eISSN 2311-2468
Том 9, № 12. 2021
Vol. 9, no. 12. 2021

электронное периодическое издание
для студентов и аспирантов

Огарёв-онлайн
Ogarev-online

https://journal.mrsu.ru

ПОЛЯНИЧКО К. С.

ОБЗОР И ТЕСТИРОВАНИЕ СИСТЕМ РАСПОЗНАВАНИЯ РЕЧИ

Аннотация. В данной статье описываются и сравниваются характеристики, таких наиболее распространённых систем по распознаванию речи, как CMU Sphinx, Google Speech, HTK, Julius, iAtros, RWTH ASR. Сравнение происходит по следующим характеристикам: поддерживаемые языки распознавания, поддерживаемые операционные системы, наличие документации, открытость кода, наличие или отсутствие офлайн режима, показатели WER, WRR, SF. Также описан процесс и результаты тестирования наилучших по различным показателям систем.

Ключевые слова: распознавание речи, CMU Sphinx, Google Speech, HTK, Julius, iAtros, RWTH ASR, точность распознавания, скорость обработки речи.

POLYANICHKO K. S.

OVERVIEW AND TESTING OF SPEECH RECOGNITION SYSTEMS

Abstract. This article describes and compares the characteristics of such most common speech recognition systems as CMU Sphinx, Google Speech, HTK, Julius, iAtros, RWTH ASR. The comparison is based on the following important characteristics: supported recognition languages, supported operating systems, availability of documentation, open source code, presence or absence of offline mode, WER, WRR, SF indicators. It also describes the process and results of testing the best systems for various indicators.

Keywords: speech recognition, CMU Sphinx, Google Speech, HTK, Julius, iAtros, RWTH ASR, recognition accuracy, speech processing speed.

Системы распознавания речи чаще всего применяются для воссоздания более удобного взаимодействия человека с техникой, например, для голосового управления. На сегодняшний день распознавание речевых сигналов обладает очень обширным спектром применения, как пример, голосовые помощники, которые могут использоваться в современных телефонах, автомобилях и т.д. [1]. В связи с актуальностью и широким распространением систем распознавания, была поставлена задача определить наиболее эффективную программу по распознаванию речи из выбранных с целью дальнейшего применения в робототехнической лабораторной платформе с техническим зрением, предназначенной для обучения робототехнике, а также участия в робототехнических соревнованиях.

Основываясь на некоторые данные из работы Беленко М. В. и Балакшина П. В. «Сравнительный анализ систем распознавания речи с открытым кодом» были рассмотрены существующие системы распознавания речи, такие как CMU Sphinx, HTK, iAtros, Julius, Kaldi и RWTH ASR, Google. Выбор именно этих систем обусловлен частотой упоминания в научно-исследовательских журналах и популярности среди индивидуальных разработчиков программного обеспечения [2]. В отличии от выше указанной работы рассматривались программы не только с открытым кодом, а также подробно были описаны этапы тестирования программ и результаты испытаний.

При анализе систем были рассмотрены следующие показатели: поддерживаемые языки распознавания, поддерживаемые операционные системы, наличие документации, открытость кода, наличие или отсутствие офлайн режима, показатели WER, WRR, SF.

WER – это точность распознавания, которая является показателем качества и определяется как процент неправильно распознанных слов (Word Error Rate), а показатель WRR (Word Recognition Rate) наоборот отражает процент правильно распознанных слов. Второй важный критерий распознавания речи, который был рассмотрен – скорость обработки речи, выраженный в показатели скорости SF (Speed Factor) [3].

В таблице 1 представлены описанные характеристики различных систем по распознаванию речи [2].

Таблица 1

Обзор систем по распознаванию речи

Система	**Поддерживаемые языки распознавания**	**Поддерживаемые ОС**	**Наличие документации**	**Открытость кода**	**Офлайн режим**	**Показатели: WER(%), WRR(%), SF**
CMU Sphinx (pocketsphinx)	Множество языков, в том числе экзотические	Linux, Mac OS, Windows, Android	Подробная онлайн документация	+	+	21.4/22.7, 78.6/77.3, 0.5/1
HTK	Английский	Linux, Solaris, HPUX, IRIX, Mac OS, FreeBSD, Windows	HTK Book – исчерпывающая информация	+	-	19,8, 80,2, 1.4

Продолжение таблицы 1

Система	**Поддерживаемые языки распознавания**	**Поддерживаемые ОС**	**Наличие документации**	**Открытость кода**	**Офлайн режим**	**Показатели: WER(%), WRR(%), SF**
Julius	Японский, Английский	Linux, Windows, FreeBSD, Mac OS	Julius Book – аналогично HTK Book	+	-	16.1, 83.9, 2.1
iAtros	Английский, Испанский	Linux	Отсутствие документации	+	-	16.1, 83.9, 2.1
RWTH ASR	Английский	Linux, Mac OS	Неподробная документация	+	-	15.5, 84.5, 3.8
Google	Множество языков	Linux, Mac OS, Windows, Android	Онлайн документация	-	-	4.9, 95.1, 0.5

Из перечисленных вариантов для тестирования было решено выбрать системы SMU Sphinx и Google. Данное решение было сделано в связи с тем, что CMU Sphinx единственная из перечисленных может работать в офлайн режиме, а также, не смотря на не очень высокую точность распознавания, обладает одной из лучших скоростей распознавания из всех указанных, большим количеством поддерживаемых языков, распространяется под лицензией BSD, которая разрешает встраивание в коммерческие проекты, а также возможностью внесения изменений в открытый код. Выбор второй системы был обусловлен наилучшими показателями WER и WRR. И одним из важнейших критериев была поддерживаемая ОС, это ОС Windows.

Далее было проведено тестирование систем, которое осуществлялось с применением языка python и выбранных ранее в обзоре систем: PocketSphinx API и Google Speech API и библиотек SpeechRecognition и Pyaudio.

Первый тест по распознаванию текста из аудио файла был сделан с использованием Google Speech API и библиотек SpeechRecognition.

Сам аудио файл содержал следующий текст: «Who’s the low to the left shoulder take the winding path reach the lake no closely the size of the gas tank wipe degrees off is dirty face men to call

before you go out the Redwood valley strain and hung limp the stray cat gave birth to kittens the young girl gave no clear response the meal was cooked before the bell rang what Joy there is a living». Данный аудио файл взят с интернет-ресурса и текст, записанный в нем произносится диктором, для которого английский язык является родным, запись сделана в хорошем качестве без посторонних шумов.

Настройка и тестирование первой системы осуществлялась по следующему алгоритму:

1. Установка библиотеки SpeechRecognition в командной строке компьютера.
2. Вызов python в командной строке компьютера.
3. Импорт только, что загруженной библиотеки.
4. Создание экземпляра класса Recognizer.
5. Создание объекта класса AudioFile модуля speech_recognition.
6. Преобразование аудиофайла в объект AudioData методом record() класса Recognizer. Передача объекта AudioFile методу record().
7. Проверка типа переменной.
8. Передача объекта audio_content методу recognize_google() объекта класса Recognizer(), и аудиофайл будет преобразован в текст. Полный текст программы представлен на рисунке 1.

main.py

```
1
2   import speech_recognition as speech_recog
3   recog = speech_recog.Recognizer()
4   sample_audio = speech_recog.AudioFile('D:/001.wav')
5   with sample_audio as audio_file:
6   audio_content = recog.record(audio_file)
7   type(audio_content)
8   recog.recognize_google(audio_content)
```

Рис. 1. Программа по настройке и запуску распознавания речи из аудио файла с помощью Google Speech API.

Пример корректной работы программы с распознаванием из аудио файла с применением Google Speech API с полученным текстом представлен на рисунке 2.

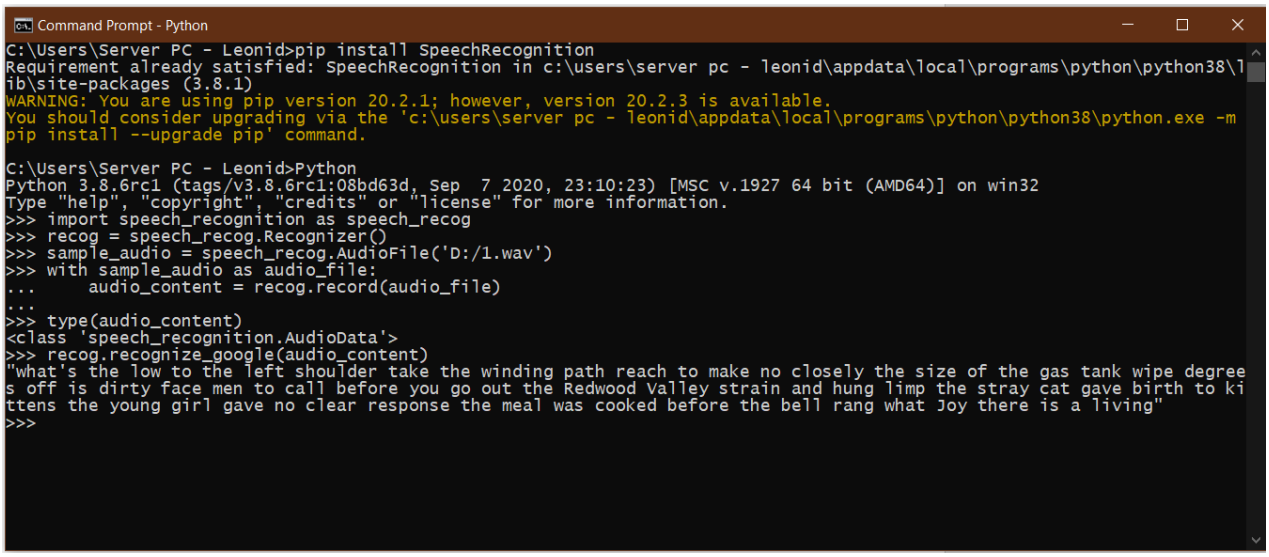


Рис. 2. Пример корректной работы программы с распознаванием из аудио файла с применением Google Speech API.

Второй тест по распознаванию текста из аудио файла был сделан с использованием PocketSphinx API и библиотек SpeechRecognition. Настройка и тестирование первой системы осуществлялась по прежнему алгоритму из приложения Б, только необходимо заменить метод

recognize_google() на метод recognize_sphinx(). PocketSphinx API может работать без подключения к сети интернет.

Пример корректной работы программы с полученным текстом представлен на рисунке 3.

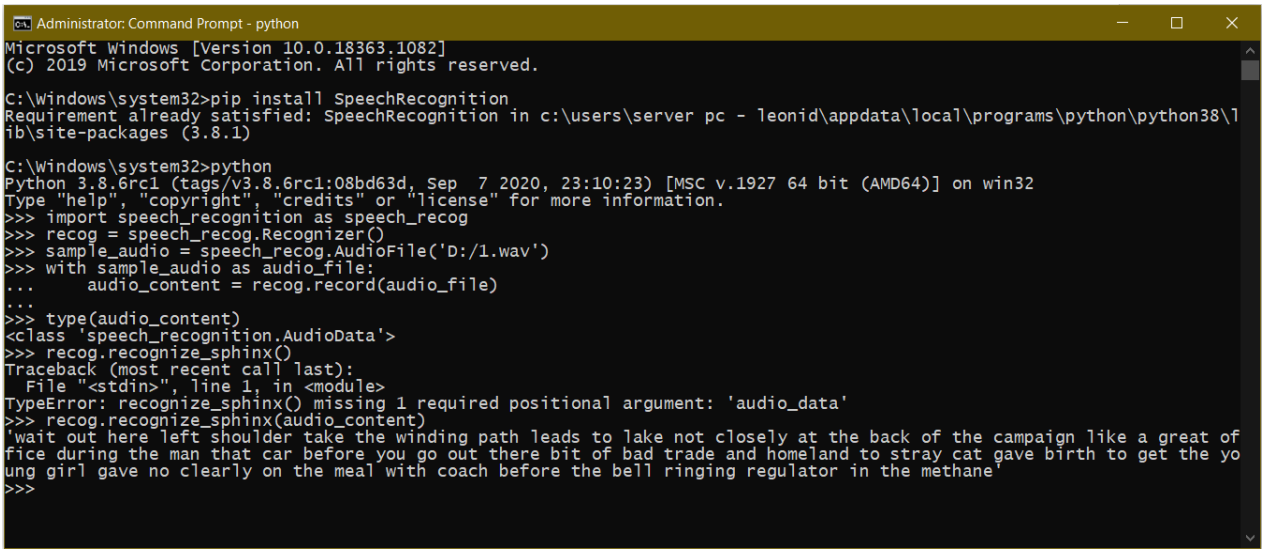


Рис. 3. Пример корректной работы программы с распознаванием из аудио файла с применением PocketSphinx API.

Результаты проверки программ показали следующие результаты:

1.Система по распознаванию Google из записанных в аудио файле 71 слова правильно распознала 68 слов.

2. Система по распознаванию PocketSphinx из записанных в аудио файле 71 слова правильно распознала 28 слов.

Для третьего теста, то есть распознавания речи с живого микрофона с использованием Google Speech API и PocketSphinx API, а также библиотеки Pyaudio была установлена VisualStudio – это линейка продуктов компании Microsoft, включающих интегрированную среду разработки программного обеспечения и ряд других инструментальных средств [4]. Распознавание речи с живого микрофона и перевод ее в текст осуществлялся последующему алгоритму программы, показанном на рисунке 4.

main.py

```
import speech_recognition as speech_recog
import pyaudio
import pocketsphinx

recog = speech_recog.Recognizer()
mic = speech_recog.Microphone()
with mic as audio_file:
    print("Speak Please")

    recog.adjust_for_ambient_noise(audio_file)
    audio = recog.listen(audio_file)

    print("Converting Speech to Text...")
    print("You said with google: " + recog.recognize_google(audio))
    print("You said with sphinx: " + recog.recognize_sphinx(audio))
```

Рис. 4. Программа по настройке и запуску распознавания речи с микрофона с помощью Google и PocketSphinx.

Программа задействует сразу две библиотеки Google Speech API и PocketSphinx API и выводит результаты в двух разных строках. Пример корректной работы программы с полученным текстом, распознанным с микрофона представлен на рисунке 5, где произносилось: «What’s going on». Необходимо отметить, что текст произносился в условиях отсутствия посторонних шумов и с использованием довольно качественного конденсаторного микрофона.

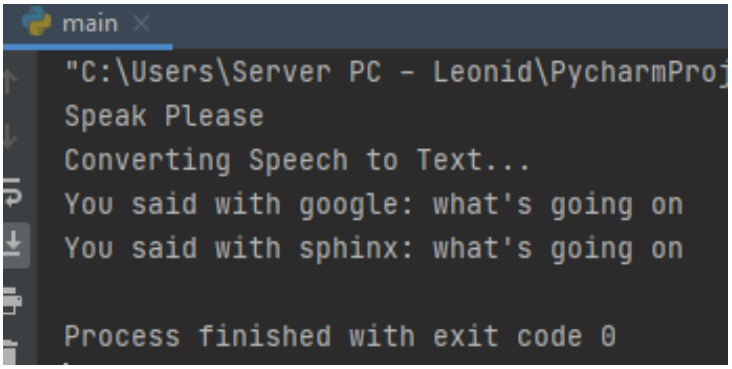


Рис. 5. Пример работы программы с распознаванием короткой фразы с микрофона.

Следующий пример корректной работы программы с полученным текстом, распознанным с микрофона представлен на рисунке 6, где произносилось: «Who's the low to the left shoulder take the winding path reach the lake no closely the size of the gas». Также следует отметить, что текст произносился в условиях отсутствия посторонних шумов, с использованием довольно качественного конденсаторного микрофона и со скоростью примерно 45 слов в минуту и не носителем английского языка.

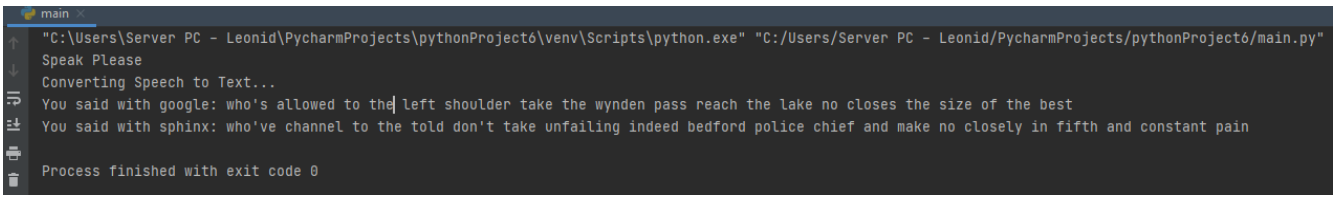


Рис. 6. Пример работы программы с распознаванием длинной фразы с микрофона.

По результату третьего теста было выявлено, что Google практически всегда правильно распознает короткие фразы, в то время, как PocketSphinx делает это с переменным успехом, для того, чтобы он корректно распознал фразу требуется очень четкого и правильного произношения. В случае с длинными фразами Google из произнесенных 23 слов корректно распознал 16 слов, а PocketSphinx всего 6 слов. Также следует отметить, что на качество распознавания влияет множество факторов и в связи с этим достаточно сложно выявить определенную зависимость.

Делая выводы, Google Speech API по качеству распознавания показал намного лучшие результаты, что и подтверждается параметрами, указанными в таблице, но в некоторых случаях и проектах, где может быть важна автономность и возможность применения в коммерческих

проектах, также может быть использован PocketSphinx API, который к тому же имеет возможность усовершенствования за счёт открытости кода.

СПИСОК ЛИТЕРАТУРЫ

1. Бабаринов С. Л., Будникова М. А. О распознавании речи // Научные ведомости Белгородского государственного университета. Серия История. Политология. Экономика. Информатика. – 2014. – № 21(192), выпуск 32/1. - С. 182-185.
2. Беленко М. В., Балакшин П. В. Сравнительный анализ систем распознавания речи с открытым кодом // Международный научно-исследовательский журнал. – 2017. – № 4(58), Часть 4. – С. 13–18 [Электронный ресурс]. – Режим доступа: https://researchjournal.org/technical/sravnitelnyj-analiz-sistem-raspoznavaniyarechi-s-otkrytym-kodom/ (дата обращения: 14.01.2021).
3. Карпов А. А., Кипяткова И. С. Методология оценивания работы систем автоматического распознавания речи // Известия высших учебных заведений. Приборостроение. – 2012. – Т. 55, №. 11. – С. 38-43.
4. Visual Studio [Электронный ресурс] // Официальная страница Visual Studio компании Microsoft. – Режим доступа: https://visualstudio.microsoft.com/ (дата обращения: 14.01.2021).

ЕГОРОВА Д. К., ГАРИН М. А., САЙФЕТДИНОВ С. Ф.

СОЗДАНИЕ WORKFLOW АНАЛИТИЧЕСКОЙ ПЛАТФОРМЫ KNIME ДЛЯ АНАЛИЗА ДАННЫХ НА ПРИМЕРЕ ВАКАНСИЙ САЙТА HEADHUNTER

Аннотация. В статье приведено создание рабочего процесса аналитической платформы KNIME Analytics Platform. Кластеризация вакансий выполнена методами k-means и density-based.

Ключевые слова: KNIME, workflow, функциональный узел, кластеризация, k-means, density-based алгоритм.

EGOROVA D. K., GARIN M. A., SAIFETDINOV S. F.

CREATION OF WORKFLOW FOR ANALYTICAL PLATFORM KNIME FOR DATA ANALYSIS ON THE EXAMPLE OF VACANCIES ON THE SITE HEADHUNTER

Abstract. The process of creating a KNIME Analytics Platform workflow was explored in the article. Clustering of vacancies was performed by k-means and density-based methods.

Keywords: KNIME, workflow, functional node, clustering, k-means, density-based algorithm.

В настоящее время существует множество инструментальных средств машинного обучения позволяющих осуществлять анализ данных. В работе приведено создание рабочего процесса workflow аналитической платформы KNIME Analytics Platform [1] на примере данных, предоставляемых HeadHunter – одним из самых крупных сайтов по поиску работы и сотрудников в мире [2]. KNIME Analytics Platform – это Java-кроссплатформенное приложение с открытым исходным кодом для анализа данных, объединяющее различные компоненты машинного обучения и интеллектуального анализа посредством модульной конвейерной обработки данных «Lego of Analytics». Выбор HeadHunter в качестве источника данных обуславливается наличием открытого API [3].

Решим задачу кластеризации данных по вакансии «Программист» в населенном пункте «Саранск» по состоянию на 23 мая 2021 года, которая покажет разбиение данных по величине предлагаемой заработной платы. Для этого создадим рабочий процесс (Workflow), считаем данные, воспользовавшись соответствующим узлом (Node) GET Request (Tools & Services→REST Web Services→GET Request), перенесем его из репозитория узлов на рабочее пространство. В конфигурации данного узла размещаем следующий код на языке python, осуществляющий запрос.

Фрагмент кода:

```
https://api.hh.ru/vacancies?text=программист&area=63&per_page=100&only_with_salary=true
```

Для извлечения вакансий воспользуемся узлом JSON Path (Structured Data → JSON→ JSON Path) и соединяем его с GET Request (рис. 1).

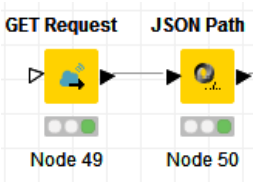


Рис. 1. Узлы, осуществляющие извлечение данных.

Данные были получены в виде таблицы (Рисунок 2), в том числе содержащей поля «from» и «to», означающие границы измерения зарплаты.

Table "default" - Rows: 1 | Spec - Columns: 7 | Properties | Flow Variables

Row ID	Status	Content type	body	from	to
Row0	200	application/json; charset=UTF-8	{ "items": [{ "id": "44770732", "premium": false,	[80000,80000,30000,...	[100000,100000,50000,...

Рис. 2. Таблица извлеченных данных.

Данные в таком виде использовать для кластеризации нельзя. Необходимо выполнить элементы так называемого разведочного анализа данных, а именно осуществить разгруппировку данных по столбцам, фильтрацию этих столбцов, вычисление основных статистик, заполнение пропущенных значений в полях с размером заработной платы (например, медианным значением), так как не все работодатели указали на сайте значения полей «from» и «to» и, наконец, их нормализацию. Для этого были использованы узлы Ungroup (Manipulation→Row→Transform→Ungroup), Column Filter (Manipulation→Colum→ Filter→ Column Filter) и Missing Value (Manipulation→Colum→ Transform→ Missing Value). Узел Box Plot (Views→JavaScript→Box Plot) позволяет построить «ящичковые диаграммы» данных полей «from» и «to». На диаграммах отображаются статистические параметры: минимум, нижний квартиль, медиана, верхний квартиль (соответственно, 25-й, 75-й 50-й процентили) и максимум. Эти параметры называются надежными, поскольку они нечувствительны к экстремальным выбросам. На рисунке 3 представлены данные до нормализации и заполнения пропущенных значений, из которого видно, что данные параметра «to» имеют несколько «выбросов» (точка) и экстремальное значение (крест).

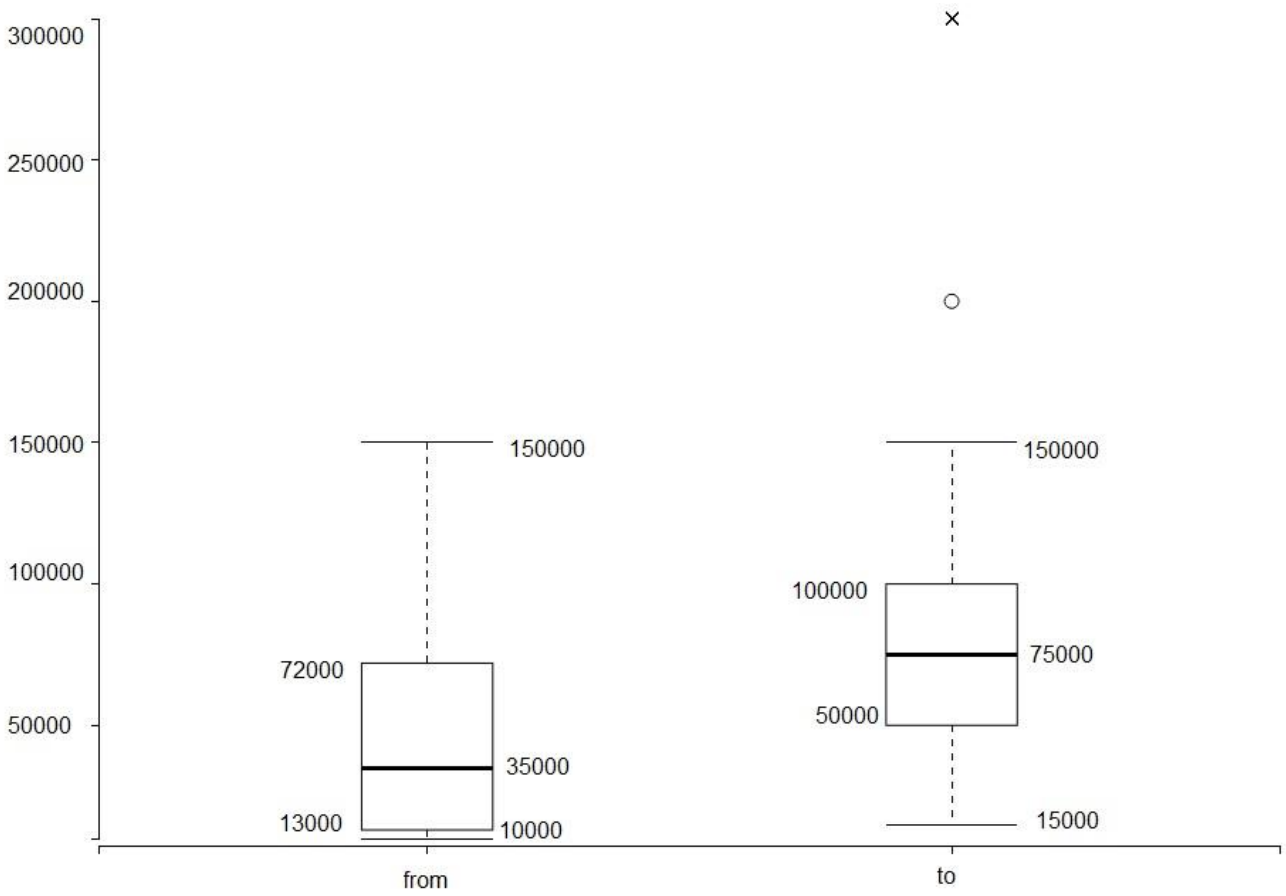


Рис. 3. Ящичковые диаграммы данных полей «from» и «to».

Узел Statistics (Statistics→Hypothesis Testing→Statistics) позволяет найти основные характеристики данных (таблица 1).

Таблица 1

Значения статистик

Статистика	Данные до нормализации		Данные после нормализации и заполнения пропущенных значений	
	«from»	«to»	«from»	«to»
Минимум	10000	15000	0	0
Максимум	150000	300000	1	1
Среднее значение	43288	89453	0,235	0,228
Стандартное отклонение	35421	65296	0,247	0,133
Асимметрия	1,123	1,19	1,18	3,724
Эксцесс	0,472	4,711	1,685	19,591
Общая сумма	2554000	1878520	14,564	14,118
Количество пропущенных значений	3	41	0	0
Количество строк	62	62	62	62

Далее была выполнена кластеризация методом k-means и алгоритмом на основе плотности (density-based алгоритм или DBSCAN). Для этого использовались узлы k-Means (Analytics→Mining→Clustering→k-Means) и DBSCAN(Analytics→Mining→Clustering→ DBSCAN). Параметры кластеризации подбирались путем использования метода силуэтов, который реализован посредством узла Silhouette Coefficient (Analytics→Scoring→ Silhouette Coefficient). Общий коэффициент силуэтов Overal, рассчитанный для каждого метода кластеризации, должны быть не менее 0,5, иначе текущее разбиение на кластеры является нецелесообразным. На рис. 4 представлены результаты работы метода силуэтов для методов k-means (рисунок 4 а) и DBSCAN (рисунок 4 б).

Row ID	D Mean Si...
cluster_2	0.319
cluster_0	0.517
cluster_1	0.757
Overall	0.57

а)

Row ID	D Mean Si...
Cluster_2	0.451
Cluster_4	0.476
Cluster_3	0.975
Noise	-0.272
Cluster_0	0.766
Cluster_1	1
Overall	0.629

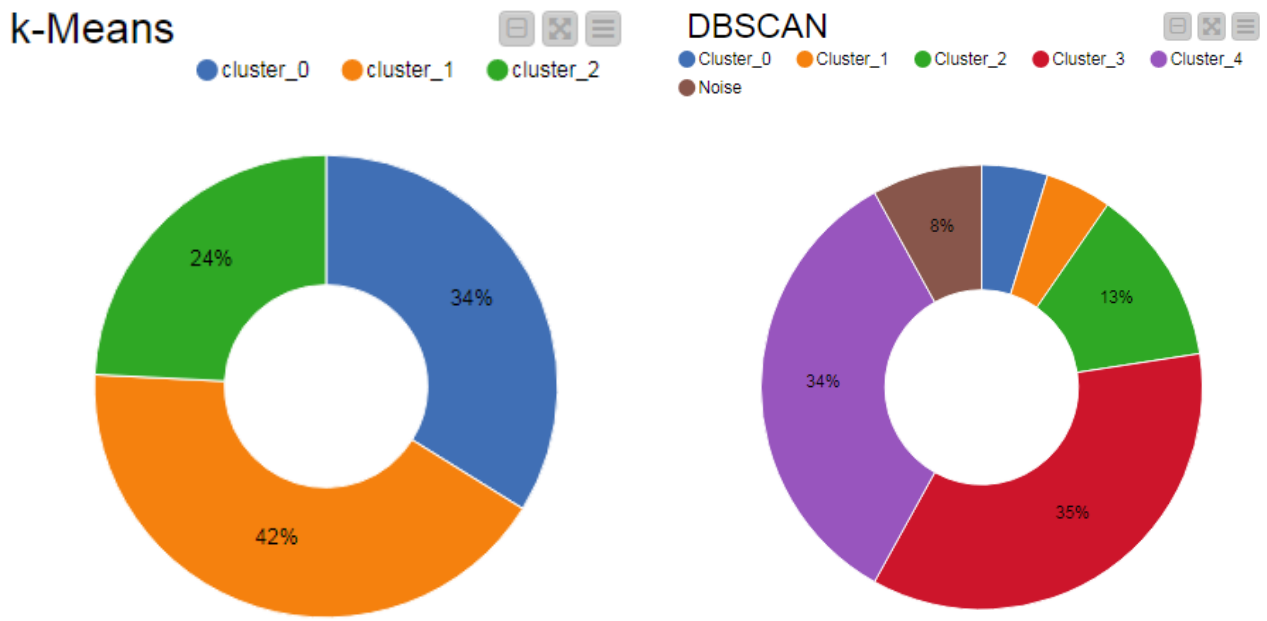
б)

Рис. 4. Коэффициенты силуэтов. а) Метод k-means, б) Метод DBSCAN.

