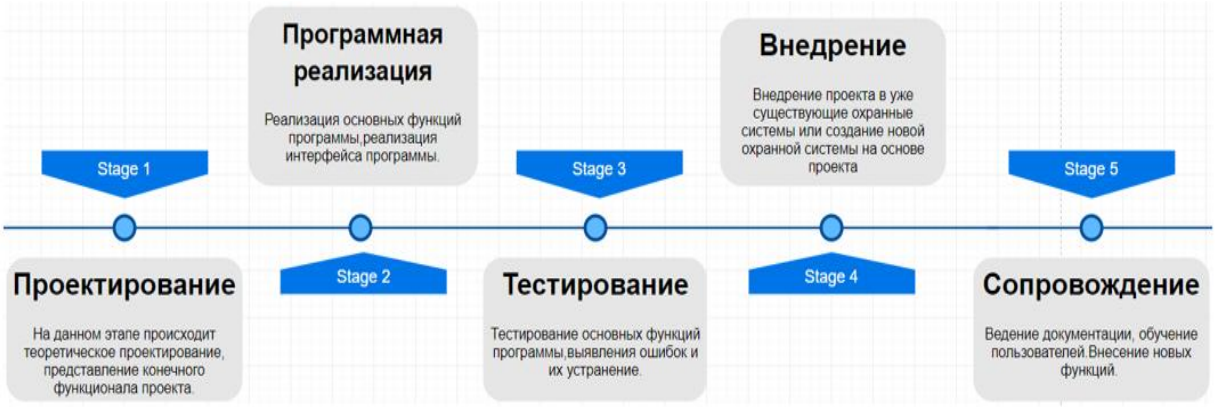


Рис 1. Схема этапов создания проекта

Программа имеет универсальный «каркас», который может в кратчайшие сроки быть адаптирован для нужд какого либо предприятия или же организации.

Рассмотрим реализацию проекта на примере нашего университета. Интеллектуальная система состоит из одного окна с двумя вкладками, в каждой из которых расположены соответствующие элементы взаимодействия администратора с аппаратной частью, и одного окна, которое позволяет оператору следить за результатом распознавания людей. Первая вкладка «загрузка профиля студента» содержит пять кнопок, область для отображения фотографии, три поля ввода для ФИО, календарный выбор даты и два выпадающих списка «институт», «должность». Они созданы для загрузки данных о человеке, который должен иметь доступ в помещения университета и последующей отправке в базу данных. Загрузка фотографии может происходить несколькими способами. Первый подразумевает загрузку уже выполненного изображения в размеченную область интерфейса приложения, по нажатию на кнопку «распознать лицо» мы убеждаемся, что фотография корректна и подходит для загрузки. Второй-открытие камеры, заранее подключенной к компьютеру. По нажатию на соответствующую кнопку, открывается дополнительное окно, в котором, в режиме реального времени распознается лицо человека. У пользователя есть возможность зафиксировать изображение, которое автоматически помещается в корневую папку приложения с наименованием, соответствующим дате и времени выполненного фотографирования. С последующей вставкой изображения по первому способу. Если изображение получилось не подходящим для загрузки в базу, т.е. лицо не было распознано, реализована возможность удалить фотографию нажатием на кнопку и попробовать снова. Демонстрация работы программы с первой вкладкой представлена на рис. 2:

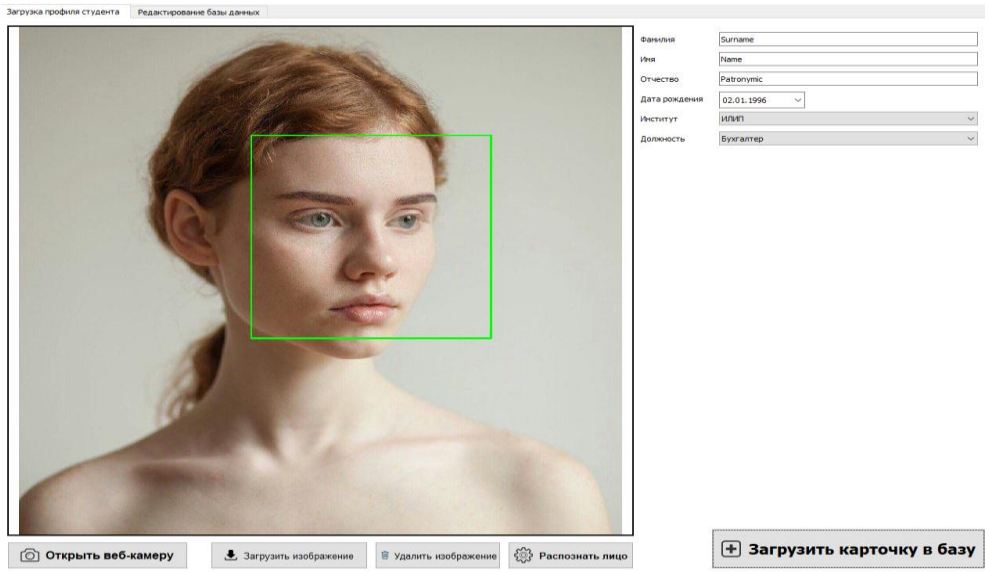


Рис. 2. Демонстрация работы с первой вкладкой

Когда всё готово, мы нажимаем кнопку «Загрузить карточку в базу» и переходим ко второй вкладке нашего приложения. В ней мы можем посмотреть все загруженные ранее карточки. А также выполнить поиск и их отредактировать с последующим изменением. Демонстрация работы пользователя со второй вкладкой программы представлен на рис. 3:

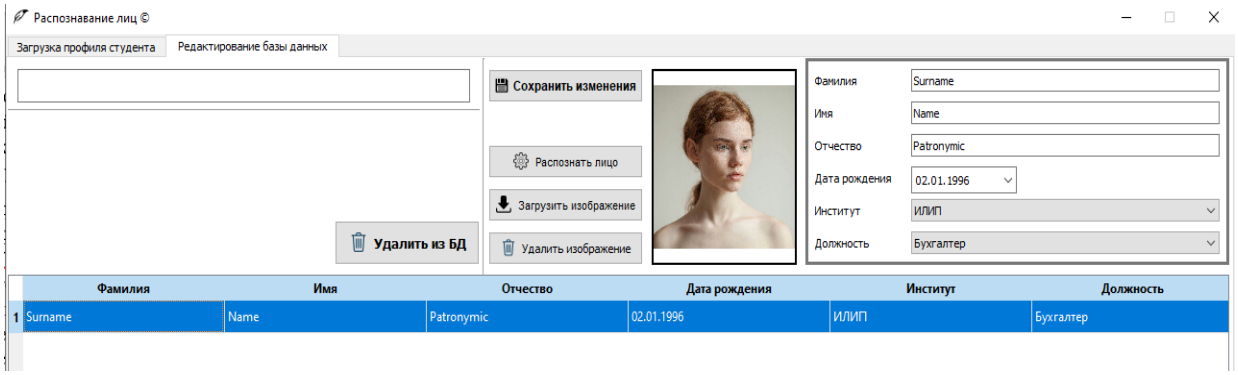


Рис. 3. Демонстрация работы с второй вкладкой

Когда данные человека занесены в базу данных, появляется возможность распознавания этого человека. Оператору необходимо всего лишь открыть окно с программой распознавания и система начнет свою работу. Демонстрация работы программы распознавания показана на рис. 4:

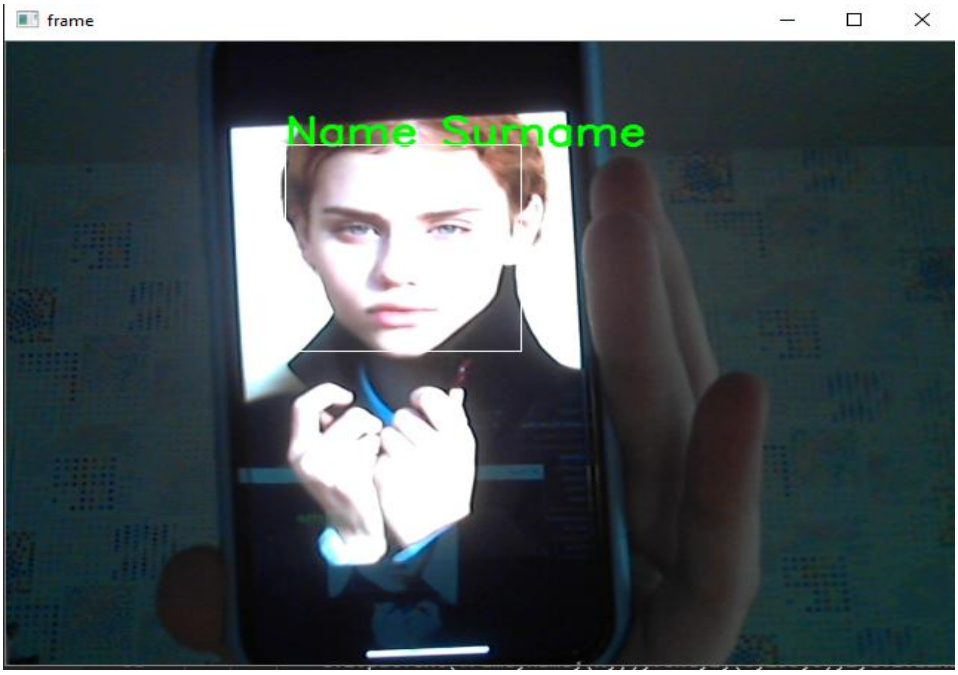


Рис. 4. Демонстрация работы программы распознавания лиц

В основе нашего распознавания лежит построенный нами алгоритм машинного обучения. Его суть заключается в следующем: при каждом распознавании лица, кадр отсылается в базу данных, таким образом, с каждым распознаванием определенного человека повышается точность работы алгоритма.

Таким образом, проект призван обеспечить безопасность организациям/предприятиям. Универсализм проекта в том, что можно создавать как полностью новую охранную систему на основе проекта.

Использована технология, позволяющая обеспечить наиболее точное и быстрое распознавание лиц и образов. Проект позволяет производить сравнение объекта с заранее подготовленным в базе данных, с последующим вынесением решения от оператора. Главной инновацией и отличительной особенностью от аналогов станет распознавание объектов и лиц в режиме реального времени, анализ и выдача информации об объекте/человеке, путём обучения программы распознавать каждого человека, предварительно занесённого в базу данных. Оператору остаётся только принять решение в том случае, если угроза, по мнению оператора, вполне реальна. В противном случае - программа продолжает работать в фоновом режиме. Это лишь один из возможных способов интеграции в охранную систему предприятия.

Потенциальным потребителем могут быть любые фирмы, предоставляющие охранные услуги. Также программа может быть интегрирована в пропускные системы, таким образом снизить вероятность проникновения на объект нежелательных лиц.

Биографический список

1. Методы и средства распознавания образов;А.В. Меженин 2012г.

2. Learning OpenCV 2008; G.Bradski, A.Kaehler.
3. Базы данных концепция баз данных, реляционная модель данных, языки SQL и XML 2010г.
4. Кириллов В.В., Громов Г.Ю. Введение в реляционные базы данных.

С.Ю. Тепляков, магистрант 2 курса
СПб ГЛТУ им. С.М.Кирова
teplyakovsergey1996@gmail.com
С.П. Хабаров, кандидат технических наук, доцент
Кафедра информационных систем и технологий
СПб ГЛТУ им. С.М.Кирова
serg.habarov@mail.ru

ИСПОЛЬЗОВАНИЕ ПРОТОКОЛА WEBSOCKET ДЛЯ ОРГАНИЗАЦИИ ПРЕЗЕНТАЦИЙ В ЛОКАЛЬНОЙ СЕТИ

Введение. В настоящее время практически каждый офис или студенческая аудитория, где проводятся какие-либо либо совещания, лекции или презентации оборудованы собственной локальной вычислительной сетью (ЛВС). Они, как правило, поддерживают Web протоколы [1,2,3] и позволяют использовать в своем составе узлы с разными операционными системами (ОС). Они могут иметь доступ к серверу, как по проводным, так и беспроводным каналам связи [4.5]. В этих условиях представляется актуальным размещение и управление презентационным материалом на сервере с возможность его синхронного или асинхронного отображения на всех устройствах, имеющих связь с сервером (рис. 1).

Рис. 1. Структурная организация ЛВС

Актуальна организация работы системы в обоих этих режимах, когда клиент может не только наблюдать за серверными слайдами, но и возвращаться к уже не активным слайдам сервера или знакомиться с будущими слайдами. Это возлагает на клиента дополнительные функции по возможности выбора режимов, либо “Докладчик”, либо “Слушатель”, В первом случае, управление последовательностью отображения слайдов осуществляется от сервера. Во втором, этим управляет сам клиент.

Такой подход требует организации практически в реальном времени двунаправленного потока данных между клиентом и сервером. Это можно организовать на базе WebSocket протокола. Он представляет собой протокол связи поверх TCP-соединения, и предназначен для обмена сообщениями между браузером и веб-сервером в режиме реального времени.

Постановка задачи. В данной статье рассмотрен подход к построению клиент серверного приложения для проведения конференций и презентаций в ЛВС посредствам синхронного и асинхронного переключения контента. При этом в качестве контента (слайда), который сервер отправляет клиенту, рассматривается байтовый поток графической информации в формате base64. Для этого потребовалось на сервере обеспечить возможность предварительного преобразования исходной информации в формате *.ppt, *.pdf или *.doc в требуемый формат сетевого обмена.

Для написания кода серверной части был выбран язык программирования Python, для клиентской части приложении, которая базируется на работу в среде стандартного браузера, совместно с html разметкой был использован программный код на JavaScript. Упрощенная блок–схема, демонстрирующая принцип работы серверной части приложения имеет вид, представленный на рис 2.

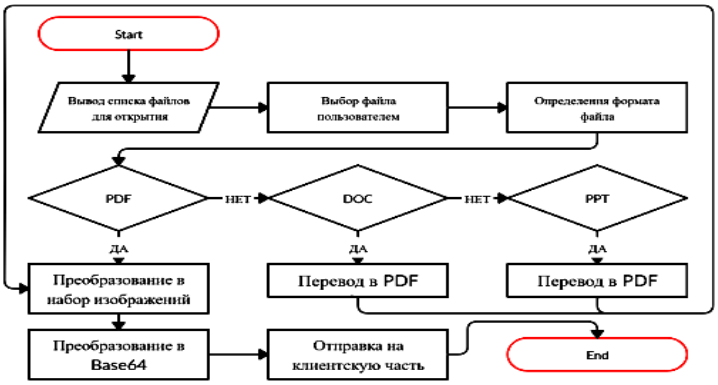


Рис. 2. Упрощенная блок-схема серверной части

Из блок-схемы видно, что для работы северной части приложения необходимо любой исходный документ перевести сначала в формат pdf, затем в набор изображений, и далее в base64. Эти манипуляции необходимы для того, чтобы обеспечить быстрый обмен сообщениями между клиентом и сервером в процессе работы приложения.

