**ЗАДАЧИ ПРОЕКТИРОВАНИЯ АДАПТИВНЫХ ИНТЕРФЕЙСОВ САПР**

На любой стадии создания проекта с использованием систем автоматизации проектирования (САПР) проектировщику целесообразно располагать некоторым набором средств, упрощающих и ускоряющих его работу, средств, помогающих ему в его информационной и конструктивной деятельности по пониманию и анализу сущности проектной задачи. Такой набор средств должен предоставить проектировщику помощь в решении как сложных задач проектирования или построения смет проекта, так и в решении простых задач. При этом решение простых, рутинных задач, должно проходить при минимальном участии проектировщика. В рамках жизненного цикла промышленных изделий, САПР решает задачи автоматизации работ на стадиях проектирования и подготовки производства. Основная цель создания САПР - повышение эффективности труда инженеров. К сожалению, несмотря на наличие достаточно серьезных комплексных САПР, специалисты вынуждены решать многие узкоспециализированные задачи в слабо автоматизированном режиме. Современные САПР предоставляют богатые возможности настройки и наращивания функционала.

Сегодня можно чётко выделить два подхода к кастомизации САПР.

-настройка существующего функционала - это набор средств, позволяющих подключать и отключать существующие функции систем автоматизированного проектирования. Этот набор средств, как правило, прост в использовании и не требует от специалиста дополнительных навыков в области программирования, при этом не предоставляется возможность добавления функций, не предусмотренных разработчиками;

-наращивание функционала представляет собой набор средств, позволяющих расширить систему автоматизированного проектирования функциями, не предусмотренными разработчиками системы. Как правило, этот набор средств, представлен в виде сложного логико-математического аппарата. Работа с этим аппаратом предполагает наличие у специалиста глубоких знаний об архитектуре конкретной САПР и навыков программирования, наращивание функционала отнимает много времени.

Исследования в области адаптивных интерфейсов автоматизированных систем актуальны, поскольку позволяют создать набор общих принципов позволяющих реализовать аппарат, который позволит специалистам в области САПР добавлять новые, необходимые им функции к системам автоматизированного проектирования, при этом, не требуя от специалиста владения навыками программирования. Новый аппарат позволит создавать новые функции системы, быстрее, чем классические логикоматематические аппараты САПР типа LISP. В изложенном контексте автором ставится задача разработки абстрактной модели построения адаптивного интерфейса пользователя САПР и определения способов интеграции средств адаптивного интерфейса пользователя, в современные САПР.

При этом следует эффективно решать следующие общие задачи:

определить роль и место адаптивного интерфейса пользователя в современных САПР:

выделить области задач САПР, требующие наличия адаптивного интерфейса пользователя;

отыскать вспомогательные средства, позволяющие интегрировать адаптивный интерфейс пользователя в современные САПР;

построить реальные предметные модели адаптивного интерфейса пользователя, и изучить возможности их применения в САПР.

В качестве объекта исследования целесообразно рассматривать системное строительное проектирование как вид деятельности, в котором реализуются все возможные действия над объектом проектирования на всех его жизненных этапах. Предмет исследования становятся системотехнические и информационные особенности представления объектов и процессов в автоматизированном строительном проектировании, логико-математический аппарат систем автоматизированного проектирования, эргономика пользовательского интерфейса и удобство эксплуатации.

Исследования необходимо строить в рамках классической математической логики,

системного анализа систем кастомизации различного уровня, используемых в рамках современных САПР и автоматизированных систем управления (АСУ) наиболее распространенных классов в строительстве.

В качестве новых результатов подобного исследования можно позиционировать:

методы построения абстрактных и предметных моделей адаптивных интерфейсов пользователя САПР, не требующих наличия сложного логико-математического аппарата;

предметную классификацию различных типов моделей адаптивного фейса пользователя, по типам решаемых задач;

требования к логико-математическому аппарату САПР, предусматривающие возможность создания надмножества моделей адаптивного интерфейса пользователя уровня практических приложений.

**ОЦЕНКА СТЕПЕНИ УДОБСТВА ИСПОЛЬЗОВАНИЯ ПОЛЬЗОВАТЕЛЬСКИХ ИНТЕРФЕЙСОВ В ЛОГИКЕ ТАЙЛОВ**

В стандарте ISO/IEC 25010 введен термин «юзабилити» (от англ. usability), означающий «удобство и простоту использования, степень удобства использования», а также пригодность использования, эргономичность (способность приложения быть понимаемым, изучаемым, используемым и привлекательным для пользователя в заданных условиях для достижения установленных целей с необходимой результативностью, эффективностью и удовлетворенностью). Далее вместо термина «юзабилити» будем использовать словосочетание «степень удобства использования».

Объект исследования настоящей работы — степень удобства использования пользовательских интерфейсов, обеспечивающих взаимодействие пользователя с программными приложениями и осуществляемых в среде одной или совокупности веб-страниц посредством веб-браузера.

Существует много традиционных средств оценки степени удобства использования, среди которых проверка по контрольному списку [1], эвристическая оценка [2], мысленный прогон [3], множественный прогон [4], оценки функциональности [5], анализ задач [6], фокус группы [7], обзорные исследования и интервью, вопросники и анкетирование, фиксация

зрительной активности пользователей [8] и др. Однако все эти оценки степени удобства использования довольно трудоемки, что проявляется в затратах времени на подготовку к сравнительному анализу, отбор респондентов и дальнейший анализ полученных результатов специалистами че- ловекомашинного взаимодействия, в некоторых случаях необходимо наличие специального оборудования. Чтобы снизить сложность и объем работ, выполняемых вручную, актуально направление автоматизации оценки степени удобства использования. В этом направлении существует три основных подхода к автоматизации оценки степени удобства использования визуальных пользовательских интерфейсов.

Подход 1. Анализ свойств пользовательских логов. Анализ осуществ- ляется в хронологическом порядке по записи действий пользователя в процессе взаимодействия с реальными приложениями или их динамическими прототипами. Как правило, такими действиями являются перемещение курсора мыши, время, затраченное пользователем для взаимодействия с одной и той же страницей, переходы между страницами и пр.

Наиболее популярными сервисами, предоставляющими услуги по сбору пользовательских логов и их анализу, являются сервисы Соое АпауИс$ и Уапаех Менчсз. Эти сервисы обрабатывают собранные данные и на основе их могут визуализировать данные в виде:

— карты кликов, отображающей клики по всем элементам страниц приложения, тем самым помогая выявлять элементы, к которым приме- нено наибольшее число кликов (максимально кликабельные), определять, какие элементы принимаются пользователем за ссылку, но таковой не являются. По картам кликов могут быть сформированы «тепловые карты», на которых места, соответствующие максимально кликабельным элементам, отображаются теплыми цветами, наименее кликабельным — холодными, а все другие — промежуточными цветами, зависящими от соответствующего им числа кликов;

— карты ссылок, отображающие число переходов по ссылке, долю пе- реходов относительно других ссылок на странице;

— карты скроллинга, отображающие время внимания посетителей определенным элементам страницы сайта; позволяют эксперту человеко- машинного взаимодействия подбирать оптимальную длину страницы и

правильно размещать важную информацию;

— карты путей по приложению, отображающие перемещение пользо-

вателей по страницам приложения; по этим картам эксперт человеко- машинного взаимодействия анализирует особенности последовательностей перемещения пользователей по страницам приложения.

Сервисы логирования представляют собой инструменты для сбора и визуализации данных для дальнейшего ручного анализа человеком, который чаще всего не является экспертом человекомашинного взаимодействия и не всегда может правильно интерпретировать полученные данные. Анализ человеком этих данных является менее трудоемким, чем анализ аудиовизуальной информации, получаемой при проведении традиционных исследований степени удобства использования. Для того чтобы собирать данные о взаимодействии пользователя с приложением, оно должно быть запущено в рабочую эксплуатацию, что при его низком качестве влечет за собой потерю потенциальных клиентов. Трудности со сбором данных испытывают приложения с низкой посещаемостью, что приводит к большим срокам проведения оценки степени удобства использования. При использовании логирования практически невозможно точно выяснить причины ухода пользователя с приложения. Уход пользователя может быть как следствием того, что он достиг своей цели в результате работы с приложением, так и, наоборот, он не смог достичь цели.

Один из главных недостатков указанных сервисов логирования — недостаточная автоматизация процесса анализа логов. Существуют работы, в которых степень автоматизации логирования существенно выше. Вот некоторые из наиболее известных.

Метод анализа, который позволяет обнаружить в логах различных пользователей одинаковые последовательности действий, характерные для большинства из них, представлен в работе [9]. Эти одинаковые для большинства пользователей последовательности рассматриваются как шаблоны, из которых складывается взаимодействие каждого пользователя с приложением. Затем, используя критерий скорости достижения пользователем цели и генетический алгоритм, выявляются эффективные и неэффективные шаблоны взаимодействия пользователя с приложением и на основе этого делается вывод о степени удобства использования приложения (чем больше число использованных эффективных шаблонов, тем выше степень удобства использования приложения).

Инструмент, который позволяет вручную задать наилучшую с точки

зрения ее создателя последовательность действий пользователя в процессе взаимодействия с приложением (эталонный лог), рассмотрен в работе [19]. После этого реальный лог, полученный в результате взаимодействия пользователя с приложением, сравнивается с эталонным логом по определенной мере. Чем ближе эталонный и реальный логи друг к другу, согласно этой мере, тем выше степень удобства использования приложения. Недостатком такого подхода является необъективность и интуитивность создания эталонного лога, а также увеличение временных затрат на его создание по мере повышения сложности приложения и числа его

модификаций.