Метод k-means позволяет определить центры кластеров (центроиды) для заранее определенного количества кластеров (его определили с помощью метода силуэтов). K-means выполняет четкую кластеризацию, которая назначает вектор данных ровно одному кластеру и в качестве метрики использует Евклидово расстояние. Алгоритм завершается, когда назначения кластера больше не меняются.

Метод DBSCAN определяет три типа точек в наборе данных. Базовые точки имеют количества соседей большее чем минимальное значение (выбрано MinPts=3) в пределах указанного расстояния (выбрано eps=0,1). Граничные точки находятся в пределах «eps» от основной точки, но имеют меньше соседей «MinPts». Точки шума «Noise» не являются ни основными, ни граничными точками. Кластеры создаются путем соединения основных точек друг с другом. Если основная точка находится в пределах «eps» от другой базовой точки, они называются непосредственно достижимыми по плотности. Все точки, которые находятся в пределах «eps» от основной точки, называются доступными по плотности и считаются частью кластера. Все остальные считаются шумом. Метод DBSCAN требует определения метрики, было выбрано Евклидово расстояние (Analytics→Distance Calculation→Numeric Distances).

Визуализация результатов работы алгоритмов кластеризации приведена на рис. 5 (а – k-means, б – DBSCAN). Кольцевые диаграммы (см. рисунок 5) выполнены на основе библиотеки NVD3 при помощи узла Pie/Donut Chart (Views→JavaScript→Pie/Donut Chart).



а) б)

Рис. 5. Кольцевые диаграммы визуализирующие работу методов кластеризации.

а) Метод k-means, б) Метод DBSCAN.

Так же с помощью узла Scatter Plot (Views→JavaScript→Scatter Plot) может быть выполнена визуализация координат центроидов.

Анализируя результаты кластеризации отметим, что метод k-means разбил вакансии на три кластера, соответствующие средним значениям по заработной плате 13000₽, 40000₽ и 100000₽ соответственно. Метод DBSCAN – на 5 кластеров, выделив при этом порядка 8% зашумленных данных.

На рис. 6 представленных итоговый рабочий процесс анализа вакансий.

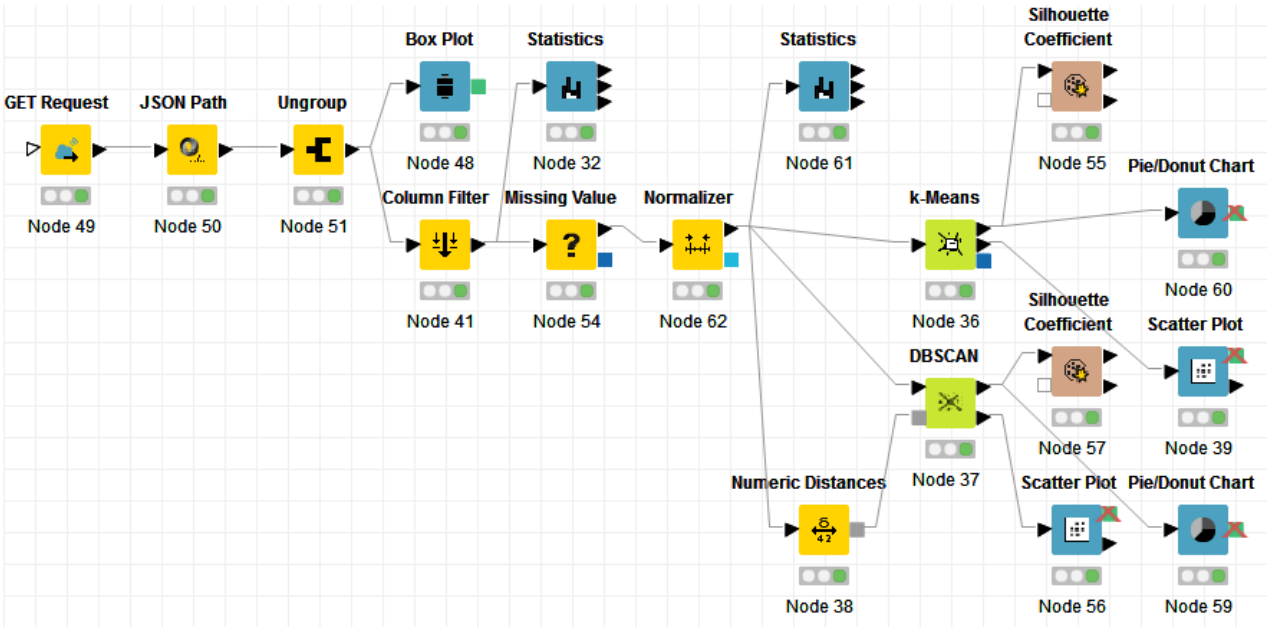


Рис. 6. Рабочий процесс анализа вакансий.

Данный процесс может быть использовать для создания шаблонов отчетов, экспортируемых в такие форматы документов, как doc, ppt, xls, pdf и другие, что может весьма эффективно использоваться для отслеживания информации на рынке вакансий в режиме реального времени.

СПИСОК ЛИТЕРАТУРЫ

1. KNIME Analytics Platform [Электронный ресурс]. – Режим доступа: https://www.knime.com/knime-analytics-platform (дата обращения: 10.09.2021).
2. HeadHunter [Электронный ресурс]. – Режим доступа: https://hh.ru (дата обращения: 10.09.2021).
3. HeadHunter API [Электронный ресурс]. Режим доступа: https://dev.hh.ru (дата обращения 10.09.2021).

АЛЕКСЕЕВА Е. С., РАССАДИН А. Э.

НОВЫЕ ДВУСТОРОННИЕ ОЦЕНКИ ПОЛНЫХ ЭЛЛИПТИЧЕСКИХ ИНТЕГРАЛОВ ПЕРВОГО И ВТОРОГО РОДА

Аннотация. В работе доказаны две теоремы, в которых установлены оценки снизу и сверху для полных эллиптических интегралов первого и второго рода, выражающиеся через элементарные функции. Эти теоремы применены к модельному геометрическому примеру. Кроме того, в статье представлены графики зависимостей от модуля этих интегралов как полученных оценок, так и их относительных погрешностей. Применимость найденных аппроксимаций для технических приложений также обсуждена.

Ключевые слова: неравенство Гёльдера, гипергеометрическая функция, круговой цилиндр, объём, инженерная точность.

ALEKSEEVA E. S., RASSADIN A. E.

NEW DOUBLE-SIDED BOUNDS FOR COMPLETE ELLIPTIC INTEGRALS OF THE FIRST AND THE SECOND KINDS

Abstract. Two theorems are proved in this paper, in which lower and upper bounds are established for complete elliptic integrals of the first and second kind, expressed in terms of elementary functions. These theorems are applied to a model geometric example. In addition, the article presents graphs of dependencies on the modulus of these integrals of both the estimates obtained and their relative errors. The applicability of the approximations found for technical applications is also discussed.

Keywords: Hölder inequality, hypergeometric function, circular cylinder, volume, engineering precision.

Необходимость в вычислении значений полных эллиптических интегралов первого:

$$\mathrm{K}(k) = \int\limits_0^{\pi/2} \frac{d\varphi}{\sqrt{1-k^2\sin^2\varphi}} \tag{1}$$

и второго рода:

$$\mathrm{E}(k) = \int\limits_0^{\pi/2} \sqrt{1-k^2\sin^2\varphi}\cdot d\varphi \tag{2}$$

часто возникает в самых разнообразных задачах (см., например, [1, 2]).

При фиксированном $k \in (0,1)$ существует целый ряд алгоритмов для определения величин интегралов (1) и (2). Например, полный эллиптический интеграл первого рода можно вычислять методом арифметико-геометрического среднего, который предложил ещё К.Ф. Гаусс [3, с. 160], а полный эллиптический интеграл второго рода можно найти

способом, описанным в [4]. Однако для теоретического исследований выражений, содержащих функции (1) и (2), эти методы неудобны. Более эффективными в такой ситуации являются двусторонние оценки интегралов (1) и (2). Примеры таких оценок дают следующие теоремы.

Теорема 1. При $k \in [0,1]$ полный эллиптический интеграл второго рода удовлетворяет неравенствам:

$$\mathrm{E}^{-}(k) \le \mathrm{E}(k) \le \mathrm{E}^{+}(k), \tag{3}$$

где

$$\mathrm{E}^{-}(k) = \frac{\pi}{2} - \frac{\pi}{2} \cdot \frac{1-(1-k^2)^{\frac{3}{4}}}{\sqrt{6}} \tag{4}$$

и

$$\mathrm{E}^{+}(k) = \frac{\pi}{2} \cdot \sqrt{1-\frac{4 \cdot k^2}{\pi^2}}. \tag{5}$$

Доказательство. При $k \in [0,1)$ производная от функции (2) по модулю k имеет вид:

$$\frac{d\mathrm{E}(k)}{dk} = -k \cdot \int_0^{\pi/2} \frac{\sin^2\varphi \cdot d\varphi}{\sqrt{1-k^2\sin^2\varphi}}. \tag{6}$$

Представим подинтегральное выражение в правой части формулы (6) как произведение двух функций: $f(\varphi) = \sin^2\varphi$ и $g(\varphi) = (1-k^2\sin^2\varphi)^{-1/2}$, и применим к этому интегралу как прямое, так и обратное неравенство Гёльдера [5, с. 11].

Сначала выберем следующие значения показателей Гёльдера: $p = q = 2$, тогда прямое неравенство Гёльдера есть не что иное, как неравенство Коши-Буняковского, а равенство (6) при $k \in (0,1)$ даёт:

$$-\frac{1}{k}\frac{d\mathrm{E}(k)}{dk} = \int_0^{\pi/2} \frac{\sin^2\varphi \cdot d\varphi}{\sqrt{1-k^2\sin^2\varphi}} \le \sqrt{\int_0^{\pi/2} \sin^4\varphi \cdot d\varphi} \cdot \sqrt{\int_0^{\pi/2} \frac{d\varphi}{1-k^2\sin^2\varphi}}. \tag{7}$$

Интегралы в правой части этого неравенства вычисляются элементарно. Таким образом, неравенство (7) сводится к следующей оценке для производной от функции (2):

$$-\frac{d\mathrm{E}(k)}{dk} \le \frac{\pi}{4} \cdot \sqrt{\frac{3}{2}} \cdot \frac{k}{\sqrt[4]{1-k^2}}, \qquad k \in (0,1). \tag{8}$$

Интегрируя неравенство (8) по модулю эллиптического интеграла от 0 до k, используя свойства интеграла Римана и учитывая, что $\mathrm{E}(0) = \pi/2$, получим, что при $k \in (0,1)$ $\mathrm{E}^{-}(k) \le \mathrm{E}(k)$ с функцией (4) в левой части этого неравенства. Справедливость неравенства при $k = 0$ и $k = 1$ проверяется непосредственно.

Далее, применим к выражению (6) обратное неравенство Гёльдера [5, с. 16] с показателями Гёльдера $p = 1/2$ (для функции $f(\varphi)$) и $q = -1$ (для функции $g(\varphi)$):

$$-\frac{1}{k}\frac{d\mathrm{E}(k)}{dk} = \int_0^{\pi/2} \frac{\sin^2\varphi \cdot d\varphi}{\sqrt{1-k^2\sin^2\varphi}} \geq \left[\int_0^{\pi/2} (\sin^2\varphi)^{1/2}\, d\varphi\right]^2 \cdot \left[\int_0^{\pi/2} \sqrt{1-k^2\sin^2\varphi} \cdot d\varphi\right]^{-1}. \qquad (9)$$

Неравенство (9) даёт для производной функции (2) следующую оценку:

$$-\frac{d\mathrm{E}(k)}{dk} \geq \frac{k}{\mathrm{E}(k)}, \qquad k \in (0,1). \qquad (10)$$

Обращаясь с неравенством (10) так же, как и в предыдущем случае, для $k \in (0,1)$ получим неравенство $\mathrm{E}(k) \leq \mathrm{E}^+(k)$ с функцией (5) в правой части этого неравенства. Справедливость неравенства при $k = 0$ и $k = 1$ проверяется непосредственной проверкой. Теорема доказана.

Теорема 1 иллюстрируется рис. 1, на котором представлены графики двусторонней оценки (3). Они демонстрируют, что при $k \in (0,1)$ функции (4) и (5) приближают функцию (2) весьма неплохо.

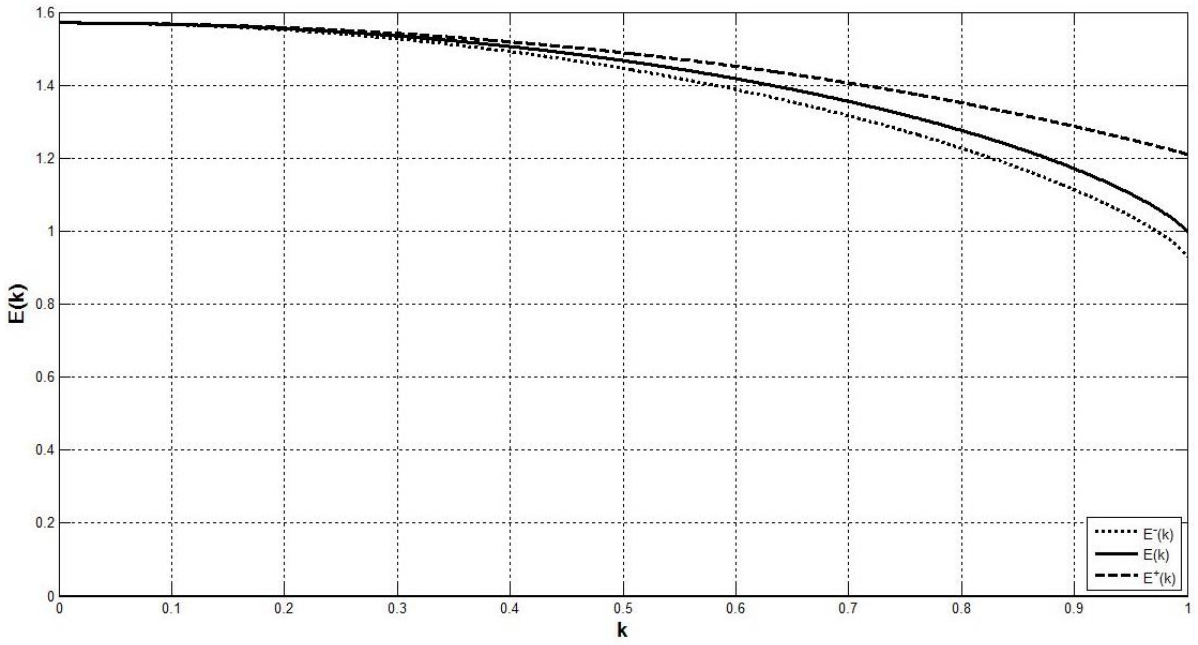


Рис. 1. Оценки полного эллиптического интеграла второго рода.

Для того, чтобы выразить количественно, насколько хорошо приближают полный эллиптический интеграл второго рода его оценки снизу (4) и сверху (5), введём относительные погрешности:

$$\delta\mathrm{E}^-(k) = \frac{\mathrm{E}(k) - \mathrm{E}^-(k)}{\mathrm{E}(k)}, \qquad \delta\mathrm{E}^+(k) = \frac{\mathrm{E}^+(k) - \mathrm{E}(k)}{\mathrm{E}(k)}. \qquad (11)$$

Графики функций (11) в зависимости от модуля k полного эллиптического интеграла приведены на рис. 2. Из этого рисунка видно, что $\delta\mathrm{E}^{-}(k) < 0{,}05$ при $k \in [0,\ 0{,}9]$, а $\delta\mathrm{E}^{+}(k) < 0{,}05$ при $k \in [0,\ 0{,}76]$, следовательно, при $k \in [0,\ 0{,}76]$ функции (4) и (5) можно применять при решении прикладных задач для оценки полного эллиптического интеграла второго рода с инженерной точностью.

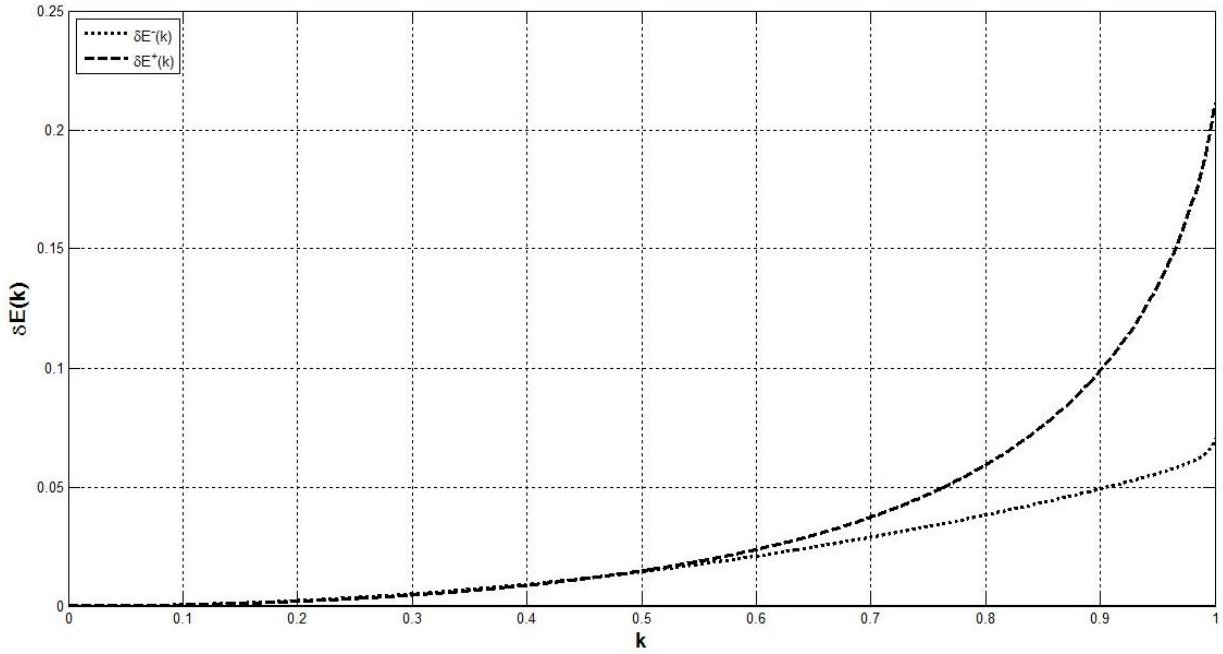


Рис. 2. Относительная погрешность оценок полного эллиптического интеграла второго рода.

Аналогичная теорема может быть доказана и для функции (1).

Теорема 2. При $k \in [0,1)$ полный эллиптический интеграл первого рода удовлетворяет неравенствам:

$$\mathrm{K}^{-}(k) \le \mathrm{K}(k) \le \mathrm{K}^{+}(k), \tag{12}$$

где

$$\mathrm{K}^{-}(k) = \frac{\pi}{2} - \frac{\pi}{6} \cdot \frac{1-(1-k^2)^{\frac{3}{4}}}{\sqrt{6}} - \frac{\pi}{4} \cdot \left(1 - \frac{1}{\sqrt{6}}\right) \cdot \ln(1-k^2) \tag{13}$$

и

$$\mathrm{K}^{+}(k) = \frac{\pi}{2} - \frac{\pi}{4} \cdot \sqrt{1-\frac{4}{\pi^2}} \cdot \ln(1-k^2) + \frac{\pi}{2} \cdot \sqrt{1-\frac{4}{\pi^2}} \cdot \ln \frac{\sqrt{1-4k^2/\pi^2} + \sqrt{1-4/\pi^2}}{1+\sqrt{1-4/\pi^2}}. \tag{14}$$

Доказательство. В работе [6] выведена следующая нелокальная связь между функциями (1) и (2), справедливая при $k \in [0,1)$:

$$\mathrm{K}(k) = \mathrm{E}(k) + \int_0^k \frac{q \cdot \mathrm{E}(q) \cdot dq}{1 - q^2}. \qquad (15)$$

Комбинируя соотношение (15) с двусторонней оценкой (3) с функциями (4) и (5) и выполняя элементарное интегрирование, придём к двусторонней оценке (12) с функциями (13) и (14). Теорема доказана.

На рис. 3, который иллюстрирует утверждение теоремы 2, приведены графики двусторонней оценки (12).

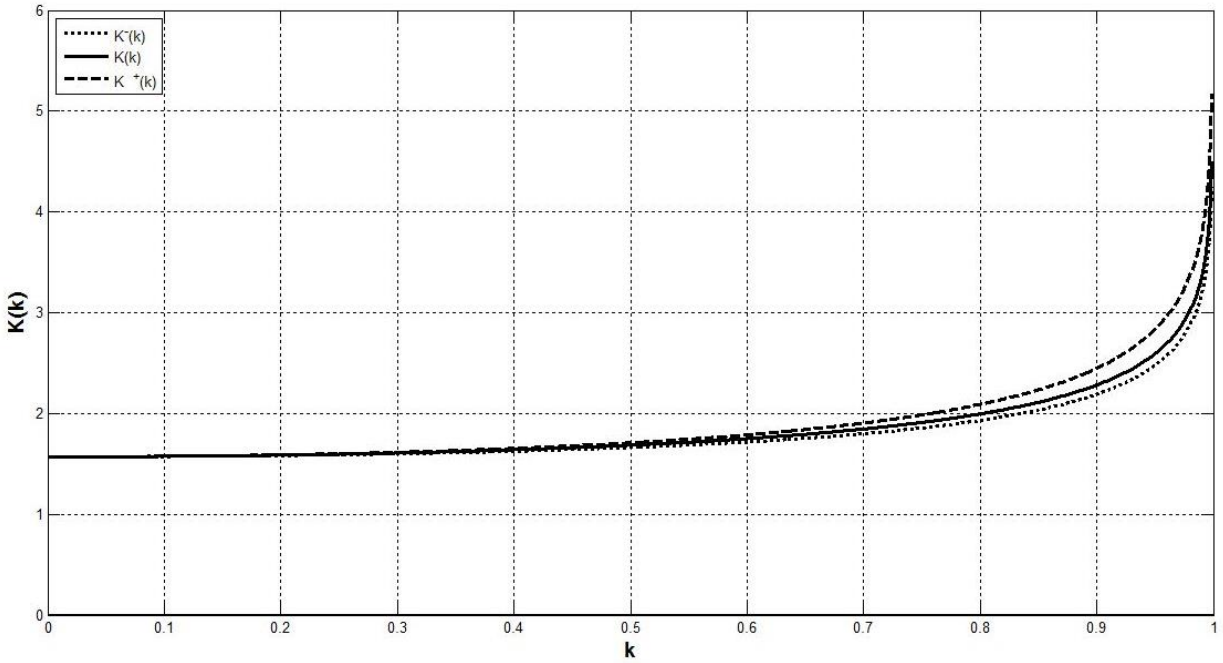


Рис. 3. Оценки полного эллиптического интеграла первого рода.

Из этого рисунка видно, что при $k \in [0,1)$ функции (13) и (14) приближают функцию (1) довольно хорошо.

Для того, чтобы выразить количественно, насколько хорошо приближают полный эллиптический интеграл первого рода его оценки снизу (13) и сверху (14), введём относительные погрешности:

$$\delta\mathrm{K}^-(k) = \frac{\mathrm{K}(k) - \mathrm{K}^-(k)}{\mathrm{K}(k)}, \qquad \delta\mathrm{K}^+(k) = \frac{\mathrm{K}^+(k) - \mathrm{K}(k)}{\mathrm{K}(k)}. \qquad (16)$$

Графики зависимостей функций (16) от модуля k полного эллиптического интеграла приведены на рис. 4. По нему можно заметить, что $\delta\mathrm{K}^-(k) < 0{,}05$ при $k \in [0,\ 0{,}99]$, а $\delta\mathrm{K}^+(k) < 0{,}05$ при $k \in [0,\ 0{,}81]$, значит, при $k \in [0,\ 0{,}81]$ функции (13) и (14) можно применять при решении прикладных задач для оценки полного эллиптического интеграла первого рода с инженерной точностью.

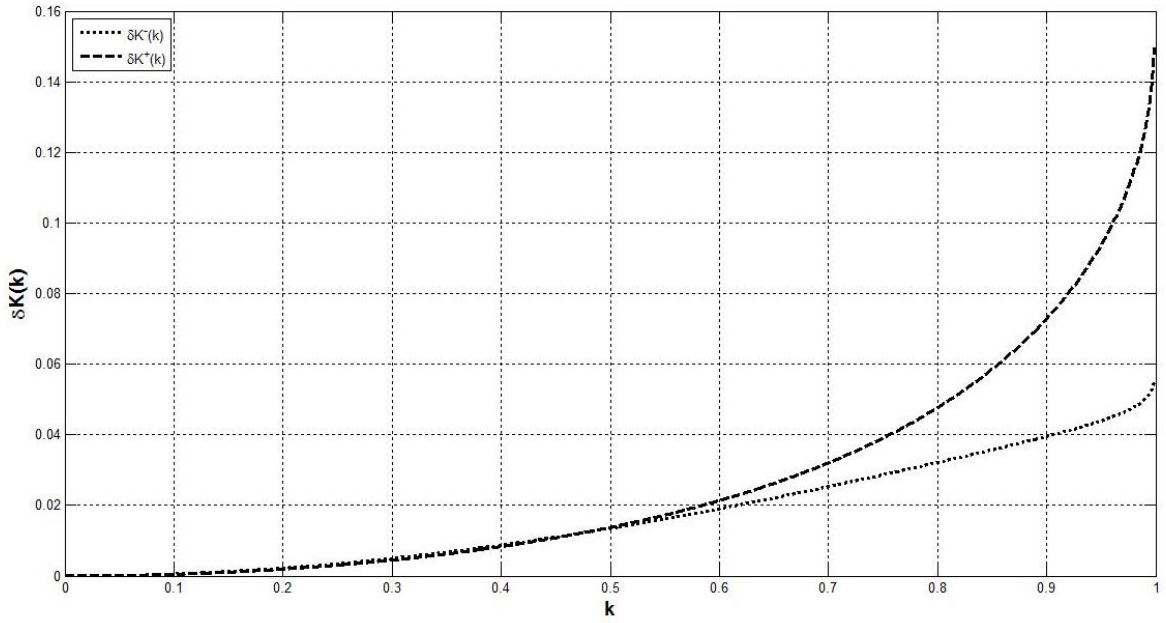


Рис. 4. Относительная погрешность оценок полного эллиптического интеграла первого рода.

Продемонстрируем применение доказанных теорем на следующем примере.

Рассмотрим два круговых цилиндра радиусов 1 и k $(0<k<1)$, оси которых пересекаются под прямым углом (см. рис. 5).

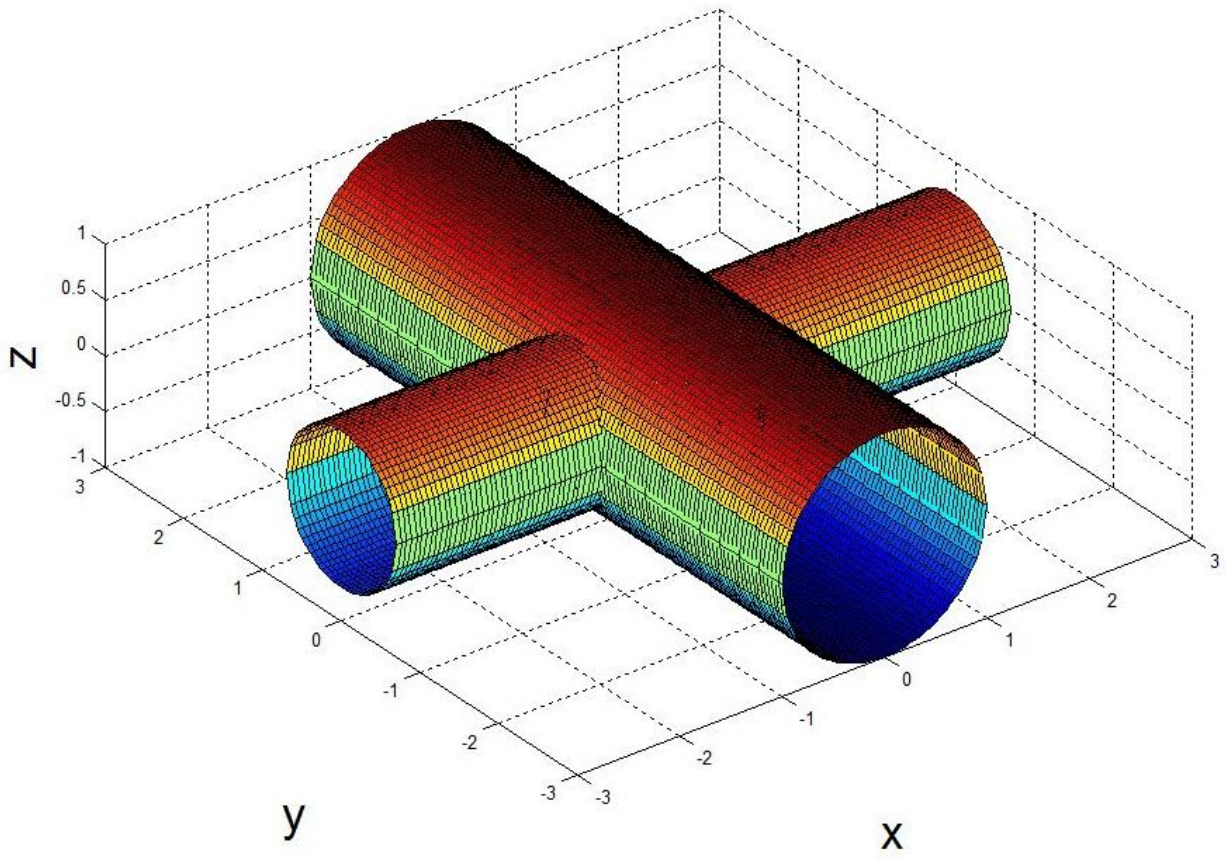


Рис. 5. Круговые цилиндры разных радиусов с осями, пересекающимися под прямым углом.