Формирование кода серверной части приложения. На первом этапе встает задача получения списка исходных для презентации файлов, которые могут быть использованы в дальнейшей работе (рис. 3).

```
files = os.listdir(os.path.abspath(os.curdir)+'\\INPUT_FILES') # файла каталога
i=0 # устаналиваем начальное значение для цикла
pdf=[] # создаем список для хранения имен файлов PDF
doc=[] # создаем список для хранения имен файлов DOC или DOCX (MS Word)
ppt=[] # создаем список для хранения имен файлов PPT (MS PowerPoint)
# цикл от 0 до количества файлов в каталоге
for i in range(len(files)):
    s = str(files[i])
    if s.find(".pdf",0,len(s)) != -1:
        pdf.append(s)
    elif ( s.find(".doc",0,len(s)) != -1) or ( s.find(".docx",0,len(s)) != -1):
        doc.append(s)
    elif ( s.find(".ppt",0,len(s)) != -1) or ( s.find(".pptx",0,len(s)) != -1):
        ppt.append(s)
```

Рис. 3. Формирование списка исходных файлов

По выполнению этого фрагмента кода будут получены три списка pdf, doc, ppt, которые на следующем шаге добавляются в словарь. Словарь в Python это неупорядоченные коллекции произвольных объектов с доступом по ключу. Пара ключ-значение определяет формат файлов и их количество в списке. После сортировки по количеству элементов можно вывести таблицу исходных файлов (рис. 4).

Список файлов		
PDF	DOC	PPT
Forest.pdf	Academician.docx	FINAL.PPTX
Inspiration.pdf	Art.docx	
Mastery.pdf	Space.docx	
Science.pdf		

Рис. 4. Вывод таблицы исходных файлов

Далее программа просит пользователя скопировать из таблицы название файла, который будет использован при докладе. После чего начнется процесс его конвертирования. Фрагмент кода для файла в формате MS PowerPoint (*.ppt) представлен на рис. 5.

```
    # если файл в формате PPT или PPTX продолжаем работу в блоке
elif ((FILENAME.split(".")[1]).upper() == 'PPT') or ((FILENAME.split(".")[1]).upper() == 'PPTX'):
    ''' FILENAME - название файла, который ввел пользователь
        os.path.abspath(os.curdir) - рабочий каталог
        str(FILENAME.split(".")[0] - ищем точку в строке FILENAME и берем все до нее
    '''
    # если дирректория FILE_FOR_WORK\\ + имя файла создана, удаляем ее и все компоненты, затем создаем новую дирректорию
    if os.path.exists(os.path.abspath(os.curdir)+'\\FILE_FOR_WORK\\'+FILENAME.split(".")[0]) == True:
        shutil.rmtree(os.path.abspath(os.curdir)+'\\FILE_FOR_WORK\\'+FILENAME.split(".")[0])
        os.makedirs(os.path.abspath(os.curdir)+'\\FILE_FOR_WORK\\'+FILENAME.split(".")[0])
    else: #иначе создаем дирректорию FILE_FOR_WORK\\ + имя файла
        os.makedirs(os.path.abspath(os.curdir)+'\\FILE_FOR_WORK\\'+str(FILENAME.split(".")[0]))
    print('Производится работа с файлом '+FILENAME+', пожалуйста подождите') # вовод сообщения
    FILENAME1 = os.path.abspath(os.curdir)+'\\INPUT_FILES\\'+FILENAME # полный путь до файла
    APPLICATION = win32com.client.Dispatch("PowerPoint.Application") # связывание с PowerPoint
    ''' Открываем PowerPoint, ReadOnly = False только чтение - нет, видимо - нет
        path - полный путь до файла pdf, который мы создаем
        convert_from_path - перевод pdf в набор изображений
    '''
    PRESENTATION = APPLICATION.Presentations.Open(FILENAME1, ReadOnly = False, WithWindow = False )
    path = os.path.abspath(os.curdir)+'\\FILE_FOR_WORK\\'+str(FILENAME.split(".")[0])+'\\'+str(FILENAME.split(".")[0])+'.pdf'
    PRESENTATION.SaveAs(path, 32) # сохранить как pdf
    APPLICATION.Quit() # закрыть PowerPoint
    print('Работа с файлом '+FILENAME+' окончена. Создан файл '+str(FILENAME.split(".")[0])+'.pdf')
    print('Производится процесс преобразования файла '+str(FILENAME.split(".")[0])+
    '.pdf в набор изображений, пожалуйста подождите') # вовод сообщения
    images_from_path = convert_from_path(path, output_folder=os.path.abspath(os.curdir)+
    '\\FILE_FOR_WORK\\'+str(FILENAME.split(".")[0]),fmt='jpeg')
    print('Набор изображений из файла '+FILENAME+' создан') # вовод сообщения
```

Рис. 5. Код конвертации файлов

Следующий этап – это перевод изображения в base64, добавление его в список, для дальнейшей пересылки клиенту (рис. 6).

```
for i in range (len(files)): # запускаем цикл for от 0 до len(files) (количество файлов)
    if files[i].endswith(".jpg") == True: # проверяем имеет ли файл формат JPG
        image = os.path.abspath(os.curdir)+'\\FILE_FOR_WORK\\'+str(FILENAME.split(".")[0])+'\\'+files[i]
        with open(image, "rb") as img_file:  #открытие файла
            my_string = base64.b64encode(img_file.read()) # чтение файла
            my_string = my_string.decode('utf-8') # перевод изображения в base64
        spisok.append(my_string) # добавление полученного значения в формате base64 в список
```

Рис. 6. Перевод изображения в формат base64

Так как программа предназначена для работы в ЛВС, то достаточно полезным будет получение списка всех активных узлов сети, подключенных к серверу. Фрагмент кода, реализующий этой функции, приведен на рис. 7. Для получения списков компьютеров в сети необходимо выполнить консольную команду cmd ‘net view’, перехватить результат и записать его в строку. Пример строки, полученной после перехвата, может иметь вид:

```
b'\x88\xac\xef\xe1\xa5\xe0\xa2\xa5\xe0\xa0\x87\xa0\xac\xa5\xe2\xa
a\xa8\r\n\r\n----------------------------------\r\n\\\\IGOR
\r\n\\\\TEPLYAKOV
```

```
\r\n\x8a\xae\xac\xa0\xad\xa4\xa0\xa2\xeb\xaf\xae\xab\xad\xa5\xad\
xa0 \xe3\xe1\xaf\xa5\xe8\xad\xae.\r\n\r\n',
```

где IGOR и TEPLYAKOV – имена компьютеров. Заметим, что перед именем компьютера в сети всегда есть подстрока ‘\r\n\\\\’, а после имени компьютера пробелы, по этим критериям и будет произведен поиск имен компьютеров (рис. 7).

```
# выполняем команду net view, перехватываем резуьтат
local_pc = str(subprocess.check_output("net view"))
array = [] # обьявление словаря
i=0 # начальное значение для цикла
string_host='/r/n////' # искомая строка
string_host= string_host.replace("/",'\\') # замена / на \
while i<= len(local_pc): # цикл от 0 до конца строки local_pc
    # поиск подстроки string_host в строке local_pc
    poisk = local_pc.find(string_host, i, len(local_pc))
    if poisk == -1: # если поиск не дал результатов выходим из цикла
        break # выход из цикла
    else: # иначе: подстрока есть в строке, мы получили индекс ее начала
        i = poisk+8 # увеличиваем счетчик на 8 символов (\r\n\\\\)
        while local_pc[i] != ' ': # пока local_pc[i] не равно ' ' цикл
            pc_name=pc_name+local_pc[i] # прибавляем к строке pc_name один символ
            i+=1 # прибавляем счетчик на 1
        array.append(pc_name) # добавляем строку в список
        pc_name='' # присвоивем строке pc_name пустую строку
        i+=1 # прибавляем счетчик на 1
```

Рис. 7. Поиск активных компьютеров в сети

По имени компьютера может быть получен локальный IP адрес. Это позволяет использовать полученные данные для определения кто из участников конференции вошел в сеть, а кто еще нет. Результат выполнения данного фрагмента кода представлен на рис. 8.

```
-----------------------------------------
|            Компьютеры в сети          |
-----------------------------------------
|    имя компьютера    |     IP-адрес    |
-----------------------------------------
|         IGOR         |  192.168.0.105  |
-----------------------------------------
|       TEPLYAKOV      |  192.168.0.103  |
-----------------------------------------
```

Рис.8. Вывод списка компьютеров в сети

Следующий шаг работы серверного программного кода – это подготовка информации для ее отправки клиенту. Данные, которые будет переданы клиенту: элемент списка, содержащий строку в кодировке base64 (текущий слайд), номер текущего слайда, количество слайдов, количество пользователей, переменная отвечающая за права пользователей. Все эти данные записываются в JSON файл и отправляются клиенту.

Заключительная часть программного кода сервера, следующая после определения всех ранее описанных функций, предполагает выполнение запуска многопоточного WebSocket сервера:

```
# host_ip – локальный IP адрес компьютера, 8081 – порт.
start_server = websockets.serve(counter, host_ip, 8081)
asyncio.get_event_loop().run_until_complete(start_server)
asyncio.get_event_loop().run_forever()
```

Отличительной особенностью приведенного выше программного кода организации WebSocket сервера является то, что в них реализован асинхронный подход к программированию.

Формирование кода клиентской части приложения. Стремление к реализации клиента в среде стандартного браузера с поддержкой API протокола WebSocket находит свое отражение и в подходе к реализации программного кода для клиентской части приложения. В его основе html разметка страницы со скриптом на JavaScript (рис. 9).

Рис. 9. Скриншот html-страницы клиента

На рис. 9 представлен скриншот работы клиентской части приложения, где использованы следующие обозначения: 1 – текущий слайд, 2 – сообщение о количестве подключенных пользователей, 3 – номер текущего слайда, 4 – переключить слайд назад, 5 – переключить слайд вперед, 6 – режим просмотра. При выборе режима просмотра “Слушатель” кнопки переключения слайдов становятся неактивными, и смена слайдов осуществляется автоматически по командам с сервера. Фрагмент JS скрипта, связанный с обработкой WebSocket соединения имеет вид:

```
.  .  .
// Подключение к WebSocket серверу
websocket = new WebSocket(uri="ws://192.168.0.103:8081/");
```

```
websocket.onopen = function (event) {
websocket.send(JSON.stringify({action:'null',name:name}));
}
minus.onclick = function (event) {
  //отправка на сервер - клиент нажал слайд назад
  websocket.send(JSON.stringify({action: 'minus'}));  }
plus.onclick = function (event) {
  //отправка на сервер - клиент нажал слайд вперед
  websocket.send(JSON.stringify({action: 'plus'}));   }
websocket.onmessage = function (event) {
 data = JSON.parse(event.data);
 switch (data.type) {
 case 'state':
   value.textContent = data.value;
   if (value.textContent <= 1){
      minus.style.setProperty('--element-height', 20+'%')
      value.textContent = 1;
      data.value = 1;  }
   else if (value.textContent > 1){
     minus.style.setProperty('--element-height', 100+'%')
   }
   if (value.textContent <= data.kolichestvo){
     var pc = document.getElementById("pic_cntr");
     // создаем динамический объект, добавляем строку
     // base64 для получения изображения
     pc.innerHTML='<img src="'
                    + data.slide + '" >';
   }
   break;
.  .  .
```

После того, как пользователь открыл клиентскую html страницу, ему предлагают ввести имя. Далее происходит процесс отправки сообщения на сервер. Сервер получает данные и выводит информацию. По окончанию процесса взаимодействия клиента и сервера необходимо предусмотреть достаточно четкое закрытие соединения с обеих сторон:

```
websocket.onclose = function(event) {
  if (event.wasClean) {
    alert('Соединение закрыто чисто');
  } else {
    alert('Обрыв соединения, '+'код: ' + event.code );
  }
```

В процессе функционирования системы важным является мониторинг как процесса подключения узлов сети к серверу, так и режимов работы каждого из клиентов. С этой целью в системе предусмотрен вывод в консоли сервера информации обо всех подключениях, количестве пользователей и всех их действиях в процессе работы (рис. 10).

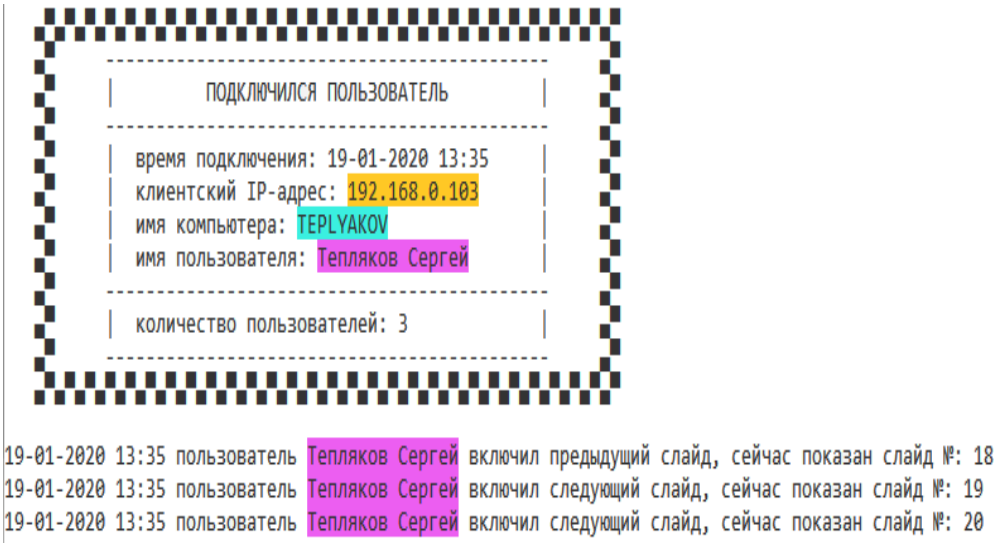


Рис. 10. Вывод информации о действиях пользователей

Библиографический список

1. Хабаров С.П., Шилкина М.Л. Вычислительные машины, системы и сети: Учебное пособие. — СПб.: СПбГЛТА, 2017.— 240 с.

2. Бойцов А.К., Хабаров С.П. Современные беспроводные технологии в лесном хозяйстве. В сборнике: Актуальные вопросы в лесном хозяйстве Материалы III международной научно-практической конференции молодых ученых. 2019. С. 138-141.

3. Вагизов М.Р., Тепляков С.А. Разработка web-картографического сервиса GIS-Aero. В сборнике: Информационные системы и технологии: теория и практика Сборник научных трудов научно-технической конференции . Том 11. – СПб.:СПбГЛТУ, 2019 С. 40-44.

4. Хабаров С.П. Доступ к беспроводным Ad Hoc сетям средствами ОС Windows 10. В сборнике: Информационные системы и технологии: теория и практика Сборник научных трудов. Вып 10. Ч. 2. – СПб.:СПбГЛТУ, 2018. С. 50-60.

5. Хабаров С.П. Использование утилиты WebSocketd в среде ОС Ubuntu Server. // Информационные системы и технологии: теория и практика. Сборник научных трудов научно-технической конференции: Сборник трудов научно-технической конференции. Том 11. – СПб.:СПбГЛТУ, 2019. – С. 95-106.

С.П. Хабаров, кандидат технических наук, доцент
Кафедра информационных систем и технологий
СПб ГЛТУ им. С.М.Кирова
serg.habarov@mail.ru

ИСПОЛЬЗОВАНИЕ ПАКЕТА OCTAVE ДЛЯ ОБРАБОТКИ ПОСТУПАЮЩИХ ПО СЕТИ ДАННЫХ

Для выполнения сложных математических расчетов, обработки экспериментальных данных и моделирования технических систем в настоящее время, наряду с системой MATLAB и SimInTech [1-4], широкое применение находят и такие свободно распространяемые специализированные математические пакеты, как GNU Octave, Scilab, Maxima, R и Sage. Среди них наибольшую популярность получил пакет GNU Octave, который использует совместимый с MATLAB язык высокого уровня и представляет собой интерактивный командный интерфейс для решения сложных математических задач и проведения различных вычислительных экспериментов.