Подход 2. Параметрический анализ свойств страниц. Осуществляет-

ся проверка отдельных параметров страниц приложения на соответствие рекомендациям по дизайну, визуальному шуму и сложности интерфейса, а также выявление областей страницы, на которые пользователь прежде всего обратит внимание при просмотре страницы.

Оценивать страницы интерфейса на основе параметров графики и текста предложено в работе [11]. Здесь утверждается, что соотношение графики и текста играет важную роль в улучшении внешнего вида страниц, что, в свою очередь, оказывает влияние на эстетические чувства пользователей, которые они испытывают при работе с приложением. Для эстетического анализа в работе [2] построена нейронная сеть, анализирующая, как конкретная комбинация элементов пользовательского интерфейса вебстраницы влияет на эстетические чувства пользователей при работе с ней. Результатом применения нейронной сети является оценка страницы по пя- тибалльной шкале (нейросеть имеет пять выходов, каждому из которых соответствует одна из пяти взаимоисключающих оценок), которая помогает веб-дизайнерам оценить степень удобства использования приложения. Для построения нейронной сети применялась обучающая выборка, содержащая 96 страниц с их эстетическими оценками. Эстетические оценки определялись субъективно экспертами человекомашинного взаимодействия с различным опытом работы. Задачи, для решения которых предназначались страницы, не учитывались. Для построения нейронной сети использовались только статические страницы.

Метод определения значимости элементов на страницах по движению глаз пользователя на различных страницах приложения предложен в работе [12]. Полученная информация может дать представление о том, как воспринимается интерфейс страницы и где находится самый важный или релевантный контент. Инструмент позволяет предсказывать важность различных элементов интерфейса, но без учета различных категорий пользователей и контекстов.

Подход 3. Моделе-ориентированное доказательство наличия свойств взаимодействия. Создается архитектура системы взаимодействия и формулируются свойства, которым эта архитектура должна удовлетворять. Затем эти свойства проверяются применительно к созданной архитектуре. Так, в работе [13] рассмотрены свойства мультимодальных интеллектуальных интерфейсов, реализация которых обеспечивает естественное «безбарьерное» взаимодействие пользователя с приложением. Если какое-либо свойство взаимодействия должно быть реализовано, то, во-первых, необходима его точная спецификация и, во-вторых, возможность формальной проверки этой спецификации применительно к конкретному взаимодействию. Кроме того, формулировка этих свойств должна носить объективный характер, признанный мировым сообществом.

Из известных реализаций моделе-ориентированного подхода доказательства наличия свойств можно также назвать работу [14], в которой представлен инструмент МЕМО, содержащий модель приложения и модель пользователя. Модель пользователя правильно взаимодействует с моделью приложения, если правила генерации ошибок, которые экспертно задает разработчик, не возникают в процессе выполнения задачи (в процессе взаимодействия). Другими словами, модель приложения не переходит в состояние, которое нежелательно для идеального взаимодействия. Задаваемые правила генерации ошибок являются субъективными правилами эксперта. Чем масштабнее интерфейс приложения (модель приложения), тем больше требуется времени, затрачиваемого на определение правил генерации ошибок. При внесении изменений в интерфейс приложения также необходимо редактировать правила генерации ошибок, добавляя новые и удаляя неактуальные.

В первом столбце таблицы перечислены, согласно приведенному аналитическому обзору [9], критерии для сравнения качества различных подходов оценки степени удобства использования приложений. В остальных столбцах для каждого из трех подходов знак «+» указывает на то, что подход удовлетворяет критерию, а знак «-» — не удовлетворяет. Согласно данным, приведенным в таблице, наибольшему числу критериев из рас- смотренных подходов удовлетворяет моделе-ориентированное доказательство наличия свойств. Остаются неудовлетворенными три критерия: 1) возможность объяснения поведения пользователей; 2) отсутствие необходимости приглашения экспертов человекомашинного взаимодействия; 3) отсутствие субъективного фактора.

Даже с учетом неудовлетворенности перечисленных трех критериев

подход 3 является наилучшим. Этот подход использует модели архитектуры и поведения системы, в которой реализуется взаимодействие. Если создатель какой-либо модели полагает, что ее архитектура и поведение обладает всеми необходимыми объективными свойствами, обеспечивающими необходимую степень удобства использования, то остается только доказать, что архитектура и поведение удовлетворяют этим свойствам.

Авторы настоящей работы полагают, что устранение указанных недостатков неудовлетворенности критериев (возможность объяснения поведения пользователей, отсутствие необходимости приглашения экспертов человекомашинного взаимодействия, отсутствие субъективного фактора) возможно на пути автоматизированного моделе-ориентированного доказательства (вывода) наличия объективных свойств архитектуры и поведения взаимодействующих моделей пользователя и приложения в рамках адек- ватного исчисления, включающего в себя:

— формальный язык адекватного описания шаблонов поведения пользователя и исполнителя в процессе взаимодействия;

— формальное описание на этом языке объективных свойств архитектуры и поведения взаимодействующих моделей, гарантирующих высокую степень удобства использования приложений;

— множества состоятельных правил вывода в этом языке;

— стратегию вывода, гарантирующую полноту доказательства нали-

чия свойств архитектуры и поведения взаимодействующих моделей.

Формальное описание объективных свойств архитектуры и поведе-

ния взаимодействующих моделей, гарантирующих высокую степень удобства использования приложений, позволит избавиться от необходимости приглашения экспертов человекомашинного взаимодействия и,

как следствие, от их субъективного мнения.

Стратегия вывода, гарантирующая полноту доказательства наличия

свойств архитектуры и поведения взаимодействующих моделей на основе состоятельных правил вывода, позволит выявлять степень удобства использования и причины неудобства.

Автоматизация стратегии вывода обеспечит простоту оценки степени удобства приложения после внесения изменений в процесс взаимодействия.

По мнению авторов работы, наиболее адекватным для автоматизации оценки степени удобства использования приложений является язык логики тайлов. Это мнение основано, в частности, на том, что тайл, как кон- струкция, позволяет, с одной стороны, естественным образом отображать элементы страниц, а с другой, — интерфейсы между элементами, осуществляемыми в результате определенных действий пользователя. Кроме того, из тайлов по определенным правилам могут компоноваться допустимые последовательности взаимодействия пользователя с системой, которые служат средством оценки степени удобства использования.

Принципы доказательства каких-либо свойств систем на основе логики тайлов известны достаточно давно [15]. Для того чтобы создать язык логики тайлов для рассматриваемого случая, необходимо, прежде всего, создать базовое множество тайлов, позволяющих представлять поведение приложения (агента-исполнителя) и респондента (агента-пользователя).

Далее приведены основные понятия, связанные с тайлами, затем представлена суть формального описания моделей приложения (агентаисполнителя) и респондента (агента-пользователя) с помощью тайлов. Введены критерии, по которым будет оцениваться степень удобства приложения. Приведен простой пример выявления степени удобства приложения в простом языке логики тайлов, а также направления дальнейших работ в области автоматизации оценки удобства приложений на ос- нове логики тайлов.

Тайлы и операции над ними. Достоинства логики тайлов связаны со структурой ее правил вывода, каждое из которых может быть представлено в виде тайла (Ше) (рис. 1)

В символическом виде тайл может быть представлен следующим образом:

Каждый тайл описывает отдельный элемент поведения системы в терминах возможных взаимодействий с внешней средой. В целом поведение системы выглядит как скоординированное взаимодействие отдельных тайлов. Тайлы могут композироваться горизонтально, вертикально и параллельно для того, чтобы задавать более сложное поведение систем. Та или иная логика тайлов получается в результате введения множества базовых тайлов и некоторой совокупности вспомогательных тайлов, над которыми могут совершаться операции композиции.

Модель оценки степени удобства. Для демонстрации принципов методологии автоматизации процесса оценки степени удобства пользовательских интерфейсов использована архитектура, моделирующая пользовательский интерфейс и состоящая из агента-исполнителя и агентапользователя. Агент-исполнитель имеет два канала: 1) входной канал ОЁ, являющийся выходным для агента-пользователя; 2) выходной канал ЕИ, являющийся входным для агента-пользователя (рис. 2). Во входной канал агента-исполнителя агент-пользователь может помещать сообщение, которое может забирать агент-исполнитель. В выходной канал агент- исполнитель может помещать сообщение для агента-пользователя. Здесь и далее агента-исполнителя и агента-пользователя будем называть исполнителем и пользователем соответственно.

Пара [ОЕ(т„), ЕО (т.)], где ОЕ(т„) означает, что в канале ОЕ находится сообщение ти, а в канале ЕО — сообщение ть, является в рас- сматриваемом случае конфигурацией, упоминаемой при определении тайла (см. рис. 1). Если безразлично, какое сообщение находится в том или ином

канале, то вместо указания конкурентного сообщения будем ставить знак «\_ ». Если в канале ничего нет, то будем использовать знак «©».

Взаимодействие исполнителя и пользователя начинается после получения от пользователя уведомления о начале взаимодействия. Если исполнитель дал согласие на взаимодействие, то оно продолжается. В противном случае взаимодействие прекращается. Пользователь прекращает взаимодействие путем отправки соответствующего сообщения исполнителю.

Архитектура пользовательских интерфейсов. Как уже было отмечено выше, пользовательский интерфейс — процесс, обеспечивающий взаимодействие пользователя с исполнителем (программным приложением). Для того чтобы формализовать это взаимодействие, введем следующие понятия и обозначения.