Объём тела, ограниченного ими, равен [7, с. 214]:

$$V(k)=\frac{8}{3}\cdot[(1+k^2)\cdot \mathrm{E}(k)-(1-k^2)\cdot \mathrm{K}(k)]. \tag{17}$$

Комбинируя неравенства (3) и (12), для величины (17) легко получить следующую двустороннюю оценку:

$$V^{-}(k)\le V(k)\le V^{+}(k)\,, \tag{18}$$

где

$$V^{-}(k)=\frac{8}{3}\cdot[(1+k^2)\cdot \mathrm{E}^{-}(k)-(1-k^2)\cdot \mathrm{K}^{+}(k)] \tag{19}$$

и

$$V^{+}(k)=\frac{8}{3}\cdot[(1+k^2)\cdot \mathrm{E}^{+}(k)-(1-k^2)\cdot \mathrm{K}^{-}(k)]. \tag{20}$$

Графики функций (17), (19) и (20) в зависимости от модуля k полного эллиптического интеграла приведены на рис. 6.

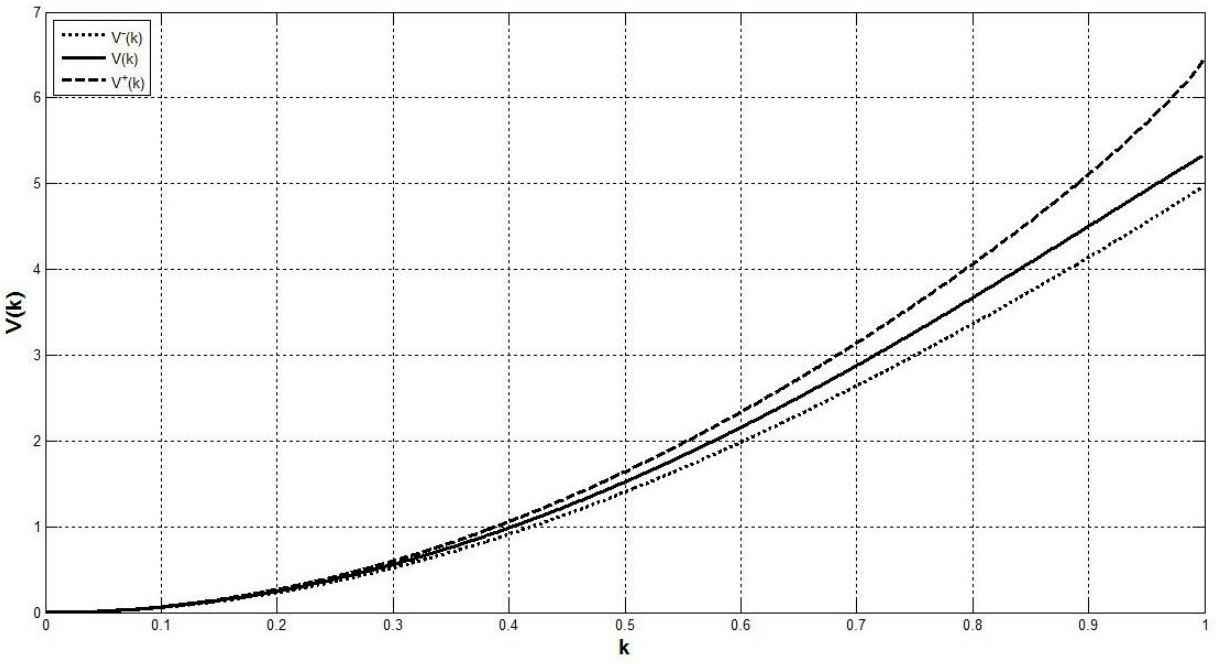


Рис. 6. Оценки объёма пересечения двух круговых цилиндров разных радиусов с осями, пересекающимися под прямым углом.

Для количественной характеризации качества приближения объёма (17) величинами (19) и (20), построенными с помощью функций (4), (5), (13), (14), как и ранее, введём относительные погрешности:

$$\delta V^{-}(k)=\frac{V(k)-V^{-}(k)}{V(k)}\,,\qquad \delta V^{+}(k)=\frac{V^{+}(k)-V(k)}{V(k)}\,. \tag{21}$$

Графики зависимостей функций (21) от модуля k полного эллиптического интеграла приведены на рис. 7.

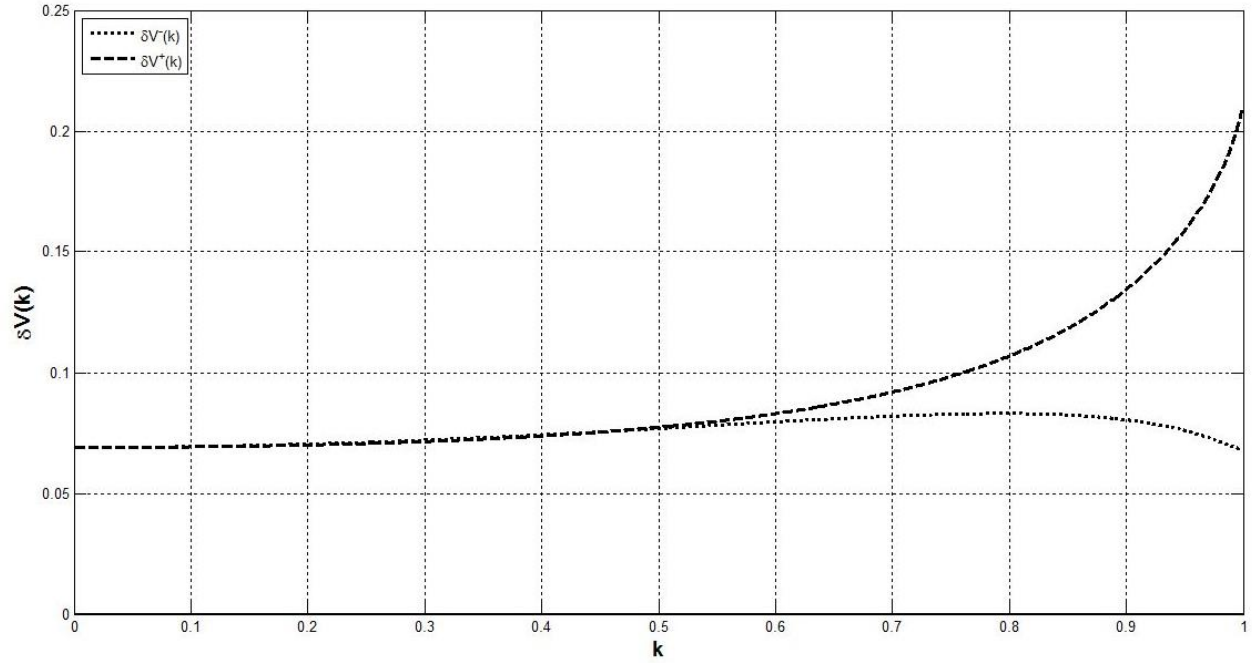


Рис. 7. Относительная погрешность оценок объёма пересечения двух круговых цилиндров разных радиусов с осями, пересекающимися под прямым углом.

Из этого графика видно, что $\delta V^{\pm}(k) > 0{,}05$ при всех $k \in (0{,}1)$, при этом кривая $\delta V^{+}(k)$ уходит вверх до заметных значений. Однако кривая $\delta V^{-}(k)$ остаётся ограниченной и снизу, и сверху: $0{,}068 < \delta V^{-}(k) < 0{,}084$, следовательно, формула (19) вполне пригодна для определения по заданному k значения объёма (17) с инженерной точностью.

Таким образом, теоремы 1 и 2, доказанные в этой работе, применимы для оценки величин как полных эллиптических интегралов первого и второго рода, так и их линейных комбинаций, с инженерной точностью.

В заключение отметим следующее обстоятельство: как хорошо известно, полные эллиптические интегралы первого и второго рода выражаются через гипергеометрические функции [3, с. 151]:

$$\mathrm{E}(k) = \frac{\pi}{2} \cdot F\left(-\frac{1}{2}, \frac{1}{2}, 1, k^2\right), \qquad \mathrm{K}(k) = \frac{\pi}{2} \cdot F\left(\frac{1}{2}, \frac{1}{2}, 1, k^2\right), \tag{22}$$

следовательно, утверждения теорем 1 и 2 означают, что специальные функции (22), являющиеся решениями линейных дифференциальных уравнений второго порядка с переменными коэффициентами, могут быть эффективно приближены с помощью элементарных функций.

СПИСОК ЛИТЕРАТУРЫ

1. Карпухин М. А. Немаксимальность экстремальных метрик на торе и бутылке Клейна // Математический сборник. – 2013. – Т. 204, № 2. – С. 31-48.
2. Rassadin A. E., Agalarov A. M.-Z. The modified Whitham modulation theory for transmission line with ferroelectric capacitors // Ferroelectrics. – 2021. – vol. 576, No. 1. – pp. 40-49.
3. Ахиезер Н. И. Элементы теории эллиптических функций. – М.: Наука, 1970. – 304 с.
4. Алексеева Е. С., Рассадин А. Э. Новый метод вычисления полного эллиптического интеграла второго рода // Алгебра, теория чисел и дискретная геометрия: современные проблемы, приложения и проблемы истории: Материалы XVIII Международной конференции, посвящённой столетию со дня рождения профессоров Б. М. Бредихина, В. И. Нечаева и С. Б. Стечкина. – Тула: Тул. гос. пед. ун-т им. Л. Н. Толстого, 2020. С. 249-251.
5. Соболев С. Л. Некоторые применения функционального анализа в математической физике. – М.: Наука, 1988. – 336 с.
6. Алексеева Е. С. Двусторонняя оценка полного эллиптического интеграла первого рода // Международный научно-практический журнал «Глобальная наука и инновация 2020: Центральная Азия». Серия «Физико-математические науки». – 2020. – № 5 (10). – С. 63-66.
7. Фихтенгольц Г. М. Курс дифференциального и интегрального исчисления. Том II. – М.: Наука, 1966. – 800 с.

БУБЛИК В. А., ГУЩИНА О. А.

РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ДЛЯ СПЕЦИАЛИЗИРОВАННОГО СЧЕТЧИКА ЭЛЕКТРИЧЕСКОЙ ЭНЕРГИИ С ДИСТАНЦИОННОЙ ПЕРЕДАЧЕЙ ИНФОРМАЦИИ

Аннотация. Статья описывает процесс разработки веб-приложения для отображения информации полученной от специализированного счетчика электрической энергии с дистанционной передачей информации. Продемонстрированы основные этапы при создании веб-приложения: постановка задачи, проектирование, макетирование, программная реализация созданного макета и тестирование.

Ключевые слова: разработка программного обеспечения, специализированный счетчик, веб-приложение, дистанционная передача информации.

BUBLIK V. A., GUSHCHINA O. A.

DEVELOPMENT OF SOFTWARE FOR A SPECIALIZED ELECTRIC POWER METER WITH REMOTE INFORMATION TRANSMISSION

Abstract. The article contains a description of the process of developing a web application for displaying information received from a specialized electricity meter with remote transmission of information. There are demonstrated the main stages of creating a web application: setting the problem, designing, prototyping, software implementation of the created layout and testing.

Keywords: development of software, specialized counter, web application, remote transmission of information.

Введение. Энергетические ресурсы являются очень важными для человечества, но также необходимо помнить и то, что их основные источники являются не возобновляемыми. Представленное ниже веб-приложение «Smart Meter Assistant» (SMA) позволяет пользователю постоянно контролировать количество потребляемой энергии в квартире или доме. Это даёт ему возможность анализировать и оптимизировать процесс электропотребления. Кроме того, спроектированный счетчик позволит владельцу использовать тариф, дифференцированный по зонам суток, что даст возможность экономии финансовых средств. Безусловно, что подобные программные продукты уже имеются на ринке, но в ходе анализа стало понятно, что их цена в несколько раз выше, чем цена спроектированного нами изделия (без потери в качестве и функционале), что делает его конкурентно способным.

Целью данного проекта является разработка и создание программного обеспечения для специализированного счетчика электрической энергии с дистанционной передачей информации.

Для достижения указанной цели необходимо было решить следующие задачи:

– выполнить анализ литературы;

– выбрать программные средства для реализации;

– определить исходные данные для проектируемой системы (ПС);

– разработать интерфейс ПС;

– разработать алгоритмы ПС;

– реализовать ПС.

Программные средства для реализации. Для реализации данного проекта использовался следующий список программных продуктов:

– интегрированная среда для разработки Arduino IDE [1] позволяет программировать микроконтроллер и его платы расширения;

– среда для разработки Visual Studio Code [2], на которой было написано веб-приложение;

– портативный локальный WAMP/WNMP сервер Open Server [3], на котором запускается веб-приложение;

– графический редактор для веб-дизайна Figma [4], применялся при разработке макетов дизайна веб-приложения;

– объектно-ориентированный язык программирования JavaScript [5], на котором написана логика работы веб-приложения.

Определение исходных данных и вариантов использования для ПС. Перед началом разработки интерфейса были определены функциональные требования ПС, на основе которых была составлена диаграмма использования (рис. 1).

Далее был проанализирован минимальный набор данных, с которым пользователь имеет возможность начать работу с ПС.

Для того чтобы клиент имел возможность приступить к началу работы с проектируемой системой необходимо определить тип пользователя:

- поставщик;
- потребитель.

Рис. 1. Диаграмма вариантов использования.

В зависимости от типа пользователя будет отличаться функциональный набор аккаунта.

Также должно быть определено: зарегистрирован ли пользователь в системе.

Если клиент ранее не использовал проектируемую систему, то ему необходимо пройти процедуру регистрации. Для этого ему следует указать:

- логин;
- пароль;
- ФИО пользователя;
- адрес электронной почты.

Стоит отметить, что потребитель может быть зарегистрирован своим поставщиком электроэнергии, если предоставит ему необходимые данные.

После авторизации каждому типу пользователя необходим минимальный набор данных, чтобы начать использование системы «SMA».

а) для поставщика этот набор данных включает:

- база потребителей численностью не менее одного человека, которая содержит в себе следующие данные:

– ФИО потребителя;

– адрес, где используется счетчик;

– уникальный номер счетчика;

– срок действия свидетельства после последней проверки;

– показатели количества использованного электричества.

• тарифный план, дифференцированный по зонам суток, при этом в нем должны быть указаны данные:

– название тарифного плана;

– время, в течение которого он является актуальным;

– стоимость одной единицы электрической энергии.

б) для потребителя этот набор данных включает следующие данные:

• один и более подключенный к системе счетчик со следующей информацией:

– ФИО потребителя;

– адрес, где используется счетчик;

– уникальный номер счетчика;

– срок действия свидетельства, после последней проверки;

– показатели количества использованного электричества.

Разработка интерфейс ПС. На рис. 2 представлен вид главного окна программы. Это окно содержит краткую информацию о самых важных аспектах, которые должен видеть пользователь. В нем также есть возможность подробно посмотреть информацию о счетчиках или же добавить новый. Кроме того, в верхней части окна доступно меню, с помощью которого можно переместиться в другие разделы.

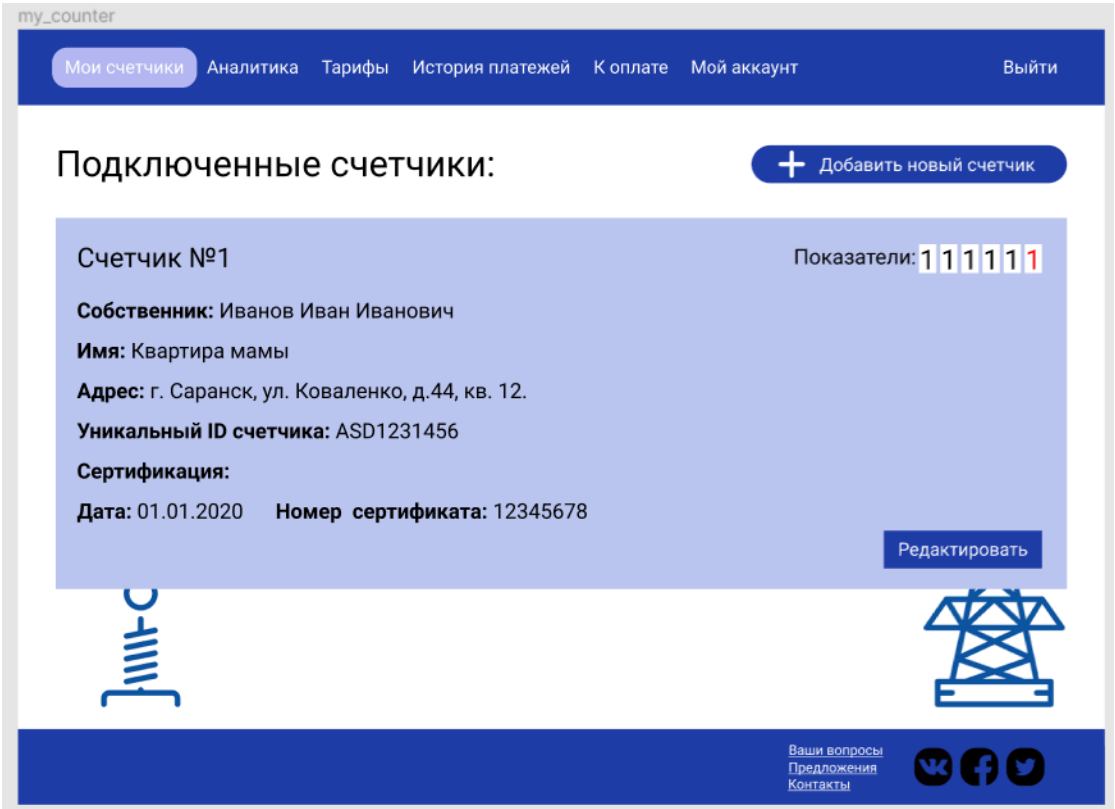


Рис. 2. Раздел «Мои счетчики».

На рис. 2 продемонстрирован раздел «Мои счетчики». Пользователь может детально ознакомиться со всей информацией о подключенных счетчиках; при необходимости он может отредактировать ее или добавить новый счетчик.

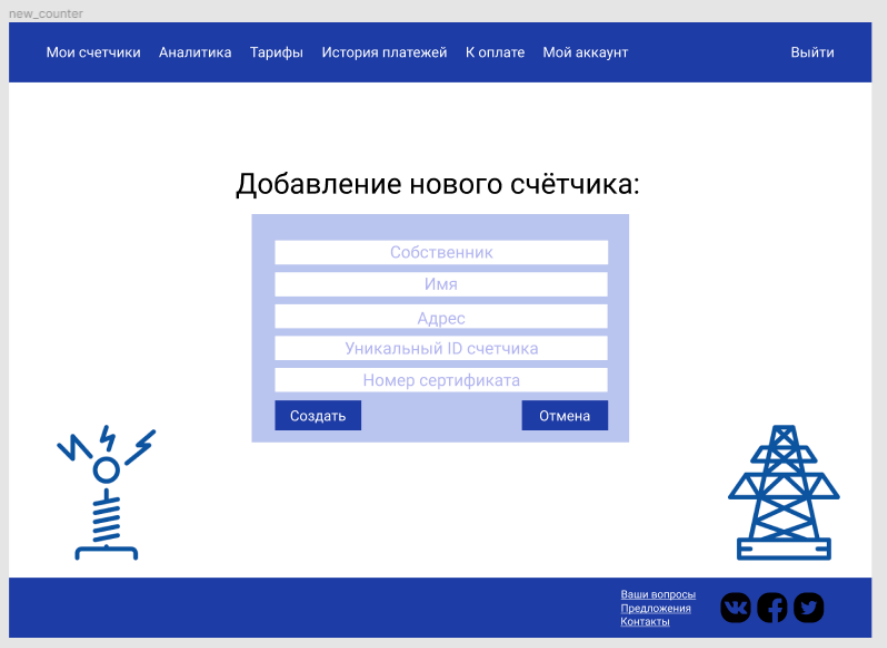


Рис. 3. Форма для добавления нового счетчика.

На рис. 3 продемонстрирована форма для добавления нового счетчика. В ней необходимо указать ФИО собственника, имя счетчика, адрес по которому он установлен, его уникальный ID и номер сертификата, который свидетельствует о том, что данный счетчик был проверен и соответствует всем необходимым стандартам.

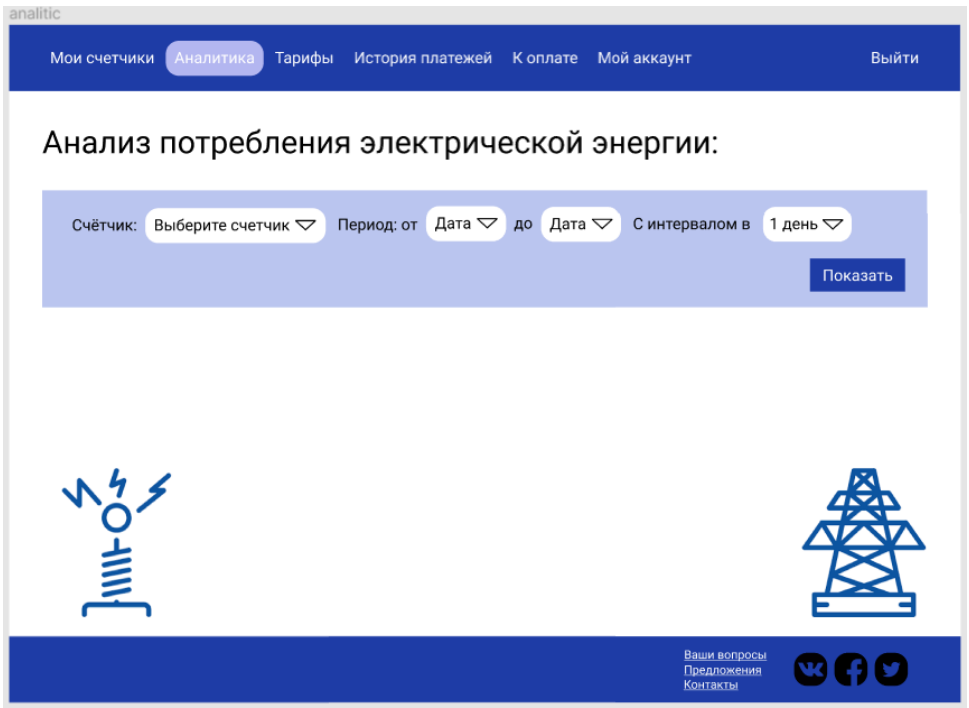


Рис. 4. Раздел «Аналитика».

На рис. 4 изображен раздел «Аналитика». В этом разделе можно просмотреть информацию о количестве и динамике потребления электроэнергии в определенный отрезок времени. Для удобства можно выбрать период дискретизации.

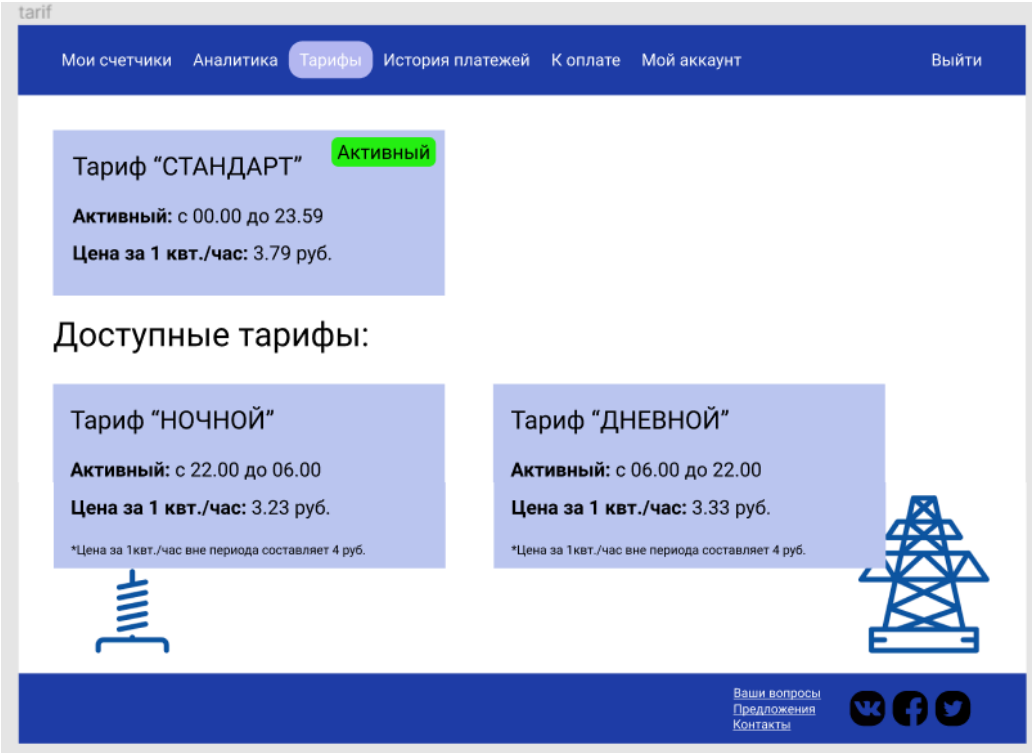


Рис. 5. Раздел «Тарифы».

На рис. 5 изображен раздел «Тарифы». В данном разделе отображается информация о действующих тарифах, которые выбрал пользователь. Здесь также можно поменять текущий тариф. В описании тарифов указано название тарифа, время (в течение которого он является актуальным) и стоимость 1 квт. / час.

На рис. 6 изображен раздел «История платежей». В данном разделе в виде таблицы отображается информация о финансовых операциях, которые были выполнены на данном аккаунте. В таблице можно просмотреть: суммы предыдущих платежей, время, в которое они были осуществлены, и за какой период была произведена оплата.

На рис. 7 изображен раздел «К оплате». В данном разделе отображается информация о количество использованной электрической энергии и сумму, которую необходимо внести для погашения задолженности.

Также на сайте есть раздел «Мой аккаунт», в котором указана информация о пользователе. При необходимости ее можно отредактировать.

Рис. 6. Раздел «История платежей».

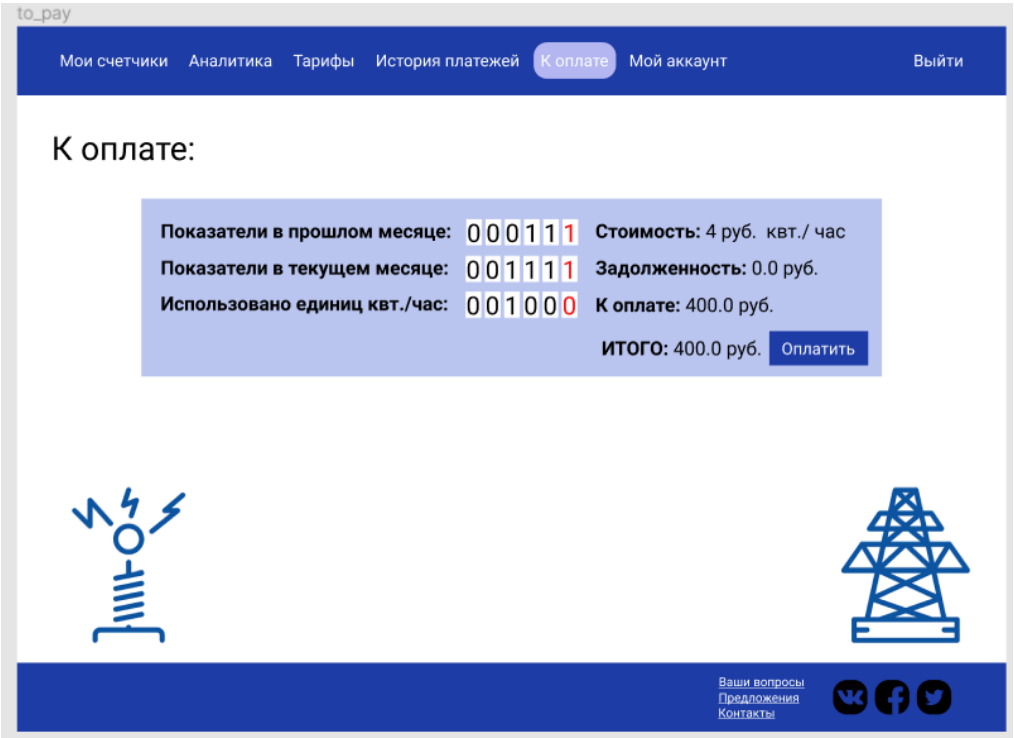


Рис. 7. Раздел «К оплате».

Результаты работы. После этапа разработки макета в графическом редакторе Figma, было реализовано веб-приложение, представленное на рис. 8. Весь функционал соответствует ранее поставленным задачам.

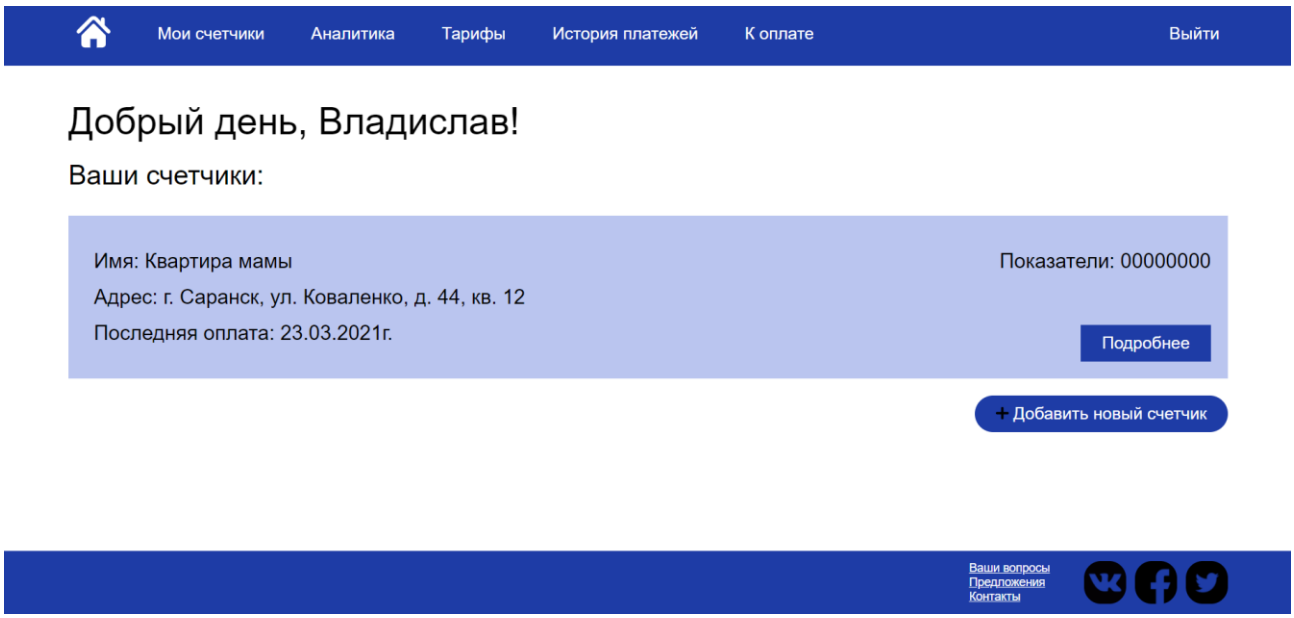


Рис. 8. Главная страница веб-приложения.