Отличительной чертой этого пакета является его возможность вести обработку данных, которые поступают от реальных технических систем или лабораторных макетов непосредственно по компьютерной сети. При этом появляется возможность не только обрабатывать поступающие данные, но и управлять наблюдаемым объектом в реальном масштабе времени. В этих условиях представляется актуальной разработка технологии использования пакета GNU Octave для обработки данных, которые поступают в него от внешнего объекта по проводной или беспроводной [5,6] компьютерной сети.

Пусть имеется некоторый наблюдаемый объект, включающий в свой состав UDP сервер, который по запросу клиента может выдать ему нужную для оценки и обработки текущую информацию о состоянии конкретных переменных состояния этого объекта. Поставим задачу по реализации на базе пакета GNU Octave клиента, который должен обеспечивать:

- подключение клиента к UDP серверу;
- передачу запроса на аутентификацию и список требуемых ему данных;
- прием байтового потока данных от сервера;
- распаковку байтового потока данных в массив вещественных чисел (single или double), соответствующих запрашиваемому клиентом списку наборов данных с указанием метки времени их возникновения на сервере;
- обработка принятых временных наборов данных

Для реализации поставленной задачи первое, что надо сделать, это с сайта https://octave.sourceforge.io/packages.php загрузить пакет sockekets-1.2.0.tar.gz, а затем установить его в среду GNU Octave с помощью консольной команды:

```
pkg install C:\my_folder\sockekets-1.2.0.tar.gz,
```

где C:\my_folder\ – это папка, в которой находится скачанный архив модуля пакета. После установки необходимо перезапустить GNU Octave. Следует особо отметить тот факт, что так как загрузка модулей не автоматизирована, то при каждом сеансе работы в среде GNU Octave требуется новый ввод консольной команды вида:

```
pkg load sockets
```

Для реализации в среде GNU Octave программного кода, который поддерживал бы все отмеченные выше функции UDP клиента, может быть использован достаточно простой по структуре скрипт, который представлен на рис.1.

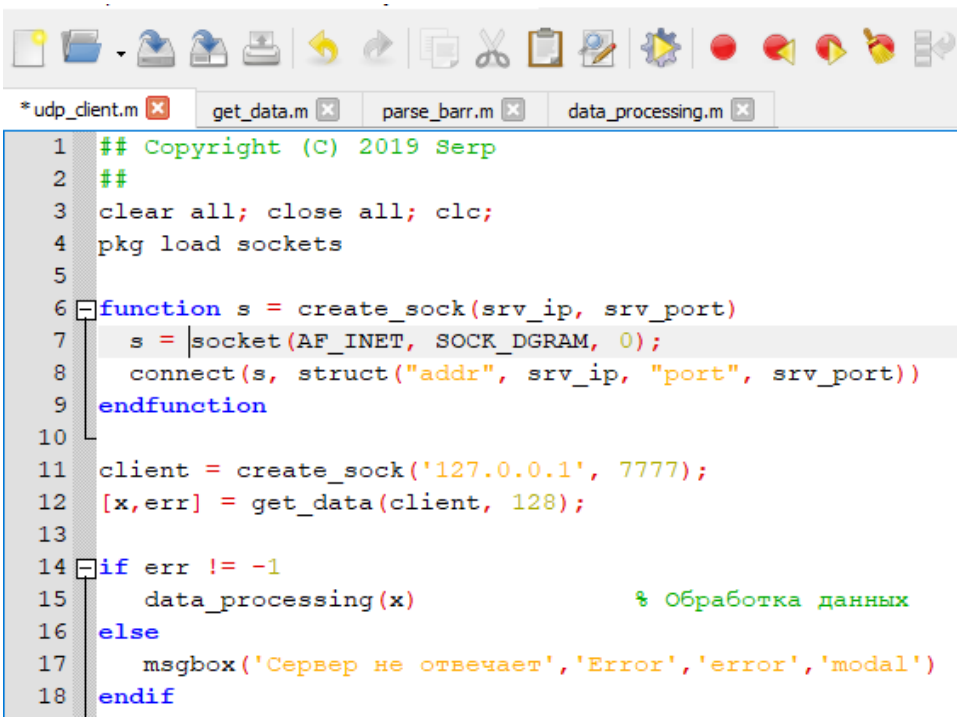


Рис. 1. Исходный код файла udpClient.m

Если принять во внимание тот факт, что данные с сервера в ответ на запрос клиента поступают как запакованная в байтовый поток последовательность данных вещественного типа двойной точности, то процедура формирования полученного от сервера массива данных может иметь вид:

```
function [x,len] = get_data(client, buff)
    send(client,'serp_query');  % Запрос на получение данных
    x=[]                        % Получение данных
    while true
      [barr,len]=recv(client,buff)
      if len==-1 || (len==8 && char(barr)=='serp_end')
        break
```

```
      endif
      [x] = [x; parse_barr(barr, 8)]
    endwhile
    disconnect(client);
endfunction
```

Данная процедура формирует стандартный для сервера запрос, в ответ на который сервер в произвольные моменты времени посылает, а клиент принимает данные о состоянии объекта. Процесс обмена сервер завершает символьным однобайтовым сообщением. Структура дейтаграмм сервера – это набор упакованных 8-байтовых вещественных чисел (double), соответствующих текущему серверному времени и значениям переменных состояния объекта.

Первой в дейтаграмме передается метка времени, а вычислив $k = len/8 - 1$, можно определить количество переменных, посылаемых сервером. При этом процедура преобразования полученного от сервера байтового потока данных в массив 8-байтовых вещественных чисел может иметь следующий вид:

```
function [x] = parse_barr(barr, n)
    if n==4, type="single"; endif
    if n==8, type="double"; endif
    for i=1:length(barr)/n
        z = barr([(i-1)*n+1:i*n]);
        x(i) = bitpack(bitunpack(z),type);
    endfor
endfunction
```

В качестве иллюстрации процесса обработки клиентом принятого от сервера потока данных используем стандартную для GNU Octave процедуру аппроксимации полученных данных в виде полиномов второго и третьего порядка для двух разных переменных состояния, генерируемых на стороне UDP сервера:

```
    function data_processing(x)
        plot(x(:,1),x(:,2),x(:,1),x(:,3))
        grid on
        % Коэффициенты полинома y=a1*x^2+a2*x+a3
        [a] = polyfit(x(:,1),x(:,2),2);
        % Коэффициенты полинома y=a1*x^3+a2*x^2+a3*x+a4
        [b] = polyfit(x(:,1),x(:,3),3);
        title(['A = [',num2str(a),']; B = [',num2str(b),']'])
    endfunction
```

После запуска реализованного на C++ UDP сервера, который имитирует работу системы, и подключения к нему UDP клиента, реализованного в среде GNU Octave, вид консоли UDP сервера будет иметь вид, как на рис. 2.

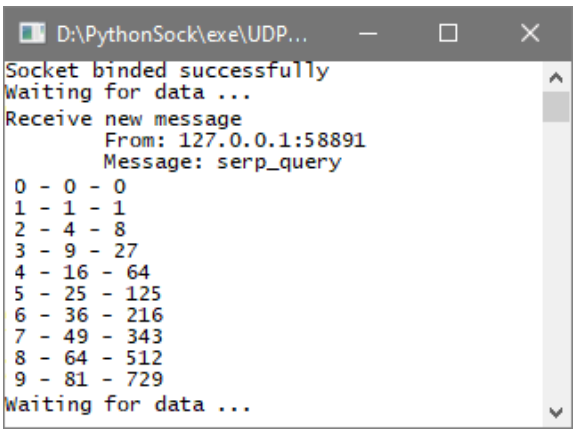


Рис. 2. Вид консоли UDP сервера при подключении GNU Octave UDP клиента

При этом результат работы на стороне клиента процедуры *data_processing()* позволит получить графики полученных от сервера и изменяющихся во времени значений двух переменных наблюдаемого объекта, а также провести их полиномиальную аппроксимацию. Как видно из заголовка графика рис. 3, на стороне клиента были получены аппроксимирующие полиномы $x=t^2$ *и* $y=t^3$, которые точно соответствуют тем наборам данных, которые формировались сервером.

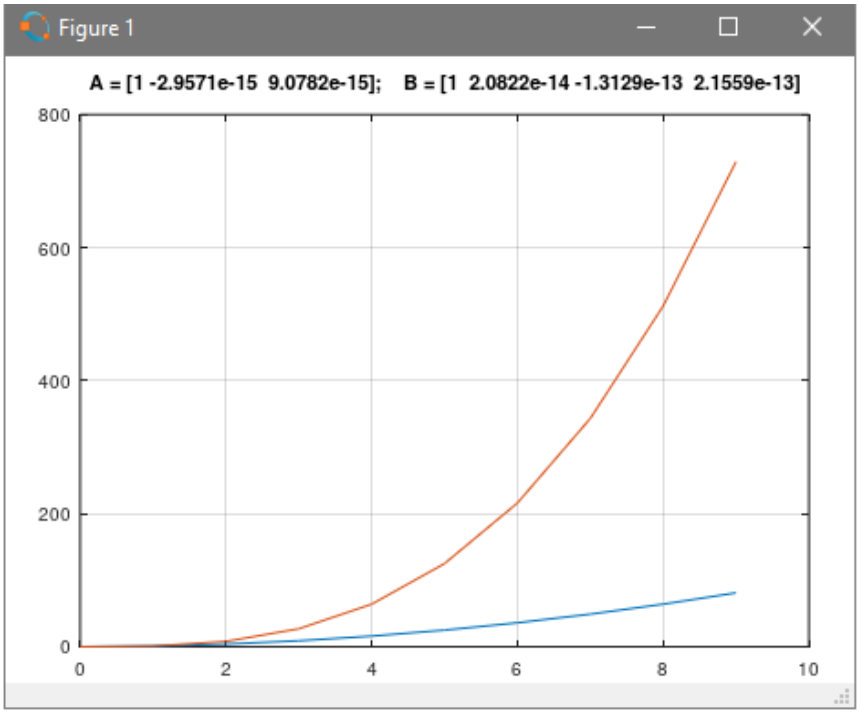


Рис. 3. Вид консоли UDP клиента после получения по сети данных от сервера

Однако рассмотренный ранее пример предполагает, что UDP клиент за одно обращение к серверу получает весь необходимый набор данных. Значительно больший интерес имеет задача обновления данных в реальном масштабе времени. Технологию использования GNU Octave для решения этой задачи рассмотрим для случая, когда UDP клиент и UDP сервер функционируют в среде GNU Octave.

Пусть сервер генерирует поток данных, соответствующий сумме двух гармоник синусоидальных колебаний с частотами 0.5 и 2 Гц, реализация которого с шагом дискретизации *dt* в течение периода *tmax* посылается клиенту. Эти два параметра сервер получает от клиента при стартовом запросе. Это запрос формируется в виде байтового потока, в который клиент упаковывает два вещественных числа одинарной точности *single ([TMAX, DT])*. Одна из возможных реализаций скрипта UDP сервера в среде GNU Octave может иметь вид:

```
% Файл udpServer.m
clear all; close all; clc;
pkg load sockets;
% Создание UDP сокетов
sck_send = create_sock('send',9001,'127.0.0.1');
sck_recv = create_sock('recv',9002);
% Ожидание запроса от клиента
while true
    disp('Waitig query from client ...')
    [barr,len]=recv(sck_recv,128);
    z=parse_barr(barr, 4);
    tmax = z(1);
    dt = z(2);
    for i=0:1:tmax/dt;
        t = i*dt;
        x = sin(2*pi*0.5*t) + 0.8*sin(2*pi*2*t);
        msg_ = double([t, x]);
        msg = bitpack(bitunpack (msg_),"uint8");
        send(sck_send,msg);
        pause(dt)
    endfor
    disp('--> End sending')
endwhile
% Закрытие UDP сокетов
disconnect(sck_send);
disconnect(sck_recv);
```

На UDP клиента возлагаются функции приема поступающих в реальном времени от сервера дейтаграмм, распаковка полученных данных и их обработка, которая состоит в построении скользящего графика отображение полученных данных и построении спектра принимаемого сигнала. Одна из возможных реализаций скрипта UDP клиента может иметь вид:

```
% Файл _udpClient.m
  clear all; close all; clc;
  pkg load sockets;
% Исходные данные (в сек)
  TMAX = 10; DT = 0.1; WIN = 5;
% Создание UDP сокетов
  sck_send = create_sock('send',9002,'127.0.0.1');
  sck_recv = create_sock('recv',9001);
```

```
% Отправка запроса
  msg = bitpack(bitunpack(single([TMAX, DT])),"uint8");
  send(sck_send,msg);
% Ожидание ответа от сервера
  x=[];
  while true
      % Прием дейтаграммы
      [barr,len]=recv(sck_recv,128);
      % Распаковка данных
      [x] = [x; parse_barr(barr, 8)];
      % Обработка полученных данных
      plot_move_signal(x,WIN/DT);
      plot_spektr(x(:,2), DT);
      % Признак окончания процесса
      if x(size(x,1),1) >= TMAX, break, endif
      pause(DT/4);
  endwhile
% Закрытие UDP сокетов
disconnect(sck_recv);
disconnect(sck_send);
```

Результат совместной работы UDP клиента и UDP сервера, запущенных на одном и том же компьютере, но из разных консолей будет иметь вид, аналогичный приведенному на рис. 4.