Архитектурой пользовательского интерфейса назовем пятерку

1= {Р,В,Е,ТЕ,Ти}, где Р — множество страниц р интерфейса, реР;

В(р) — множество блоков В страницы р, БеВ(р); Е) — множество элементов е блока В, ее Е(Б); Те, То — множества тайлов, описывающих поведение исполнителя и пользователя. Страницы, блоки и элементы имеют уникальное обозначение р, В, е с числовыми индексами, играющее

роль объектной (индивидной) переменной, которую будем называть просто объектом. Значение объектной переменной — объектная константа, являющаяся набором значений таких атрибутов, как пояснение, поле ввода, поле вывода, кнопка, изображение, список, видеоматериал, аудиоматериал, всплывающая подсказка, форма, цвет, яркость и др. Каждый атрибут может иметь достаточно сложную структуру. Как правило, многие атрибуты в их явном виде скрыты от пользователя.

Так, на рис. 3 показан простой пример страницы р! пользовательского интерфейса, содержащий блок В! = Преобразование систем счисления (ПОС) и элементы ев! = Исходная система счисления (ИСС), е› = Исходное число (ИЧ), ез = Результирующая система счисления (РСС), ед = Результирующее число (РЧ). Здесь текст «Преобразование систем счисления» поясняет назначение блока В!, тексты «Исходная систем счисления», «Исходное число» являются полями ввода элементов ет, е2, а тексты «Результирующая система счисления», «Результирующее число» — полями вывода элементов ез, е4.

Тайлы для описания поведения исполнителя. Поведение исполнителя описывается двумя типами тайлов.

Тайл «Указание исполнителем объекта, находясь в конфигурации, осуществляет восприятие сообщения т,, имену- ющее тот или иной объект, к которому пользователю необходим доступ, выдает реакцию !7., указывающую на объект !1., к которому пользователю разрешается доступ, устанавливает конечный выходной интерфейс

100 = те. Конечная конфигурация тайла «Указание исполнителем объекта» при этом становится конфигурацией: Тайл «Обработка объекта исполнителем (ООИ) всегда выполняется

после тайла «Указание исполнителем объекта», согласно горизонтальной композиции тайлов, и позволяет помещать реакцию исполнителя (сооб-

щение исполнителя) ть в выходной канал ЕП в ответ на восприятие собственной реакции те и в ответ на восприятие сообщения пользоваеля т„ тайлом «Указание исполнителем объекта», а также при условии, что его начальный и конечный входные интерфейсы совпадают с начальным и конечным выходными интерфейсами тайла «Указание исполнителем объекта». Начальная конфигурация тайла «Обработка объекта исполнителем» переходит в конечную конфигурацию:

'Тайлы для описания поведения пользователя. Поведение исполнителя описывается тремя типами тайлов.

Тайл «Инициализация пользователем взаимодействия (ИПВ)» позволяет начать взаимодействие пользователя и исполнителя. Согласно этому тайлу, его начальной конфигурацией является (каналы

пустые), его начальные входной и выходной интерфейсы и восприятие также являются пустыми. Тайл позволяет выдавать реакцию !ти, имену- ющую объект, с которого следует начинать взаимодействие. В результате конечная конфигурация тайла становится равной

а конечный выходной интерфейс о, =ти, означающий разрешение доступа к объекту Ти: Тайл «Указание пользователем объекта» позволяет формировать выходной конечный интерфейс пользователя в ответ на восприятие пользователя (сообщение исполнителя). Тайл, находясь в конфигурации , осуществляет восприятие сообщения т, именующее тот или иной объект, к которому разрешен доступ исполнителю с помощью тайла «Указание исполнителем объекта», начальный выходной интерфейс. ? о; = ть х выдает реакцию !т., запрашивающую доступ к объекту

те. Конечная конфигурация тайла «Указание исполнителем объекта» при этом становится конфигурацией

Тайл «Обработка объекта пользователем» выполняется после тайла «Указание пользователем объекта», согласно вертикальной композиции тайлов, и выдает реакцию пользователя ти (результат обработки объекта

т,) в выходной канал ОЕ в ответ на восприятие собственной реакции ! т, подтверждающей восприятие сообщения т, пользователя тайлом «Указание исполнителем объекта» и при условии, что его начальные входной и выходной интерфейсы совпадают с конечными входным и выходным интерфейсами тайла «Указание пользователем объекта». Начальная конфигурация тайла «Обработка объекта пользователем» пе- реходит в конечную конфигурацию

Пример взаимодействия пользователя и исполнителя в логике тайлов. Для страницы, пример которой приведен на рис. 3, введем сле-

дующие тайлы.

Тайл «Инициализация пользователем взаимодействия», задающий

конечный выходной интерфейс о, = ри, т. е. страницу ру, с которой должно начаться взаимодействие:

'Тайл «Указание исполнителем объекта», подтверждающий своей реакцией !р1 согласие исполнителя на обработку пользователем страницы |1:

'Тайл «Указание пользователем объекта», запрашивающий своей реакцией !Ё! доступ к блоку в:

'Тайл «Указание исполнителем объекта», подтверждающий своей реакцией !Б! согласие исполнителя на обработку пользователем блока в:

'Тайл «Указание пользователем объекта», запрашивающий своей реакцией !е, доступ к элементу е1:

Тайл «Указание исполнителем объекта», подтверждающий своей реакцией !е, согласие исполнителя на обработку пользователем элемента еи:

Тайлов (7)-—(12) достаточно для моделирования поведения пользователя и исполнителя в процессе навигации от страницы р! к элементу е!, так как на этом этапе никакой обработки объектов не осуществляется. Это поведение представлено композицией тайлов (рис. 4). Для компактного представления композиции тайлы представлены несколько в ином виде, а именно в виде трех строк, где первая строка содержит начальный входной и выходной интерфейсы и начальную конфигурацию, вторая строка — восприятие и реакцию, а третья строка — конечный входной и выходной интерфейсы и конечную конфигурацию.

Для дальнейшего моделирования взаимодействия пользователя и исполнителя требуется введение тайлов обработки объектов. Для всех элементов, показанных на рис. 3, схема обработки идентична и заключается в следующем. После указания исполнителем пользователю с помощью тайла «Установка исполнителем объектов» элемента, к которому разре- шен доступ, исполнитель с помощью тайла «Обработка объекта исполнителем» указывает поле для ввода информации пользователем. Пользо-ватель с помощью тайла «Обработка объекта пользователем» вводит в это поле информацию. Исполнитель после восприятия этой информации указывает пользователю на следующий элемент, к восприятию которого пользователь может перейти. Введем требуемые тайлы только для элемента е, и покажем, как они композируются для моделирования поведения пользователя и исполнителя.

Тайл «Обработка объекта исполнителем» выполняется после тайла (12), согласно горизонтальной композиции тайлов, и позволяет помещать реакцию исполнителя ИСС в выходной канал Е( в ответ на восприятие собственной реакции е! тайла (12) и при условии, что его начальный и конечный входные интерфейсы совпадают с начальным и конечным выходными интерфейсами тайла «Обработка объекта исполнителем». Начальная конфигурация тайла «Обработка объекта исполнителем» переходит в конечную конфигурацию

Тайл «Обработка объекта пользователем» выполняется после тайла «Обработка объекта исполнителем», согласно вертикальной композиции тайлов, и формирует реакцию пользователя в выходной канал ОЕ в ответ на восприятие реакции ИСС тайла (13), указывающую значение поля

ИСС (объекта ИСО). Например, этой реакцией может быть Двоичная система счисления (ДСО). Начальная конфигурация тай- ла «Обработка объекта пользователем» переходит в конечную конфигурацию

'Тайл «Указание исполнителем объекта» своей реакцией !е› указыва-

ет, что следующим элементом, подлежащим обработке пользователем,

является элемент е>:

ботке элемента е1. Переходы к другим элементам рассматриваемого примера и их обработке аналогичны продемонстрированным. Для компактной ком- позиции тайлы представлены в несколько другом виде (см. рис. 4).

Оценка степени удобства использования пользовательских интерфейсов в логике тайлов Критерии оценки степени удобства. Критерии оценки степени удоб-

ства пользовательских интерфейсов могут быть качественными и количественными и позволяют делать вывод, насколько интерфейс удобен, а также сравнивать его с другими проектами пользовательских интерфейсов.

Простейшие критерии оценки степени удобства пользовательских интерфейсов — оценки достижения пользователями своих целей при работе с приложениями. Дж. Нильсен (]. №е]5еп, [5]) предложил три такие оценки:

1) пользователь достиг своей цели;

2) пользователь столкнулся с проблемами, но достиг своей цели;

3) пользователь не достиг цели.

Цели в логике тайлов могут формулироваться как целевые тайлы, до-

стижимость которых проверяется наличием композиции, в результате

которой целевой тайл достигается. Если такой композиции не существует, то цель считается недостигнутой. В случае достижения цели простейшей оценкой степени удобства может быть, например, число тайлов, требуемых для достижения цели. Чем меньше тайлов для этого требуется, тем выше степень удобства использования. Более точные критерии оценки степени удобства использования требуют учета различных качественных и количественных свойств пользовательских интерфейсов, таких как сложность достижения конечной цели, сложность достижения промежуточных целей, выявление причин, приводящих к слишком высокой сложности достижения целей, причина достижения цели и т. п. Все это требует создания значительных библиотек тайлов, их классификации по группам пользователей, введения частных критериев оценки выполнения отдельных тайлов, по которым формируются общие оценки, и т. п.

Задача, которая поставлена в настоящей работе, не предполагает детального рассмотрения всех критериев оценки степени удобства использования пользовательских интерфейсов. Здесь рассмотрен только один критерий, который оценивает степень удобства использования приложения по результату выполнения задачи (выполнена — приложение имеет высокую степень удобства, в противном случае низкую).