Заключение. В результате выполнения данного проекта была изучена предметная область специализированных счетчиков электрической энергии с дистанционной передачей информации, проанализированы существующие аналоги и предложено свое решение поставленной цели, разработано программное обеспечение с применением Visual Studio Code, Open Server и Arduino IDE, создана система «Smart Meter Assistant», позволяющая получать, хранить и обрабатывать данные, полученные от специализированного счетчика.

СПИСОК ЛИТЕРАТУРЫ

1. Среда разработки Arduino [Электронный ресурс]. – Режим доступа: http://arduino.ru/Arduino_environment (дата обращения 13.09.2021).
2. Visual Studio Code - Code Editing. Redefined [Электронный ресурс]. – Режим доступа: https://code.visualstudio.com/ (дата обращения 13.09.2021).
3. Open Server Panel [Электронный ресурс]. – Режим доступа: https://ospanel.io/ (дата обращения 13.09.2021).
4. Что такое Figma: возможности и принципы работы [Электронный ресурс]. – Режим доступа: https://skillbox.ru/media/design/chto_takoe_figma/ (дата обращения 13.09.2021).
5. Современный учебник JavaScript [Электронный ресурс]. – Режим доступа: https://learn.javascript.ru/ (дата обращения 13.09.2021).

БУТКИНА А. А., ШАМАЕВ А. В.

РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ДЛЯ ИНТЕГРАЦИИ CAD, CAE-СИСТЕМ, УСТАНОВЛЕННЫХ НА ТЕРРИТОРИАЛЬНО-УДАЛЕННЫХ КОМПЬЮТЕРАХ

Аннотация. В статье описано разработанное авторами программное обеспечение. Оно предназначено для организации процесса обмена информацией между территориально-удаленными компьютерами, на которых функционируют различные системы автоматизированного проектирования и инженерного анализа.

Ключевые слова: CAD, CAE, веб-приложение, база данных, MySQL, PHP.

BUTKINA A. A., SHAMAEV A. V.

DEVELOPMENT OF SOFTWARE FOR INTEGRATION OF CAD, CAE-SYSTEMS INSTALLED ON REMOTE COMPUTERS

Abstract. The article describes the software developed by the authors. The software is designed to organize the process of electronic data interchange between remote computers on which different computer-aided design and engineering analysis systems are installed.

Keywords: CAD, CAE, web application, database, MySQL, PHP.

Введение. Данное исследование посвящено разработке программного обеспечения для интеграции CAD, CAE-систем, организующего эффективный и удобный обмен информацией о проектируемых изделиях с возможностью разделения прав доступа. Его актуальность обусловлена необходимостью организации процессов обмена информацией между территориально-удаленными компьютерами, на которых функционируют различные системы автоматизированного проектирования и моделирования.

Для достижения указанной цели в работе были поставлены задачи:

– выполнить анализ предметной области;

– исследовать аналоги систем обмена и хранения информации;

– осуществить выбор инструментов для проектирования и реализации разрабатываемой системы;

– определить ключевые прецеденты системы;

– реализовать программное обеспечение;

– выполнить проверку сценариев использования разработанного программного обеспечения на конкретных примерах.

Анализ предметной области. Применяемые в настоящее время системы автоматизированного проектирования и моделирования представляют собой несколько взаимодействующих комплексов программ, которые включают как системы автоматизированного проектирования (CAD-системы), так и системы, предназначенные для решения различных инженерных задач (CAE-системы). Однако, эти системы не всегда хорошо интегрированы друг с другом. Поскольку в современном производстве наблюдается тенденция к увеличению доли аутсорсинга и созданию расширенных компаний, становится очевидной необходимость использования новых инструментов для организации коллективного доступа к информации о разрабатываемых изделиях и соответствующей технологической документации. Это приводит к необходимости решения проблем интеграции информации, полученной из различных CAD/CAE-систем, путём создания единой информационной среды для их успешного совместного функционирования.

Разработанное в рамках выполнения данного исследования веб-приложение предназначено для создания такой единой информационной среды (ЕИС), которая обеспечит возможности организованного хранения и доступа к информации, генерируемой в различных CAD/CAE-системах, размещённых на территориально-удаленных компьютерах, с возможностью разграничения прав доступа к ней.

Анализ аналогов. Рассмотрим известные аналоги разрабатываемой системы:

1. Dropbox [1] – это облачное хранилище для различных файлов с расширенным функционалом. Особенностями данного сервиса являются:

– синхронизация файлов любых размеров и типов;
– различные уровни доступа к разным папкам;
– синхронизация файлов;
– автоматическое резервное копирование файлов.

2. Google Drive (Google Диск) [2] – это облачный сервис для хранения файлов, который позволяет выполнять обмен и совместное редактирование файлов, с использованием любого устройства, подключенного к сети Интернет.

Недостатками данных аналогов по сравнению с разрабатываемой системой являются:

– отсутствие модуля «общение»;
– наличие ограничения на размер бесплатного хранилища данных.

Технологии реализации. Для разработки веб-приложения использовался следующий стек технологий:

– портативно-серверная платформа OpenServer;
– система управления базами данных MySQL;

– языки программирования и создания веб-контента HTML5 и PHP, а также каскадная таблица стилей CSS.

Проектирование. В разрабатываемом программном обеспечении имеются две роли – администратор и пользователь. Их базовые функции представлены на рис. 1 и не требуют пояснений.

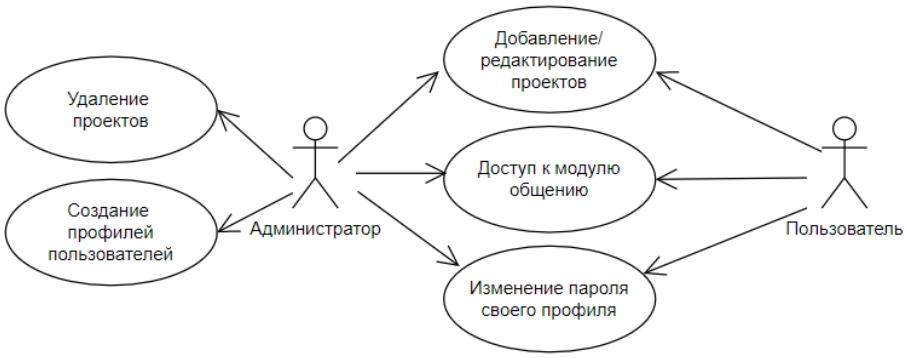


Рис. 1. Диаграмма вариантов использования.

Эффективность любого приложения, использующего в своей работе базу данных, зависит от того, насколько оптимально спроектирована ее схема. Она напрямую влияет на такие характеристики, как общая архитектура системы, набор классов, скорость обработки данных и удобство обращения к хранилищу.

В процессе разработки веб-приложения спроектирована следующая схема базы данных (БД), используемой в качестве хранилища проектов (рис. 2).

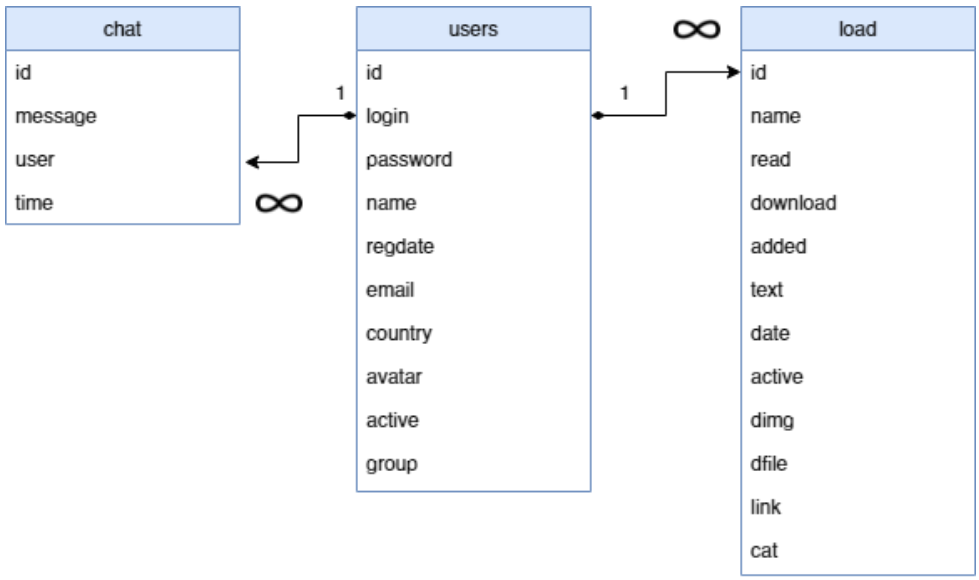


Рис. 2. Схема БД разработанного веб-приложения.

Рассмотрим структуру разработанной базы данных:

1. Таблица **Users** – содержит информацию обо всех пользователях разработанного веб-приложения. В частности, в ней содержится следующая информация: идентификатор пользователя (id), имя учетной записи пользователя (login), пароль (password), имя пользователя (name), дату регистрации (regdate), адрес электронной почты (email), фотография пользователя (avatar), поле active (результат активации учетной записи с помощью отправки письма на электронную почту пользователя), статус профиля: администратор или обычный пользователь (group).

2. Таблица **Load** – включает в себя информацию о загруженных в систему проектах. Для каждого проекта хранится идентификатор (id), имя проекта (name), количество просмотров (read), количество скачиваний (download), имя пользователя, который добавил проект (added), текст описания проекта (text), дата добавления проекта (date), полный путь к каталогу, в котором хранятся файлы проекта (dfile), ссылка на Интернет-ресурсы, содержащие дополнительные материалы по проекту (link), категория проекта (cat).

3. Таблица **Chat** – содержит информацию сообщениях, которыми обмениваются пользователи. Она включает следующие поля: идентификатор сообщения (id), текст сообщения (message), имя отправителя (user), время отправки сообщения (time).

Таблица **Users** связана с таблицей **Chat** отношением один ко многим, так как один пользователь может отправить несколько сообщений. С таблицей **Load** данная таблица связана отношением один ко многим, потому что один пользователь может загружать несколько проектов.

Обзор основных модулей реализованного приложения. Серверная часть разработанного веб-приложения включает:

1. Модуль регистрации пользователей.

2. Модуль авторизации пользователей, включающий проверку для различения компьютеров и людей с использованием технологии CAPTCHA.

3. Модуль добавления проекта в БД. В зависимости от принадлежности пользователя к определенной группе пользователей он будет иметь соответствующие права доступа.

4. Модуль редактирования проекта. Права доступа к возможности редактирования проекта определяются принадлежностью пользователя к определенной группе.

Рассмотрим разделы интерфейса клиентской части разработанного веб-приложения:

1. Раздел **Каталог проектов**.

Этот раздел предназначен для хранения информации о проектах в системе. В нем пользователю предоставляется возможность выбрать файл архива проекта для его загрузки в хранилище, ввести его название и краткое описание (вид интерфейса страницы каталога

проектов представлен на рис. 3). В данном случае файл архива проекта используется для того, чтобы сохранить все файлы с информацией об одном проекте в сжатом виде в едином файле. Каждый файл архива проекта может быть отнесен к одной из существующих категорий хранилища.

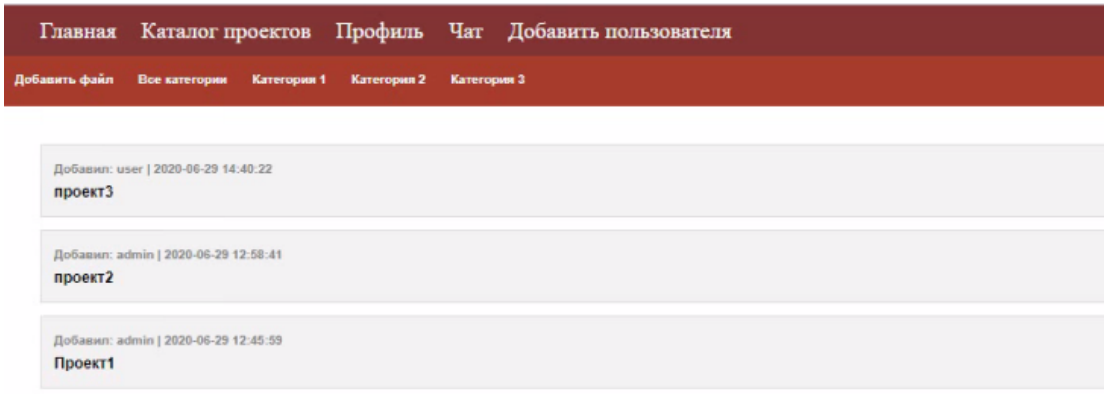


Рис. 3. Вид страницы каталога проектов для администратора.

Следует отметить, что, как уже было указано ранее, права пользователя и администратора отличаются. Пользователь может только добавлять и редактировать проекты, а администратор также может удалять их (рис. 4). Все пользователи системы имеют доступ к информации о дате размещения файла и учетной записи пользователя, который его добавил.

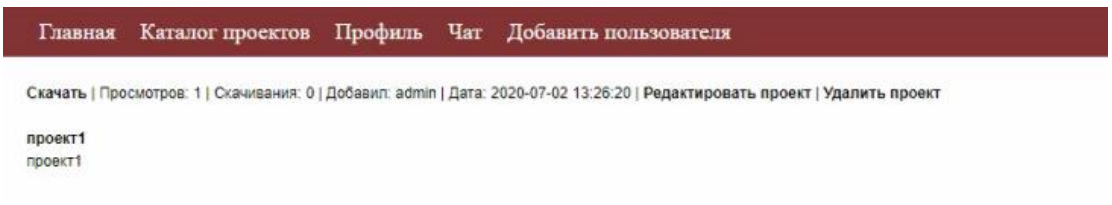


Рис. 4. Страница просмотра проекта для администратора.

2. Раздел **Редактировать.**

При выборе пункта **Редактировать проект**, пользователь переходит на страницу редактирования названия проекта и его текстового описания (рис. 5). Кроме того, с помощью данной страницы пользователь может изменить категорию проекта и при необходимости записать новую версию файла архива проекта, нажав кнопку **Выбрать файл**. При этом старая версия проекта будет удалена.

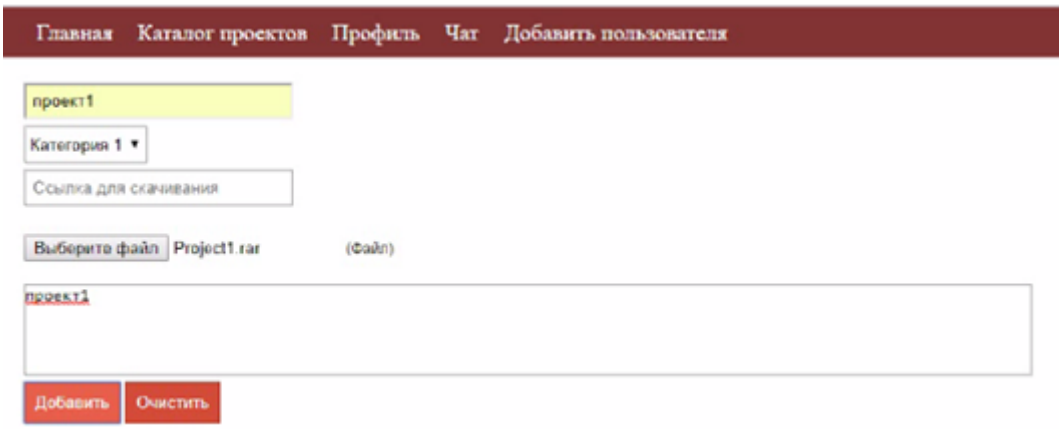


Рис. 5. Редактирование проекта от имени администратора.

3. Раздел **Удалить** (доступен только администратору).

Выбрав определенный проект, администратор может удалить его. Результат выполнения данной операции показан на рис. 6.

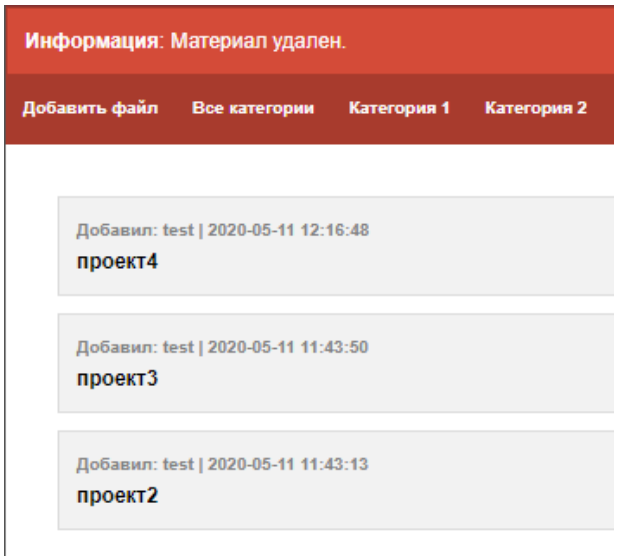


Рис. 6. Информация об удалении материала.

4. Раздел **Профиль**.

Каждому пользователю, вошедшему в систему, доступен раздел **Профиль** (рис. 7). На этой странице он может изменить имя и пароль своей учетной записи, загрузить фотографию и просмотреть дату регистрации и статус своего профиля (администратор/пользователь).

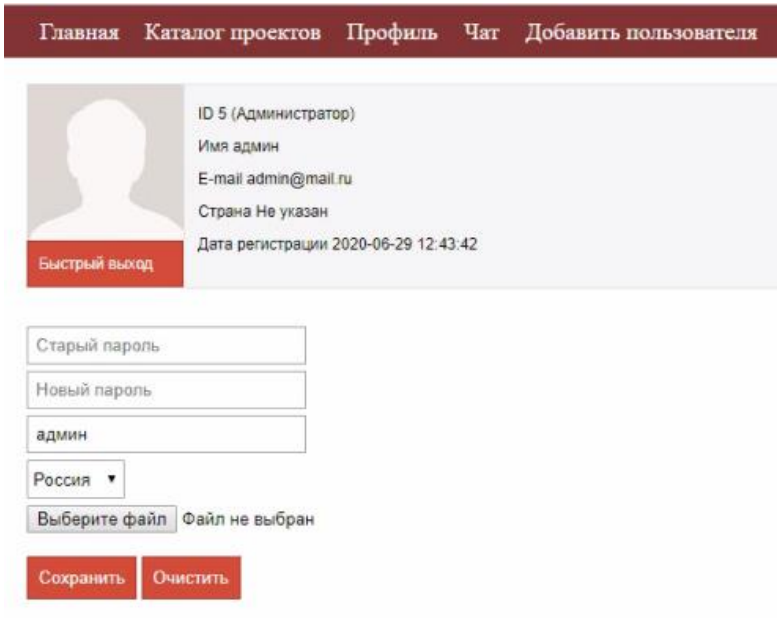


Рис. 7. Страница профиля для администратора.

5. Раздел **Чат**.

Авторизованные в системе пользователи имеют возможность обмениваться друг с другом сообщениями, используя чат (рис. 8). При этом они могут как отправлять сообщения, так и просматривать их историю. Для каждого сообщения отображается дата его отправки и имя учетной записи отправителя.

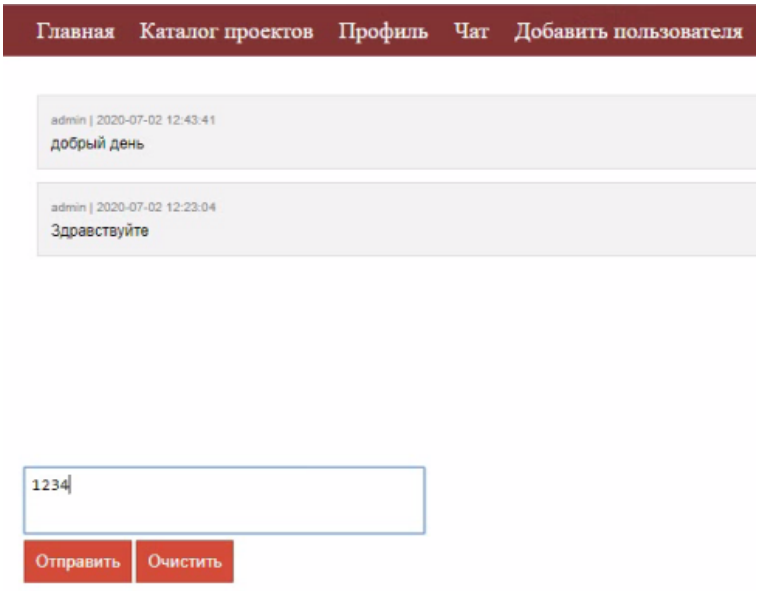


Рис. 8. Страница обмена сообщениями для администратора.

6. Раздел **Добавить пользователя.**

Администратор имеет возможность **добавления новых пользователей** в систему. Для этого ему нужно перейти на соответствующую страницу (рис. 9). После чего необходимо заполнить соответствующие поля и нажать кнопку «Регистрация».

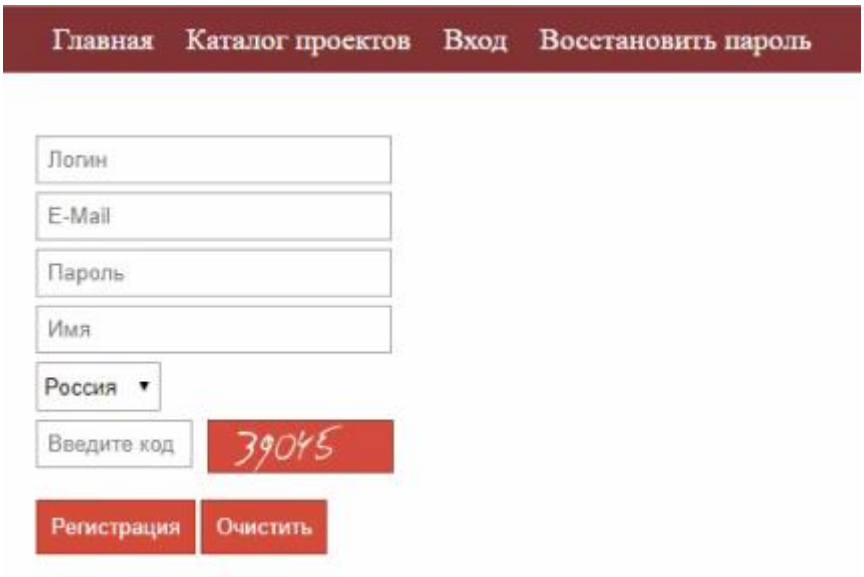


Рис. 9. Страница регистрации нового пользователя.

Продемонстрируем работу данного веб-приложения на конкретном примере. Пусть в качестве пользователя CAD-системы выступает инженер-проектировщик, работающий с соответствующей системой автоматизированного проектирования, находящейся в одном здании предприятия, а в качестве пользователя CAE-системы выступает специалист в области инженерного анализа, работающий с соответствующей автоматизированной системой, находящейся в другом здании этого предприятия (или его филиале в другом городе).

Таким образом, мы рассматриваем ситуацию взаимодействия CAD-системы и CAE-системы, расположенных на территориально-удаленных компьютерах. Рассмотрим в качестве примера следующий сценарий использования разработанного веб-приложения:

1. Пользователь CAD-системы завершает процесс проектирования одной из 3D моделей проекта и загружает ее в хранилище, после чего сообщает об этом в чате приложения для того, чтобы остальные разработчики имели возможность получить к ней доступ и выполнить ее тестирование.

2. Получив сообщение о добавлении модели к проекту, пользователь CAE-системы загружает ее на свой компьютер и выполняет ее тестирование, используя соответствующее

приложение данной системы. Далее он сообщает о результате тестирования в общий чат и добавляет в проект информацию об обнаруженных дефектах протестированной модели.

3. Пользователь CAD-системы реагирует на сообщение в чате и вносит корректировки в данный проект после редактирования соответствующего файла с 3D моделью в CAD-системе и вновь сообщает об этом в чат.

Описанный процесс может продолжаться итерационно до тех пор, пока не будут достигнуты требуемые характеристики проекта. Таким образом, может происходить взаимодействие пользователей CAD- и CAE-систем, расположенных на территориально-удаленных компьютерах.

Заключение. Все задачи, поставленные в рамках данного исследования, были решены. Результатом работы является разработанное веб-приложение, которое организует процесс обмена информацией между территориально-удаленными компьютерами, на которых функционируют различные системы автоматизированного проектирования и инженерного анализа. Разработанное программное обеспечение позволит повысить скорость и удобство реализации проектов, а также оптимизирует процесс обмена данными по проектам между разработчиками.

СПИСОК ЛИТЕРАТУРЫ

1. Dropbox – обзор сервиса. [Электронный ресурс] // Startpack. – Режим доступа: https://startpack.ru/application/dropbox-box (дата обращения 25.08.2021).
2. Google Диск – обзор. [Электронный ресурс] // Startpack. – Режим доступа: https://startpack.ru/application/google-drive (дата обращения 25.08.2021).

ФИРСОВА С. А., МЕЛЕШКИН А. А.

РАЗРАБОТКА СИСТЕМЫ ДЛЯ АВТОМАТИЧЕСКОГО РАСПОЗНАВАНИЯ АККОРДОВОЙ ПОСЛЕДОВАТЕЛЬНОСТИ В ЦИФРОВЫХ АУДИОФАЙЛАХ

Аннотация. В статье рассматривается алгоритм распознавания аккордовой последовательности в аудиофайлах с помощью разделения частот нот на частотные классы. Описываются принципы работы и реализация разработанной авторами системы, основанной на данном алгоритме.

Ключевые слова: быстрое преобразование Фурье, распознавание аккордов, гармоники, спектрограмма, теория музыки, музыкальный строй, хромограмма, алгоритмическое распознавание аккордов, частотно-временное представление, частотные классы.

FIRSOVA S. A., MELESHKIN A. A.

DEVELOPMENT OF A SYSTEM FOR AUTOMATIC CHORD SEQUENCE RECOGNITION IN DIGITAL AUDIO FILES

Abstract. The article considers an algorithm for recognizing a chord sequence in audio files by dividing the frequencies of notes into frequency classes. The principles of operation and implementation of the system developed by the authors based on this algorithm are described.

Keywords: fast Fourier transform, chord detection, harmonics, spectrogram, music theory, musical scale, chromogram, algorithmic chord recognition, time-frequency domain, pitch class profile.

Введение. В настоящее время вся музыкальная индустрия интенсивно использует информационные технологии для создания музыки. Для начинающих музыкальных продюсеров открылось множество способов испытывать свои художественные идеи дешево и быстро. Больше не нужно нанимать целый оркестр, чтобы добавить звучание дополнительных инструментов в создаваемый трек. Существует большое количество библиотек, содержащие сэмплы любых музыкальных инструментов, которые можно воспроизвести на midi-клавиатуре. Использование цифрового звука также позволяет анализировать звуковую дорожку путем вывода спектрограммы и при этом позволяет изменять интенсивность некоторых частот. Музыкальные инженеры имеют в своем распоряжении утилиты для настройки записанного вокала. Они могут увидеть всю последовательность нот, которую спел вокалист, а также корректировать их. Инструмент, который будет распознавать аккорды может сильно упростить анализ неизвестной ранее музыкальной записи звуковому инженеру или аранжировщику для дальнейшей работы с треком.

Информационные технологии не только упростили работу для профессиональных музыкантов. Обучение музыке дома может дать все более полное представление о музыкальной теории и может стать более эффективным. Существуют приложения для тренировки музыкального слуха, изучения нотации, демонстрации высотности звука вокалиста в реальном времени. Приложение, распознающее аккорды в музыке, поможет избавить новичков от поиска нотации в интернете. Особенно это актуально, когда песня недавно вышла, и профессиональные музыканты еще не успели записать и выложить нотацию к ней.

Анализ цифрового звука для распознавания аккордов. Для начала рассмотрим основные понятия теории музыки. Мы представляем ноту как восприятие человеком звуковой волны, колеблющейся на определенной частоте. Например, в современной западной музыке инструменты принято настраивать по ноте – «Ля» первой октавы», которая имеет стандартизованную частоту 440 Гц.

Аккорд – это сочетание одновременно звучащих трёх и более звуков разной высоты.

Чтобы получить из цифрового аудиофайла информацию о содержащихся в нем частотных классах, нужно перевести этот файл в частотное представление. Для этого может использоваться быстрое преобразование Фурье. На рис. 1 представлено преобразование Фурье для ноты «Ля малой октавы» [1]:

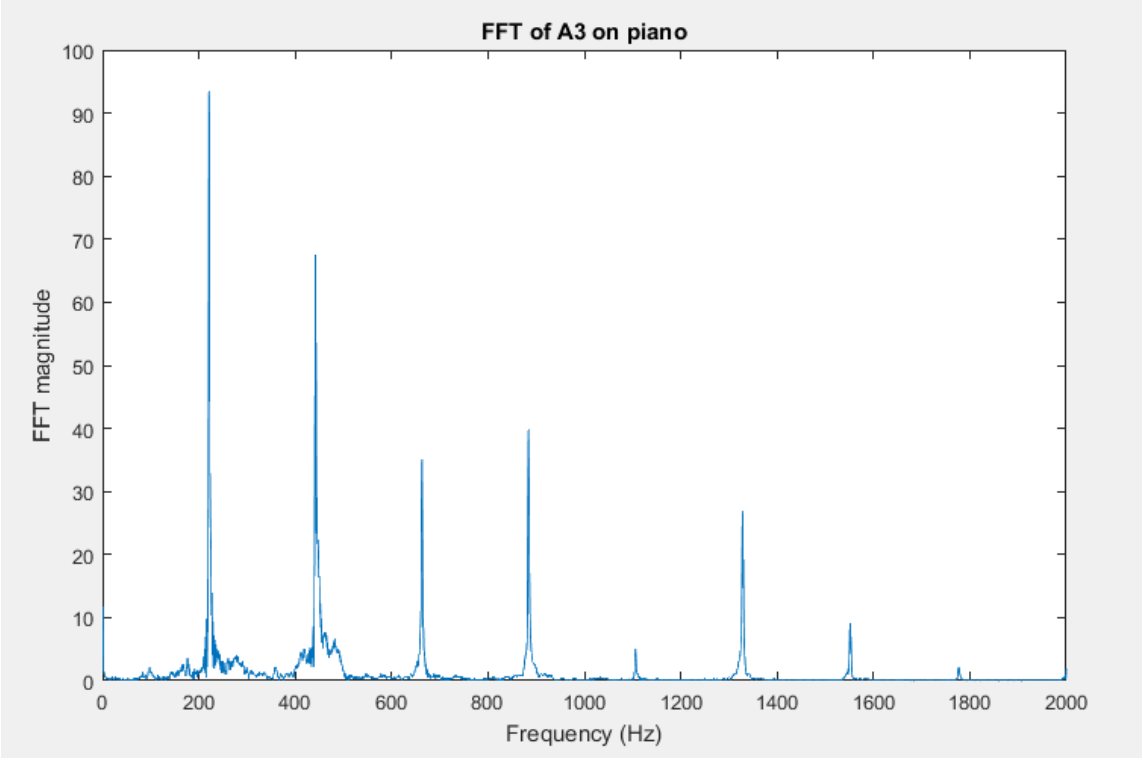


Рис. 1. Преобразование Фурье для записи, на которой звучит нота Ля, сыгранная на фортепиано.