При этом в составе программного кода UDP клиента используется обращение к внешней функции *plot_move_signal(y,win)*, которая обеспечивает скользящее отображение графика принятого от сервера сигнала в диапазоне, определяемым значением переменной *win*, которое задает длину окна, выраженную в единицах времени. Один из возможных вариантов реализации такой функции может иметь вид:

```
function plot_move_signal(y,win)
    X = y;
    if (size(X,1)-win > 0)
        X = X(length(X)-win:length(X),:);
    endif
    subplot(2,1,1)
    plot (X(:,1), X(:,2));
    axis([X(1,1) ,axis()(2:4)]);
    grid, title('Сигнал'), xlabel('Время (с)')
endfunction
```

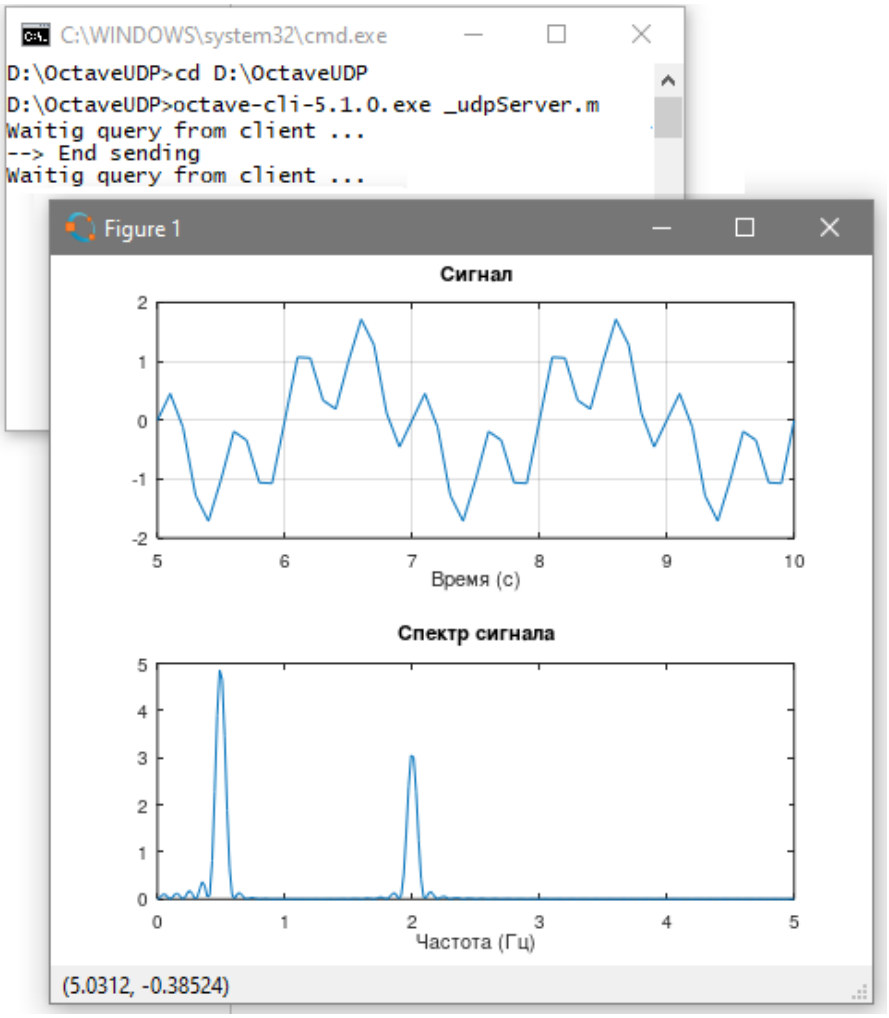


Рис. 4. Результат совместной работы UDP клиента и UDP сервера

Что касается необходимости расчета и построения на стороне клиенте спектра исследуемого сигнала, то для этих целей легко воспользоваться обращением к стандартной, реализованной среде GNU Octave, процедуре быстрого преобразования Фурье.

```
function plot_spektr(y, dt)
    X=fft(y,512);
    Pxx=X.*conj(X)/512;
    f = 1/dt*(0:256)/512;
    subplot(2,1,2)
    plot(f,Pxx(1:257))
    title('Спектр сигнала'); xlabel('Частота (Гц)');
endfunction
```

На рис. 4 приведен результат работы клиента и сервера на конечной стадии приема сигнала, при полностью заполненном окне наблюдений. Несколько иная картина наблюдается в процесс заполнения окна наблюдений (рис. 5)

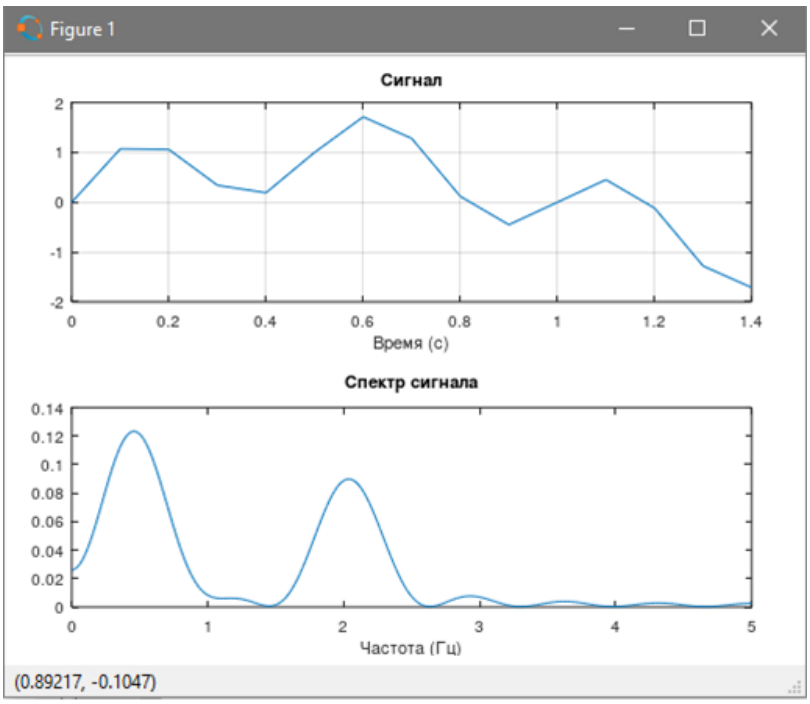


Рис. 5. Результат работы UDP клиента и сервера в первые моменты времени

Более гладкое отображение сигнала и более точное вычисления и построения спектра сигнала будет наблюдаться при уменьшении шага дискретизации до 0.05 секунд (рис. 6). Это полностью соответствует теореме Котельникова для рассматриваемого типа сигнала.

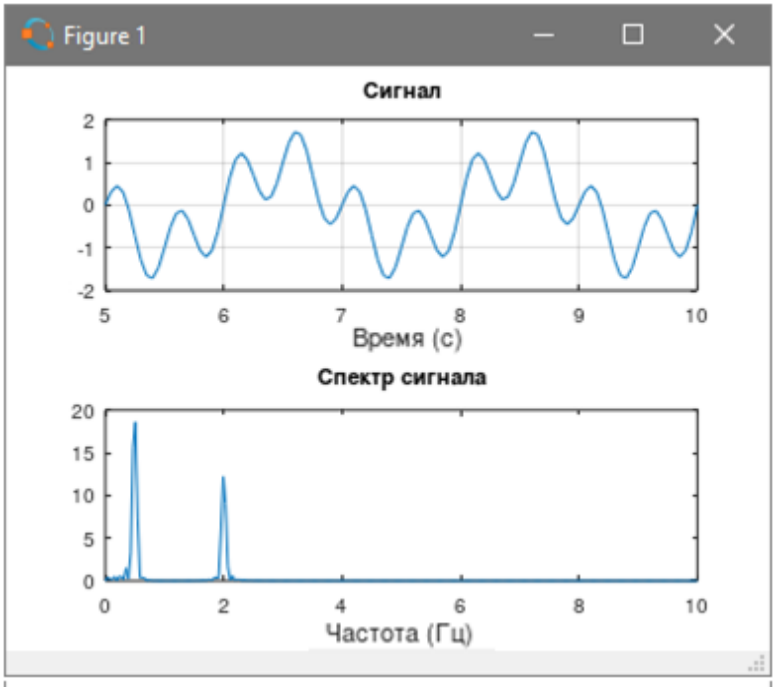


Рис. 6. Результат работы UDP клиента и сервера при dt=0.05

Но проблема состоит в том, что уже и этом случае явно ощущается отставание работы UDP клиента в среде GNU Octave от режима реального времени. Это связано не только с сетевым обменом, но и с отрисовкой обновле-

ний. При этом также становится ясно, что обновления с такой частотой не в состоянии воспринять и человек-оператор.

Встает новая задача, как обеспечить высокий шаг дискретизации процесса с достаточно адекватным восприятием человеком-оператором режима реального времени при заданной пропускной способности сети.

Добиться этого можно за счет того, что клиент в начальном запросе кроме параметров времени наблюдения процесса (Tmax) и шага дискретизации (Td), аналогичных предыдущему примеру, будет передавать на сервер еще один параметр (Tnew), который задает нужный клиенту период обновления данных.

Это потребует от сервера формирования для клиента дейтаграмм, каждая из которых будет содержать уже по Tnew/Td временных отсчетов значений состояний объекта. При этом общее число посылаемых по сети дейтаграмм сократиться до Tmax/Tnew штук, существенно сократив, как нагрузку на сеть, так загрузку клиента по отрисовке обновленной информации.

```
% Файл _udpServer.m
clear all; close all; clc;
pkg load sockets;
try
  % UDP sockets for sending and receiving
  sck_send = create_sock('send',9001,'127.0.0.1');
  sck_recv = create_sock('recv',9002);
  % Waitig query from client
  while true
      disp('... Waitig query from client ...')
      [barr,len]=recv(sck_recv,128);
      client_packet=parse_barr(barr, 8);
      tmax = client_packet(1)
      dt   = client_packet(2)
      tnew = client_packet(3)
      msg=[]; tic
      for i=0:1:tmax/dt;
          t = i*dt;
          x = sin(2*pi*0.5*t) + 0.8*sin(2*pi*2*t);
          msg = [msg, double([t; x])];
          if (mod(t,tnew)==0 && t>0)
              pause(tnew);
              msg = bitpack(bitunpack (msg),"uint8");
              send(sck_send,msg);
              disp(['-> Send ',num2str(length(msg)),' bytes'])
              msg = [];
              toc, tic
          endif
      endfor
      disp('--> End sending')
  endwhile
catch
  disp(['---> Last Error: ',lasterr])
```

```
end_try_catch
disconnect(sck_send);
disconnect(sck_recv);
```

В данном примере серверного кода для возможности его тестирования в режиме реального времени предусматривается между каждыми посылками данных в сеть выполнение паузы на время равное Tnew. Это позволит реально смоделировать частоту отправки дейтаграмм с сервера в сеть. Кроме этого в данном серверном коде в отличие от ранее приведенного отсутствует передача клиенту финального сообщения и сразу осуществляется закрытие сокетов.

Естественно, что такой подход приводит и к реализации некоторых дополнительных функций, возлагаемых на работу UDP клиента. В частности, это потребует динамического вычисления длины буфера приема дейтаграмм в зависимости от длины формируемых сервером данных единичного временного отсчета (data_len). Кроме этого существенно изменится процедура распаковки полученной от сервера дейтаграммы на совокупность данных, соответствующих отдельным временным отсчетам. Ввиду отсутствия финальной посылки от сервера, потребуется сформировать признак окончания процесса обмена информации для нормального завершения программы со своевременным закрытием сокетов.

```
% Файл _udpClient.m
clear all; close all; clc;
pkg load sockets;
try
  % Создание UDP socket для приема и передачи
  sck_send = create_sock('send',9002,'127.0.0.1');
  sck_recv = create_sock('recv',9001);
  % Исходные данные (в сек)
  Tmax = 6; Td = 0.1; Tnew = 1; Win = 5;
  data_len = 16;
  buff = (int16(Tnew/Td )+1)*data_len
  % Отправка запроса на сервер
  query = bitpack(bitunpack(double([Tmax,Td,Tnew])),"uint8");
  send(sck_send, query);
  % Ожидание ответа от сервера
  x=[];
  while true
      % Прием дейтаграммы
      [barr,len]=recv(sck_recv, buff);
      % Распаковка данных
      y = [];
      for i=1:len/data_len
          z = barr([(i-1)*data_len+1:i*data_len]);
          y = [y; parse_barr(z, 8)];
      endfor
      [x] = [x; y]
      % Обработка полученных данных
      plot_move_signal(x, Win/Td);
```

```
      plot_spektr(x(:,2), Td);
      % Признак окончания процесса
      if x(size(x,1),1) >= Tmax, break, endif
      pause(Tnew/10);
  endwhile
catch
  disp(['Last Error: ',lasterr])
end_try_catch
disconnect(sck_recv);
disconnect(sck_send);
```

Результат совместной работы в двух консолях GNU Octave на одном компьютере UDP клиента и UDP сервера при начальном запросе с клиента, соответствующем параметрам Tmax=6, Td=0.1 и Tnew=1, приведен на рис. 7. На консоли сервера видно, что он успешно распаковал начальный запрос от клиента, после чего сформировал 6 дейтаграмм. Первая из них длиной 176 байт соответствует 11 временным отсчетам по 16 байт каждый. Остальные пять имеют длину 160 байт (по 10 временных отсчетов).

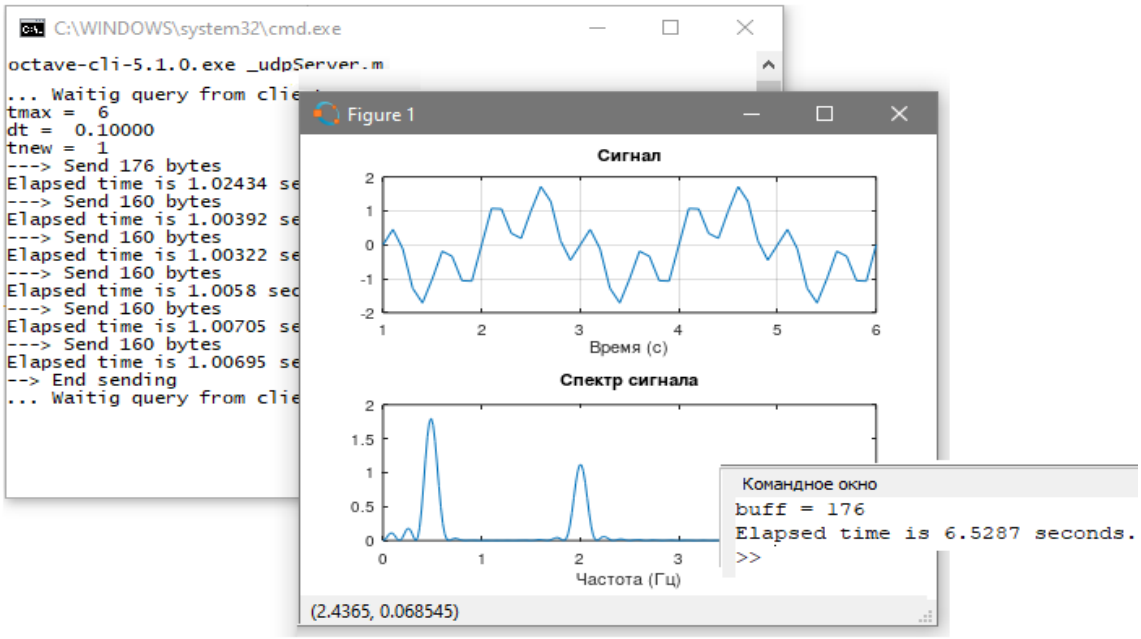


Рис. 7. Взаимодействие UDP клиента и сервера при Tmax=6, Td=0.1 и Tnew=1

Добавленные в программные коды клиента и сервера операторы tic и toc позволили отследить реальное время выполнения (Elapsed time) отдельных фрагментов программ. Из консоли сервера видно, что период между отправкой дейтаграмм составляет действительно около 1с, а общее время работы программы клиента составило около 6.5 секунд.

В другом эксперименте, когда в программе клиента задавалось Tmax = 20с, Td = 0.01с, Tnew = 0.5с и Win = 5с, расчетная длина буфера была 816 байт, а общее время работы программы составило 21.3407 секунды. При этом периодичность отправки данных с сервера была в диапазоне от 0.505 до 0.509 секунд. Таким образом, можно сделать вывод, что разработанные программы

могут обеспечить сетевую работу практически в реальном масштабе времени.