Заключение. Приведен аналитический обзор подходов к автоматизации оценки степени удобства использования приложений, в рамках которого выявлены преимущества и недостатки каждого подхода. Предложен формальный моделе-ориентированный метод оценки степени удобства использования, основанный на использовании логики тайлов и позволяющий устранить недостатки известных подходов. В логике тайлов создается модель взаимодействия пользователя и исполнителя (приложения). Мо-

В.В. Девятков, Е.А. Типсин

дель композируется из тайлов в соответствии с определенными правилами, которые служат средством оценки удобства использования. Принципы методологии проиллюстрированы на примере одностраничного приложения. В дальнейшем будет создана полномасштабная модель взаимодействия в логике тайлов, учитывающая особенности поведения различных групп пользователей при взаимодействии с реальными приложениями, увеличено число критериев оценки степени удобства использования. Фактически, оценка степени удобства использования в логике тайлов — извлечение знаний о поведении взаимодействующих агентов, крайне востребованных как предмет аналитики на современном этапе развития интеллектуальных информационных систем. Изложены принципы извлечения знаний о поведении взаимодействующих агентов, основанные на использовании паттернов локального поведения агентов в логике тайлов, состав которых формирует общее поведение взаимодействующих агентов. Представление этих принципов осуществляется на уровне композиции тайлов, но без формального доказательства свойств реального поведения взаимодействующих агентов. Конкретизация такого доказательства свойств для случая выявления наличия определенных свойств интеллектуальных интерфейсов в языке логического программирования РКОГОС дана в работе [13].

В основе процедур доказательства свойств взаимодействия агентов на языке РКОГОС лежит механизм унификации, который может служить средством автоматизированной проверки свойств поведения агентов, представляемого различными композициями тайлов. Кроме того, по аналогии с

той же работой интересующие свойства поведения взаимодействующих агентов могут быть формально выражены на модальных языках для последующей их формальной проверки.

**ЛИНГВИСТИЧЕСКАЯ ЭКСПЕРТОЛОГИЯ**

**КАК НАПРАВЛЕНИЕ ТЕОРЕТИЧЕСКОЙ**

**И ПРИКЛАДНОЙ ЛИНГВИСТИКИ**

современной науке выделяют лингвист-

тику теоретическую (научную) и приклад-

ную. Объектом исследований в теории-

чешской лингвистики является лингвистическая

теория, основанная на построении системы ар-

аргументов и научных концепций, доказывающих

определенную гипотезу; прикладная лингвис-

тика специализируется на разработке методов

решения практических задач, связанных с изу-

чением языка, а также использования лингви-

стических теорий на практике [1].

Областью исследований теоретического язы-

сознания является изучение языка в теорети-

ческом аспекте, синтезирующее данные о язы-

ке. Теоретическое языкознание является осно-

вой, базой для практического, прикладного язы-

кознания.

Результатом теоретического знания являет-

ся практическое изучение отдельного языка, ко-

торый оказывает воздействие на выработку прин-

ципов и методов изучения отдельного конкрет-

ного языка, изучение и преподавание языка.

Методологической основой для прикладно-

го языкознания является теоретическое языкоз-

нание. П

е Прикладное языкознание использует на

практике выработанные и полученные теорети-

ческим языкознаниям анания о языке.

Целью первой задачи является существен-

ное повышение уровия объективности реауль-

татов лингвистических исследований, т.

т.е. экс-

пертизы должны быть доступны и носсоздавае-

мы другими нсследонателями.

Прикладная лингвистика

область науч-

ных ананий, научающая строение и функцио-

нирование и моделирование функций языка в

нелингвистических областях знаний и в различ-

ных областях деятельности человека, т.е. это

направление в науке, занимающееся разработ-

кой и осуществлением практического ли

глингвис-

тического обесшечения для равнообразных для

сфер производства, науки, культуры и др.

Прикладная лингвистика

совокупность

многих языковедчесних наук, данные которых

используют различные отрасли че

человеческой

деятельности [2].

Задачи прикладной лингвистики базируют-

ся на исследованиях [2] теоретического харак-

тера, формулирующих общие принципы, а по-

лучаемые результаты используются в качестве

исходных оснований для теоретических пост-

роений.

Область изучения языка прикладной линг-

пистики

это решение определенных практи-

ческих аадач.

В последние годы возникли повые отрасли

прикладной лингвистики, таковой является и

юридическая лингвистика, предметом изучения

которой является соотношение языка и права.

Данное от

ное отношение организует подотраель

юридической линганстики

линганстическую

экспертологию.

Лингаистическая экспертология занимает-

ся теоретическим и методическим обесшечени-

ем производства лингаистических экспертиа.

На примере линганстической экспертологии

можно увидеть тесное переплетение теоретичес-

кой и прикладной линганстики, оба направле-

ния лингвнстики в

к взаимосвяваны и дополияют

друг друга.

Недостаточно исследованные лингаистичес-

кие теории, на которых основываются совре-

менные экспертные исследования, подробно

исследуются и разрабатываются в конкретные

методики производства лингвистических экс-

пертиа в рамках различных категорий дел.

Процесс развития теории лингвистической

экспертизы нераарывно связан с направлени-

ями теоретического и прикладного характе-

ра, а также е необходимостью решения важ-

ных задач.

Целью второй задачи является деятельность,

связанная с переводом юридической термино-

логии в лингвистические термины, и наоборот.

Третьей немаловажной падачей яв

ейявляется

исследование объективных и устойчивых вааи-

мосвязей между элементами вер

вербальной и

ауальной составляющих текста в семиотичес-

ком понимании.

По исследуемым объектам пыделяют экспер-

тизы звучащей речи, письменного текста и вер-

бально-визуальные экспертизы. Э

Это комплеке

исследований, направленный на ус

на установление

принадлежности звучащей речи ко

конкретному

лицу; в акспертивах используются специальные

инструментарии; в

в письменных эк

ы экспертилах

применяются различные методики, такие, как

морфологический анализ, синтактический ана-

лип и семантический.

Вербально-вилуальные экспертизы сочетают

вербальную и невербальную информацию, на-

пример, анализ изображения и текста.

Судебная лингвистическая экспертиза свое

развитие получила во второй половине 90-х го-

дов ХХ в.

В лингвокриминалистике сформировались

такие виды, как: экспертиза по делам диффа-

мации (распространения сведений, порочащих

честь и

и достоинство), экспертиза недобросовес-

тной и недостоверной рекламы, пр

противоза

ной агитации, экспертива словесного экстремиз-

ма и др.

На современном этапе развития лингвокри-

миналистика обладает широкой научно-методи-

ческой базой для производства судебно-лингвн-

стических экспертиа. Ее основу составляют об-

щая теория судебной экспертивы (экспертоло-

гия) и частная теория лингвистической экспер-

тизы, являющаяся направлением прикладной

лингпистики.

Необходимо отметить, что судебная лингви-

стическая экспертива продолжает развиваться

и совершенствоваться. Учеными детально раз-

работаны и продолжают разрабатываться типо-

вые перечни вопросов, необходимых при про-

изводстве лингвистических экспертиз, изложе-

ны теоретические основы анализа текста.

В современном обществе фи

фиксируется мно-

жество повых речевых деликтов, которые тре-

буют законодательного оформления, в связи с

чем возникает потребность в создании принцип-

пиально новых экспертиз.

Например, вырабатывается основа для произ-

водства лингвистических экспертиз по ст. 205.2

УК РФ - публичные призывы к осуществлению

террористической деятельности, публичное оправ-

здание терроризма или пропаганда терроризма.

Анализируя количество возросших речевых

правонарушений и преступлений, начиная от

распространения клеветы, оскорблений и до

пропаганды экстремизма и терроризма, на дан-

ном этапе к виду судебных экспертиз устанав-

сливаются высокие стандарты в плане научно-

сти и методологической адекватности.

**Информативность N-грамм в пределах слова**

1

Взаимная информация

Элементы теории информации интенсивно используются в

лингвистике. Понятие взаимной информации использовано в данном

сообщении для изучения информативности отдельных букв, биграмм и

триграмм в слове.

Пусть L - язык с алфавитом А, и V - список слов wi (i = 1,m)

одинаковой длины n. С каждым словом свяжем число fi - относительную

частоту встречаемости слова w1. Предположим, что 0 < f $ 1 иf = 1.

Последовательность длины s, состоящую из букв алфавита А, будем

называть n-граммой g\*. В данном сообщение рассматриваются n-граммы

размеров 1, 2 и 3. N-граммы g1 также будут называться отдельными

буквами, g2 - биграммами, g3 - триграммами. Позиция n-граммы в слове

определяется позицией первой буквы входящей в п-грамму. Например, в

слове "домашний", биграмма "аш" находятся на позиции 4, и отмечается

как

Пусть W - случайная величина, значением которой является какое-

либо слово из списка V с вероятностью p(wi) = fi. Пусть Gf- другая

случайная величина, значениями которой являются n-граммы размера s,

которые могут появиться в k-й позиции в слове wi.

Количество информации, получаемой о величине W после подсчёта

величины Gf будем называть взаимной информацией этих величин. В

согласии с [Ash, 1990, раздел 1.5], количество взаимной информации

определяется по формуле

I(WiGE) = 1(GEIW) = H(GE) - H(GEIW).

(1)

Здесь H(G) - это неопределённость по поводу значения величины G

которая выражается как

H(Gp) =- p(g3)logp(gj).

где p(8j) - вероятность того, что G примет значение g\* в слове wj.