Данная нота имеет основную частоту 220 Гц. Но тогда откуда берутся остальные пики? Одна из основных проблем распознавания сложных аккордов заключается в том, что музыкальные ноты также содержат частоты, кратные основной частоте, известной как обертоны или гармоники. Например, «Ля малой октавы» при частоте 220 Гц имеет гармоники на 440 Гц, 660 Гц, 880 Гц, 1100 Гц и так далее. Чем больше гармоник содержит инструмент, тем он кажется «богаче» на слух. Большинство частот образуют интервал «октава» с основным тоном, которые не мешают распознаванию аккорда, но также в обертоновом ряду присутствуют «квинта», «большая терция» и «малая септима» так, что первые 6 гармоник образуют мажорное трезвучие, а первые 8 – малый мажорный септаккорд. При этом в аккорде звучит более одной ноты, и каждая имеет свою последовательность обертонов.

Даже когда мы имеем идеальную запись без шумов и дополнительных инструментов, нельзя напрямую выбрать все частоты нот и провести соответствие со словарем аккордов.

Так как современная стандартная настройка инструментов неидеальна, можно попытаться находить частоты, которые отклоняются от равномерно-темперированного строя, и не учитывать их. Однако в записи присутствуют такие инструменты, как гитара, которую часто невозможно идеально настроить, и тогда нужные ноты будут теряться.

Другой вариант – нахождение основного тона, как тона с самой низкой частотой, и игнорирование тонов, которые могли бы является его обертонами, так как мы можем их рассчитать. Этот вариант тоже не подходит, так как в современной музыке основной тон аккорда дублируется на 1 или 2 октавы ниже. И в данном случае будут удаляться уже основные тона аккорда, так что точность распознавание будет понижаться.

В данной разработке был использован третий вариант. Было принято решение распознавать лишь мажорные и минорные трезвучия путем накопления интенсивности каждого частотного класса. Частотный класс представляет собой множество всех частот ноты на каждой октаве. Например «До первой октавы» и «До второй октавы» относятся к одному частотному классу.

Разработка алгоритма распознавания аккордов. Идея разделять частоты нот на частотные классы для распознавания аккордов была предложена Т. Фуджишимой в 1999 г. [2]. В этой работе рассматривалось распознавание с использованием скрытых марковских моделей для учета последовательности, в которой аккорды звучат. Для этого способа нужно большой объем музыки с полной нотацией аккордов и времени, в течение которого они звучат. Также способ не сильно повышал точность распознавания, если использовались отличные от обучающей выборки группы или жанры музыки.

Наиболее близкий к примененному в данной работе алгоритм был рассмотрен в работе Кристофа Хауснера [3].

Разработанная авторами система предназначена не для использования в сценариях, работающих в режиме реального времени, то есть примененный алгоритм сначала считывает данные всей песни, а затем выводит все аккорды. Обнаружение аккордов в реальном времени считается сложной задачей, поскольку аккорды необходимо оценивать «на лету» без информации о нескольких следующих моментах времени.

Набор аккордов, оцениваемых данной системой, ограничен 24 мажорными и минорными аккордами (C, Cm, C#, C#m, D, ..., B, Bm) и символом NC (no chord – если считается, что аккорд отсутствует). Такой довольно ограниченный набор объясняется несколькими причинами. Во-первых, это, безусловно, самые распространенные аккорды, присутствующие в популярной музыке. Во-вторых, точность обнаружения значительно падает, когда распознаются слишком много аккордов. Кроме того, часто можно функционально заменить более сложные аккорды простым мажорным или минорным аккордом, имеющим общий основной тон (например, Gm7 на Gm, Dmaj7 на D и т. д.).

Рассмотрим стадии преобразования сигнала (см. рис. 2).

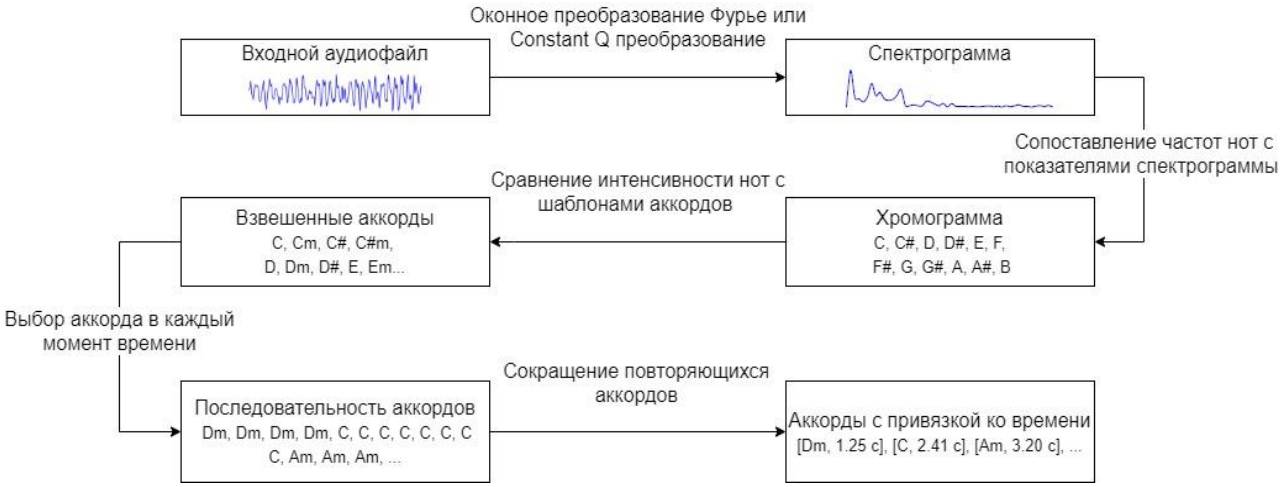


Рис. 2. Стадии преобразования сигнала для распознавания аккордов.

Алгоритм принимает на вход несжатые аудиофайлы в формате*.wav. На выходе алгоритма получается список предполагаемых аккордов и их продолжительность. Необработанные входные данные очень избыточны, поэтому, учитывая, что нас интересуют только аккорды, и нет способа извлечь эту информацию непосредственно из аудиофайла, было принято решение последовательно преобразовывать эти данные и затем распознавать в них частоты, отвечающие за конкретные аккорды.

После поступления аудиофайла он с помощью оконного преобразования Фурье либо преобразования постоянной добротности преобразуется в данные спектрограммы [4].

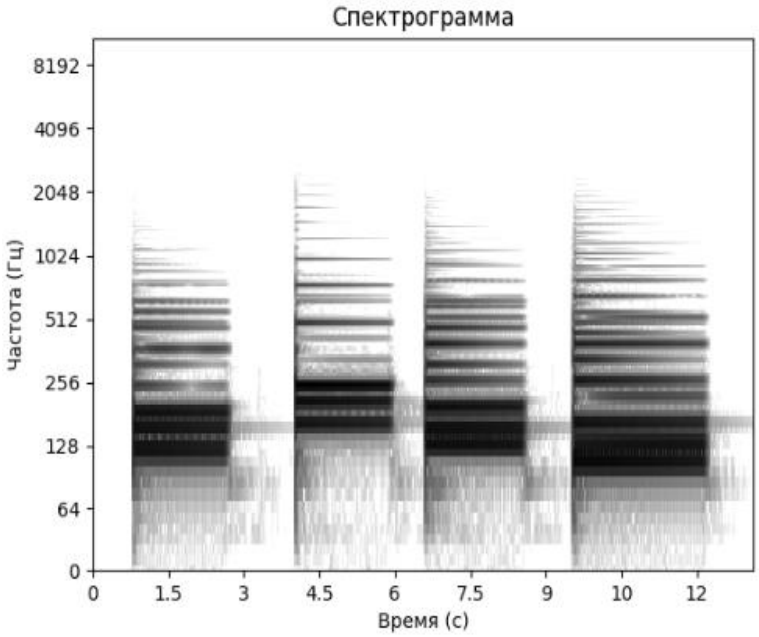
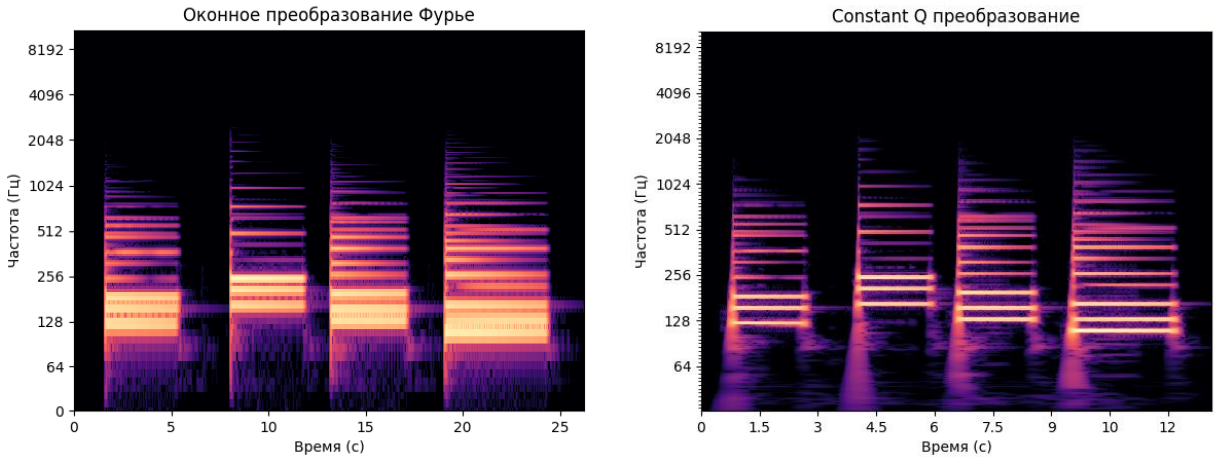


Рис. 3. Построенная диаграмма спектрограммы.

В данном случае спектрограммой называется двумерная структура данных, одна ось которой представляет измерение времени, а другая – измерение частоты. Значение каждого элемента структуры, показывает интенсивность частоты (см. рис. 3).

В связи с тем, что частоты нот расположены в логарифмическом ряду, при обычном оконном преобразовании Фурье разрешение на низких частотах ниже, чем на высоких. Поэтому было принято решение использовать преобразование Constant Q (см. рис. 4) [5].



а) б)

Рис. 4. Сравнение оконного преобразования Фурье (а)
и Constant Q преобразования (б).

Следующим шагом алгоритма является вычисление так называемой хромограммы.

Как и спектрограмма, хромограмма представляет собой двумерную структуру, однако вместо частотного спектра в ней содержится информация об интенсивности 12 классов частот. Каждый класс соответствуют 12 нотам хроматической гаммы: C, C#, D, D#, E, F, F#, G, G#, A, A#, B. При этом не игнорируется номер октавы. Например, частоты нот C1 и C4 соответствуют одному и тому же классу, и их интенсивности усредняются (см. рис. 5).

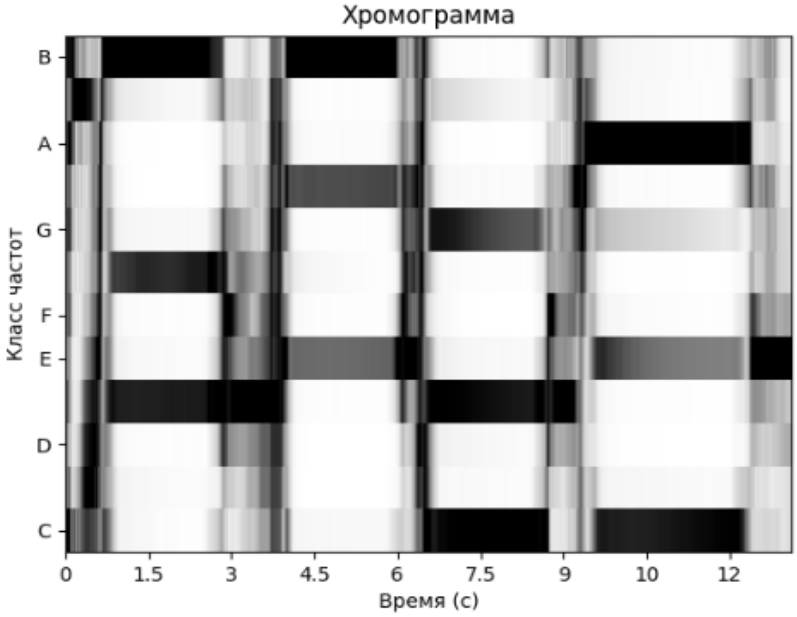


Рис. 5. Построенная диаграмма хромограммы.

После получения классов хромограммы есть возможность получить аккорды, которые состоят из данных классов. В хромограмме присутствует множество обертонов, а также тонов, не относящихся к гармоническому инструменту, например вокал. Поэтому, во-первых, необходимо рассмотреть, насколько хорошо данные хромограммы в каждый момент времени подходят под шаблон каждого аккорда, а во-вторых, рассматривать небольшое количество аккордов. В данном случае будут использоваться простые минорные и мажорные аккорды.

В итоге список шаблонов, будет состоять из 24 аккордов и обозначения «NC» – отвечающие за случай, когда ни один аккорд не подходит: C, Cm, C#, C#m, D, Dm, D#, D#m, E, Em, F, Fm, F#, F#m, G, Gm, G#, G#m, A, Am, A#, A#m, B, Bm, NC.

Шаблон для каждого аккорда будет состоять из 12 весовых коэффициентов, каждый из которых равен 1 либо 0, в зависимости от того входит ли данная нота в состав аккорда. Для обнаружения отрывков без аккордов (N.C.) используется шаблон, в котором весовые коэффициенты всех частотных классов равны 1 (см. табл. 1).

Структуры аккордовых шаблонов занесены в таблицу 1.

Таблица 1

Структуры аккордовых шаблонов

Название аккорда	Классы частот											
	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
C	1	0	0	0	1	0	0	1	0	0	0	0
Cm	1	0	0	1	0	0	0	1	0	0	0	0
C#	0	1	0	0	0	1	0	0	1	0	0	0
C#m	0	1	0	0	1	0	0	0	1	0	0	0
D	0	0	1	0	0	0	1	0	0	1	0	0
Dm	0	0	1	0	0	1	0	0	0	1	0	0
D#	0	0	0	1	0	0	0	1	0	0	1	0
D#m	0	0	0	1	0	0	1	0	0	0	1	0
E	0	0	0	0	1	0	0	0	1	0	0	1
Em	0	0	0	0	1	0	0	1	0	0	0	1
F	1	0	0	0	0	1	0	0	0	1	0	0
Fm	1	0	0	0	0	1	0	0	1	0	0	0
F#	0	1	0	0	0	0	1	0	0	0	1	0
F#m	0	1	0	0	0	0	1	0	0	1	0	0
G	0	0	1	0	0	0	0	1	0	0	0	1
Gm	0	0	1	0	0	0	0	1	0	0	1	0
G#	1	0	0	1	0	0	0	0	1	0	0	0
G#m	0	0	0	1	0	0	0	0	1	0	0	1
A	0	1	0	0	1	0	0	0	0	1	0	0
Am	1	0	0	0	1	0	0	0	0	1	0	0
A#	0	0	1	0	0	1	0	0	0	0	1	0
A#m	0	1	0	0	0	1	0	0	0	0	1	0
B	0	0	0	1	0	0	1	0	0	0	0	1
Bm	0	0	1	0	0	0	1	0	0	0	0	1
NC	1	1	1	1	1	1	1	1	1	1	1	1

Обозначим через a – вектор интенсивностей частотных классов в единичный момент времени, b – вектор весовых коэффициентов для определенного шаблона аккорда. Чтобы оценить, насколько вектор a соответствует вектору b, нужно найти коэффициенты,

показывающие степень совпадения прозвучавшего аккорда и шаблона. В работе Хауснера [3] было предложено использовать косинусное сходство, которое определяется следующей формулой:

$$\cos(\theta) = \frac{(a,b)}{|a|\cdot|b|} = \frac{\sum_{i=1}^{n} a_i \cdot b_i}{\sqrt{\sum_{i=1}^{n}(a_i)^2} \cdot \sqrt{\sum_{i=1}^{n}(b_i)^2}},$$

где $n = 12$, $a_i -$ интенсивность i-го частотного класса в единичный момент времени, $b_i -$ весовой коэффициент i -го частотного класса для определенного шаблона аккорда.

Преимущество данного выражения состоит в том, что его значения изменяются в диапазоне от 0 до 1.

После применения косинусного сходства для вектора хромограммы в каждый момент времени будет получена двумерная структура, в которой будут отображаться коэффициенты, показывающие насколько совпадает прозвучавший аккорд с шаблонами эталонных аккордов в каждый момент времени.

После этого нужно найти, для какого аккорда коэффициент будет максимальным во все моменты времени. Затем сократить повторяющиеся в последовательности аккорды, записывая время начала звучания аккорда. В композиции с несколькими партиями неизбежно будут появляться лишние ноты, и такая комбинация частот будет распознаваться как другой аккорд. Однако в большинстве случаев такие аккорды будут длиться достаточно короткий промежуток времени.

Алгоритм может эффективно применяться для музыки с относительно простым аккомпанементом и не слишком быстрым темпом, поэтому целесообразно отбрасывать аккорды, которые длятся менее половины секунды. Также необходимо повторно применить функцию, которая уберет дубликаты записей аккордов так, чтобы длительность их звучания сохранялась.

Максимально точно распознавание проходит в тех случаях, когда все ноты аккорда звучат одновременно. При усложнении звука и наполнении его другими партиями точность алгоритма будет падать. Поэтому перед применением алгоритма была использована обученная модель Spleeter от компании Deezer [6]. Она позволяет разделять музыкальный трек на стэмы – группы музыкальных инструментов.

Реализация алгоритма. Для реализации системы автоматического распознавания аккордовой последовательности в цифровых аудиофайлах на основе вышеизложенного алгоритма был выбран язык программирования Python. В нем присутствуют такие библиотеки как NumPy, MatPlotLib, помогающие в работе с математическими операциями и построением графиков. В пакете Librosa существует много удобных функций и инструментов для работы с звуком, например, функция, выполняющая оконное преобразование Фурье, а также структура

для постройки спектрограммы, автоматически сопоставляющая ноты и частоты.

Первым этапом работы системы было получение спектрограммы сыгранного аккорда (см. рис. 6).

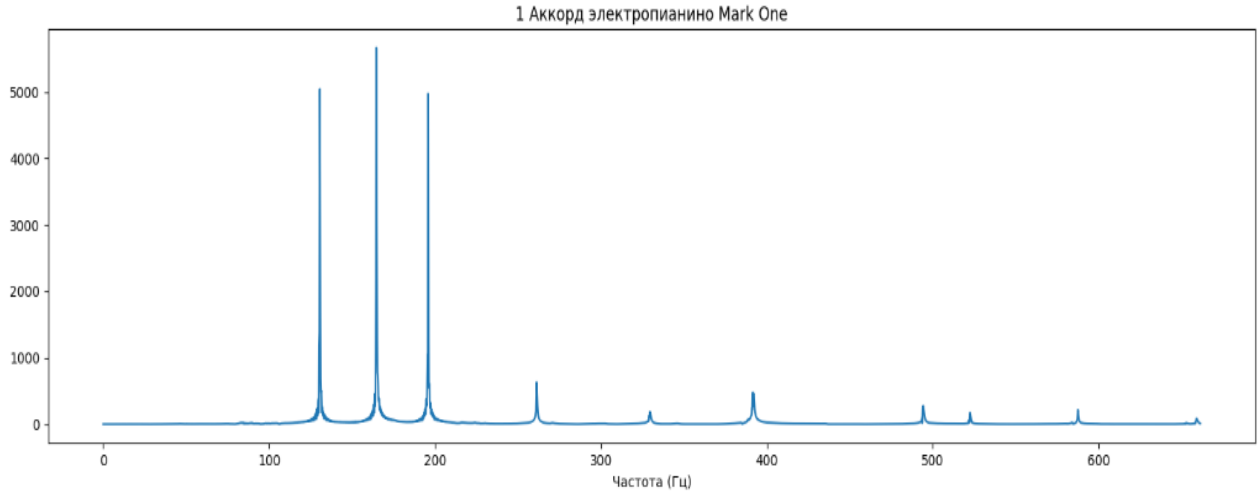


Рис. 6. Спектрограмма аккорда, сыгранного на электропианино.

Можно заметить, что на графике легко можно выделить три основных тона аккорда, а гармоники выражены не сильно. Это связано с тем, что для тестирования работы алгоритма было выбрано электропианино, имеющее более мягкий и джазовый звук.

Затем был построен график хромограммы, где отображаются не частоты, а интенсивности отдельных частотных классов (см. рис. 7).

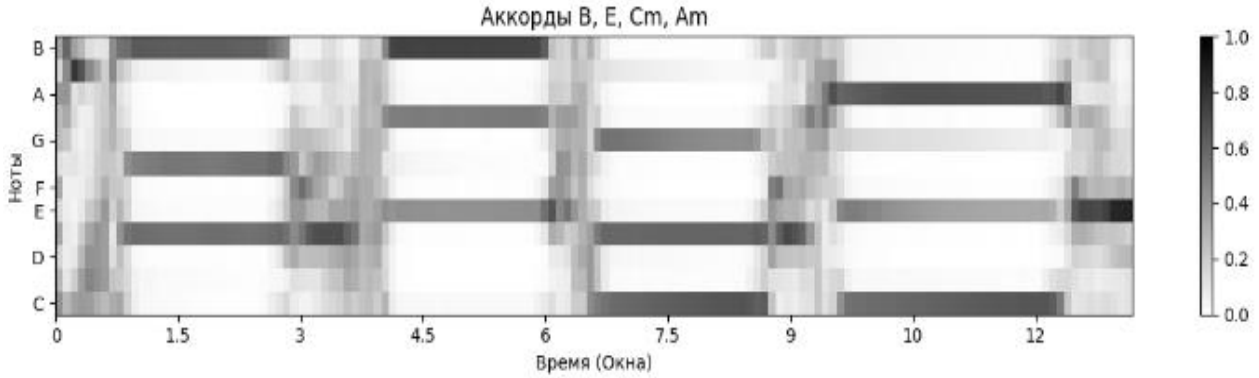


Рис. 7. Хромограмма последовательности аккордов, сыгранного на электропианино.

Благодаря пакету Librosa нет необходимости вручную прописывать коррекцию настройки инструмента и отображать интенсивности частот в интенсивности частотных классов. Поэтому далее можно найти сходство интенсивности нот в каждый промежуток времени с шаблонами аккордов. При этом получится последовательность аккордов (см. рис. 8).

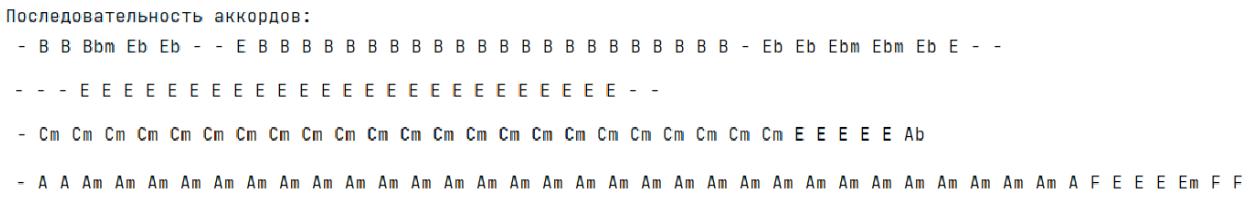


Рис. 8. Вывод последовательности аккордов.

Во время смены аккордов в коротких промежутках времени неизбежно распознавание неправильных аккордов. Поэтому нужно сократить повторяющиеся аккорды, присвоить каждому аккорду начало его звучания и удалить аккорды, звучащие небольшое количество времени (см. рис. 9).

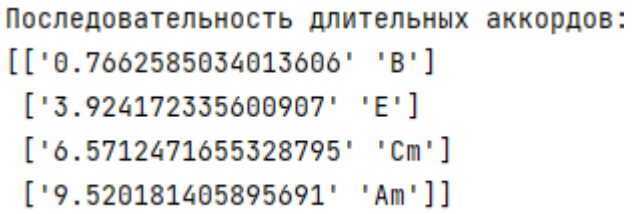


Рис. 9. Вывод последовательности длительных аккордов.

В результате работы алгоритма получается правильная последовательность сыгранных аккордов.

Описание разработанной системы. Чтобы решить проблему с конфликтующими версиями библиотеки (к примеру, Spleeter имеет в зависимостях более старую версию пакета Librosa, по сравнению с той, что используется в проекте) в данном случае было принято решение использовать систему управления пакетами в виртуальной среде Conda. Была создано виртуальное окружение в Conda с названием «ChordRec» для версии Python 3.8, и в ней запущены команды для установки пакетов: PyQt5, Librosa, Numpy, Matplotlib и Spleeter. В итоге все библиотеки и их зависимости были установлены. Для разработки интерфейса была использована программа QtDesigner. Интерфейс состоит из двух окон – главного (см. рис. 10) и окна с отображением хромограммы (см. рис. 11).

На рис. 10-11 продемонстрирована работа приложения на записанном тестовом аудиофайле (4Chords.wav) – четырех сыгранных аккордах на фортепиано.

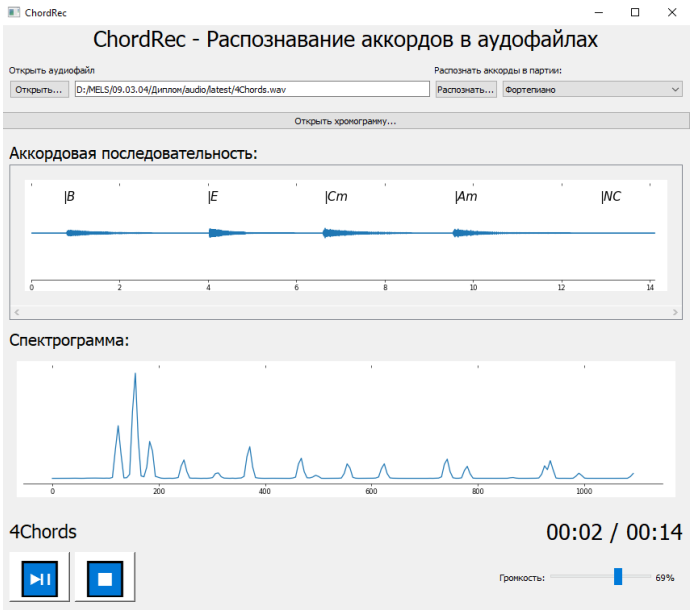


Рис. 10. Демонстрация работы главного окна приложения.

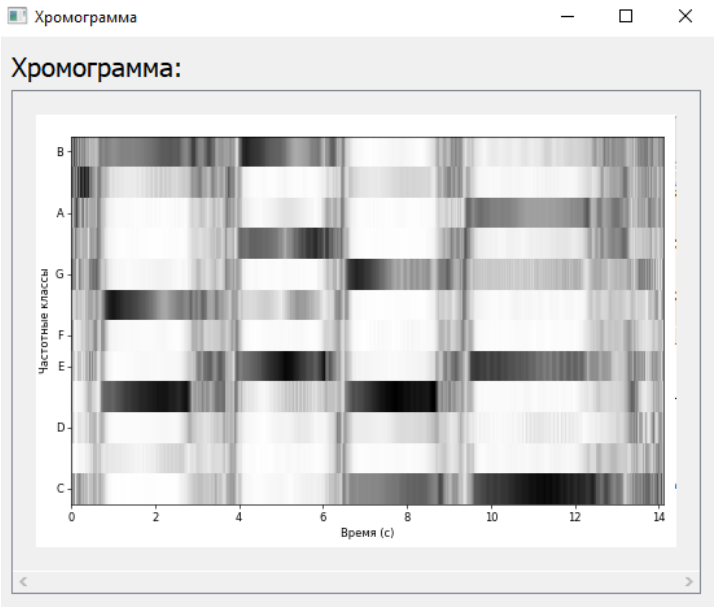


Рис. 11. Демонстрация работы отображения хромограммы.

Для повышения точности распознавания аккордов в системе присутствует функция разделения аудиофайлов на отдельные составляющие.

Пакет Spleeter предоставляет три варианта разделения музыкального аудиофайла. При этом осмысленно определять аккорды лишь в некоторых компонентах. В группе «вокал / аккомпанемент» – в аккомпанементе, в группе «вокал / барабаны/ бас /все остальное» – во всем остальном, а в группе «вокал / барабаны / бас / фортепиано / все остальное» – в фортепиано.

В разрабатываемой системе это можно сделать, выбрав в выпадающем меню

необходимую группу инструментов и нажав кнопку «Распознать…» (см. рис. 12).

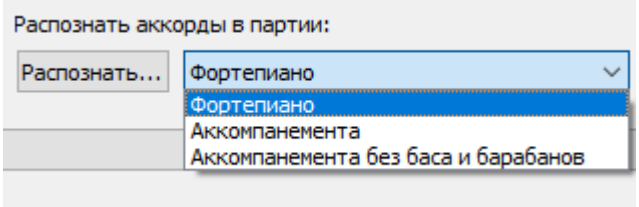


Рис. 12. Выпадающее окно для распознавания аккордов
в определенных партиях аккомпанемента.

При выборе группы инструментов программа предложит пользователю выбрать папку для сохранения стэмов. В папке сохранятся компоненты аудиофайла в соответствии с выбранным вариантом разделения. Сразу после этого нужный аудиофайл автоматически загрузится в систему.

Заключение. Рассмотренная в статье система умеет достаточно точно распознавать аккорды в аудиофайлах, в которых присутствует только звук фортепиано, и все ноты звучат одновременно. В системе присутствует пакет для предварительной фильтрации сигнала путем выделения сигнала определенного музыкального инструмента. Тем не менее данный подход не позволит распознавать аккорды сложнее, чем стандартные трезвучия. Проблема заключается в большом количестве обертонов в звуках разных инструментов. В дальнейшем можно попытаться определять гармоники для разных музыкальных инструментов при помощи машинного обучения [7,8].