Библиографический список

1. Хабаров С.П., Шилкина М.Л. Основы моделирования технических систем. Среда SimInTech: Учебное пособие. – СПб: Издательство "Лань", 2019. – 120 с.
2. Хабаров С.П., Шпекторов А.Г. Проектирование и исследование распределенных судовых систем управления техническими средствами в среде SimInTech // Морские интеллектуальные технологии. 2019. №4 (36) Т.2. С. 181-188.
3. Хабаров С.П. Использование SimInTech для анализа систем автоматического управления. // В сборнике: Информационные системы и технологии: теория и практика Сборник научных трудов научно-технической конференции. 2019. С. 106-116.
4. Хабаров С.П., Шилкина М.Л. Технология построения распределенной модели системы управления поливом на лесном питомнике в среде SimInTech. // В книге: Леса России: политика, промышленность, наука, образование. Материалы IV научно-технической конференции. 2019. С. 184-187.
5. Хабаров С.П. Доступ к беспроводным Ad Hoc сетям средствами ОС Windows 10. // В сборнике: Информационные системы и технологии: теория и практика. Сборник научных трудов. 2018. С. 50-60.
6. Хабаров С.П. Организация программных точек доступа средствами ОС Windows 10. // В сборнике: Информационные системы и технологии: теория и практика Сборник научных трудов. 2018. С. 60-73.
7. Заяц А.М., Хабаров С.П. Организация доступа к беспроводным AD HOC сетям информационных систем мониторинга лесных территорий из среды Windows 10. // Известия Санкт-Петербургской лесотехнической академии. – СПб, 2018, Вып. 223. - с. 285-299.
8. Бойцов А.К., Хабаров С.П. Современные беспроводные технологии в лесном хозяйстве. // В сборнике: Актуальные вопросы в лесном хозяйстве Материалы III международной научно-практической конференции молодых ученых. 2019. С. 138-141.

М.Л.Шилкина, кандидат технических наук, доцент
Кафедра информационных систем и технологий
СПб ГЛТУ им. С.М.Кирова
mchernobay@inbox.ru

СОЗДАНИЕ ПРОСТОГО ЧАТА С ИСПОЛЬЗОВАНИЕМ WTBSOCKET СЕРВЕРА НА СЕРВЕРНОЙ ПЛАТФОРМЕ NODE.JS

В последнее время популярным направлением разработок при проектировании клиент-серверных систем является использование технологии WebSocket – двунаправленной асинхронной симметричной связи между браузером и сервером, самого революционного расширения протокола HTTP с момента его появления. Протокол WebSocket поддерживается практически всеми современными браузерами, в нем клиент и сервер являются равноправными участниками обмена данными, в отличие от HTTP протокола, построенного по модели «запрос – ответ».

В данной работе предлагается реализовать клиентский и серверный код на JScript с использованием серверной платформы Node.js для создания социального чата.

Мощная современная программная платформа Node.js была создана, чтобы расширить возможности JScript и превратить его в язык общего назначения. Дать ему возможность взаимодействовать с устройствами ввода-вывода через свой API, подключать внешние библиотеки, написанные на разных языках, обеспечивая вызовы к ним из JavaScript-кода. В ее основе лежит событийно-ориентированное и асинхронное программирование. Node.js применяется преимущественно на сервере, выполняя роль веб-сервера. Поэтому актуальной является задача написания для Node.js веб-сервера, который обменивается данными со своими клиентами по протоколу WebSocket, например, в таком востребованном приложении, как социальный чат.

Прежде чем приступать к написанию серверного кода, надо установить Node.js, загрузив его с официального сайта https://nodejs.org/en/download/. Затем надо подключить дополнительные модули, которые позволяют Node.js реализовать WebSocket протокол. Для этого в меню Пуск в списке установленных программ надо найти в папке Node программу Node.js command prompt, запустить ее и выполнить в командной строке следующую последовательность команд:

```
npm install node-static
npm install ws
```

Теперь можно создать программный код, реализующий функции сервера, и сохранить его в файле server.js, например, в папке C:/ws/Nod:

Файл server.js

```
//'http' и 'node-static' нужны только для обычного сервера, если обычный
//(статика) сервер не нужен, то можно их не подключать
var http = require('http');
```

```
var Static = require('node-static');
var WebSocketServer = new require('ws');
// подключенные клиенты
var clients = {};

// WebSocket-сервер на порту 8081
var webSocketServer = new WebSocketServer.Server({port: 8081});
webSocketServer.on('connection', function(ws) {

    var id = Math.random();
    clients[id] = ws;
    console.log("новое соединение " + id);

    ws.on('message', function(message) {
          console.log('получено сообщение ' + message);
          for(var key in clients) {
                clients[key].send(message);
          }
   });

   ws.on('close', function() {
         console.log('соединение закрыто ' + id);
         delete clients[id];
    });

});

// обычный сервер (статика) на порту 8080 (если надо, раскомментировать)
//var fileServer = new Static.Server('.');
//http.createServer(function (req, res) {
//    fileServer.serve(req, res);
//}).listen(8080);

console.log("Сервер запущен на портах 8080, 8081");
```

В первых строках кода server.js подключаются только что добавленные к платформе модули, осуществляющие обмен по WebSocket протоколу, создается пустой глобальный ассоциативный массив clients = {} для хранения подключенных клиентов и сам WebSocket сервер, который будет слушать 8081 порт. Если придет сообщение, то он отошлет его всем подключенным на текущий момент клиентам (хранящимся в массиве clients) и выведет его в лог сервера. Индекс (ID) для новых клиентов, записываемых в ассоциативный массив clients, генерируется с помощью встроенного датчика случайных чисел. При отключении клиента соответствующий элемент массива clients корректно удаляется.

Приведенный выше код WebSocket сервера с использованием платформы Node.js достаточно прост для понимания, и чтобы запустить сервер, нужно в окне Node.js command prompt перейти при помощи команды cd в каталог C:/ws/Nod и выполнить команду

```
node server.js
```

После чего вид консоли сервера примет вид, аналогичный тому, что представлен на рис.1.

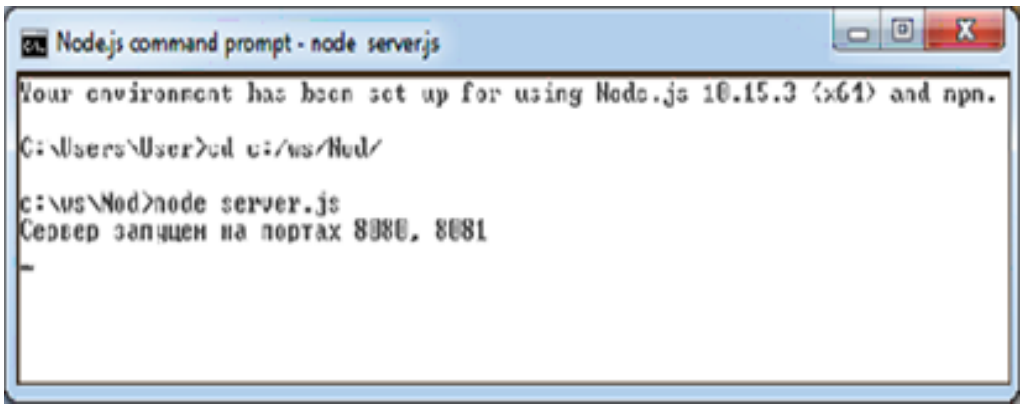


Рис.1. Вид консоли работающего сервера server.js

Теперь надо создать код клиентской страницы C:/ws/Nod/Chat.html:

Файл Chat.html

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
</head>
<body>

<!-- форма для отправки сообщений -->
<form name="publish">
     <input type="text" name="message"/>
     <input type="submit" value="Отправить"/>
</form>

<!-- здесь будут появляться входящие сообщения -->
<div id="subscribe"></div>

<script src="browser.js"></script>

</body>
</html>
```

Страница Chat.html использует скрипт browser.js:

Файл browser.js

```
if (!window.WebSocket) {
      document.body.innerHTML = 'WebSocket в этом браузере не
поддерживается.';
}
// создать вебсокет-подключение к 8081 порту локального хоста
var socket = new WebSocket("ws://localhost:8081");

// отправить сообщение из формы publish
document.forms.publish.onsubmit = function() {
  var outgoingMessage = this.message.value;

  socket.send(outgoingMessage);
  return false;
};

// обработчик входящих сообщений
socket.onmessage = function(event) {
  var incomingMessage = event.data;
  showMessage(incomingMessage);
```

```
};

// показать сообщение в div#subscribe
function showMessage(message) {
  var messageElem = document.createElement('div');
  messageElem.appendChild(document.createTextNode(message));
  document.getElementById('subscribe').appendChild(messageElem);
}
```

Чтобы протестировать работу сервера и клиентов чата, надо при запущенном сервере (рис.1) открыть файл Chat.html одновременно в двух браузерах, например, в Google Chrome и Firefox. Теперь можно поочередно отправлять сообщения то из одного, то из другого браузера (рис.2).

В консоли сервера во время обмена сообщениями клиентов будут появляться сообщения аналогично тому, как это представлено на рис.3.

Из полученных результатов следует вывод, что реализовать WebSocket сервер для протипа чата достаточно просто с использованием популярной программной платформы Node.js. При этом код разрабатываемого сервера будет легким для чтения и модификации при необходимости добавления нового функционала.

Например, в разрабатываемый проект можно добавить сохранение истории чата на сервере – если не в базу данных, то для простоты хотя бы в текстовый файл с фиксированным количеством сохраняемых строк. Как только всеми пользователями чата будет введено это количество строк, все эти строки добавляются в конец файла, который тут же перезаписывается.

Новому подключающемуся к чату клиенту надо будет выгружать последнюю версию сохраненного текстового файла, а также строки пользователей чата, которые еще не записаны в файл, а временно сохранены в массиве. Историю чата надо выгружать клиентам при подключении их к серверу.

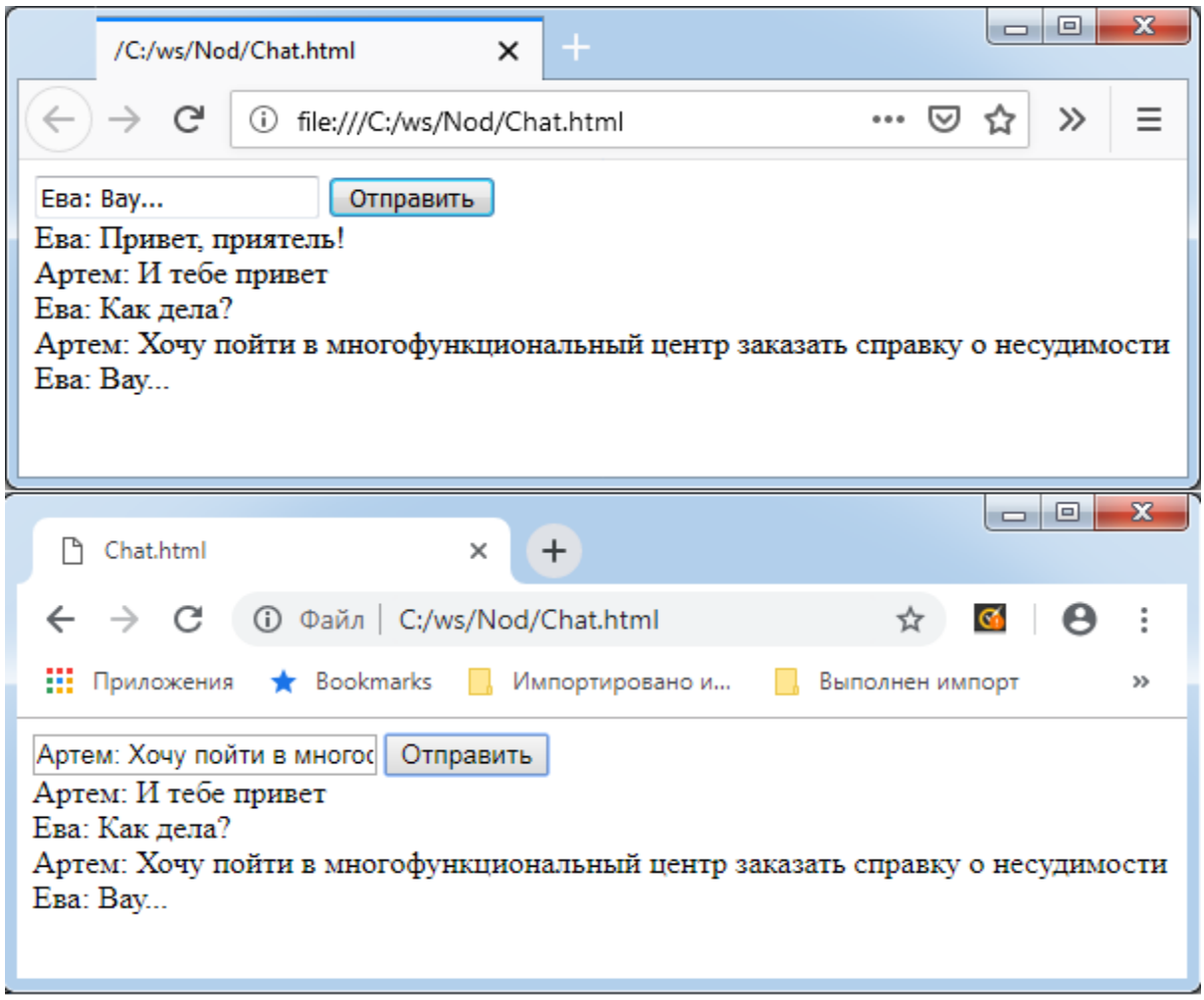


Рис.2. Файл Chat.html, одновременно открытый в двух браузерах

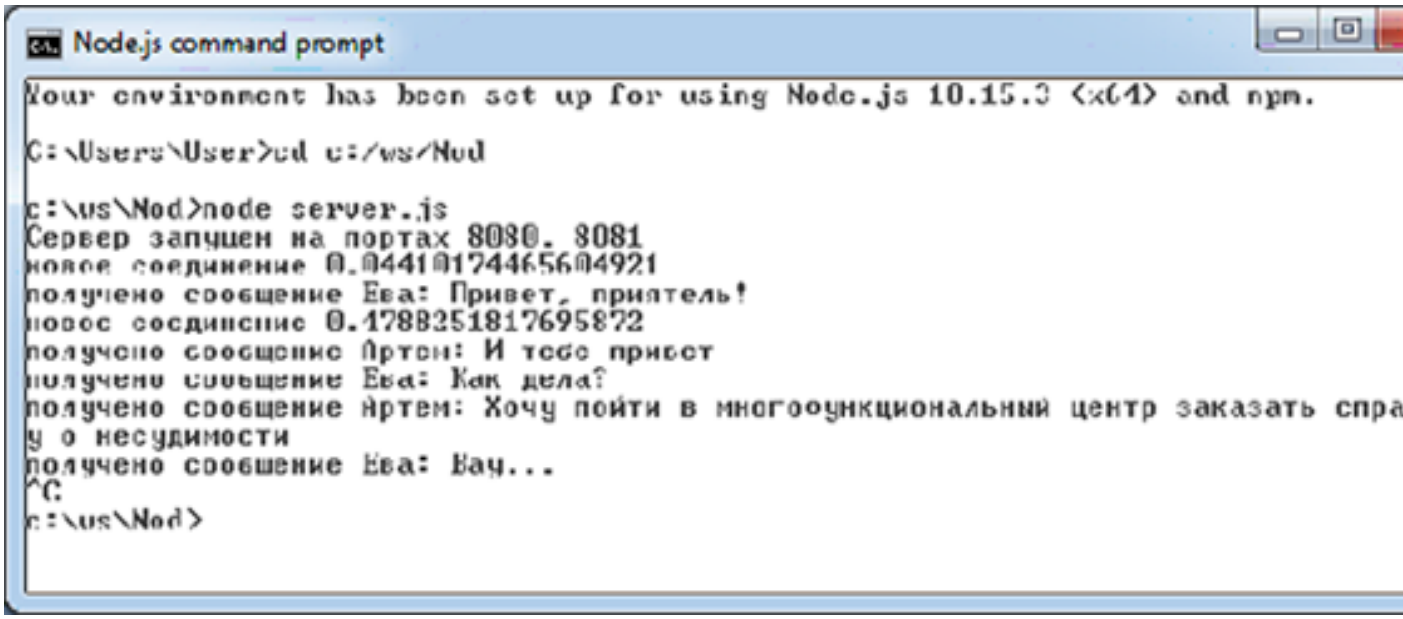


Рис.3. Вид консоли сервера с логом текущих событий чата

Создание WebSocket сервера электронных торгов на серверной платформе Node.js

Можно модифицировать разработанное простейшее распределенное приложение-чат с использованием протокола WebSocket и платформы Node.js, чтобы разработать распределенное Web приложение, реализующее электронные торги. С этой целью требуется на виртуальной машине с гостевой ОС Ubuntu Server разместить WebSocket сервер электронных торгов и организовать доступ к нему с клиентских html страниц, спроектированных для этого сервера.

В рамках данной работы клиент-серверная модель приложения может быть упрощена в предположении, что торги начинаются с момента поступления первого предложения цены и заканчиваются, когда с момента последнего поступившего предложения прошло 50000мс.

- За основу модели предлагается взять клиент-серверное приложение прототипа чата, разработанное в разделе 1. Клиентская часть приложения существенно не изменяется, только на html странице должно размещаться фото продаваемого товара, а в скрипте browser.js в строке с адресом WebSocket сервера должен быть указан IP-адрес виртуальной машины с Ubuntu Server, на которой сохранен и запущен сервер.
- Серверная часть требует модификикации. Связано это с тем, что если в чате любые сообщения рассылались сразу всем клиентам, то в случае электронных торгов сообщения для разных клиентов могут быть, вообще говоря, разные. Например, если клиент предлагает цену ниже текущей, то только ему надо отослать сообщение о неверной цене, при окончании торгов победителю отправляется сообщение о победе в торгах, а остальным – об окончании торгов.

Следует напомнить, что серверная часть приложения в разделе 1 была реализована в ОС Windows с использованием популярной программной платформы Node.js, используемой для создания веб-серверов, а в рассматриваемом приложении сервер располагается на виртуальной машине с ОС Ubuntu Server. Поэтому перед началом разработки сервера требуется установить Node.js на виртуальной машине с Ubuntu Server.

Для этого лучше всего использовать PPA (персональный архив пакетов), который поддерживается компанией NodeSource. В нем содержатся более новые версии Node.js, чем в официальных репозиториях ОС Ubuntu. Прежде всего, необходимо обновить локальный индекс пакетов и установить сам PPA для получения доступа к его содержимому: то есть надо перейти в домашнюю (корневую) папку, а затем использовать curl для получения установочного скрипта необходимой версии Node.js:

```
    sudo apt update
    cd ~
curl -sL https://deb.nodesource.com/setup_8.x -o
nodesource_setup.sh
```

В последней команде надо заменить 8.x на желаемую (последнюю) версию Node.js, а затем запустить скрипт:

```
sudo bash nodesource_setup.sh
```

Теперь PPA включен в конфигурацию, и локальный кэш пакетов обновился автоматически. После выполнения установочного скрипта от Nodesource, можно установить Node.js так же, как все прочие установочные пакеты в Ubuntu:

```
sudo apt install nodejs
```

Пакет nodejs содержит и nodejs и npm, поэтому нет никакой необходимости в дополнительной установке npm, можно проверить версии установленных пакетов, выполнив команды:

```
nodejs -v
npm -v
```

Затем надо установить в nodejs дополнительные модули, позволяющие организовать обмен данными с клиентом по протоколу websocket:

```
npm install node-static
npm install ws
```

Далее надо создать папку для проекта сервера: например, ~ws/nod, а в ней уже создать файл server.js.

Файл server.js

```
var http = require('http');
var Static = require('node-static');
var WebSocketServer = new require('ws');
// подключенные клиенты хранятся в массиве clients
var clients = {};
var price = 0;
var winner = 0;
var BargainStart = true;

// создается WebSocket-сервер на порту 8081
var webSocketServer = new WebSocketServer.Server({port: 8081});
webSocketServer.on('connection', function(ws) {

  var id = Math.floor((1 + Math.random()) *
0x10000).toString(16).substring(1);
  clients[id] = ws;
  console.log("новое соединение " + id);
  clients[id].send ("Ваш ID: " + id);

  ws.on('message', function(message) {
       console.log('получено сообщение ' + message);
       var SendToAll = true;
       if (!BargainStart) { //торги закончены, ставки не принимаются
            clients[id].send(" Сделка уже заключена. Торги закончены.");
       }
//функция Number() пытается конвертировать строку-аргумент в число
       if ((Number(message) != NaN) && BargainStart) {
            var new_price = Number(message);
            if (new_price > price) {
```

```
                price = new_price;
                winner = id;
                clients[id].send("You are the winner. You have to pay " +

price.toString());
                setTimeout(function() {
                      if (winner == id) {
                            //торги закончились:
                            BargainStart = false;
                            for(var key in clients) {
                            if (key != winner)   //для проигравших:
                            clients[key].send("All sold to " +
id.toString() +

" by price " + price.toString() +

". Thanks for participating.");
                            else     //для победителя:
                            clients[key].send("You have bought by price " +

price.toString() + " Congradulations!");
                             }
                }
                }, 50000);
           }
           else {
//если предложенная цена меньше текущей цены торгов, она отклоняется
                clients[id].send(
                                    "The bet isn't accepted. Your suggestion
must be more, than " +

price.toString());
//сообщение об этом предложении рассылать не надо
                SendToAll = false;
           }
     }

  if (SendToAll && BargainStart) {
    for(var key in clients) {
           if (key != winner)   //сообщение для всех, кроме текущего победи-
теля:
                clients[key].send(id.toString() + ": " + message +
                      ". Делайте ответные предложения.");
           else                              //сообщение для текущего победи-
теля
                clients[key].send(id.toString() + ": " + message);
    }
  }
  });     //конец обработки нового сообщения от клиента ws.on('message', ...

  ws.on('close', function() {
    console.log('соединение закрыто ' + id);
    delete clients[id];
  });

});   //конец обработки нового соединения webSocketServer.on('connection',
...

// обычный сервер (статика) на порту 8080
var fileServer = new Static.Server('.');
http.createServer(function (req, res) {
       fileServer.serve(req, res);
}).listen(8080);
```

```
console.log("Сервер запущен на портах 8080, 8081");
```

В клиентской части приложения, оставшейся на хостовой машине Windows, надо в файл Chat.html добавить изображение продаваемого товара, а в файле browser.js изменить строку подключения к серверу, где надо вписать ip-адрес виртуальной машины Ubuntu, узнав его, выполнив в виртуальной машине Ubuntu команду ifconfig. Например, строка подключения может иметь вид:

Файл browser.js

```
...
var socket = new WebSocket("ws://192.168.56.101:8081");
...
```

Чтобы протестировать работу приложения, надо сначала перейти с помощью cd в директорий, где находится server.js, и запустить его на виртуальной машине с Ubuntu, выполнив команду:

```
node server.js
```

Затем в браузере хостовой машины надо открыть клиентскую страницу Chat.html в трех окнах, можно в разных браузерах, и вводить поочередно в клиентских окнах в поле для отправляемых на сервер сообщений различные суммы для торгов (рис.4).

- В ответ сервер в случае корректного размера предлагаемой суммы рассылает всем клиентам новый размер ставки с просьбой повысить предложение, а клиенту-отправителю последнего принятого предложения посылает уведомление о том, что его ставка лидирует.
- Если сумма предложения ниже текущего значения суммы торгов, то клиенту, отправившему некорректное предложение цены, отправляется уведомление о непринятии заниженной ставки.
- Если в течение заданного таймаута в 50000 мс не приходит ни одного нового корректного предложения, то клиент, сделавший последнее принятое сервером предложение, становится победителем.
- Если сервер закончил торги, то далее он будет отправлять соответствующие сообщения об этом клиенту, если клиент попытается отправить на сервер новые сообщения о ставках (рис.4).

На основе полученных результатов можно сделать вывод о том, что реализованный в ОС Ubuntu веб-сервер с поддержкой обмена данными между клиентами по WebSocket протоколу, эффективно и производительно функционирует в приложении, которое работает в условиях реального времени и при интенсивном обмене текстовыми сообщениями. Связано это с асинхронностью и дуплексностью данного протокола.

При определенных навыках работы с WebSocket протоколом можно организовать быстрый клиент-серверный обмен не только текстовой, но и графической и, вообще говоря, любой информацией.

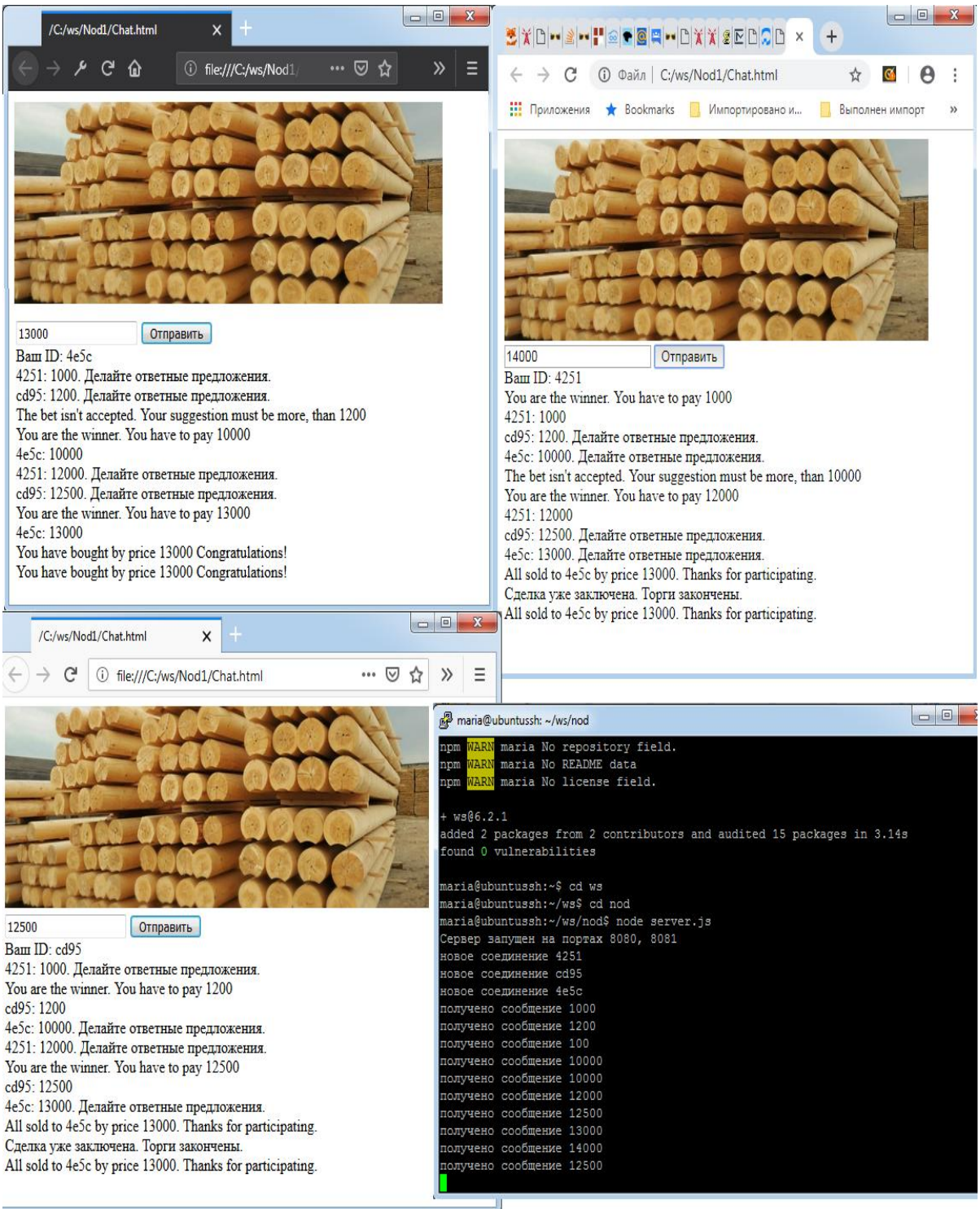


Рис.4. Вид окон клиентов и сервера после проведения электронных торгов

Р.М. Яковлев, кандидат ф.-м .наук., ведущий научный сотрудник
Санкт-Петербургское отделение Пагуошского движения
по нераспространению ядерного оружия
E-mail:Robertgood2019@gmail.com
И.А. Обухова, кандидат технических наук, доцент
Кафедра информационных систем и технологий
lobukhova@inbox.ru

СИТУАЦИОННЫЙ АНАЛИЗ ЭКОЛОГИЧЕСКОЙ И ЭНЕРГЕТИЧЕСКОЙ БЕЗОПАСНОСТИ В РОССИИ И В МИРЕ

Введение

Ядерная энергетика крупных масштабов, обеспечивающая подавляющую часть энергопотребления всех видов, явится величайшим благом для человечества и разрешит целый ряд острых проблем /А.П.Александров, 1977 г.[1]

Любой атомный реактор с размещённым в активной зоне ядерным топливом и образовавшимися в нём радиоактивными продуктами является носителем весьма высокого уровня опасности, что стало очевидным после двух атомных катастроф и ставших известными многочисленных аварий на различных объектах атомной энергетики.

Обеспеченность энергией во всём мире пытаются осуществлять более безопасными и дешёвыми энергоносителями. Только в России почти всю энергетику предполагается сделать до конца века ядерной. Согласно атомному стратегическому плану 2008 года мощности атомных станций России предполагалось увеличить к середине столетия до 300 ГВт. Но в связи с мировым кризисом и падением цен на нефть и газ финансирование было сокращено. Такое количество реакторов предполагается сделать к концу столетия. Но и по откорректированному плану 2014 года его варианту 2018 года развитие атомной энергетики является приоритетным и размеры бюджетного финансирования сотни триллионов рублей. Быстрые реакторы с плутониевым опасным топливом после доработки предполагается строить после 2030 года. Комбинат по переработке ОЯТ для получения МОКС-топлива построен под Красноярском. Сейчас осуществляется строительство реакторов на тепловых нейтронах, при этом строится много реакторов за рубежом в основном при финансировании их Россией при льготной нулевой ставке кредитного процента. Россия также по договорам осуществляет подготовку специалистов, поставляет ядерное топливо и возвращает облучённое топливо обратно в РФ для хранения и переработки с целью расширенного производства строящихся реакторов с МОКС-топливом. Атомный путь развития страны, является весьма опасным и не только для России.

После Чернобыльской катастрофы для общественности стали известны многие другие аварии в атомной промышленности и открылись весьма серьёзные проблемы атомной энергетики, которые делают её распространение весьма опасным. Но наибольшую опасность представляет не крупная авария на реакторе, а война с применением ядерного оружия, которого изготовлено сейчас в количестве достаточном для многократного уничтожения всей жизни на Земле. Чтобы не забыть этого, напоминаем.

1. Ядерное оружие и его опасность для жизни на Земле

Энергия, выделяющаяся при делении атомных ядер нейтронами, в миллионы раз превосходит ту, которая образуется в любых химических процессах. Так, во взрывных устройствах при делении всего 1кг ядер урана или плутония выделяется энергия 81700 ГДж, эквивалентная взрыву 18 тысяч тонн тринитротолуола. Такая энергия выделяется при сжигании в топке электростанции 2,5 тысяч тонн угля или 1,7 тысячи тонн нефти. Этой энергии достаточно для работы в течение года АЭС с мощностью 1 МВт (эл.).

В США министерством обороны США была подготовлена и издана американской комиссией по атомной энергии в 1962 году книга “THE EFFECTS OF NUCLEAR WEAPONS” [2], в которой приведены данные о действии атомного оружия, полученные в результате теоретических исследований и практических испытаний, а также на основании изучения последствий взрывов атомных бомб в Хиросиме и Нагасаки. В 1963 году перевод с английского появился в СССР в военном издательстве МИНИСТЕРСТВА ОБОРОНЫ [2]. После внимательного ознакомления с этой книгой любой разумный человек поймёт ужас ядерной войны. Например, при воздушном взрыве бомбы эквивалентной 10 Мегатоннам тринитротолуола на расстоянии 9,4 км от эпицентра разрушаются здания с бетонными стенами и на расстоянии 13,9 км все многоквартирные кирпичные с несущими стенами (табл.12.21 на стр.621). Велика опасность от образующихся при взрыве радиоактивных продуктов. На стр.451-453 приведена уникальная информация о радиоактивном заражении на Маршальских островах в результате испытательного взрыва «Браво», произведённого у атолла Бикини 1 марта 1954 года. Общая мощность взрыва соответствовала 15 Мегатоннам. В результате взрыва возникло сильное заражение района вдоль направления ветра протяженностью более 530 км и шириной около100 км. На полосе длиной 260 км и шириной 50 км вдоль направления ветра за 96 часов превышала 700 рентген и была смертельно опасна.

При другом взрыве неожиданно очень высокая доза была зарегистрирована на атолле «Ронгелап». На северо-западной оконечности атолла в 160 км от эпицентра взрыва суммарная доза за 96 часов после начала выпадения радиоактивных продуктов составила 3300 рентген. Мы привели всего два эпизода из книги, в которой 680 страниц весьма интересного текста с многими фотоснимками, схемами и диаграммами, чтобы читатель проникся ужасом войны ядерной. Ведь всего не самой большой мощности взрыв уничтожает большой город, например, Нью-Йорк или Петербург, а хвост радиоак-

тивный убивает окрестности. В ядерной войне подвергаются смертельной опасности не только объекты бомбардировок.

В 1970-х и 1980-х годах в США под руководством К.Сагала [3], и в СССР под руководством акад. Н.Н.Моисеева[4,5] были выполнены расчёты, которые показали, что помимо локального разрушения и высокого уровня радиоактивного загрязнения, ядерная война между США и СССР приведёт к гибели жизни на Земле. Мы приводим здесь довольно длинную цитату[6] , чтобы представлять насколько большую опасность заключает в себе ядерное оружие:

«Даже если в ядерной войне будет использовано всего лишь 100–150 мегатонн ядерного горючего, но оно будет распределено надлежащим образом по основным городам Европы, Азии и Америки, то эти города также сгорят в огненных вихрях. Это количество ядерного оружия немногим превышает то, которое носят на себе атомные подводные лодки. Такие числа дают представления о том, над краем какой бездны сейчас оказалось человечество". Результаты этих расчётов подтвердили гипотезу и дали первые количественные оценки эффекта ядерной зимы [6].

Самые большие запасы ядерного оружия в России и США. Они на два порядка превосходят ядерное оружие остальных стран. Сверхвысокая опасность большой ядерной войны была осознана политиками после разъяснения этой опасности учёными высокого ранга в двух странах. После чуть несостоявшейся войны во время Карибского кризиса наступил период улучшения отношений между странами, было приняты нужные решения о сокращении ядерных зарядов и о прекращении ядерных испытаний. Для других стран было принято решение о недопущении ядерного оружия, кроме 5 ядерных держав. Было принято затем решение о сокращении запасов оружейного плутония в США и СССР по 34 тонны в каждой стране.

Но после 2014 года ситуация изменилась. Программы по взаимному сокращению запасов ядерного оружия и сокращению запасов оружейного плутония не выполняются. США размещают тактическое ядерное оружие в странах Европы. В последние годы оно размещено и в Польше. Как 60 лет назад в США и НАТО разрабатываются планы **«разоружающего удара»** по России или КНР. Этот удар без применения ядерного оружия (ЯО) должны нанести крылатые ракеты наземного, морского и воздушного базирования, авиация и иные средства. Они должны уничтожить штабы, командные пункты, узлы связи, пусковые установки межбаллистических ракет (МБР), аэродромы стратегической авиации. Военно-морские силы США и НАТО, обладая многократным перевесом в кораблях, подводных лодках и авиации, должны уничтожить все российские подводные лодки - носители МБР.

Но, в любом случае уничтожить всё не удастся. Конец не только человечества , но и высокоорганизованной жизни наступит на планете не только от высокого радиоактивного загрязнения, но и поражающих всё живое факторов климатических, которые носят глобальный характер. «Ядерная зима» и «ядерная ночь», отсутствие света, пищи, пресной воды, отравление атмо-

сферы токсичными газами затронут всю планету в равной степени. Поэтому заявления некоторых высокопоставленных представителей военно-промышленного комплекса США, являются авантюрными высказываниями, которыми они пытаются обосновать необходимость получения огромных финансовых средств из бюджета на вооружение. Прямое ядерное столкновение между Россией и США вместе со странами НАТО исключены.

Но любой острый конфликт между государствами, создавшими у себя, несмотря на запреты ядерное оружие, может явиться запуском большой ядерной войны с втягиванием в конфликт других государств, обладающих им. В этой войне не может быть не только победителей и побежденных, но даже нейтральных. Причем роковым может оказаться даже и сравнительно небольшой ядерный конфликт. Значит, приобщение каждой новой страны к ядерным арсеналам увеличивает угрозу не только для ее потенциальных противников, но и для всего мира.

2. Мирная ядерная энергетика и ВИЭ в России и в других странах

Атомные реакторы, устройства по обогащению урана делящимся изотопом ураном-235, выделение плутония из облучённого топлива из ВПК напрямую перешли и используются в мирной атомной энергетике. В любой стране, где появляется атомный реактор, а также специалисты, обученные ядерной физике и радиохимии в тех странах, которые для них реакторы, открывается возможность при использовании давно опубликованных технологий получить сначала ядерную взрывчатку, и, имея её, без особых проблем перейти к атомной бомбе. Это и произошло сначала в Израиле, а затем в ЮАР, Индии, Пакистане, а затем уже в КНДР при помощи специалистов из тех стран, которые создали ядерное оружие. Только ЮАР при смене политического режима отказался от атомного оружия и ликвидировал все военные ядерные разработки.

Сейчас значительно расширяется география размещения атомных станций по всему миру и главная роль в этой опасной деятельности принадлежит России. Россия сейчас заключила контракты на сооружение 36 атомных реакторов за рубежом и строит реакторы в Турции, Иране, Египте, Индии, Вьетнаме, Бангладеш, Венгрии Белоруссии. Расширение географии увеличивает опасность создания ядерного оружия и его применения со стороны тех стран, руководители которых могут попытаться использовать для уничтожения ненавидимого противника.

При поставке МОКС-топлива в зарубежные реакторы извлечь из него несколько кг плутония для ядерного заряда не представит особых проблем. Взрывное устройство с высоким КПД создать сложно, но если из 10 кг вовремя взрыва разделится хотя бы 1 г, то это эквивалентно взрыву 2 тонн тринитротолуола, а распылившийся остальной нанесёт непоправимый вред городу, где произошёл взрыв. А теперь от вреда, который может принести МОКС топлива .

В соответствии со стратегическим планом развития атомной энергетики РФ 2008-го года, который мы привели в предыдущей статье, к середине

столетия предполагалось общее довести количество атомных реакторов мощностью 1 ГВт (эл.) в России до 300, причём к 2030 году их должно быть 100, а в 2020 году уже 53. Но из-за кризиса и отсутствия должного финансирования, грандиозный план, сорвался. В 2014 году он был откорректирован, а в 2018 году дополнительно изменён. Но всё равно, атомная энергетика считается приоритетной, и на её развитие выделены огромные деньги, в том числе и на разработку и изготовление опытных образцов реакторов на быстрых нейтронах с охлаждением расплавом свинца или синца-висмута. Научным руководителем этого направления, названного «Прорывом» является бывший глава атомного ведомства Е.О.Адамов, в своё время пытавшегося осуществить ввоз в Россию многих тонн зарубежного облучённого ядерного топлива и в конце своей министерской карьеры попавшего под суд с задержанием в Австрии. Направление работ в этом направлении является тупиковым, поскольку свинец растворяет любой металл (именно поэтому он и используется при сварке), а висмут и один из изотопов свинца при облучении нейтронами становится проникающим через любые щели радиоактивным полонием. Поэтому, казавшиеся сначала весьма перспективными свинцово-висмутовые реакторы, разработанные для подводных лодок после нескольких аварий были отменены.