Второе слагаемое в (1), H(GEIW) - это условная неопределенность

переменной Gk, порождаемая событием W. Интуитивно, это слагаемое

должно быть равно нулю, так как нет неопределенности в буквах данного

слова. В предыдущей работе [Нормантас, 2013] это утверждение

доказывается в случае отдельных букв. Посколько это доказательство

легко обобщить к случаю п-грамм, оно в данный доклад не включается.

Скомбинировав (1), (2) и приравняв H(GfIW) к 0, получим

1(WIG) =-p(e,)logp(e)).

(3)

Данная формула показывает, что количество информации об условной

вероятности события (Gf|W) равно неопределенности события,

состоящего в появлении n-граммы g\* в слове w, на позиции k.

Для подсчёта p(g1) построим подмножество слов из V, состоящее из

слов,

имеющих

п-грамму

g

B

позиции

k:

V' = {w € V|w[k…k+s - 1] = g°}. Пусть частоты f', каждого слова

подмножества V' будут равны соответствующим частотам слов fj. Тогда

вероятность p(g) может быть выражена следующим образом:

V

p(g) =

(4)

Пример

Рассмотрим некоторый язык с алфавитом A = {a,b,c,d}. Табл. 1

содержит список четырехбуквенных слов с частотой их встречаемости.

Таблица 3. Пример списка слов с частотами

Слова

Относительные

частоты

аааа

0,25

baaa

0,15

сааа

0,1

bbaa

0,1

cbaa

0,3

dba

0.1

Применим формулу (4) для построения табл. 2. Например, биграмма

ba появляется в позиции 2 в трех словах: bbaa, cbaa и dbaa. Поэтому

вероятность р(G = ba) = 0,1 + 0,3 + 0,1 = 0,5.

Таблица 4. Вероятности появления биграмм в каждой позиции

Биграммы

Позиции в слове

2

aa

0,25

0,5

ba

0,15

0,5

са

bb

cb

0,1

0,1

0,3

0

(

db

0,1

0

Теперь воспользуемся формулой (3) для подсчёта информативности

биграмм в каждой позиции. В рассматриваемом примере информативность

биграмм в первой позиции подсчитывается следующим способом:

1(W|G2) = - 0,25 log 0,25 - 0,15 log 0,15 - 0,1 log 0,1 - 0,1 log 0,1 - 0,3 log 0,3 -

0,1 log 0,1 2,4 бит

Так как все слова на второй позиции имеют аа или ba с равными

вероятностями 0,5, то мы получаем ровно один бит информации:

1(W|G2) = - 0,5log 0.5 - 0,5 log 0,5 - 0 log 0 - 0log 0 - 0 log 0 - Olog 0 = 1 бит.13

B

третьей

позиции:

I(W|G;) = -1 log1 - Olog 0 - 0log0 - 0log 0 - 0log0 - 0log 0 = 0 бит.

Этот

результат следует из того факта, что все слова в нашем примере

заканчиваются на биграмму аа. Следовательно, узнав её, мы не получаем

полезной информации о слове.

3

Статистическое исследование

Для определения информативности п-грамм в словах английского,

литовского, русского, таджикского и узбекского языков и искусственного

языка эсперанто использовались те же самые частотные словари, как и

предыдущей работе [Нормантас, 2013] и [Усманов и Нормантас, 2012] с

добавлением данных узбекского языка, который прежде не изучался.

При подготовке данных к статистической обработке слова каждой

коллекции разделялись на группы, составленные из слов одинаковой

длины. Затем были подсчитаны вероятности появления каждой программы

на каждой позиции с помощью формулы (4). Далее с помощью (3) была

подсчитана взаимная информация.

Статистическое исследование было осуществлено с помощью

компьютерной программы, написанной автором

на

языке

программирования Scala. Код программы может быть выслан желающим.

4 Результаты

Результаты исследований в случае отдельных букв представлены на

рис. 1, 2 и 3 по отдельности для слов, состоящих из 5, 10, 15 букв. Они

показывают информативность букв в различных позициях в словах для 6

упомянутых языков.

На рис. 1, 2 и 3 видно, что вторая буква менее информативна по

сравнению с первой, третьей и следующими буквами до середины слова.

Например, буквы на позициях 1, 3 и 4 слов английского языка, состоящих

из 7 букв, проявляют около 4,2 бит информации о слове. Буква на позиции

2 несёт 3,8 бит. Для слов таджикского языка такого же размера эта разница

еще значительней: позиции 1, 3 и 4 доставляют 4-4,3 бит информации, а

позиция 2 - около 2,8 бит. Похожая картинка имеет место для всех языков

и длин слов.

Отметим также, что взаимная информация начинает постепенно

падать после середины слова (для слов более чем из 3 букв). Эта картина

особенно характерна для литовского, таджикского и эсперанто языков,

менее характерна для слов английского языка и почти не заметна для

русского языка. Для всех языков, кроме русского, последняя буква слова

несет наименьшую информацию в сравнении с другими позициями, за

исключением позиции 2.

На рис. 4 видно, что биграммы на позициях 3, 4 и 5 несут наибольшую

информацию в словах, состоящих из 10 букв всех шести языков. В случае

триграмм (см. рис. 5) самыми информативными являются триграммы на

позициях 3 и 4. Информативность биграмм и триграмм постепенно падает

после середины слова.

Выводы

Во всех исследованных языках наблюдаются

похожие

закономерности информативности отдельных букв: вторая буква слова

несет меньше информации по сравнению с первой и третьей буквами;

информативность букв постепенно понижается во второй половине слова.

Информативность биграмм и триграмм показывает похожие

закономерности, но в сглаженной форме.

В данной работе в первые изучалась информативность букв и n-грамм

узбекского языка. Из изученных языков, узбекский является единственным

неиндоевропейским языком (кроме эсперанто). Результаты показывают,

что картинка информативности этого языка заметно отличается только в

начале слова. Например, первые три биграммы и первые три триграммы

слов передают больше информации в сравнении с другими языками. В

случае отдельных букв, различия между узбекским и другими языками

незначительны.

Следует заметить, что хотя эсперанто - сконструированный язык, для

него имеют место те же закономерности, как и для других естественных

языков.

Необходимо проведение дальнейших исследований для объяснений

закономерностей, обнаруженных в данной статье. Автор предполагает, что

понижение информативности второй буквы может быть связано с высокой

частотой встречаемости гласных на этой позиции. Понижение

информативности между серединой и концом слова может иметь

отношение к статистическим свойствам суффиксов и окончаний.

Результаты этих исследований можно использовать, например, при

разработке или улучшении алгоритмов для поиска в текстовых данных.

Другая возможная область приложения - облегчение чтения.

# СРАВНИТЕЛЬНЫЙ АНАЛИЗ САПР НА ПРИМЕРЕ ПРОЕКТИРОВАНИЯ ТРЕХМЕРНОЙ МОДЕЛИ КОРПУСА СУДНА

Развитие высокотехнологичной судостроительной отрасли предусматривается государственной програм- мой Российской Федерации «Развитие судостроения и техники для освоения шельфовых месторождений» [1] и обеспечивается разработкой новых технологий и проектов гражданской морской техники и техники для освоения шельфовых месторождений. Вместе с тем одним из приоритетов научно-технологического развития Российской Федерации [2] является переход к передовым цифровым производственным технологиям, новым материалам и спо- собам конструирования, который позволит получить научные и научно-технические результаты и создать техноло- гии, являющиеся основой инновационного развития внутреннего рынка продуктов и услуг, устойчивого положения России на внешнем рынке.

Согласно указа Президента РФ «О национальных целях развития Российской Федерации на период до 2030 года» одной из таких целей является «Цифровая трансформация», в задачи которой входит увеличение объема и скорости применения информационных технологий в отечественных решениях по сравнению с показателем 2019 года.

Необходимость изучения систем автоматизированного проектирования подтверждается значительным упрощением труда инженеров-конструкторов, что также позволит обучающимся совершенствовать свои компетен- ции как в профессии, так и в процессе обучения в вузе. Федеральные государственные образовательные стандарты высшего образования поколения 3++, в частности, по специальности 26.05.01 Проектирование и постройка кораб- лей, судов и объектов океанотехники [3], обязуют вузы определять профессиональные компетенции на основе про- фессиональных стандартов, соответствующих профессиональной деятельности выпускников.

Так в основной образовательной программе высшего образования по специальности 26.05.01 Проектиро- вание и постройка кораблей, судов и объектов океанотехники (специализация: Проектирование и постройка судов и объектов океанотехники) Севастопольского государственного университета из профессионального стандарта «Спе- циалист по проектированию и конструированию в судостроении» [4] при формировании профессиональной компе- тенции ПК-3 «Способен использовать современные системы автоматизированного проектирования (САПР) при со- здании сложных систем, трехмерного моделирования судов и объектов океанотехники, выполнения инженерных расчетов с обеспечением электронного документооборота на всех стадиях жизненного цикла» частично выделены следующие обобщенные трудовые функции, соответствующие профессиональной деятельности выпускников:

* C – «Разработка и модернизация проектов, техническое сопровождение производства судов, плавучих сооружений, аппаратов и их составных частей», в части трудовых действий, касающихся использования САПР, тру- довых функций «C/01.6 Разработка и согласование комплектов технологической документации при проведении тео- ретических и экспериментальных» и «C/02.6 Разработка эскизных, технических проектов судов, плавучих сооруже- ний, аппаратов и их составных частей»;
* D – «Организация и выполнение плана по разработке комплектов проектно-конструкторской докумен- тации на постройку и модернизацию судов, плавучих сооружений, аппаратов и их составных частей», в части трудо- вых действий, касающихся использования САПР, трудовой функции «D/02.6 Организация и выполнение плана по разработке комплектов проектно-конструкторской документации на постройку и модернизацию судов, плавучих сооружений, аппаратов и их составных частей»;
* E – «Руководство инновационными конструкторскими исследованиями, созданием и модернизацией проектов судов, плавучих сооружений, аппаратов и их составных частей», в части трудовых действий, касающихся использования САПР, трудовой функции «E/01.6 Руководство исследованиями в области создания новых образцов судов, плавучих сооружений, аппаратов и их составных частей в соответствии с техническим заданием».