СПИСОК ЛИТЕРАТУРЫ

1. Sisk P., Hughes B., Paul R., Aggarwal A., Patankar M. Chordination Detection Algorithm // University of Michigan [Электронный ресурс]. – Режим доступа: http://www-personal.umich.edu/~rbpaul/ (дата обращения: 11.03.2021)
2. Fujishima T. Realtime Chord Recognition of Musical Sound: a System Using Common Lisp Music // Proc. ICMC. – 1999. – pp. 464–467 [Электронный ресурс]. – Режим доступа: https://ci.nii.ac.jp/naid/10013545881/en/ (дата обращения: 14.12.2020)
3. Hausner C. Design and Evaluation of a Simple Chord Detection Algorithm // University of Passau, Faculty of Computer Science and Mathematics. – 2014. – 59 p [Электронный ресурс]. – Режим доступа:https://www.fim.uni-passau.de/fileadmin/dokumente/fakultaeten/fim/lehrstuhl/sauer/geyer/BA_MA_Arbeiten/BA-HausnerChristoph-201409.pdf (дата обращения: 11.10.2020).
4. Lenssen N., Needell D. An Introduction to Fourier Analysis with Applications to Music // Journal of Humanistic Mathematics. – 2014. – Vol. 4. – pp. 72–91 [Электронный

ресурс]. – Режим доступа:https://www.researchgate.net/publication/314918556_An_Introduction_to_Fourier_Analysis_with_Applications_to_Music (дата обращения: 14.12.2020).

5. Brown Judith C. Calculation of a constant Q spectral transform // Journal of the Acoustical Society of America. – 1991. – Vol. 89(1). – pp. 425–434 [Электронный ресурс]. – Режим доступа: http://academics.wellesley.edu/Physics/brown/pubs/cq1stPaper.pdf (дата обращения: 12.03.2021).
6. Deezer Research – Spleeter [Электронный ресурс]. – Режим доступа: https://research.deezer.com/projects/spleeter.html (дата обращения: 25.04.2021).
7. Глазырин Н. Ю. Алгоритмическое распознавание аккордов в цифровом звуке: автореф. дисс. … канд. физ.-мат. наук. – Екатеринбург, 2015. – 22 с [Электронный ресурс]. – Режим доступа: https://elar.urfu.ru/handle/10995/27846 (дата обращения: 12.12.2020).
8. Дудырев Е. О. Исследование методов распознавания аккордов, основанных на машинном обучении [Электронный ресурс] // Всероссийский форум научной молодежи «Богатство России»: Сборник докладов, Москва, 4–6 дек. 2017 г. – 2018. – С. 56–57. – Режим доступа: https://elibrary.ru/item.asp?id=32874106 (дата обращения: 06.04.2021).

КОРЫТИН С. И., МАМЕДОВА Т. Ф.

МОДЕЛИРОВАНИЕ ПРОЦЕССА ОПТИМИЗАЦИИ СТРУКТУРЫ ПРОИЗВОДСТВА

Аннотация. В статье рассматривается задача построения математической модели для расчета оптимальной структуры выпуска продукции и соответствующей структуры цен, поддерживающих максимальный темп роста производства. Оптимальная траектория экономического роста производства определяется как долгосрочное состояние равновесия рынка.

Ключевые слова: математическая модель, оптимизация, функция Лагранжа, темпы роста производства, теория спроса и предложения.

KORYTIN S. I., MAMEDOVA T. F.

MODELING THE PRODUCTION STRUCTURE OPTIMIZATION PROCESS

Abstract. The problem of constructing a mathematical model for calculating the optimal structure of output and the structure of prices supporting the maximum rate of growth of production is considered in the article. The optimal trajectory of economic growth in production is defined as a long-term state of equilibrium in the market.

Keywords: mathematical model, optimization, Lagrange function, production growth rates, supply and demand theory.

Актуальность проблемы. Оптимальное распределение имеющихся ресурсов – основная задача менеджмента производства. Основной интерес связан с программами, которые помогают обеспечит оптимизацию. Условия оптимальности этих программ, построенные на математических законах, дают возможность изучить траекторию экономического роста как ряд отдельных равновесных состояний рынка, при котором объем предложения соответствует объему спроса.

Спрос – это выражение потребности участниками рынка, представляющее собой способность и желание приобретать экономические блага.

Предложение – это отражение поведения товаропроизводителя на экономическом рынке, его готовность предложить (произвести) определенное количество товара за ограниченный период времени при обозначенных условиях. Предложение характеризуется ценой, количество экономических благ (часто им выступают деньги), за которые продавец (покупатель) готов произвести продажу (покупку) товара.

Одним из важнейших факторов, влияющих на спрос, является цена. Наряду с данным компонентом, изменение спроса каждого товара отражает, например, возможность у покупателя удовлетворить ту же потребность другим, близким по назначению продуктом.

Из-за того, что стремления продавцов и потребителей удовлетворяются с помощью друг друга, они встречаются в экономических взаимоотношениях, образуется так называемый рынок – экономическая площадка взаимодействий между покупателями и поставщиками товара, в ходе которых экономическими агентами совершаются сделки.

При оптимальности траектории пропорции спроса, предложения и цен на большей ее части сохраняются постоянными, если экономический рост рассматривать в долгосрочной перспективе. Оптимальная траектория экономического роста производства определяется как долгосрочное состояние равновесия рынка [1].

Постановка задачи. Рассмотрим производителя, который использует $x = \{x_1, \dots, x_m\}$ ресурсов для производства s видов товара $q = \{q_1, \dots, q_s\}$. Преобразование исходных ресурсов в товар описывается технологической функцией производителя $\varphi^n(q, x)$. С помощью этой функции записываются ограничения, которые описывают область допустимых связей между сырьем и товаром

$$\varphi(q, x) = 0.$$

Процедура оптимизации производства осуществляется в два этапа. На первом этапе определяется оптимальный вектор x^* для используемых ресурсов, при котором получаемый «товарный» вектор q^* имеет минимальные затраты [2; 3].

Если цена исходных продуктов $c_1, \dots, c_m$, то стоимость сырья определяется по формуле:

$$C = \sum_{i=1}^{m} c_i x_i.$$

Таким образом, на первом этапе решается следующая задача:

$$C(c, x) = \sum_{i=1}^{m} c_i x_i \overset{x}{\Rightarrow} min, \quad \varphi(q, x) = 0.$$

Для ее решения введем функцию Лагранжа производителя:

$$L(x, c, q, \theta) = C(c, x) - \mu\varphi(q, x).$$

Условия стационарности функции Лагранжа по прямым переменным x и двойственной переменной μ приобретают следующий вид

$$c_i + \mu \frac{\partial \varphi^n(q,x)}{\partial x_i} = 0;\ \ i = 1, \dots, m;\ \ \varphi(q,x) = 0\,.$$

Эти уравнения определяют оптимальные объемы сырьевых компонентов x^* в зависимости от объемов товаров q и уровней цен c, т. е.

$$x^* = g(q,c).$$

В результате получаем функцию затрат для производителя:

$$C(q,c) = \sum_{i=1}^{m} c_i g_i(q,s).$$

На втором этапе предполагается, что производитель выбирает свой вектор производства q таким образом, чтобы максимизировать свою прибыль.

Прибыль П определятся как разница между общей выручкой от продаж всех товаров и всеми производственными затратами:

$$\Pi(q,c,p) = \sum_{i=1}^{s} p_i q_i - C(q,c) > 0,$$

где p – цена продаж.

Если нет никаких ограничений на величину или норму прибыли (процент от выручки), то условия максимума прибыли определяются следующей системой уравнений

$$p_i - \frac{\partial C(q,c)}{\partial q_i} = 0, \qquad (1)$$

$$-\frac{\partial^2 C(q,c)}{\partial q_i \partial q_j} < 0; \qquad i,j = 1, \dots, s. \qquad (2)$$

Заметим, что в условие (1) производные

$$\frac{\partial C(q,c)}{\partial q_i},$$

представляют собой по смыслу некоторые предельные цены производства товара. Тогда из первого условия следует, что цена продаж должна равняться предельной цене товара.

Решая уравнения (1) относительно количеств $q_1, \dots, q_s$ производимых товаров, получим оптимальное, с точки зрения производителя, предложение товаров на рынок. Оно, как нетрудно заметить, будет зависеть только от цен приобретения сырья и цен продаж:

$$\widetilde{g_i} = Y_i(p, c), \qquad i = 1, \dots, s.$$

Рассмотрим поведение производителя в период времени t и сформулируем задачу для поиска максимального темпа роста производства при указанном распределении выпуска $q_{i,t-1}$.

Найти $p_{i,t}$, $q_{i,t}$ при $i = 1,2, \dots, m$ и λ такие, что

$$\begin{cases} \sum_{i=1}^{m} p_{i,t} q_{j,t} \le c_{j,t} q_{j,t-1}\ , & j = 1,2, \dots, m; \\ \sum_{i=1}^{m} c_i q_{j,t} \le q_{i,t-1}\ , & j = 1,2, \dots, m; \\ q_{j,t} \ge \lambda\, q_{i,t-1}, & j = 1,2, \dots, m. \end{cases} \tag{3}$$

где $p_i -$ набор цен на товары проиводителя, λ – скорость роста экономики производителя.

Решение задачи (3) позволяет найти оптимизированное распределение ресурсов производства по наборам товаров и вычислить цены, обеспечивающие максимальный рост производства. Так же найденная скорость роста характеризует сложившейся на момент времени t объем выпуска продукции и служит количественной характеристикой потенциала роста производства в состоянии q_i.

Предположим, что скорость выпуска продукции от периода к периоду неизменна. Тогда, используя модель (3) для отыскания максимального темпа роста и оптимальной структуры выпуска, получим

$$q_{j,t} = \lambda q_{j,t-1} = \lambda^{\,t} q_j; \quad j = 1,2, \dots, m.$$

Окончательно задачу можно сформулировать следующим образом.

Определить $\lambda > 0, \bar{q}$ и $\bar{p}$ такие, что выполняются условия:

$$\begin{aligned} &\lambda \sum_{j=1}^{m} c_i \bar{q}_j - \bar{q}_i \le 0; && i = 1, 2, \dots, m; \\ &\bar{p} - \lambda \sum_{i=1}^{m} c_i \bar{p}_i \le 0; && j = 1, 2, \dots, m; \end{aligned} \tag{4}$$

$$\bar{p}(\lambda \sum_{j=1}^{m} c_i \bar{q_j} - \bar{q_i}) = 0; \quad i = 1, 2, \dots, m;$$

где $\bar{q}$ −вектор оптимальных пропорций выпуска продукции, $\bar{p}$ − вектор оптимальных цен.

Для автоматизации представленной выше методики было создано программное обеспечение на языке программирования C++.

В качестве исходных данных рассматриваются результаты мониторинга цен на апельсиновый сок марки «Добрый» и данные, полученные из опроса потребителей. В результате опроса были получены данные, приведенные в таблице 1.

Таблица 1

Результаты опроса потребителей

№	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Цена, руб.	95	60	75	50	80	55	65	100	45	75	10	50	75	40	65	85	56	55	50	75
№	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40
Цена, руб.	40	50	85	40	45	60	35	90	45	75	80	50	55	30	85	50	75	80	75	85

В ходе анализа цен, за которые супермаркеты предлагают покупателям приобрести сок «Добрый» получены данные, представленные в таблице 2.

Таблица 2

Результат анализа цен в супермаркетах

№	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Цена, руб.	88	92	93	87	98	93	95	92	88	95	93	98	92	95	87	98	93	95	90	93
№	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40
Цена, руб.	92	93	90	90	95	98	93	95	87	95	98	92	95	93	90	95	93	87	93	88

Из расчетов программы, представленных на рис. 1, видно, что равновесный объем будет равен 7 и равновесие наступит при значении цены равной 88 руб.

Цена, руб.	Спрос, шт.	Предложение, шт.
87	7	4
88	7	7
90	6	13
92	4	19
93	4	22
95	3	29
98	1	38
100	0	44

Рис. 1. Зависимость спроса и предложения на товар от цены.

Для рассмотрения задачи оптимизации ресурсов на предприятии апельсинового сока предположим, что имеются следующие ресурсы и подсистемы производства, представленные на рис. 2.

Нормы расхода ресурсов	Объект производства 1	Объект производства 2	Объект производства 3	Запасы ресурсов
Ресурс 1, кг.	120	45	98	520
Ресурс 2, кг.	85	110	0	350
Ресурс 3, л.	155	0	75	360
Ресурс 4, шт.	12	26	18	180
Ресурс 5, шт.	38	66	120	490

Рис. 2. Нормы расходов ресурсов для производства апельсинового сока.

На рис. 2 представлены нормы расходов ресурсов необходимых для соответствующих объектов производства и запасы этих ресурсов на предприятии

С каждого объекта производства предприятие в целом получает следующие доходы, представленные на рис. 3.

	Объект производства 1	Объект производства 2	Объект производства 3	Доход общий	Объем общий
Доход с объекта, руб.	28	25	12	65	
Объем, л.	1	2	2		5

Рис. 3. Доходы с объектов производства и объемы объектов производства.

Изучив рынок и смоделировав равновесный объем, добавим его в ограничение производства, исходя из того, что количество выпускаемых объемов не может быть больше равновесного. На основе этого ограничения будет производиться поиск оптимального распределения ресурсов между подсистемами производства.

В результате работы программы было получено распределение ресурсов, представленное на рис. 4.

Задействовано ресурсов	
Ресурс 1, кг.	406
Ресурс 2, кг.	305
Ресурс 3, л.	305
Ресурс 4, шт.	100
Ресурс 5, шт.	410

Рис. 4. Необходимое количество соответствующих ресурсов задействованных для оптимального производства товара.

Заключение. Для нахождения оптимального плана распределения ресурсов необходимо учитывать состояние и поведение рынка на промежутке времени (в динамике), что и отражает нахождение равновесного объема за прошедший период реализации товара, и применение этой величины для нахождения объемов производства уже на будущий этап продаж. В ходе работы созданной программы удалось получить план производства апельсинового сока, который удовлетворяет принципам и идее, построенным в ходе описания математической модели задачи.

СПИСОК ЛИТЕРАТУРЫ

1. Бахтизин А. Р. Вычислительная модель «Россия: Центр – Федеральные округа». – М.: ЦЭМИ РАН, 2003. – 134 с.
2. Симонов П. М. Экономико-математическое моделирование. Динамические модели экономики: учеб. пособие: в 2 ч. – Пермь: Пермский гос. ун-т, 2009. – 274 с.
3. Мамедова Т. Ф., Каледин О. Е., Шабанова В. Г., Кирейчева Е. Ю. Математическая модель оптимизации управления хозяйственной деятельностью одного производственного предприятия // Аналитические и численные методы моделирования естественно-научных и социальных проблем : сб. ст. X Междунар. науч.-техн. конф. (г. Пенза, 28−30 октября 2015 г.) / под ред. И. В. Бойкова. – Пенза: ПГУ, 2016. – С. 125–130.

КУЗНЕЦОВА О. А., ГУЩИНА О. А.

ОБЗОР ПРОГРАММНЫХ СРЕДСТВ РЕАЛИЗАЦИИ БИОНИЧЕСКИХ АЛГОРИТМОВ

Аннотация. В статье рассматриваются бионические алгоритмы и их виды. Также представлен обзор программных средств для реализации бионических алгоритмов. Продемонстрирован пример одного из бионических методов – генетический алгоритм.

Ключевые слова: бионические методы, эволюционные алгоритмы, роевой интеллект, генетический алгоритм, принципы интеллектуальных алгоритмов.

KUZNECOVA O. A., GUSHCHINA O. A.

REVIEW OF SOFTWARE FOR IMPLEMENTATION OF BIONIC ALGORITHMS

Abstract. The article discusses bionic algorithms and their types. An overview of software tools for the implementation of bionic algorithms is also presented. An example of one of the bionic methods is demonstrated - the genetic algorithm.

Keywords: bionic methods, evolutionary algorithms, swarm intelligence, genetic algorithm, principles of intelligent algorithms.

1. Суть бионических методов. На протяжении многих веков человек, взаимодействуя с окружающим его природным миром, постигал законы функционирования живых существ. Именно такие законы легли в основу создания бионических методов. Эти методы становятся мощным инструментом для различного рода задач и принятия решения. Реализация множества вычислительных алгоритмов, которые базируются на таких методах, безусловно, можно считать одним из значительных направлений развития современной науки.

На сегодняшний день учёные выделяют две группы бионических алгоритмов [1-3].

Первая группа содержит в себе алгоритмы роевого интеллекта (РИ). Это алгоритмы, которые были созданы при помощи коллективного интеллекта или методом анализа поведения группы особей (роя) сообществ животных.

Ко второй группе принадлежат эволюционные вычисления. Это алгоритмы, которые основаны на принципах «выживания сильнейших» (так называемом «естественном отборе»).

1.1. Роевой интеллект. Под термином «рой» чаще всего понимается некоторое сообщество насекомых, объёдинённых какими-либо коллективными действиями (например, рой пчёл, ос, муравьём). Каждая особь роя двигается хаотично, т.е. каждая особь обладает непредсказуемым характером поведения. Простые правила, принятые внутри роя, которые имеют значение только внутри данного сообщества и не могут использоваться единичными особями вне роя послужили базисом для создания коллективного разума. Такое явление

получило название «роевый интеллект». Такой способ мышления помогает рою максимально эффективно использовать имеющиеся ресурсы и богатства окружающей его природной среды.

Если алгоритм удовлетворяет следующим принципам, то его можно отнести к интеллектуальным алгоритмам:

– принцип разнообразия: рой хранит все свои ресурсы в нескольких рассредоточенных локациях, а не в одном месте.

– принцип адаптируемости: в случае необходимости, рой может скорректировать своё поведение;

– принцип приближения: рой может выполнять простые вычисления (пространственные и временные);

– принцип стабильности: рой после любого изменения окружающей среды никогда не отклоняться от своей линии поведения;

– принцип качества: рой всегда реагирует на различные факторы качества окружающей среды, такие как безопасность местоположения, качество пищи и тому подобные.

Главными парадигмами роевых алгоритмов являются:

– метод роя частиц;
– муравьиный алгоритм;
– алгоритм искусственного роя пчёл и другие.

1.2. Эволюционные алгоритмы. Данная группа бионических алгоритмов основывается на третьей эволюционной теории Чарльза Дарвина. Данная теория объясняет принцип естественного отбора. Её суть заключается в том, что в любом сообществе при условии ограниченности необходимых жизненно важных ресурсов неизбежно возникновение конкуренции, которая влечет за собой к преобладанию наилучших (более сильных) особей сообщества над худшими (слабыми). Результатом является то, что продолжают функционировать («выживают») наилучшие, то есть наиболее адаптированные особи. Эволюционное развитие такого сообщества методом естественного отбора позволяет поддерживать многообразие видов различных существ в живой природе и их адаптируемость к внешним условиям. Соответственно, эволюционные алгоритмы основываются на процессах и моделях естественной биологической эволюции животного и растительного миров.

Эволюционные алгоритмы распределяют свойства адаптации по средствам итерационного процесса. Этот метод является методом проб и ошибок, но для живых

организмов он максимизирует качества особей в популяции, то есть в определенном смысле составляет перспективные наборы этих особей. Перспективными не всегда являются наилучшие варианты. Часто лучшее потомство дают «обыкновенные» особи.

В генетических алгоритмах вариации решения поставленной задачи можно представить с помощью некоторой виртуальной популяции, стремящейся к наилучшему значению фитнесс-функции (целевой функцией), аналогичной выживанию в живой природе.

Процесс повторяется многократно и соответствует смене поколений, то есть популяций особей в окружающем мире. Во многих случаях происходит не только улучшение популяции, но каждой отдельной особи.

На вышеописанных механизмах строятся генетические алгоритмы. Для их реализации наиболее часто используются соответствующие генетические операторы. Основными из них являются: мутация и рекомбинация.

Эволюционные (в частности генетические) алгоритмы позволяют успешно решать большой пласт сложноразрешимых оптимизационных задач в различных проблемных областях, для решения которых плохо подходят традиционные методы. Это может быть обусловлено: неопределенностью/неточностью/многозначностью аргументов и значения фитнесс-функции, а также непрерывностью пространства поиска.

Аналогичные рассуждения можно представить и для иммунных алгоритмов.

Бионические алгоритмы редки применяются «в чистом виде». Чаще всего для повышения продуктивности решения задачи и увеличения наглядности процесса решения их совмещают с какими-либо традиционными подходами решения оптимизационных задач (например: локальный поиск).

Бионические методы чаще всего используются для гибридных задач – задач, решение которых включает совокупность нескольких разнообразных и разнотипных подзадач. Например, одни из них могут быть на нахождение локального оптимума, другие – содержат динамические ограничений так далее.

Эволюционных методы базируются на следующих основополагающих парадигмах:

- генетические алгоритмы;
- генетическое программирование;
- эволюционное программирование;
- дифференциальная эволюция.

2. Обзор программных средств реализации бионический алгоритмов. Сделаем обзор различных средств реализации бионических алгоритмов. Рассмотрим несколько самых популярных на сегодняшний день языков программирования: C++, C# и Python.

2.1. Язык C++. C++ – алгоритмический, компилируемый, статически типизированный язык программирования общего назначения. Он основывается на следующих парадигмах программирования: процедурное программирование, объектно-ориентированное программирование, обобщённое программирование. C++ поддерживает обширную стандартную библиотеку, включающую такие возможности как: распространённые контейнеры и алгоритмы, ввод-вывод, регулярные выражения, поддержку многопоточности и многие другие особенности. C++ поддерживает свойства высокоуровневых и низкоуровневых языков программирования.

C++ широко применяется для разработки программного обеспечения. Он является самым универсальным и качественным языком программирования. Существует множество реализаций языка C++, как бесплатных, так и коммерческих. Он используется написания программного обеспечения для различных платформ (например: на платформе x86 – GCC, Visual C++, Intel C++ Compiler, Embarcadero (Borland) C++ Builder и пр.).

2.2. Язык C#. C# – объектно-ориентированный язык программирования семейства языков с C-подобным синтаксисом. Его синтаксис наиболее близок к синтаксису алгоритмических языков программирования C++ и Java. C# поддерживает статическую типизацию, поддерживает полиморфизм, перегрузку операторов и ещё большое количество полезных функции.

Переняв многое от C++, C#, опираясь на практику использования C++, исключил многие проблематичные модели при разработке программных комплексов. Например: C# в отличие от C++ не поддерживает множественное наследование классов, но допускает множественную реализацию интерфейсов.

2.3. Язык Python. Python – высокоуровневый алгоритмический язык программирования общего назначения с динамической строгой типизацией и автоматическим управлением памятью, ориентированный на повышение производительности разработчика, читаемости кода и его качества, а также на обеспечение переносимости написанных на нём программ. Python является полностью объектно-ориентированным, то есть в нем всё является объектами. Синтаксис ядра языка Python минималистичен, за счёт чего на практике редко возникает необходимость обращаться к документации. Python является интерпретируемым языком программирования. Часто он применяется для написания скриптов.

Python является мультипарадигмальным языком программирования, поддерживающим императивное, процедурное, структурное, объектно-ориентированное программирование, метапрограммирование и функциональное программирование. Задачи обобщённого программирования часто решаются за счёт динамической типизации.

Основные архитектурные черты языка – динамическая типизация, автоматическое управление памятью, полная интроспекция, механизм обработки исключений, поддержка многопоточных вычислений с глобальной блокировкой интерпретатора (GIL), высокоуровневые структуры данных. Поддерживается разбиение программ на модули, которые, в свою очередь, могут объединяться в пакеты.

Стандартная библиотека включает большой набор полезных переносимых функций, начиная от функционала для работы с текстом и заканчивая средствами для написания сетевых приложений. Дополнительные возможности могут реализовываться посредством обширного количества сторонних библиотек, а также интеграцией библиотек, написанных на Си или C++.

3. Пример реализации генетического алгоритма на языке программирования Python. Наиболее подходящим инструментом для реализации бионического алгоритма является язык программирования Python. На листинге 1 представлен пример программного кода, реализующий обобщённый генетический алгоритм.

```
from __future__ import annotations
from typing import TypeVar, Generic, List, Tuple, Callable
from enum import Enum
from random import choices, random
from heapq import nlargest
from statistics import mean
from chromosome import Chromosome

C = TypeVar('C', bound=Chromosome) #тип хромосом

class GeneticAlgorithm(Generic[C]):
    SelectionType = Enum("SelectionType", "ROULETTE TOURNAMENT")
    def __init__(self, initial_population: List[C], threshold: float, max_generations: int =
100,
                 mutation_chance: float = 0.01, crossover_chance: float = 0.7,
                 selection_type: SelectionType = SelectionType.TOURNAMENT) -> None:
        self._population: List[C] = initial_population
        self._threshold: float = threshold
        self._max_generations: int = max_generations
        self._mutation_chance: float = mutation_chance
```

```
        self._crossover_chance: float = crossover_chance
        self._selection_type: GeneticAlgorithm.SelectionType = selection_type
        self._fitness_key: Callable = type(self._population[0]).fitness
    # Используем метод рулетки с нормальным распределением,
    # чтобы выбрать двух родителей
    # Примечание: не работает при отрицательных значениях жизнеспособности
    def _pick_roulette(self, wheel: List[float]) -> Tuple[C, C]:
        return tuple(choices(self._population, weights=wheel, k=2))
    # Выбираем случайным образом num_participants и берем из них две лучших
    def _pick_tournament(self, num_participants: int) -> Tuple[C, C]:
        participants: List[C] = choices(self._population, k=num_participants)
        return tuple(nlargest(2, participants, key=self._fitness_key))
    # Замена популяции новым поколением особей
    def _reproduce_and_replace(self) -> None:
        new_population: List[C] = []
    # продолжаем, пока не заполним особями все новое поколение
        while len(new_population) < len(self._population):
        # выбор двух родителей
            if self._selection_type == GeneticAlgorithm.SelectionType.ROULETTE:
                parents: Tuple[C, C] = self._pick_roulette([x.fitness() for x in
                self._population])
            else:
                parents = self._pick_tournament(len(self._population) // 2)
        # потенциальное скрещивание двух родителей
            if random() < self._crossover_chance:
                new_population.extend(parents[0].crossover(parents[1]))
            else:
                new_population.extend(parents)
    # если число нечетное, то один лишний, поэтому удаляем его
        if len(new_population) > len(self._population):
            new_population.pop()
        self._population = new_population # заменяем ссылку
    # Каждая особь мутирует с вероятностью _titatiоп_chапce
    def _mutate(self) -> None:
```

```
            for individual in self._population:
                if random() < self._mutation_chance:
                    individual.mutate()
        # Выполнение генетического алгоритма для tax_generations итераций
        # и возвращение лучшей из найденных особей
            def run(self) -> C:
                best: C = max(self._population, key=self._fitness_key)
                for generation in range(self._max_generations):
                # ранний выход, если превышен порог
                    if best.fitness() >= self._threshold:
                        return best
                    print(f"Generation {generation} Best {best.fitness()} Avg
{mean(map(self._fitness_key, self._population))}")
                    self._reproduce_and_replace()
                    self ._mutate()
                    highest: C = max(self._population, key=self._fitness_key)
                    if highest.fitness() > best.fitness():
                        best = highest # найден новый лучший результат
        return best # лучший найденный результат из _tax_generations
```

Листинг 1. Реализация генетического алгоритма на языке Python

Представленная программа представляет собой изначальный шаблон, на основе которого можно реализовать решение любой оптимизационной задача. Для этого будет необходимо внести коррективы в нужные генетические операторы и откорректировать фитнесс-функцию. Этот шаблон прекрасно подходит для решения следующих проблем: оптимизации запросов в базах данных, задач на графах, настройки и обучения искусственной нейронной сети, задач компоновки, составления расписания, задач теории приближения, синтеза конечных автоматов и многих других задач.

Заключение. Бионические алгоритмы являются достаточно мощным инструментом современных учёных и практиков. В результате выполнения данного проекта была изучена предметная область использования алгоритмов оптимизации, основанных на принципах живой природы и процессе эволюции: рассмотрены общие принципы, лежащие в основе эволюционных вычислений и методов роевого интеллекта; выполнен обзор и проанализированы основные языки программирования для реализации бионических

алгоритмов; программно реализованы бионические алгоритмы с демонстрацией на листинге основной части генетического алгоритма на оптимальном для него языке Python.

СПИСОК ЛИТЕРАТУРЫ

1. Гладков Л. А., Курейчик В. В., Курейчик В. М., Сороколетов П. В. Биоинспирированные методы в оптимизации. – М.: ФИЗМАТЛИТ, 2009. – 384 с.
2. Семенкин Е. С., Жукова М.Н., Жуков В. Г., Панфилов И. А., Тынченко В. В. Эволюционные методы моделирования и оптимизации сложных систем. Конспект лекций. – Красноярск, 2007. – 515 с. [Электронный ресурс]. – Режим доступа: http://files.lib.sfu-kras.ru/ebibl/umkd/22/u_lectures.pdf/ (дата обращения: 16.05.2018).
3. Ashlock D. Evolutionary Computation for Modeling and Optimization. – Springer-Verlag. – Berlin. Germany, 2006. – 571 p.

ФИРСОВА С. А., ПАКШИН А. В.

ОРГАНИЗАЦИЯ ПРОЦЕССА АВТОМАТИЗИРОВАННОГО ТЕСТИРОВАНИЯ WEB-ПРИЛОЖЕНИЙ НА БАЗЕ ФРЕЙМВОРКА SELENIUM

Аннотация. В статье описана разработанная авторами программная система, позволяющая выполнять автоматизированное тестирование web-приложений на базе фреймворка SELENIUM. Подробно рассмотрена архитектура программной системы, описан процесс построения тестовых сценариев и генерации отчетов по проведенному тестированию, приведен пример тестирования конкретного web-приложения.

Ключевые слова: автоматизированное тестирование, тестовые сценарии, архитектура программной системы, UML-диаграмма, Selenium.

FIRSOVA S. A., PAKSHIN A.V.

ORGANIZATION OF THE PROCESS OF AUTOMATED TESTING OF WEB APPLICATIONS BASED ON THE SELENIUM FRAMEWORK

Abstract. The article describes a software system developed by the authors that allows automated testing of web applications based on the SELENIUM framework. The architecture of the software system is considered in detail, the process of building test scenarios and generating reports on the conducted testing is described, an example of testing a specific web application is given.

Keywords: automated testing, test scenarios, software system architecture, UML diagram, Selenium.

Введение. В современном мире программное обеспечение используется практически во всех сферах жизни, гигантские суммы тратятся на разработку разнообразных программ, востребованных в промышленности, бизнесе, индустрии развлечений, образовании и медицине [1]. Задача снижения стоимости разработки программного обеспечения и улучшения качества выпускаемой продукции является одной из наиболее актуальных в индустрии информационных технологий.