Тем не менее, после 2030 года сооружение разрабатываемых (СВБР и БРЕСТ-300) реакторов на быстрых нейтронах с замыканием ядерного топливного цикла является приоритетным. Для нас сейчас может быть особенно интересным, что основные разработки атомных реакторов - на тепловых нейтронах ВВЭР-1000, РБМК и на быстрых нейтронах БН-300, БН-600 и БН-800 были уже разработаны и некоторые уже работали в 1977 году. Все, якобы новые установки, являются незначительно измененными прежними, а работающие реакторы РБМК и ВВЭР теми же самыми. Зная это, странными являются уверения топ-менеджеров атомного ведомства о новых и потому безопасных совершенно реакторах. Все реакторы обладают и должны обладать избыточной реактивностью, которая компенсируется поглощающими стержнями или наличием поглощающего нейтроны вещества в самой охлаждающей воде. Вывести его в надкритическое состояние, если очень захочется всегда можно. Или же нарушить съём тепловыделения, что приведёт к расплаву топлива. Но строительство наиболее опасных плутониевых реакторов на быстрых нейтронах сейчас, слава Богу, задерживается. Сейчас происходит интенсивное строительство новых АЭС с реакторами ВВЭР, названных ТОИ и АЭС-2006. В 2012–2020 годы объём бюджетных ассигнований (открытая часть) на реализацию госпрограммы составил 899723770,7 тыс. рублей (900 триллионов рублей).

Список строящихся станций в РФ приводим ниже [7]:

Дмитрий Медведев 01.08. 2016 своим распоряжением Председателя Правительства РФ № 1634-р утвердил план строительства восьми новых АЭС. Согласно распоряжению, до 2030 года в России будут построены восемь крупных АЭС.

1. Кольская АЭС-2, 1 ВВЭР-600. Итого 675 МВт.
2. Центральная АЭС, 2 ВВЭР-ТОИ, по 1255 МВт. Итого 2510 МВт.
3. Смоленская АЭС-2, 2 ВВЭР-ТОИ, по 1255 МВт. Итого 2510 МВт.
4. Нижегородская АЭС, 2 ВВЭР-ТОИ, по 1255 МВт. Итого 2510 МВт.
5. Татарская АЭС, 1 ВВЭР-ТОИ, по 1255 МВт. Итого 1255 МВт.
6. Белоярская АЭС, 1 БН-1200. Итого 1200 МВт.
7. Южноуральская АЭС, 1 БН-1200. Итого 1200 МВт.

Северская АЭС, 1 БРЕСТ-300. Итого 300 М. Кроме того предполагается в РФ соорудить новые блоки на уже построенных АЭС. Всего за период с 2016 по 2030 гг. будет построено 22 энергоблока и 25,36 ГВт мощностей. За тот же период будет закрыт 21 энергоблок мощностью 13,042 ГВт. Общая мощность добавится на 46%. И за это же время будет построено 36 блоков за рубежом. Утверждается, что к жителям России и стран, где АЭС сооружаются не за их счёт, на долгий срок (теперь он 60 лет) придёт дешёвая, безопасная и самая чистая энергия. А это является неправдой**.** Безопасной её можно было называть до 1986 года только при полном сокрытии сверхсекретной информации об огромном количестве аварий и жертв в атомном ведомстве. А сейчас, когда открыты полные списки аварий и инцидентов/списки из ВИКИ/, том числе и в ВПК, и каждый может с ними ознакомиться, никто не может называть атомную энергетику безопасной. Безусловно, многим авторам, академикам и организаторам промышленности, да и большей части других уважаемых авторов были известны во всяком случае такие аварии, как взрыв одной из ёмкостей с радиоактивными отходами в закрытом г.Озёрске (Кыштым) и об опасной аварии на 1 блоке РБМК на Ленинградской АЭС и многих других с человеческими жертвами. Но, то было во время великого противостояния систем. А к 1977 году в США уже было почти 100 энергетических реакторов, а в СССР всего 15. Надо было их догонять. Ну, и была, кроме всего, уверенность, что только атомное ядро при делении спасёт человечество энергией при истощении углеводородных ресурсов. Так было в 1977 году. А сейчас? После Чернобыля и Фукусимы, при ставших известными многих других инцидентах и авариях? Заявляют: спасём человечество от потепления и всемирного потопа. Но это тоже враньё, созданное нобелевским жуком Гором, чтобы грести денежки с легковерных. Интересно, что США, создав этот миф сами в этой игре не участвуют, также как и Китай, продолжая сжигать самые большие количество угля и углеводородов. Мы не дураки, заявил Трамп, на предложение подписать на эту тему в Париже, очередное соглашение. А мы? А мы, т.е правители наши вместе с атомщиками говорят - мы, Спасители, не будем сжигать много газа и нефти в стране нашей, пускай добытое нами жгут в Германии и Китае. Будем строить новые АЭС, которые мы назовём безопасными и экологически самыми чистыми. О безопасности рассказали. Теперь о чистоте. Заявлять это надо уже не для идиотов, а совсем для дураков. Ведь даже если исключить глобальное загрязнение Сибири после Кыштыма при взрыве всего одной из сотни ёмко-

стей и не хотеть знать о Теченском каскаде, то ведь переработка на отстроенном в Железногорске не только своего, но и зарубежного ОЯТ со всех построенных Россией блоков является сверхгрязным делом.

Поскольку расходы производства электроэнергии по сравнению со всеми другими источниками, особенно если учесть кроме самого дорого строительства реактора ещё расходы на утилизацию ОЯТ и последующее захоронение радиоактивных отходов и высокой стоимости при закрытии станции. На этом мы останавливались в предыдущей работе [9]. Понимая опасность данного пути, нами давно предлагалась значительно более безопасная атомная энергетика [8-16], без накопления плутония и Чернобыля, но ввиду неполучения финансирования пока не было развито. Компьютерное моделирование данного типа установок было описано в работе [11] в данном сборнике в 2017 г.

Посмотрим, а какая ситуация сложилась во всем остальном мире, особенно после необыкновенно быстрого удешевления таких возобновляемых источников, как солнце и ветер.

Сопоставление себестоимости получения энергии из разных источников представлено на рис. 1.

А сейчас покажем, как менялась стоимость производства электроэнергии для разных носителей/ Источник WNISR-2018 [17].

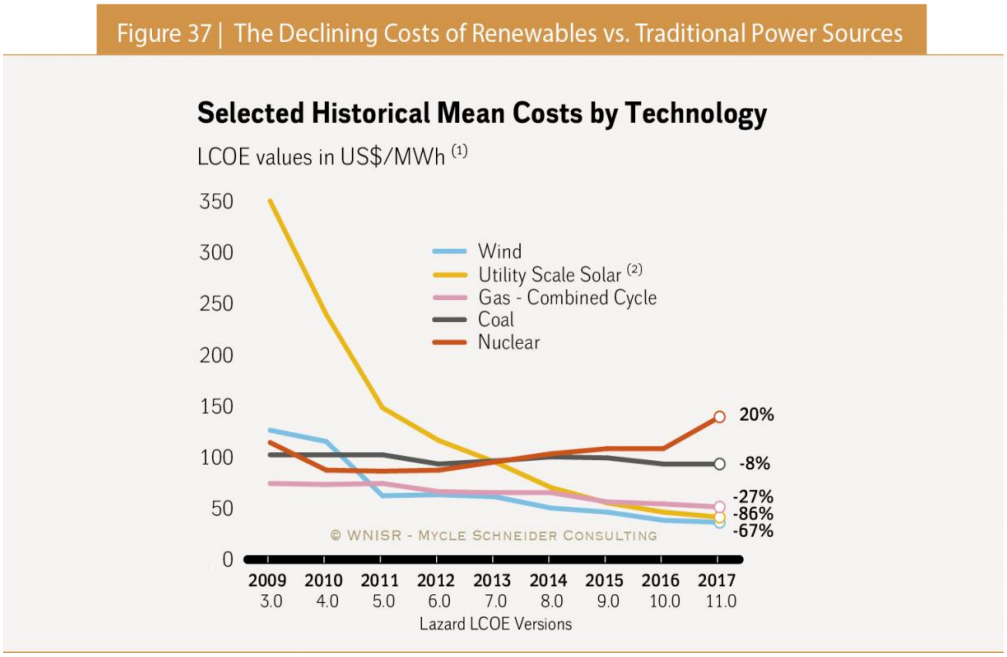


Рис.1. Себестоимость производства электроэнергии для различных источников

Стоимость единичной мощности на возобновляемых источниках уже давно значительно меньше, чем ядерная. С 2011 года себестоимость получаемой электроэнергии для ветра, а с 2013 и для Солнца, даже с учётом непостоянства их работы оказывается ниже атомной, что показано на этом рисунке.

Почти во всём мире с учётом указанных выше проблем (главная из них - высокая стоимость) атомная энергетика постепенно свёртывается. Во всём мир инвестиции в неё уже давно значительно ниже, чем в экологически чистые возобновляемые источники энергии, что видно на рис. 2 , взятом нами из того же обзора Шнайдера [17]. Ниже представлены эти инвестиции:

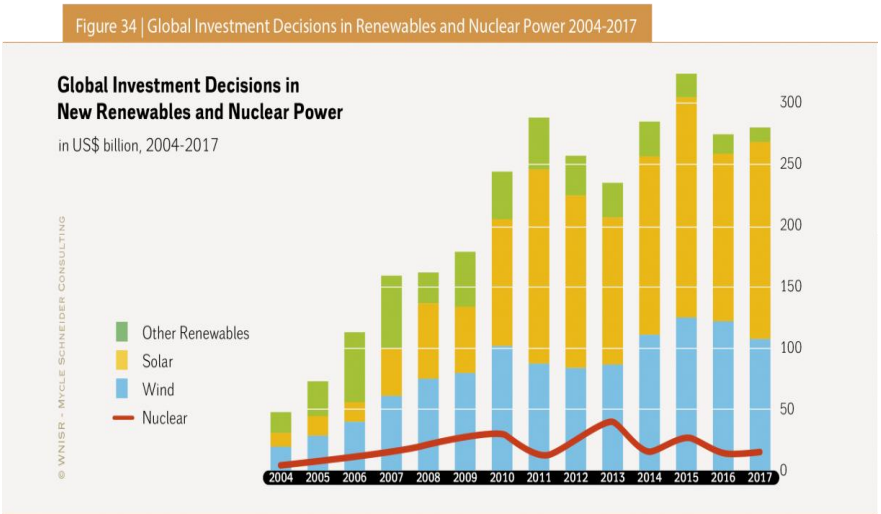


Рис.2. Инвестиции в мире в различные отрасли энергетики (в млрд. долл.)

Поскольку инвесторы намерены вкладываться в те направления, которые могут обеспечить получение и отдачу от вложенных денег при наименьших затратах, желательно через короткий промежуток времени, то естественно, в атомную энергетику с ее высокой себестоимостью, желание вкладываться отсутствует. Здесь существенным является длительное время строительства, которое для АЭС значительно больше, чем для сооружения электростанции на газе, которое составляет полгода – год и еще скорее при сооружении энергоблоков, использующих энергию Солнца. Время строительства атомных станций еще может очень затянуться после возможных крупных аварий на любой из них, последствия которых приостанавливают или отменяют вообще уже начатое строительство.

Так было после сравнительно небольшой аварии на АЭС в США, где было прекращено уже начатое строительство 28 реакторов. Правительство Картера скомпенсировало, правда, спонсорам затраченные средства, но такие компенсации в других странах вряд ли возможны. Поэтому риск потерять свои деньги, вложенные в строительство АЭС весьма велик. После Чернобыльской катастрофы во многих странах мира было прекращено строительство многих АЭС, включая Россию. В результате развернутого антиядерного движения было остановлено строительство АЭС в г. Горьком, г. Ростове, в Крыму, Башкирии, Татарстане.

Атомные электростанции сейчас наиболее интенсивно сейчас строят Китай и Россия, причём не только в своих странах. Китай не обладает большими запасами нефти и газа, уголь слишком загрязняет биосферу, а гидроэнергия не только на больших реках, но и на малых почти исчерпана . Поэтому при осуществлении программы интенсивного экономического роста с

обеспечением в максимально короткие сроки более высокого уровня жизни для своих граждан, Китай вынужден использовать опасную атомную энергию. Интересно, что стоимость строительства китайских АЭС и сроки строительства в два раза меньше российских. Но приоритетом и для Китая является строительство электростанций с использованием энергии Солнца и ветра, инвестиции в них значительно выше, чем в атомную. Производство электроэнергии с использованием ветра превышает с 2011 года производство АЭС, что видно из рис. 3:

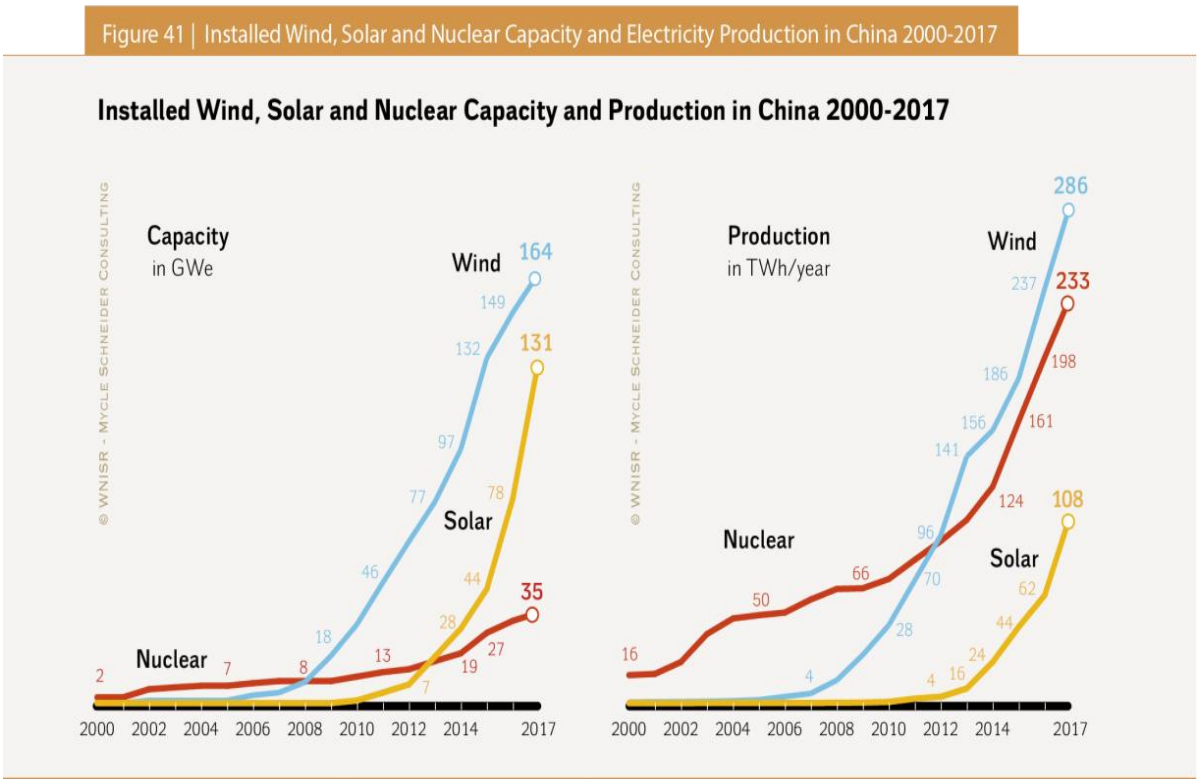


Рис. 3. Атомная энергия и возобновляемые источники энергии в Китае

Строительство возобновляемых источников энергии является приоритетным сейчас во всём мире, кроме России. Приоритетом для неё является сверх опасная и дорогая атомная энергетика. Может быть, в России. как уже почти во всех странах, возьмутся по-настоящему использовать самые большие на Земле ресурсы ВОЗОБНОВЛЯЕМОЙ ЭНЕРГИИ? Резервы России самые большие и по энергии ветра, приливной энергии океана, геотермальной. Благоприятные перспективы использования солнечной энергии есть на Северном Кавказе, в Нижнем Поволжье и в Забайкалье, т. е. в районах, где в году много ясных солнечных дней. Самые ветряные районы расположены вдоль береговой линии Северного Ледовитого океана и в Калининградской области. На Кольском полуострове, в Кислой губе есть небольшая электростанция, работающая на энергии морских приливов и отливов. Большие возможности для строительства такого рода электрических станций имеются на побережье Охотского моря, где приливы достигают 18 метров. Источники геотермальной энергии есть в сейсмически активных зонах Земли. Это Камчатка (в Долине гейзеров работает небольшая электростанция) и Курильские

острова. Но эти возможности не используются, поскольку всё внимание и средства направлены на атомную энергетику, альтернативы которой как неправильно утверждают руководители атомного ведомства, а за ними и государства -НЕТ. Как мы отметили уже в нашей предыдущей публикации [13] уровень ВИЭ в России в последние минимум 10 лет топчется на уровне 1 %, в том числе 0,6% – биомасса, 0,3% – малые ГЭС, и всего только 0,1% – это ветряная, солнечная электроэнергетика и геотермальные источники. Ситуацию необходимо менять!

Библиографический список

1. Акад. Александров А.П., Боголюбов Н.Н., Бочвар А.А., Велихов Е.П., Долежаль Н.А. и др. Атомная наука и техника в СССР М. Атомиздат, 1977, 273с.

2. Действие ядерного оружия. Перев.с англ. Изд.Мин.обор. СССР.М., 1963.

3. Carl Sagan "Nuclear war and climatic catastrophe: some policy implications.

4. Моисеев Н.Н. Экология человечества глазами математика. М., 1988. С.73-81.

5. Моисеев Н.Н., Александров В.В, Тарко А.М. Человек и биосфера. М., 1985.

6. Александров В.В., Стенчиков Г.Л. Моделирование климатических последствий ядерной войны, М., 1983.

7. Inet: geoenergetics.ru/2016/08/12/aes-v-rossii-do-2030-goda/.

8. Яковлев Р.М., Обухова И.А. Оценка перспектив применения альтернативных моделей реакторов в ядерной энергетике. Сб.Инф.сист.и технологии. Вып.10, Ч.2, СПб, СПбГЛТУ,2018, с.3.-13.

9. Яковлев Р.М., Обухова И.А. Ситуация с нераспространением ядерного оружия и проблемы развития атомной энергетики. Сб. СПбГЛТУ «Инф.сист.и техн.:теория и практ.»Вып.11, СПб, СПбГЛТУ,2019,с.153-169.

10. Яковлев Р.М., Обухова И.А. На пути к безопасной атомной энергетике (научная статья) ВАК, РИНЦ, спец.25.00.00 .Биосфера, СПб, Фонд 21 век. Т.9, N2, 2017,с.123-135.

11. Яковлев Р.М., Обухова И.А. Компьютерное моделирование жидкосолевого уран-ториего реактора Сб. СПбГЛТУ «Инф.сист.и техн.:теория и практ.»,вып.9, 2017.с.67-75.

12. Яковлев Р.М., Обухова И.А. О нераспространении ядерного оружия и проблемах развития атомной энергетики России. Эк.вестник России, N1, 2020.с.32-36.

13. Кузякин Ю.И. , Яковлев Р.М. . Патент №57040 Ядерная реакторная установка с топливом-теплоносителем в виде расплавов солей фторидов, 27.09.2006.

14. Кузякин Ю.И. , Яковлев Р.М.. Патент №64424 Моноблочная ядер-

ная реакторная установка с жидкометаллическим топливом-теплоносителем, 25.06.2007 г.

15.Яковлев Р.М., акад. РАН Данилевич Я.Б., Игнатьев М.Б., Суглобов Д.Н. Атомная энергетика без плутония и Чернобыля. Мир и Согласие, №2(35), М. 2008.

16.Суглобов Д.Н, Яковлев Р.М., акад. РАН Мясоедов Б.Ф. Торий-урановый топливный цикл для тепло- и электроэнергетики. Радиохимия, 2007;49(5):385-392.

17. Schneider M., Фроггатт A., Hazemann J.и др. World Nuclear Industry Status Report 2018. Доклад о состоянии мировой ядерной промышленности 2018 (HTML).jeudi 6 septembre 2018. WNISR-18

18.Адамов Е.О., Балашов Л.А., Ганев И.Х., Зродников И.В., Кузнецов А.К., Лопаткин А.В., Мастепанов А.М., Орлов А.В., Рачков А.В., Смирнов А.С., Солонин М.И., Ужанова В.В., Черноплеков Н.А., Шаталов Г.Г. Белая книга ядерной энергетики. М.: Изд-во ГУП НИКИЭТ; 2001.

19. Inet:REN21 2016. Renewables Global Status Report 2016 (pdf).

20.Inet:REN21 2018. Renewables Global Status Report 2018 (pdf).

Оглавление

Научное издание

О т в . р е д а к т о р
Заяц Анатолий Моисеевич

ИНФОРМАЦИОННЫЕ СИСТЕМЫ И ТЕХНОЛОГИИ: ТЕОРИЯ И ПРАКТИКА

Сборник научных трудов

В ы п у с к 1 2

В авторской редакции с готового оригинал-макета

Подписано в печать 26.03.20.
Усл.-печ. л. 8,75. Заказ № 26. С 209.

Санкт-Петербургский государственный лесотехнический университет
Издательско-полиграфический отдел СПбГЛТУ
194021, Санкт-Петербург, Институтский пер., 3