AutoCAD (Autodesk) – одна из известных и распространенных базовых САПР в мире и России с 1982 года, но не является единственной программой для проектирования изделий, судов и т.п., существуют и многие другие [5].

Цель статьи – выполнение сравнительного анализа САПР AutoCAD (Autodesk), Inventor (Autodesk) и NanoCAD Механика (ООО «Нанософт разработка») на примере проектирования трехмерной модели корпуса судна в рамках подготовки инженеров-судостроителей по дисциплине «Автоматизация проектирования корабля».

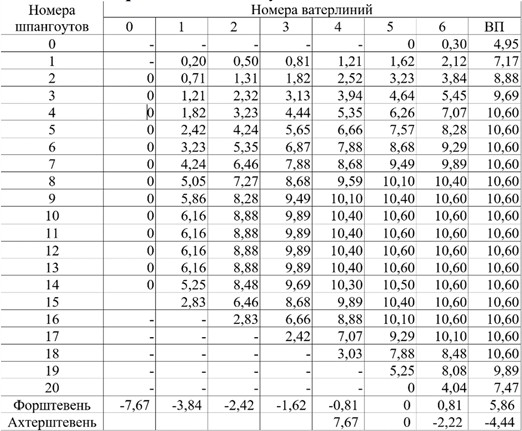
Для достижения поставленной цели были решены следующие задачи: выявление характерных особенно- стей при проектирования гражданской морской техники и техники для освоения шельфовых месторождений в дан- ных САПР; определение перспективности применения одной из двух предложенных САПР с целью снижения тру- доемкости способа построения трехмерной модели корпуса судна.

**Методика построения трехмерной модели корпуса судна в САПР AutoCAD с использованием тео- ретического чертежа.** Теоретический чертеж (ТЧ) строится на основе таблица плазовых ординат (табл. 1), который является одним из основных конструкторских документов и представляет совокупность, вычерченных в масштабе линий, образуемых пересечением теоретической поверхности корпуса судна тремя семействами взаимно перпенди- кулярных плоскостей, параллельным главным плоскостям судна. ТЧ – основа для создания трехмерной модели суд- на. Координаты точек, которые составляют каждый шпангоут (табл. 1), задают его форму [6]. Для построения шпан- гоутов (кривые линии) используются сложные примитивы – «Полилиния» и «Сплайн». «Полилиния» – соединенные единым примитивом отрезки различной длины, которые требуется редактировать (сглаживать). Однако при сглажи- вании необходимо применять исключительно функцию «Сгладить», без «Сглаживания по сплайну», иначе получае- мая кривая пройдет не по контрольным точкам, что повлечет последующую несогласованность линий ТЧ. «Сплайн»

– гладкая кривая, которая проходит через заданный набор точек. AutoCAD работает с частным случаем сплайнов – неоднородными рациональными B-сплайновыми кривыми (NURBS). Для дальнейшего построения трехмерной мо- дели судна требуется замкнутый контур всех шпангоутов, для чего вычерчивается и верхняя палуба (ВП), имеющая соответствующую погибь. Чтобы получить более сглаженную трехмерную модель судна, необходимо выполнить построение проекций «Бок» и «Полуширота» с последующим согласованием всех проекций ТЧ.

## Таблица 1

***Ординаты шпангоутов и штевней***



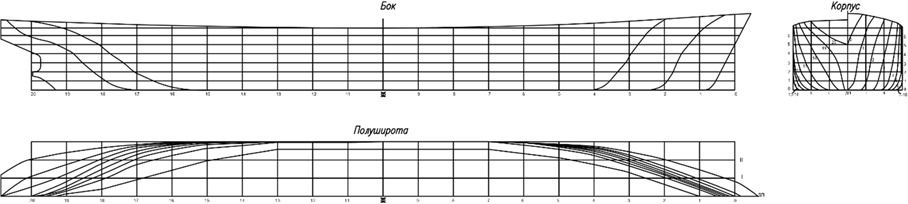
Используя ТЧ (рис. 1), как минимум две его проекции: «Корпус» и «Бок», возможно подготовить все не- обходимые части корпуса судна к их изменению посредством команд 3D моделирования [7].

Первым этапом необходимо с использованием команд «3D-поворот» и «Переместить относительно точ- ки» расположить проекции «Корпус» и «Бок» перпендикулярно друг другу, аналогично расположению плоскостей мидель-шпангоута и диаметральной плоскости (рис. 2) для дальнейшего формирования каркасной модели корпуса судна. Начало системы координат необходимо поместить в точке пересечения плоскостей мидель-шпангоута, диа- метральной и основной.

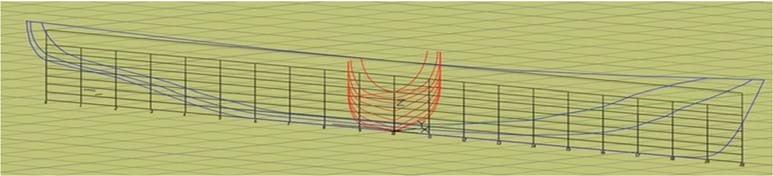
Следующий этап – с помощью команды «Зеркало» представить две проекции «Корпус» с полными сим- метричными шпангоутами носовой и кормовой оконечности (рис. 2).

Сформировать трехмерный теоретический чертеж можно различными способами и последовательностью действий, например, опираясь на теоретическую секцию проекции «Бок» используя команду «Переместить» после- довательно изменить местоположение всех шпангоутов проекции «Корпус» на соответствующие им места проекции

«Бок». В результате получаем трехмерную каркасную модель судна из 21 шпангоута, линий форштевня и ахтер- штевня проекции «Бок». Данный каркас станет основой для трехмерной модели. С целью повышения точности по- лучаемой трехмерной поверхности можно продолжить построение каркасной модели, переместив ватерлинии про- екции «Полуширота» на соответствующие места проекции «Бок». Построение каркасной модели завершим соедине- нием всех крайних верхних точек шпангоутов командой «Сплайн», тем самым получив линию борта. В результате данных преобразований получим трехмерный каркас корпуса судна (рис. 4).



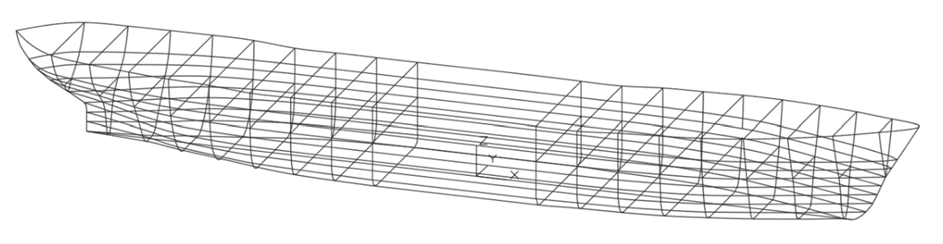
## Рис. 1. Теоретический чертеж судна



***Рис. 2. Расположение проекций***



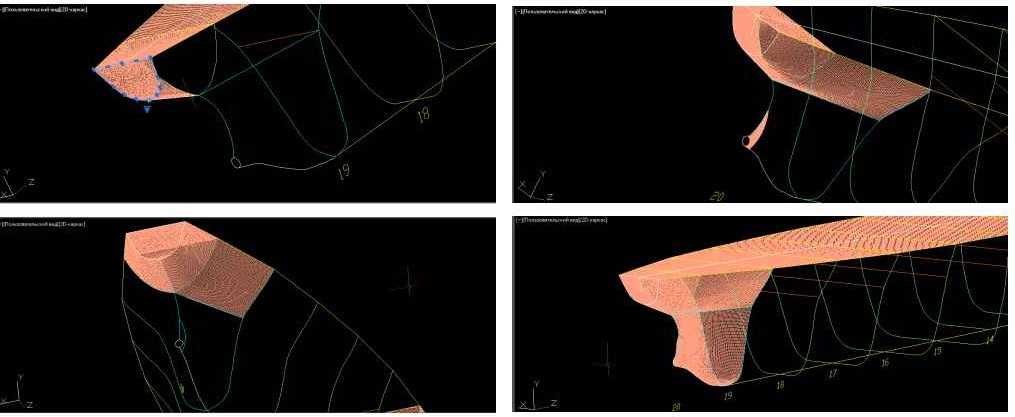
## Рис. 3. Расположение проекций



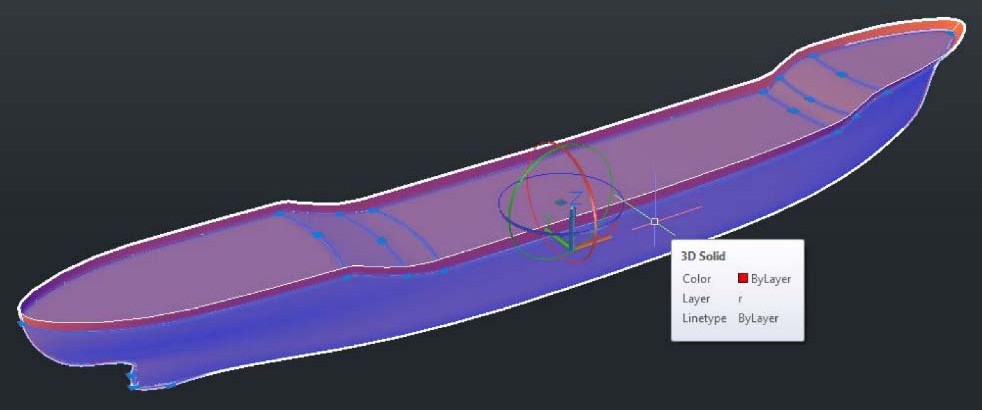
***Рис. 4. Каркасная модель корпуса судна***

Построение трехмерной поверхности судна выполняется с использованием полученной каркасной моде- ли, для этого используются различные средства AutoCAD, работающие с созданием и редактированием поверхно- стей. Наиболее точным вариантом построения является обтягивание поверхностью двух расположенных рядом шпангоутов, например, 2–3 и т.д. с использованием поверхности Кунса, реализованной функцией «П-кромка». В этом случае может потребоваться более детальная проработка оконечностей, из-за сложной кривизны поверхностей (рис. 5). Аналогично выполняется построение трехмерной поверхности палубы судна из попарно соединенных па- лубных линий, созданных путем соединения вершин одного шпангоута.