Автоматизация тестирования позволяет значительно сократить затраты компаний-разработчиков, сэкономить время и ресурсы, расходуемые на тестирование, снизить риск выпуска на рынок некачественного продукта. Поэтому технологии автоматизации тестирования набирают все большую популярность среди компаний, связанных с разработкой программных продуктов. Автоматизированное тестирование заключается в написании автотестов – программ, направленных на выполнение заложенных в них тестовых сценариев. Благодаря им разработчик может по истечению короткого периода времени получить информацию о том, были ли обнаружены в ходе выполнения данного сценария

ошибки или тест пройден успешно. Для разработки и запуска автотестов используются специальные системы автоматизированного тестирования. Главной идеей автоматизированного тестирования является полная замена человеческого труда. Программист единожды создает программу, которая проверяет все, подготовленные заранее, тестовые сценарии. Далее, сценарии только поддерживаются в актуальном рабочем состоянии. Создание таких программ позволяет снизить присутствие ручного тестирования в жизненном цикле разрабатываемого ПО [2].

Данный подход экономит время сотрудников отдела по контролю качества, а также ощутимо ускоряет работу команды разработчиков, и весь процесс разработки ПО в целом.

Недостатком автоматизации тестирования является то, что она применима только к работе с длинными проектами. Создание программы тестирования занимает много времени, и тратить это время на небольшие проекты просто бессмысленно. Как правило, автоматизированное тестирование эффективно применяется к относительно простым сценариям и тестам. Автоматизация сложных многошаговых сценариев может занять недопустимо много времени из-за непредвиденных нюансов или технических проблем [3]. Также может потребоваться много времени для поддержки тестов, при необходимости часто вносить в них правки после изменений в тестируемой функциональности. Несмотря на нюансы автоматического тестирования, этот тип не уступает, и компании внедряют его в свой рабочий процесс.

В данной статье рассматривается разработанная авторами программная система, предназначенная для построения тестовых сценариев и автоматизированного тестирования с их помощью различных web-приложений.

Разработка архитектуры программной системы. Архитектуру системы опишем в соответствии с рекомендациями Rational Unified Process. UML спецификация системы разделена на следующие части:

– Концептуальная модель. Данная модель описывает участников системы, основные сценарии и варианты использования системы;

– Логическая модель. Эта модель включает описание и диаграммы наиболее важных модулей системы, описания значимых классов, диаграммы последовательностей выполнения типовых задач и процессов приложения;

– Модель реализации. Эта модель включает описание разделения прототипа системы на отдельные компоненты, независимые задачи, подпрограммы, информационные и управляющие потоки и связи между элементами системы.

Концептуальная модель разработанной программной системы представлена диаграммой вариантов использования (см. рис. 1):

Рис. 1. Диаграмма вариантов использования.

На диаграмме вариантов использования показаны участники системы (актеры) и основные варианты работы системы (прецеденты):

– Тестировщик создает/редактирует сценарий тестирования web-приложения, используя возможности фреймворка Selenium Web Driver [4] и среды юнит-тестирования NUnit для языка C#. Сценарий сохраняется в виде исполняемого файла .exe для проведения последующих тестов.

– Программист запускает исполняемый файл, и система проводит автоматическое тестирование приложения. По завершении тестирования генерируется отчет, содержащий информацию о времени начала и завершения тестирования, количестве запущенных тестов, процент успешно пройденных тестов.

Логическая модель включает описание и диаграммы наиболее важных модулей системы, описания значимых классов, диаграммы последовательностей выполнения типовых задач и процессов приложения.

Диаграмма классов системы представлена на рисунке 2:

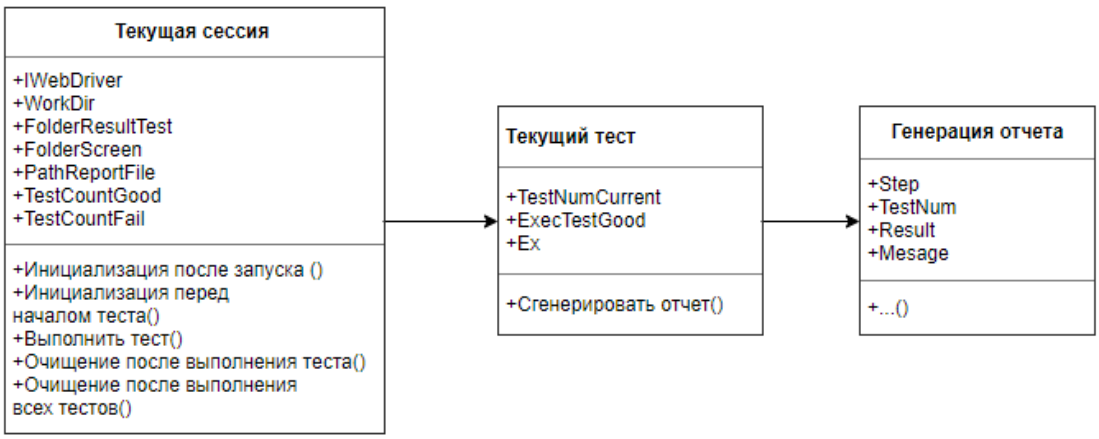


Рис. 2. Диаграмма классов.

Класс <Текущая сессия> содержит переменные для подсчета количества пройденных и не пройдённых тестов, адреса каталогов, а также методы инициации и проведения функционального тестирования:

– переменная <IWebDriver> служит указателем на интерфейс, через который пользователь управляет браузером;

– переменные <WorkDir>, <FolderResult>, <FolderScreen>, <PathReportFile> содержат информацию о путях расположения компонентов программы;

– переменные <TestCountGood>, <TestCountFail> являются счетчиками положительного и отрицательного прохождения тестов.

Класс <Текущий тест> содержит атрибуты, предоставляющие информацию о прохождении теста; инициирует и хранит такие переменные как:

– переменная <TestNumCurrent> хранит строку с наименованием протекающего тест-сценария.

– переменная <ExexTestGood> служит индикатором прохождения тестирования.

– переменная <Ex> является указателем на возникающее в случае выявления ошибки сообщение.

Класс <Генерация отчета> содержит атрибуты для создания отчета, инициирует и хранит такие переменные как:

– переменная <Step> которая является идентификатором этапа внесения информации: 0 – начало отчета, 1 – формирование отчета, -1 – завершение отчета.

– переменная <TestName> хранит строку с наименованием протекающего тест-сценария.

– переменная <Result> хранит результат прохождения тестирования: True-пройдено, False-не пройдено.

– переменная <Mesage> хранит строку с сообщением, выводимым при возникновении ошибки.

Модель реализации описывает разделение системы на отдельные компоненты, независимые задачи, подпрограммы, информационные и управляющие потоки и связи между элементами системы.

В разработанной программной системе в Visual Studio происходит компиляция кода на языке C#, который передается для исполнения утилите NUnit.consolerunner.

Скрипт теста содержит в себе команды, имитирующие взаимодействие пользователя с браузером Google Chrome при помощи драйвера Selenium WebDriver. При этом все внесенные или измененные данные записываются в существующую базу данных, связанную с тестируемым web-приложением. Диаграмма компонентов представлена на рисунке 3:

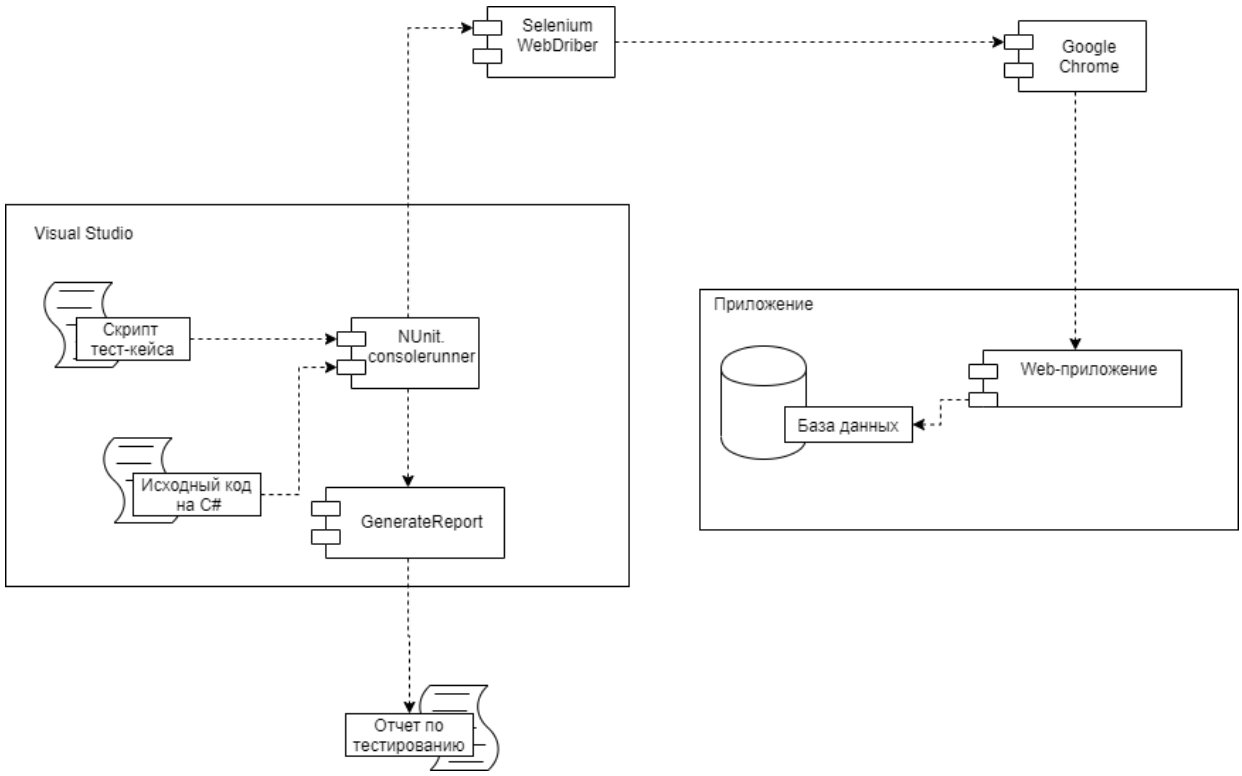


Рис. 3. Диаграмма компонентов.

Также во время выполнения тестирования, полученные результаты вносятся в формируемый при помощи метода <GeneraterReport> отчет, модель которого представлена на рис. 4:

Тест начат 20.09.2021 12:15:16

Тест	Результат	Время	Снимок	Сообщение
S1	Успех	12:15:18	-	
S2	Успех	12:15:23	-	
S3	Успех	12:15:43	open	

Всего тесто запущено: 3 || **Успешно: 2** || **Провалено: 1** || Процент пройденных тестов: 66% ||
Тест завершен: 20.09.2021 12:17:35

Рис. 4. Модель отчета.

Отчет представляет собой HTML-таблицу, в первой строке которой размещается время начала тестирования. Первый столбец хранит название тестовых сценариев, второй – отвечает за результат проведения тестирования (Успех и Провал), третий – хранит время воспроизведения тестового сценария, которое привязано к системным часам операционной системы. В четвертый столбец (в случае провала тестирования) выводится ссылка на сделанный в момент возникновения критической ситуации скриншот окна браузера. Последний столбец содержит сообщение, выводимое при возникновении ошибки тестирования. В конце отчета выводится информация об общем количестве проведенных

тестов, количестве успешных и проваленных тестов, а также процент успешно пройденных тест-сценариев, а также время завершения тестирования.

Программная реализация системы автоматизированного тестирования web-приложений. Система построена на основе NUnit-фреймворка, предназначенного для модульного тестирования. NUnit поддерживает использование атрибутов, необходимых для распознавания тестов.

Атрибут [TestFixture] означает, что все помеченные им классы образуют тестовую структуру.

Атрибут [OneTimeSetUp] означает, что помеченные им классы будут вызываться единожды перед началом тестирования.

Атрибут [OneTimeTearDown] означает, что помеченные им классы будут вызываться единожды после окончания тестирования.

Атрибут [SetUp] означает, что помеченные им классы будут вызываться перед началом каждого теста.

Атрибут [TearDown] означает, что помеченные им классы будут вызываться после окончания каждого теста.

Атрибут [Test] означает, что все помеченные ими классы, являются непосредственно тестами.

Таким образом, проект представляет собой следующую структуру:

```
[TestFixture]
class Test
{       public static void Main(string[] args){}
                [OneTimeSetUp] public void TestFixtureSetUp(){}
                [OneTimeTearDown] public void TestFixtureTearDown(){}
                [SetUp]  public void SetUp(){}
                [TearDown] public void TearDown(){}
                [Test] public void TEST_1(){}
                [Test]  public void TEST_2(){}
}
```

Листинг 1. Структура проекта.

Для корректного запуска тестов при помощи исполняемого файла будем использовать утилиту nunit3-console.exe, которая подключается к проекту при помощи встроенного в VisualStudio менеджера пакетов NuGet. Функция запуска тестов имеет вид:

```
public static void Main(string[] args)
{
string path = Assembly.GetExecutingAssembly().Location;
NUnit.ConsoleRunner.Program.Main(new[] { path });
}
```

Листинг 2. Функция запуска тестов.

Метод <GetExecutingAssembly().Location> получает путь до расположения загрузочного файла, а метод Program.Main, принадлежащий пространству имен <NUnit.ConsoleRunner>, вызывает утилиту nunit3-console.exe которой передается путь до скомпилированного файла.

Конструирование тест-кейсов. Построение тест-кейсов начинается с объявления используемых глобальных переменных:

```
public string Sname = Randomizer.CreateRandomizer().GetString(8);
public string Slogin = Randomizer.CreateRandomizer().GetString(10);
public string Sphone =
                Randomizer.CreateRandomizer().GetString(9,"1234567890");
public string Spassword = Randomizer.CreateRandomizer().GetString(6);
public string ItemName;
public string ItemCount;
public string OrderItemCount =
                Randomizer.CreateRandomizer().GetString(2, "1234567890");
```

Листинг 3. Объявление глобальных переменных.

Переменные <Sname>, <Slogin>, <Spassword>, <Sphone> хранят случайно сгенерированную информацию о пользователе сайта, которая в дальнейшем будет использоваться для доступа к приложению.

Вид класса <TestFixtureSetUp()> представлен ниже на листинге 4. Класс <TestFixtureSetUp()> имеет атрибут [OneTimeSetUp], что означает, что он будет вызываться перед стартом тестирования. При помощи метода <CreateDirectory()> создаются новые каталоги, в которых будут храниться отчет и скриншоты, а метод <GenerateReport()> вызывает функцию генерации отчета.

Определяем <driver> как указатель переменной типа <ChromeDriver>, принадлежащего пакету <Selenium.WebDriver.ChromeDriver>. Это позволит нам управлять действиями браузера Google Chrome.

Метод <Manage()> позволяет управлять настройками драйвера, с его помощью устанавливается разрешение окна приложения “На весь экран” и время ожидания до появления следующего элемента в 5 секунд.

```
[OneTimeSetUp]
public void TestFixtureSetUp()
{
if (Directory.Exists(iFolderResultTest)) Directory.Delete(iFolderResultTest, true);
DirectoryInfo dr = Directory.CreateDirectory(iFolderResultTest);
DirectoryInfo ds = Directory.CreateDirectory(iFolderScreen);
GenerateReport(0, "0", true);
ChromeOptions options = new ChromeOptions();
options.AddArguments("--ignore-certificate-errors");
options.AddArguments("--ignore-ssl-errors");
driver = new ChromeDriver(iWorkDir, options);
driver.Manage().Window.Maximize();
driver.Manage().Timeouts().ImplicitWait = TimeSpan.FromSeconds(5);
}
```

Листинг 4. Класс <TestFixtureSetUp()>.

Класс <SetUp> имеет вид:

```
[SetUp]
public void SetUp()
{                driver.Navigate().GoToUrl("http://localhost/index.php");   }
```

Листинг 5. Класс <SetUp>.

При помощи метода <Navigate()> браузер переходит на страницу тестируемого приложения перед запуском каждого теста.

Объектом для проведения тестирования было выбрано web-приложение, созданное студенткой, обучающейся по направлению подготовки «Программная инженерия», и реализующее косметику посредством сети Интернет. Вид главной страницы приложения представлен на рисунке 5:

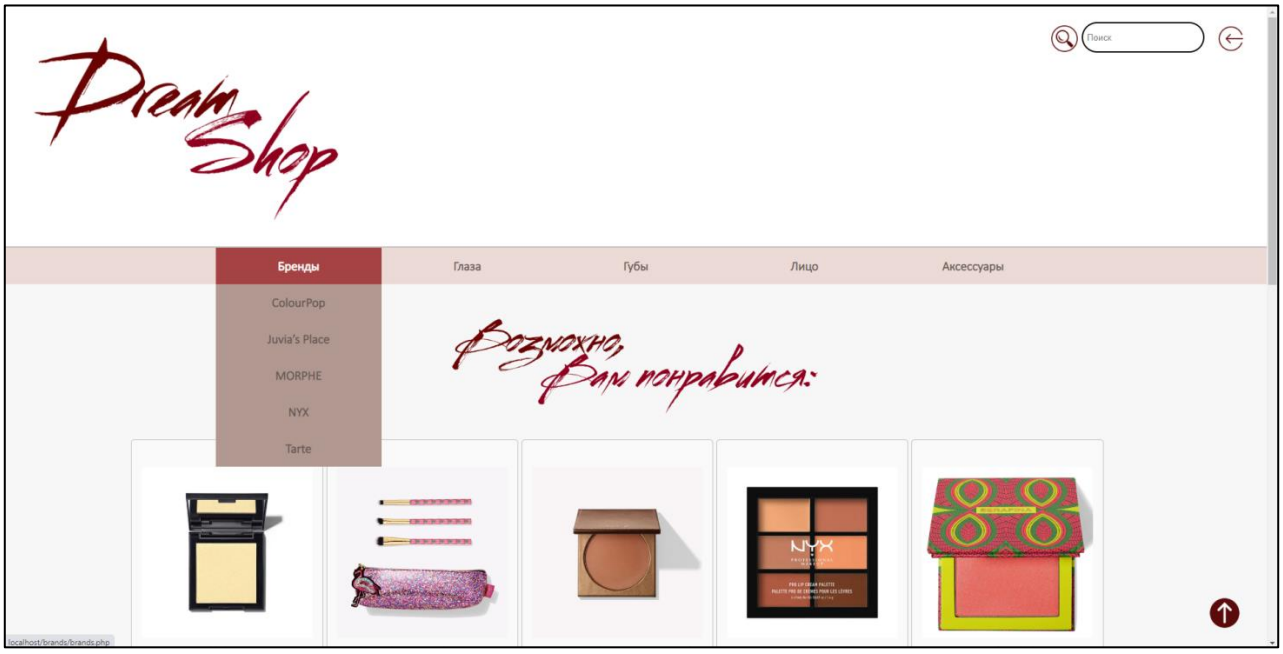


Рис. 5. Главная страница web-приложения.

Было создано 10 тест-кейсов, выполняющих проверку работоспособности основного функционала web-приложения:

- тест-кейс №1 – Проверка прохождения регистрации,
- тест-кейс №2 – Проверка прохождения авторизации,
- тест-кейс №3 – Проверка правильности отображения меню,
- тест-кейс №4 – Проверка правильности отображения товаров,
- тест-кейс №5 – Заказ товара,
- тест кейс №6 – Поиск товара,
- тест-кейс №7 – Проверка покупки повара,
- тест-кейс №8 – Проверка товара на складе,
- тест-кейс №9 – Редактирование цены товара администратором,
- тест-кейс №10 – Проверка совместимости с браузером Internet Explorer.

Подробно рассмотрим создание тестового сценария для тест-кейса №1 – Проверка прохождения регистрации.

Пакет Selenium.WebDriver предоставляет следующие методы взаимодействия с содержимым web-страницы:

Метод <FindElement()> находит на странице первый элемент, удовлетворяющий условиям поиска в зависимости от выбранного метода:

– By.Id() – поиск элемента происходит по наименованию идентифицирующего его поля Id;

– By.Name() – поиск элемента происходит по наименованию его поля Name;

– By.Xpath() – поиск элемента происходит при помощи языка запросов к элементам XML-документа XPATH

Метод <Click()> производит щелчок левой кнопкой мыши по выбранному элементу.

Метод <SendKeys()> производит ввод текста в выбранный элемент.

Код тест-сценария №1 имеет вид:

```
[Test]
public void TEST_1()
{       iTestName = "registration";
        try
{driver.FindElement(By.Id("login")).Click(); driver.FindElement(By.Id("registration")).Click();
 driver.FindElement(By.Name("FIO")).SendKeys(Sname);
 driver.FindElement(By.Name("login")).SendKeys(Slogin);
 driver.FindElement(By.Name("password")).SendKeys(Spassword);
 driver.FindElement(By.Name("phone")).SendKeys(Sphone);
 driver.FindElement(By.XPath("//button[@type='submit']")).Click();
 Assert.IsTrue(driver.FindElement(By.XPath("//*[contains(text(),'Вы успешно
зарегистрированы!')]")).Displayed);
GenerateReport(1, iTestName, true); iExecTestGood = true; }
catch (Exception ex)
{ GenerateReport(1, iTestName, false, ex.ToString()); iExecTestGood = false; }
Assert.True(iExecTestGood); }
```

Листинг 6. Код тест-сценария №1.

В результате проведения автоматизированного тестирования с помощью разработанной авторами программной системы, был получен отчет, представленный на рисунке 6. На рисунке видно, что «провалены» 4 сценария: Поиск товара, Проверка покупки товара, Проверка товара на складе и Проверка совместимости с браузером Internet Explorer:

Тест начат 20.09.2021 12:15:16

Тест	Результат	Время	Снимок	Сообщение
Registration	Успех	12:15:18	-	
Autorization	Успех	12:15:23	-	
ListCount	Успех	12:15:43	-	
ItemOnPage Count	Успех	12:16:01	-	
Order	Успех	12:16:18	-	
ItemSearch	Провал	12:16:29	open	OpenQA.Selenium.NoSuchElementException: no such element: Unable to locate element: {"method":"link text","selector":"Juvia&#65533;s 5pcs Multicolored Eye Set"} (Session info: chrome=83.0.4103.106) в OpenQA.Selenium.Remote.RemoteWebDriver.UnpackAndThrowOnError(Response errorResponse) в OpenQA.Selenium.Remote.RemoteWebDriver.Execute(String driverCommandToExecute, Dictionary`2 parameters) в OpenQA.Selenium.Remote.RemoteWebDriver.FindElement(String mechanism, String value) в OpenQA.Selenium.Remote.RemoteWebDriver.FindElementByLinkText(String linkText) в OpenQA.Selenium.By.<>c__DisplayClass17_0.<LinkText>b__0(ISearchContext context) в OpenQA.Selenium.By.FindElement(ISearchContext context) в OpenQA.Selenium.Remote.RemoteWebDriver.FindElement(By by) в TESTING_SPACE.Test.TEST_6() в C:\Users\alexp\Desktop\Новая папка\Tests1\Program.cs:строка 225
CheckOrder	Провал	12:16:41	open	OpenQA.Selenium.NoSuchElementException: no such element: Unable to locate element: {"method":"link text","selector":"Juvia&#65533;s 5pcs Multicolored Eye Set"} в OpenQA.Selenium.Remote.RemoteWebDriver.FindElementByLinkText(String linkText) в TESTING_SPACE.Test.TEST_7() в C:\Users\alexp\Desktop\Новая папка\Tests1\Program.cs:строка 248
CheckItem Count	Провал	12:17:03	open	OpenQA.Selenium.NoSuchElementException: no such element: Unable to locate element: {"method":"xpath","selector":"//article[1]//child::img"} в OpenQA.Selenium.Remote.RemoteWebDriver.FindElementByXPath(String xpath) в TESTING_SPACE.Test.TEST_8() в C:\Users\alexp\Desktop\Новая папка\Tests1\Program.cs:строка 274
Admin Change	Успех	12:17:30	-	
Internet ExplorerRun	Провал	12:17:35	open	OpenQA.Selenium.NoSuchElementException: Unable to find element with css selector == #logout в OpenQA.Selenium.Remote.RemoteWebDriver.FindElementById(String id) в TESTING_SPACE.Test.TEST_10() в C:\Users\alexp\Desktop\Новая папка\Tests1\Program.cs:строка 345

Всего тесто запущено: 10 || **Успешно: 6** || **Провалено: 4** || Процент пройденных тестов: 60% ||
Тест завершен: 20.09.2021 12:17:35

Рис. 6. Отчет о прохождении тестирования.

Благодаря сделанным при возникновении ошибки снимкам экрана (см., например, рисунок 7) были выявлены ошибки, сделанные разработчиком при написании кода web-приложения.

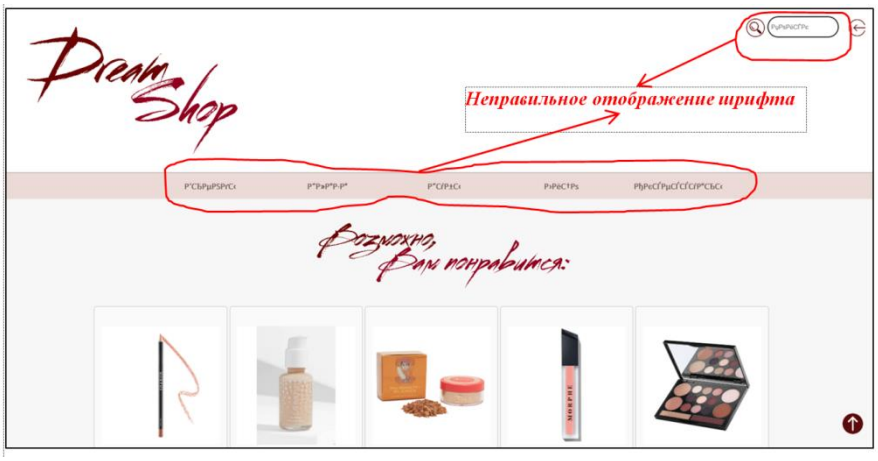


Рис. 7. Снимок экрана при возникновении ошибки тест-кейсе № 10.

После устранения выявленных ошибок было проведено повторное тестирование web-приложения, результаты которого представлены на рисунке 8:

Тест начат 20.09.2021 12:15:16

Тест	Результат	Время	Снимок	Сообщение
Registration	Успех	12:15:18	-	
Autorization	Успех	12:15:23	-	
ListCount	Успех	12:15:43	-	
ItemOnPageCount	Успех	12:16:01	-	
Order	Успех	12:16:18	-	
ItemSearch	Успех	12:16:29	-	
CheckOrder	Успех	12:16:41	-	
CheckItemCount	Успех	12:17:03	-	
AdminChange	Успех	12:17:30	-	
InternetExplorerRun	Успех	12:17:35	-	

Всего тесто запущено: 10 || **Успешно: 10** || **Провалено: 0** || Процент пройденных тестов: 100% ||
Тест завершен: 20.09.2021 12:17:35

Рис. 8. Отчет о повторном прохождении тестирования.

Заключение. В статье рассмотрена реализованная авторами программная система, построенная на базе фреймворка SELENIUM и предназначенная для проведения автоматизированного тестирования web-приложений. Разработанные в системе тест-кейсы могут успешно применяться для тестирования студенческих работ, выполняемых по дисциплине «Разработка web-приложений», а при согласовании с заказчиком и на основе его требований – и для тестирования коммерческих проектов.

СПИСОК ЛИТЕРАТУРЫ

1. Винокуров А. В., Лавлинская О. Ю. Уровни организации автоматизированного тестирования мобильных приложений для операционной системы ANDROID // Вестник Воронежского института высоких технологий. – 2020. – № 3 (34). – С. 22-26 [Электронный ресурс]. – Режим доступа: https://vivt.ru/downloads/vestnik/vestnik34.pdf (дата обращения: 15.09.2021).
2. Янгунаева Е. А., Янгунаев В. М. Сравнение автоматизированного и ручного подхода в тестировании веб-приложений // Научный альманах. – 2016. – № 1-1 (15). – С. 546-549 [Электронный ресурс]. – Режим доступа: https://ukonf.com/doc/na.2016.01.01.pdf (дата обращения: 15.09.2021).
3. Барвин С. К., Попов Д. В. Метрики автоматизированного тестирования web-приложения // Современные научные исследования и инновации. – 2019. – № 4 (96). – С. 4 [Электронный ресурс]. – Режим доступа: https://web.snauka.ru/issues/2019/04/89160 (дата обращения: 15.09.2021).

4. Янгунаев В. М., Янгунаева Е. А. Исследование средств семейства Selenium для автоматизированного тестирования веб-приложений // Вестник научных конференций. – 2016. – № 1-5 (5). – С. 218–219 [Электронный ресурс]. – Режим доступа: https://ukonf.com/doc/cn.2016.01.05.pdf (дата обращения: 15.09.2021).

ШАМАНАЕВ П. А., ПРОХОРОВ С. А.

ИССЛЕДОВАНИЕ ВЫНУЖДЕННЫХ КОЛЕБАНИЙ ЛИНЕЙНОЙ СИСТЕМЫ С ДВУМЯ СТЕПЕНЯМИ СВОБОДЫ И МАЛЫМ ПАРАМЕТРОМ МЕТОДОМ ЛЯПУНОВА–ШМИДТА

Аннотация. Работа посвящена нахождению периодических решений линейных систем с двумя степенями свободы и малым параметром методом Ляпунова-Шмидта. На основе метода Ляпунова-Шмидта разработан алгоритм в пакете Maple, построены графики компонент периодических решений и фазовых траекторий возмущенной системы.

Ключевые слова: система с двумя степенями свободы, вынужденные колебания, периодические решения, малый параметр, метод Ляпунова-Шмидта, резонанс, Maple.

SHAMANAEV P. A., PROKHOROV S. A.

INVESTIGATION OF FORCED VIBRATIONS OF A LINEAR SYSTEM WITH TWO DEGREES OF FREEDOM AND A SMALL PARAMETER BY THE LYAPUNOV-SCHMIDT METHOD

Abstract. The article is devoted to finding periodic solutions to linear systems with two degrees of freedom and a small parameter by the Lyapunov-Schmidt method. On the basis of the Lyapunov-Schmidt method, an algorithm was developed in the Maple package, graphs of the components of periodic solutions and phase trajectories of the disturbed system were constructed.

Keywords: system with two degrees of freedom, forced oscillations, periodic solutions, small parameter, Lyapunov-Schmidt method, resonance, Maple.

Введение. Математические модели в линейном приближении могут быть описаны как неоднородные линейные системы обыкновенных дифференциальных уравнений с малым параметром. Общая методика исследования периодических решений таких систем описана в работе [1]. Она основана на представлении исследуемой системы в виде операторного уравнения в банаховом пространстве и применении метода Ляпунова-Шмидта [2–4].

В работе [5] приведено исследование периодических решений одной линейной неоднородной системы второго порядка с малым линейным возмущением. В работе [6] проведено исследование вынужденных колебаний одной линейной системы двух связанных осцилляторов с малым параметром.