Возможен второй вариант построения трехмерной поверхности судна – с помощью команды «Лофт», функция которой заключается в создании 3D тела (поверхности) в пространстве между несколькими поперечными сечениями. Для построения более точной поверхности судна лучше использовать отрезки шпангоутов между двумя соседними ватерлиниями, так как используемые командой «Лофт» поперечные сечения определяют форму получае- мого тела (поверхности). Полученная таким образом трехмерная поверхность корпуса судна будет отображать его архитектурные особенности и криволинейность формы корпуса. Затем с использованием команды «Поверхнапол- нить» создается сплошное твердое тело (3D Solid) – трехмерная модель корпуса судна (рис. 6).

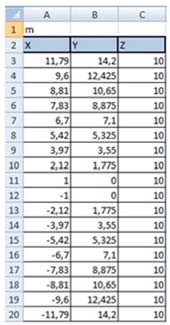


## Рис. 5. Проработка кормовой оконечности



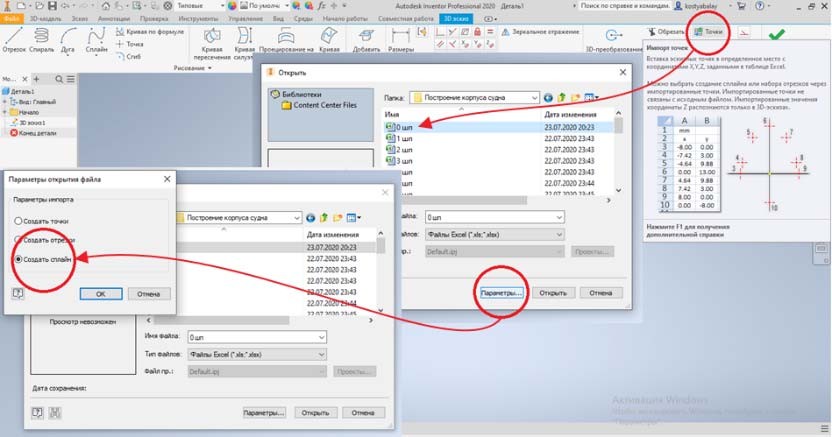
***Рис. 6. Трехмерная модель корпуса судна в AutoCAD***

**Методика построения трехмерной модели корпуса судна в САПР Inventor с использованием коор- динат точек, описывающих форму шпангоута.** Построение 3D-модели корпуса судна в Inventor начинается с фор- мирования MS Excel файлов с координатами каждого шпангоута, для чего данные таблицы плазовых ординат (табл. 1) предварительно должны быть представлены в определенном формате по трем известным координатам (X, Y, Z) (рис. 7).



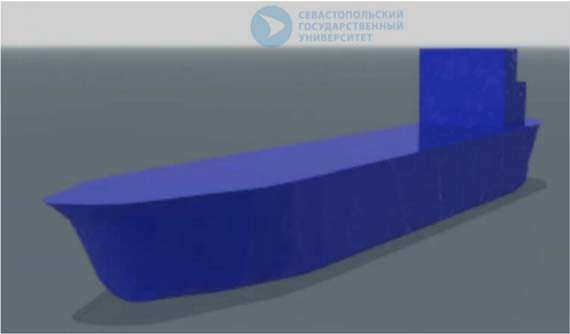
## Рис. 7. Пример файла шпангоута для импорта

В рабочее пространство Inventor последовательно от носа в корму загружаются файлы, созданные для каждого шпангоута (рис. 8).



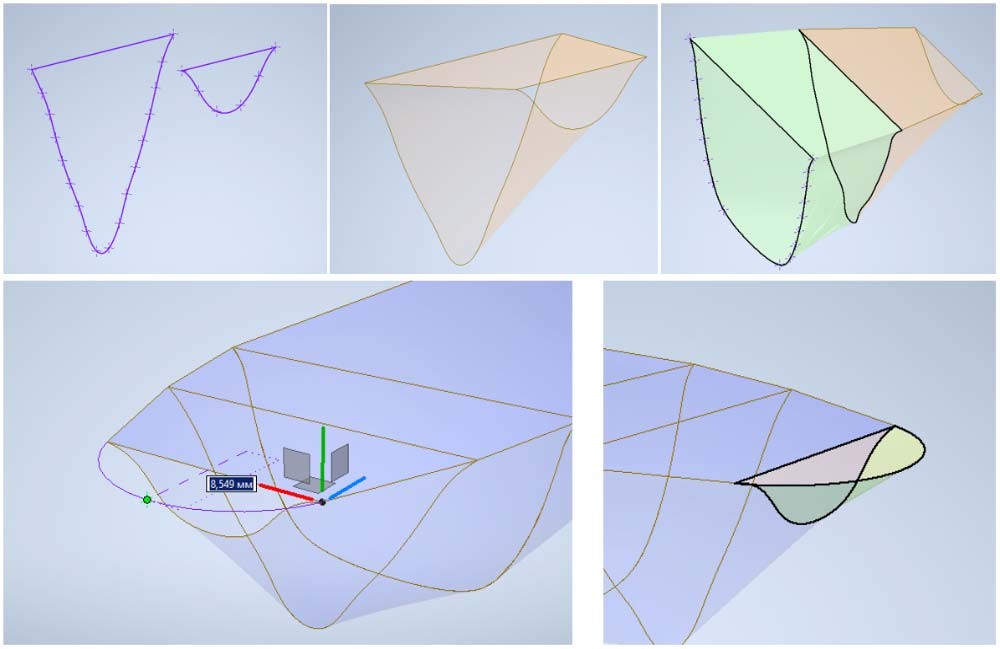
## Рис. 8. Импорт шпангоутов в Inventor

Затем шпангоуты последовательно соединяются с использованием функции «Лофт» в трехмерную модель судна (рис. 9). Также можно построить надстройку, используя команды создания и преобразования 3D-объектов.



## Рис. 9. Трехмерная модель корпуса судна в Inventor

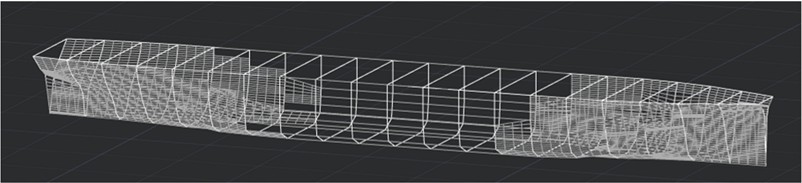
Только после построения всех шпангоутов выполняются дополнительные построения для 3D моделиро- вания ахтер- и форпиковых частей корпуса судна (рис. 10).



## Рис. 10. Построение ахтерпика

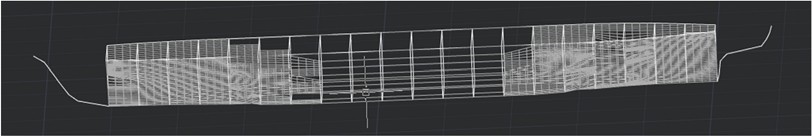
**Отечественные САПР.** Санкционная политика сказалась на САПР AutoCAD и Inventor, которые стали не доступны на территории России. Учитывая направленность на суверенизацию реализуемых проектов Севастополь- ский государственный университет реализует дальнейшую подготовку обучающихся с использованием САПР NanoCAD Механика.

В отличие от отписанных выше САПР, в NanoCAD не требуется построение теоретического чертежа и создание файлов для импорта каждого шпангоута. Необходимо подготовить только один файл с координатами точек шпангоутов на базе (табл. 1). Построение шпангоута выполняется с помощью команды «Полилиния» путем ввода координат в командную строку. Дальнейшее построение выполняется в среде «3D-инструменты» функцией «Вытя- гивание по сечениям» (рис. 11).



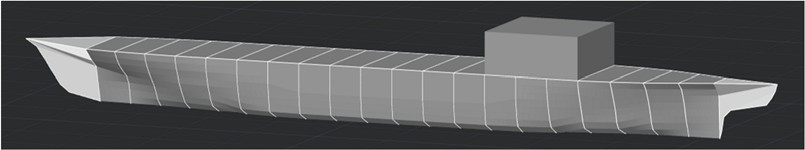
## Рис. 11. Построенная поверхность с 0 по 20 шпангоуты

Ахтерштевень и форштевень строятся аналогично шпангоутам, но требуют доработки с помощью функ- ции «3D повернуть» (рис. 12).