В настоящей работе исследуются вынужденные колебания линейной системы с двумя степенями свободы и малым параметром при условии, что на систему действует внешняя периодическая сила с двумя соизмеримыми частотами.

Математическая модель вынужденных колебаний системы с двумя степенями свободы и малым параметром. Рассмотрим математическую модель вынужденных колебания системы с двумя степенями свободы в виде [7]

$$\begin{cases} \ddot{Q}_1 + n_1^2 Q_1 - KQ_2 = F_1(t), \\ \ddot{Q}_2 + n_2^2 Q_2 - KQ_1 = F_2(t), \end{cases} \tag{1}$$

где n_1^2 и $n_2^2 -$ парциальные частоты, $K > 0$, функции $F_1(t)$ и $F_2(t)$ имеют вид

$$\begin{aligned} F_1(t) &= r_{11}\sin(\omega_1 t + \theta_{11}) + r_{12}\sin(\omega_2 t + \theta_{12}), \\ F_2(t) &= r_{21}\sin(\omega_1 t + \theta_{21}) + r_{22}\sin(\omega_2 t + \theta_{22}), \end{aligned} \tag{2}$$

где r_{ks}, θ_{ks}, $\omega_k \in R$, $k, s = 1, 2, \omega_2 = \alpha\omega_1$, $\alpha \in Z$. Обозначим $T = \frac{2\pi}{\omega_1}$.

Исследуем вынужденные колебания системы (1), когда параметры системы мало отклоняются от заданных значений и две частоты вынужденных колебаний совпадают с двумя частотами собственных колебаний (случай резонанса). В этом случае, система (1) с учетом малого параметра будет иметь вид

$$\begin{cases} \ddot{q}_1 + (n_1^2 + \varepsilon d_{11})q_1 + (-K + \varepsilon d_{12})q_2 = F_1(t), \\ \ddot{q}_2 + (-K + \varepsilon d_{21})q_1 + (n_2^2 + \varepsilon d_{22})q_2 = F_2(t), \end{cases} \tag{3}$$

Где $d_{ks} \in R$, $k, s = 1, 2$, ε – малый вещественный параметр. Заметим, что система (1) получается из системы (3), если положить $\varepsilon = 0$.

Сформулируем задачу для системы (3): при достаточно малых вещественных ε найти T–периодическое решение $q_1(t, \varepsilon)$, $q_2(t, \varepsilon)$ системы (3), удовлетворяющее условию $q_1(t, 0) = Q_1(t)$, $q_2(t, 0) = Q_2(t)$, где $Q_1(t)$, $Q_2(t)$ есть T–периодическое решение системы (1).

Периодичность решения $Q_1(t)$, $Q_2(t)$ системы (1) достигается за счет подбора амплитуд r_{ks} и начальных фаз θ_{ks} в формуле (2) для периодических функций $F_1(t)$ и $F_2(t)$.

Представление математической модели в виде системы обыкновенных дифференциальных уравнений в нормальной форме. Делая в системе (3) замену

$$\begin{aligned} x_1 = q_1,\ x_2 = \dot{q}_1 &\Rightarrow \dot{x}_1 = x_2, \\ x_3 = q_2,\ x_4 = \dot{q}_2 &\Rightarrow \dot{x}_3 = x_4, \end{aligned} \tag{4}$$

получим систему обыкновенных дифференциальных уравнений в нормальной форме

$$\frac{dx}{dt} = (B_0 - \varepsilon B_1)x - f(t), \tag{5}$$

где $x \in R^4$,

$$B_0 = \begin{pmatrix} 0 & 1 & 0 & 0 \\ -n_1^2 & 0 & K & 0 \\ 0 & 0 & 0 & 1 \\ K & 0 & -n_2^2 & 0 \end{pmatrix}, B_1 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ d_{1,1} & 0 & d_{1,2} & 0 \\ 0 & 0 & 0 & 0 \\ d_{2,1} & 0 & d_{2,2} & 0 \end{pmatrix}, \ f(t) = \begin{pmatrix} 0 \\ -F_1(t) \\ 0 \\ -F_2(t) \end{pmatrix}.$$

Системе (1) соответствует неоднородная линейная система обыкновенных дифференциальных уравнений в нормальной форме

$$\frac{dz}{dt} = B_0 z - f(t). \tag{6}$$

В этом случае задача, сформулированная для систем (1) и (3), перейдет в задачу для систем (5) и (6) в следующем виде: при достаточно малых вещественных ε найти $T-$периодическое решение $x(t,\varepsilon)$ системы (5), удовлетворяющее условию $x(t,0) = z(t)$, где $z(t)$ является $T-$периодическим решением системы (6).

В дальнейшем систему (5) будем называть возмущенной системой, а систему (6) – невозмущенной.

Вычисление периодического решения возмущенной системы методом Ляпунова-Шмидта. В работе [1] приведено решение поставленной задачи методом Ляпунова-Шмидта. Периодическое решение возмущенной системы будем вычислять с помощью разработанного на основе метода Ляпунова-Шмидта алгоритма в математическом пакете Maple.

Для иллюстрации разработанного алгоритма в пакете Maple выберем параметры системы (5) следующим образом:

$$\omega_1 = 3, \omega_2 = 9, n_1^2 = n_2^2 = 45,\ K = 36,\ d_{11} = -2,\ d_{22} = 2,\ d_{12} = d_{21} = 0.$$

Собственные значения матрицы B_0 при выбранных значениях параметров равны

$$\lambda_{1,2} = \pm 3i, \qquad \lambda_{3,4} = \pm 9i.$$

Поскольку для системы (6) частоты собственных колебаний совпадают с частотами вынужденных колебаний, то для система (6), а значит и системы (1) имеет место случай резонанса. В этом случае $T = \frac{2\pi}{3}$ и задача сводится к нахождению $\frac{2\pi}{3}-$периодического решения системы (5).

Вычислим обобщенный жорданов набор оператора

$$\mathcal{B}_0 = B_0 x - A\frac{dx}{dt}$$

по формулам

$$\mathcal{B}_0 \varphi_k^{(1)} = 0,\ \mathcal{B}_0 \varphi_k^{(j)} = B_1 \varphi_k^{(j-1)}, \tag{7}$$

где $j = \overline{2,\ p_k}, k = \overline{1,n}$.

Согласно методу Эйлера, решения уравнений (7), будем искать в виде

$$\varphi_k^{(j)} = u_k^{(j)} e^{\lambda t}, \tag{8}$$

где $\lambda \in R,\ u_k^{(j)} \in R^4$.

Подставляя формулы (8) в уравнения (7), получим следующие элементы обобщенных жордановых цепочек оператора $\mathcal{B}_0$

$$\varphi_1^{(1)}(t) = e^{i3t}\begin{pmatrix} -\frac{i}{3} \\ 1 \\ -\frac{i}{3} \\ 1 \end{pmatrix},\ \varphi_1^{(2)}(t) = e^{3it}\begin{pmatrix} -\frac{i}{54} \\ \frac{1}{18} \\ 0 \\ 0 \end{pmatrix}, \tag{9}$$

$$\varphi_2^{(1)}(t) = e^{i9t}\begin{pmatrix} -\frac{i}{9} \\ -1 \\ -\frac{i}{9} \\ 1 \end{pmatrix},\ \varphi_2^{(2)}(t) = e^{i9t}\begin{pmatrix} -\frac{i}{162} \\ \frac{1}{18} \\ 0 \\ 0 \end{pmatrix}. \tag{10}$$

Здесь число и длины жордановых цепочек равны $n = 2,\ p_1 = p_2 = 2$, соответственно.

Элементы обобщенных жордановых цепочек сопряженного оператора $\mathcal{B}_0^*$, удовлетворяющих условию биортогонализации, имеют вид

$$\psi_1^{(1)} = e^{3it}\begin{pmatrix} 81 \\ 27i \\ 81 \\ 27i \end{pmatrix},\ \psi_1^{(2)} = e^{3it}\begin{pmatrix} 0 \\ 0 \\ -\frac{9}{2} \\ -\frac{3i}{2} \end{pmatrix}, \tag{11}$$

$$\psi_2^{(1)} = e^{9it}\begin{pmatrix} 729 \\ 81i \\ -729 \\ -81i \end{pmatrix},\ \psi_2^{(2)} = e^{9it}\begin{pmatrix} 0 \\ 0 \\ -\frac{81}{2} \\ -\frac{9i}{2} \end{pmatrix}. \tag{12}$$

Коэффициенты c_{kj} в разложении искомого периодического решения вычислим по формулам

$$c_{kj} = \langle\langle f, \psi_k^{(j)}\rangle\rangle = \frac{1}{T}\int_0^T \langle f(t), \psi_k^{(j)}(t)\rangle\, dt, \qquad k, j = \overline{1,2}\,.$$

Учитывая формулы (11) и (12), получим

$$\begin{aligned} c_{11} &= \tfrac{27r_{11}e^{i\theta_{11}}}{2} + \tfrac{27r_{12}e^{i\theta_{12}}}{2}, \qquad c_{12} = \tfrac{3r_{12}e^{i\theta_{12}}}{4}, \\ c_{21} &= \frac{81r_{21}e^{i\theta_{21}}}{2} - \frac{81r_{22}e^{i\theta_{22}}}{2}, \qquad c_{22} = -\frac{9r_{22}e^{i\theta_{22}}}{4}. \end{aligned} \tag{13}$$

Для того, чтобы система (6) имела T-периодическое решение, необходимо и достаточно, чтобы $c_{11} = 0$ и $c_{21} = 0$ [1], что эквивалентно следующим условиям, соответственно,

$$r_{11}e^{i\theta_{11}} + r_{12}e^{i\theta_{12}} = 0, \tag{14}$$

$$r_{21}e^{i\theta_{21}} - r_{22}e^{i\theta_{22}} = 0. \tag{15}$$

Решения уравнений (14) и (15) можно записать в параметрической форме, соответственно,

$$r_{11} = r_{12} = r_1, \theta_{11} = \theta_1\ \theta_{12} = \theta_1 + \pi, \tag{16}$$

$$r_{21} = r_{22} = r_2,\ \theta_{21} = \theta_2\ \theta_{22} = \theta_2, \tag{17}$$

где r_k, θ_k, $k = 1,2,$ – произвольные вещественные параметры.

После подстановки формул (16) и (17) в выражения (13), получим

$$c_{11} = 0, \quad c_{12} = \frac{3r_1 e^{i\theta_1}}{4}, \quad c_{21} = 0, \ \ c_{22} = -\frac{9r_2 e^{i\theta_2}}{4}. \tag{18}$$

Тогда при $\varepsilon \neq 0$ вычисляя величины ξ_k по формулам (3.3) из работы [1], получим

$$\xi_1 = -\frac{3r_1 e^{i\theta_1}}{4\varepsilon}, \quad \xi_2 = -\frac{9r_2 e^{i\theta_2}}{4\varepsilon} \tag{19}$$

Дополнительное слагаемое $y(\varepsilon)$, входящее в $\frac{2\pi}{3}$ −периодическое решение системы (5) и принадлежащие к дополнению корневого пространства, имеет вид

$$y(t,\varepsilon) = \begin{pmatrix} -\frac{(18\varepsilon+1)}{36\varepsilon^2-36} r_1 \sin(3t+\theta_1) - \frac{(18\varepsilon-1)}{36\varepsilon^2-36} r_2 \sin(9t+\theta_2) \\ -\frac{(18\varepsilon+1)}{12\varepsilon^2-12} r_1 cos\ (3t+\theta_1) - \frac{(18\varepsilon-1)}{4\varepsilon^2-4} r_2 cos\ (9t+\theta_2) \\ -\frac{\varepsilon}{2\varepsilon^2-2} r_1 \sin(3t+\theta_1) + \frac{\varepsilon}{2\varepsilon^2-2} r_2 \sin\ (9t+\theta_2) \\ -\frac{3\varepsilon}{2\varepsilon^2-2} r_1 \cos(3t+\theta_1) + \frac{9\varepsilon}{2\varepsilon^2-2} r_2 \cos\ (9t+\theta_2) \end{pmatrix}. \tag{20}$$

Таким образом, подставляя формулы (9), (10), (18) и (19) в выражение (3.4) из работы [1], получим $\frac{2\pi}{3}$ – периодическое решение системы (5)

$$x(t,\varepsilon) = \begin{pmatrix} -\frac{1}{2\varepsilon} r_1 \sin(3t+\theta_1) - \frac{1}{2\varepsilon} r_2 \sin(9t+\theta_2) \\ -\frac{3}{2\varepsilon} r_1 \cos(3t+\theta_1) - \frac{9}{2\varepsilon} r_2 \cos(9t+\theta_2) \\ -\frac{1}{2\varepsilon} r_1 \sin(3t+\theta_1) + \frac{1}{2\varepsilon} r_2 \sin(9t+\theta_2) \\ -\frac{3}{2\varepsilon} r_1 \cos(3t+\theta_1) + \frac{9}{2\varepsilon} r_2 \cos(9t+\theta_2) \end{pmatrix}. \tag{21}$$

Построение графиков компонент периодического решения и фазовых траекторий системы (3). Учитывая замену (4) из формулу (21), получим

$$q_1(t,\varepsilon) = -\frac{1}{2\varepsilon} r_1 \sin(3t+\theta_1) - \frac{1}{2\varepsilon} r_2 \sin(9t+\theta_2),$$

$$q_2(t,\varepsilon) = -\frac{1}{2\varepsilon} r_1 \sin(3t+\theta_1) + \frac{1}{2\varepsilon} r_2 \sin(9t+\theta_2).$$

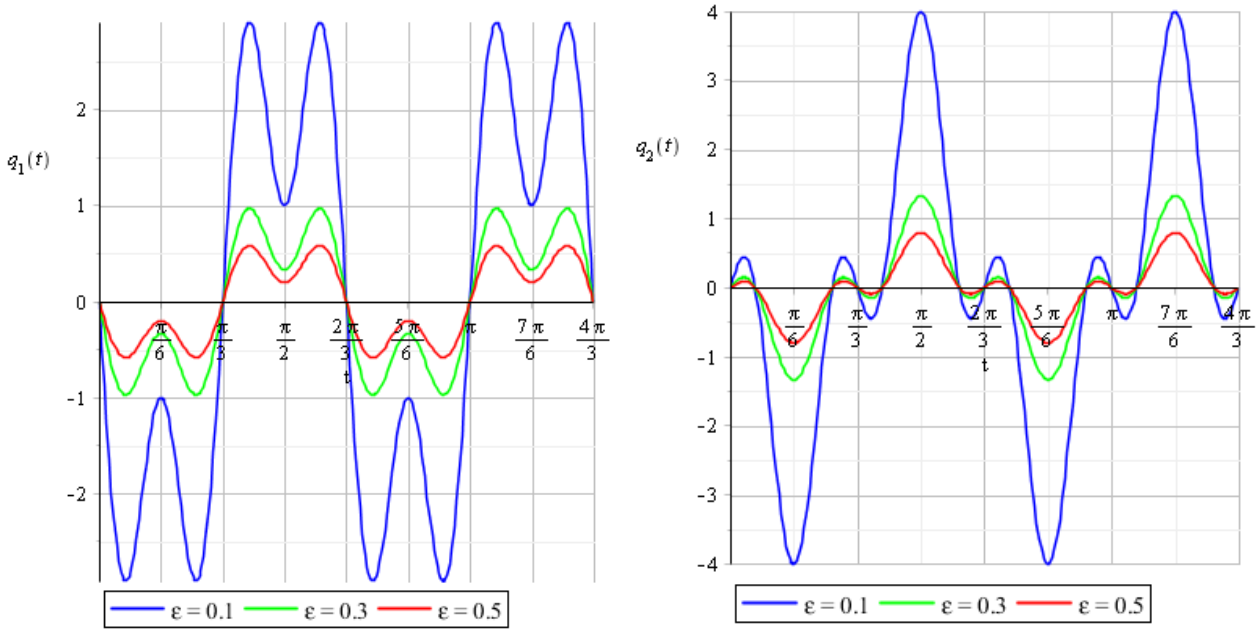
Для построения графиков компонент решений и фазовых траекторий системы (3) выберем следующие наборы параметров:

1) $r_1 = 0{,}5;\ r_2 = 0{,}3;\ \theta_1 = 0;\ \theta_2 = 0;$

2) $r_1 = 0{,}5;\ r_2 = 0{,}3;\ \theta_1 = 0;\ \theta_2 = \frac{2\pi}{5};$

3) $r_1 = 0{,}5;\ r_2 = 0{,}3;\ \theta_1 = 0;\ \theta_2 = \frac{4\pi}{5}.$

На рис. 1 и рис. 2. для случая 1) приведены графики компонент $\frac{2\pi}{3}$ −периодического решения и фазовых траекторий системы (3) при различных значениях параметра ε.



а) $q_1(t,\varepsilon)$; б) $q_2(t,\varepsilon)$.

Рис. 1. Графики компонент а) $q_1(t,\varepsilon)$ и б) $q_2(t,\varepsilon)$ $\frac{2\pi}{3}$ − периодического решения системы (3) при различных значениях параметра ε в случае 1).

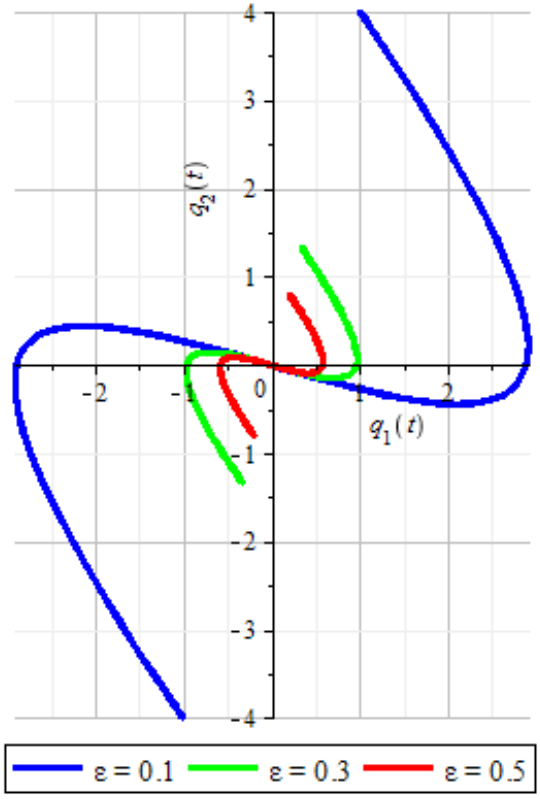
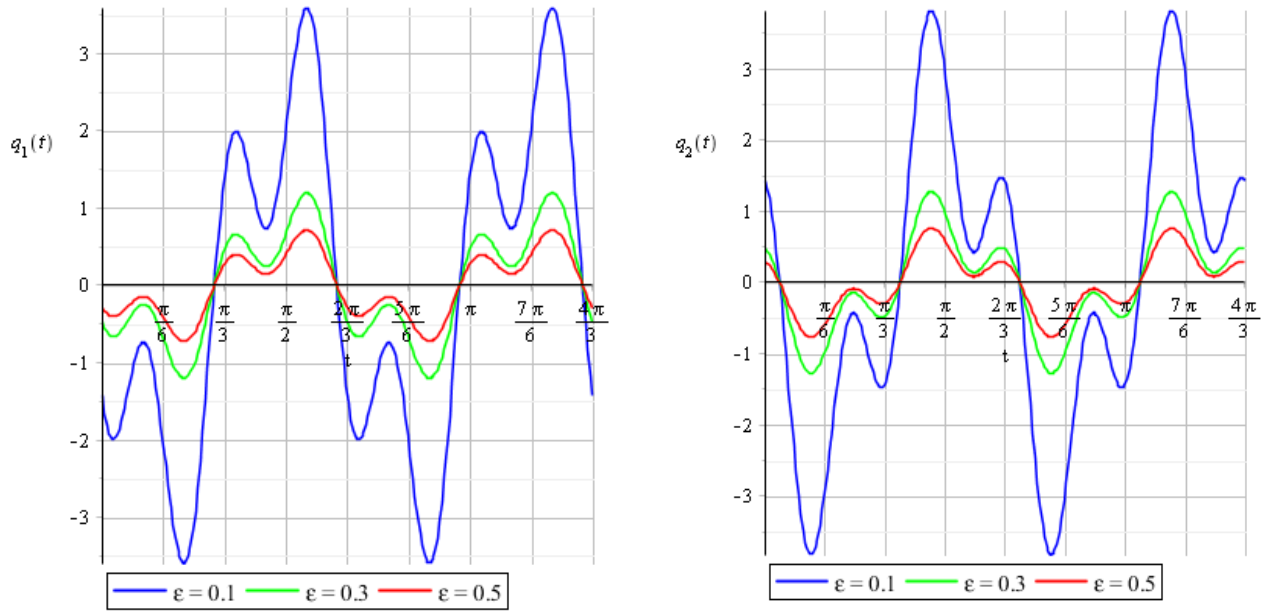


Рис. 2. График фазовой траектории системы (3) в конфигурационном пространстве Oq_1q_2 при различных ε в случае 1).

На рис. 3 и рис. 4. для случая 2) приведены графики компонент $\frac{2\pi}{3}$ −периодического решения и фазовых траекторий системы (3) при различных значениях параметра ε.



а) $q_1(t,\varepsilon)$; б) $q_2(t,\varepsilon)$.

Рис. 3. Графики компонент а) $q_1(t,\varepsilon)$ и б) $q_2(t,\varepsilon)$ $\frac{2\pi}{3}$ − периодического решения системы (3) при различных значениях параметра ε в случае 2).

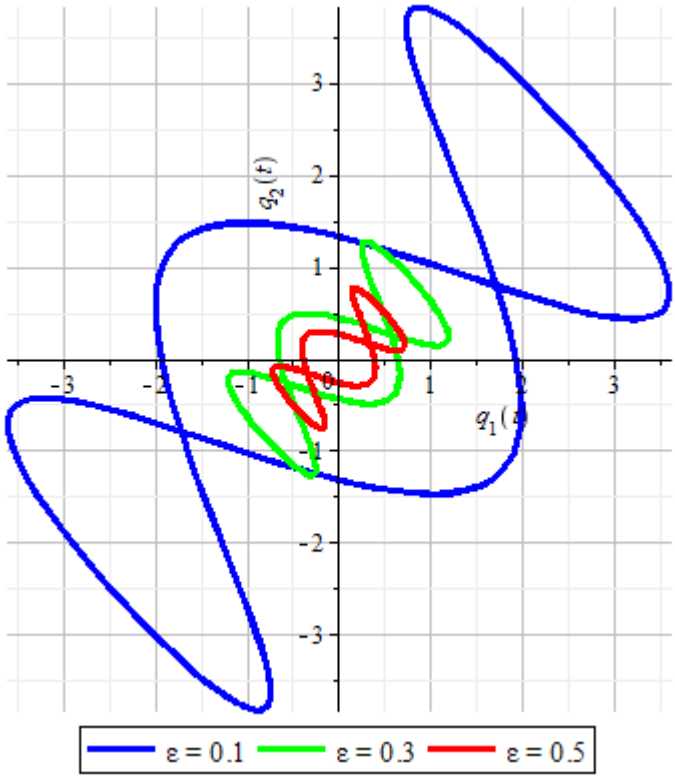
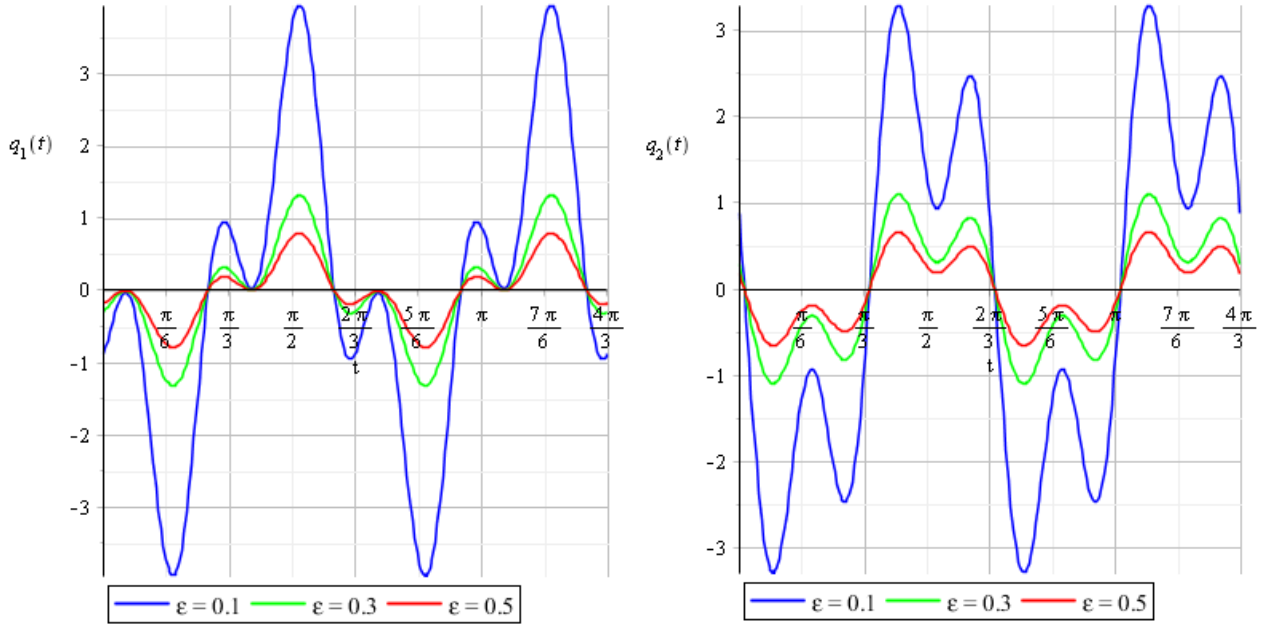


Рис. 4. График фазовой траектории системы (3) в конфигурационном пространстве Oq_1q_2 при различных ε в случае 2).

На рис. 5 и рис. 6. для случая 3) приведены графики компонент $\frac{2\pi}{3}$ −периодического решения и фазовых траекторий системы (3) при различных значениях параметра ε.



а) $q_1(t,\varepsilon)$; б) $q_2(t,\varepsilon)$.

Рис. 5. Графики компонент а) $q_1(t,\varepsilon)$ и б) $q_2(t,\varepsilon)$ $\frac{2\pi}{3}$ − периодического решения системы (3) при различных значениях параметра ε в случае 3).

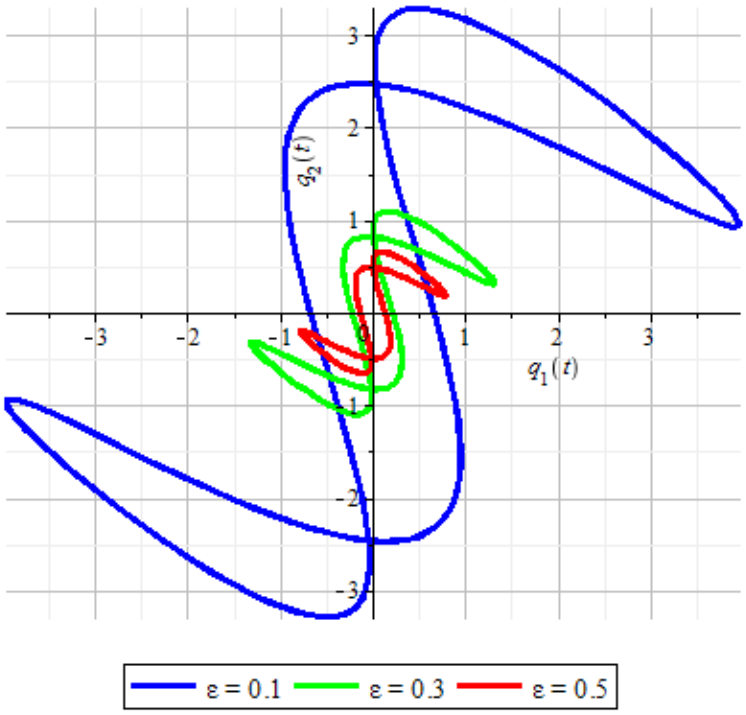


Рис. 6. График фазовой траектории системы (3) в конфигурационном пространстве Oq_1q_2 при различных ε в случае 3).

Заключение. Из формулы (21) следует, что каждая компонента $\frac{2\pi}{3}-$ периодического решения системы (3) имеют полюс первого порядка в точке $\varepsilon = 0$ и зависит от тех же начальных фаз θ_{ks} и амплитуд r_{ks}, что и периодические функции $F_1(t)$ и $F_2(t)$.

СПИСОК ЛИТЕРАТУРЫ

1. Кяшкин А. А., Логинов Б. В., Шаманаев П. А. О ветвлении периодических решений линейных неоднородных дифференциальных уравнений с вырожденным или тождественным оператором при производной и возмущением в виде малого линейного слагаемого // Журнал Средневолжского математического общества. – 2016. – Т. 18, № 1. – С. 45–53.
2. Вайнберг М. М., Треногин В. А. Теория ветвления решений нелинейных уравнений. – М.: Наука. – 1964. – 524 с.
3. Коноплева И. В., Логинов Б. В. Обобщенная жорданова структура и симметрия разрешающих систем ветвления // Вестник Самарского университета. – 2001. – № 4. – С. 56–84.
4. Коноплева И. В., Логинов Б. В., Русак Ю. Б. Симметрия и потенциальность уравнений разветвления в корневых подпространствах в неявно заданных стационарных и динамических бифуркационных задачах // Известия высших учебных заведений. Северо-Кавказский регион. Серия: Естественные науки. – 2009. – С. 115–124.
5. Кадрякова М. Р., Логинов Б. В., Шаманаев П. А. О периодических решениях одного класса линейных неоднородных систем обыкновенных дифференциальных уравнений с малым параметром в резонансном случае // Огарев-online. – 2017. – № 13. – С. 8–17 [Электронный ресурс]. – Режим доступа: http://journal.mrsu.ru/arts/o-periodicheskix-resheniyax-odnogo-klassa-linejnyx-neodnorodnyx-sistem-obyknovennyx-differencialnyx-uravnenij-s-malym-parametrom-v-rezonansnom-sluchae (дата обращения: 25.09.2021).
6. Карчиганов А. Ф., Шаманаев П. А. Исследование вынужденных колебаний одной линейной системы двух связанных осцилляторов с малым параметром // Огарев-online. – 2020. – № 13. – С. 8–17 [Электронный ресурс]. – Режим доступа: http://journal.mrsu.ru/arts/issledovanie-vynuzhdennyx-kolebanij-odnoj-linejnoj-sistemy-dvux-svyazannyx-oscillyatorov-s-malym-parametrom (дата обращения: 25.09.2021).
7. Стрелков С. П. Введение в теорию колебаний: Учебник. – 3-е изд., испр. – СПб.: Лань, 2005. – 440 с.