## Рис. 12. Форштевень и ахтерштевень

Далее выполняются дополнительные построения достраиваются носовая и кормовая оконечности с ис- пользованием команд «3Д грань» или «3Д сеть», а также – надстройка (рис. 13).



## Рис. 13. Трехмерная модель корпуса судна в NanoCAD Механика

Данные методики обучения можно применять для любых инженерных специальностей при построении различных объектов машиностроения.

**Заключение.** Навыки работы в различных САПР повышают уровень компетентности и личностных воз- можностей инженер-конструктора, благодаря чему он получает преимущество при трудоустройстве в организации, осуществляющие проектирование и разработку проектов судов и объектов океанотехники для освоения шельфовых месторождений.

Созданные трехмерные модели корпусов различных судов и морских технических сооружений при по- мощи систем автоматизированного проектирования упрощают и обеспечивают решение практических инженерных задач. Экспортируемые готовые модели из AutoCAD и nanoCAD являются численными моделями для различных программных комплексов, например, таких как Anchored Structures, в которых возможно промоделировать поведе- ние судна или объектов океанотехники для освоения шельфовых месторождений при изменчивых ветро-волновых условиях, создать оптимальные параметры системы удержания, учесть динамику якорных линий, исследовать пове- дение цифровых моделей сооружений при воздействии в длительном временном интервале нагрузок, соответствую- щих реальным ситуациям, что невозможно при использовании аналитических расчетных соотношений. Применение трехмерного моделирования обеспечивает формирование новых подходов к обоснованию проектных решений.

**СИСТЕМА ПРЕДИКТИВНОГО ВВОДА КАК СРЕДСТВО ПОВЫШЕНИЯ ЭФФЕКТИВНОСТИ НАБОРА ТЕКСТА**

В основе современного производства лежит взаимодействие человека и техники. Если брать во внимание офисных работников, то под техникой пони- мается компьютер. Для повышения экономической эффективности производ- ства необходимо повышение эффективности работы человека и компьютера.

Основными факторами, влияющими на эффективность работы за ком- пьютером, являются:

* 1. цветовая гамма рабочего места [1];
  2. эргономичная офисная мебель [2];
  3. соблюдение режимов труда и отдыха [3];
  4. планирование рабочего времени;
  5. мотивация [4];
  6. эффективное человеко-машинное взаимодействие.

Первые три пункта из представленного списка регулируются Санитарно- эпидемиологическими правилами и нормативами СанПиН 2.2.2/2.4.1340-03 [5], в которых перечислены санитарно-эпидемиологические правила и нормативы

«Гигиенические требования к персональным электронно-вычислительным машинам и организации работы».

Планирование рабочего времени производится на основе «Типовой ин- струкции по охране труда при работе на персональном компьютере ТОИ Р-45-084-01» [6], списка задач и сроков их выполнения. Эффективное планиро- вание позволит качественно и своевременно достигать поставленных целей.

Мотивация повышает производительность труда, тем самым увеличи- вает прибыль компании.

Под человеко-машинным взаимодействием понимается система взаимо- действия между пользователем и компьютером на уровне пользовательского ин- терфейса, состоящая из программного и аппаратного обеспечения [7].

Значительная часть пользователей в силу своих профессиональных обязанностей производит перевод больших объемов текстовой информации в электронный вид. В качестве основного средства ввода информации приме- няется клавиатура, но также могут быть использованы:

* речевой ввод текста [8], примером реализации которого являются программы Горыныч, Диктограф (российские разработки фирмы VM TECH), MedSpeak (система распознавания речи для врачей-рентгенологов от IBM), Sakrament ASR Engine (разработка «Сакрамент», позволяющая управлять действиями компьютера или другого электронного устройства с помощью голосовых команд, озвучивать электронный текст);
* рукописный ввод с помощью графического планшета или планшетно- го компьютера: программы PenReader, MyScript. Данные программные про- дукты позволяют производить перевод рукописного текста в печатный вид. В качестве инструмента для создания рукописных заметок может использо- ваться Microsoft Office [9].

Тем не менее клавиатура остается наиболее часто используемым сред- ством человеко-машинного взаимодействия. Современное расположение букв на клавиатуре явилось наследием печатных машинок, первое упомина- ние о которых встречается еще в XVIII в. [10]. Первоначально буквы распо- лагались в строго алфавитном порядке и при печати возникали технические проблемы – рычаги близко расположенных букв не успевали вернуться в ис- ходное положение и сцеплялись между собой. В дальнейшем конструкция машинок была переработана, и наиболее встречаемые комбинации букв в ан- глийских словах были разнесены по клавиатуре, что снизило вероятность по- ломки механизма. Современная кириллистическая раскладка клавиатуры пи- шущих машинок была придумана в США в конце XIX в. Клавиатура разраба- тывалась как эргономичная: под самыми сильными и быстрыми указатель- ными пальцами были размещены наиболее часто используемые буквы, а под слабыми безымянными пальцами и мизинцами – более редкие. С появлением электрических печатных машинок, а затем и компьютеров, клавиатура не претерпела каких-либо изменений, так как это привело бы к переподготовке большого количества персонала, обученного для работы на пишущей машин- ке. Можно сделать выводы, что, несмотря на заложенную эргономичность в русскую раскладку, расположение букв на клавиатуре не является опти- мальным, так как нагрузка по пальцам распределяется неравномерно, прихо- дится часто менять положение пальцев для нажатия той или иной клавиши, так как расположение клавиш учитывает механизм работы механической пе- чатной машинки. Но, несмотря на это, возможно повысить скорость набора текста, повысив тем самым эффективность работы. Могут быть использованы следующие методы:

1. применение методик слепой печати [11];
2. использование программ для сохранения шаблонов текста

(PhraseExpress, Flashpaste) и дальнейшей вставки их в редактируемый текст;

1. использование программ для замены шаблонов текста (Microsoft Word, Punto Switcher, Breevy) на заданные текстовые фрагменты;
2. использование систем предиктивного ввода.

Особое внимание следует обратить на системы предиктивного ввода. Данные системы позволяют завершать вводимые пользователем слова. В том случае, если слово было найдено в словаре, то оно, а также его падежные формы, будут предложены пользователю. Если слово обнаружено не было, то формируется альтернативных список на основе предположений о допу- щенной ошибке. Система позволяет исправлять распространенные граммати- ческие ошибки, что повышает уровень грамотности и внимательности поль- зователей. Автоматически повышается скорость набора текста, так как поль- зователю не придется набирать слова полностью, что будет полезно для тех, кто не владеет скоростным набором. В случае необходимости пользователь может загрузить в систему дополнительный специализированный словарь, ко- торый расширит функционал системы. Кроме того, возможно добавление фраз, которые также будут предлагаться при наборе первого слова фразы.

В рамках написания магистерской диссертации был создан прототип системы предиктивного ввода. Для тестирования системы была проведена серия экспериментов с целью определения скорости посимвольного набора текста и с использованием предиктивного ввода. В эксперименте приняло участие 15 человек. Им было предложено произвести ввод текста, состоящего из 700 знаков, при этом время ввода фиксировалось в итоговой таблице. По результатам экспериментов было получено среднее время набора текста. При наборе стандартным методом оно составило около 3,5 мин, с примене- нием предиктивного ввода – 2,5 мин. Скорость набора текста напрямую зави- сит как от подготовки участника эксперимента, так и от набираемого текста. Эффективность применения системы предиктивного ввода возрастает, если набираемые слова имеют длину более четырех букв.

Проведенные исследования показывают, что системы предиктивного вода повышают скорость набора текста, позволяют повысить эффективность человеко-машинного взаимодействия и работоспособность в целом.

**Система контроля достоверности текстовой информации на основе n-граммных парсинговых моделей**

1. **Постановка задачи контроля и коррекции текстовой информации.** Функционирование любых информа- ционных систем в существенной степени зависит от достоверности передачи сообщений, которая снижается вследствие ошибок человека-оператора, влияния помех в системах связи, сбоев электронного оборудования и по- грешностей систем сканирования и распознавания. Причем в системах, предназначенных для обработки большого объема текстовой информации, например в системах электронного документооборота (СЭД), искажения проявля- ются в основном в виде орфографических ошибок различной кратности (однократные, двукратные, *n*-кратные) [1, 2]. В научных исследованиях, посвященных компьютерной обработке текстовой информации, многократно под- черкивается (главным образом, в виде постановки задач, а не решения проблемы) эффективность использования *n*-граммной модели естественного языка (ЕЯ) для решения задач контроля достоверности передачи и обработки текстов [3]. Однако решение проблемы контроля и коррекции ошибок в текстах на основе *n*-граммной модели, хотя и представляется наиболее перспективным, мало изучено с точки зрения обеспечения качества обработки

текстовой информации, особенно представляемой на узбекском языке.

Следует отметить, что проблема контроля и коррекции ошибок в текстах на основе *n-*граммной модели ЕЯ связана с решением комплекса теоретических и практических задач, среди которых наиболее важными являются: исследование вероятностей появления ошибок для получения априорной базы *n-*грамм; разработка методик оцен- ки достоверности информации при равномерных и неравномерных моделях *n-*кратных искажений; парсинговое моделирование структуры слова на основе словоформ, разработка вероятностных моделей кластеризации и поиска объектов контроля; компьютерная реализация моделей и алгоритмов контроля и коррекции *n-*граммных ошибок, оптимизация параметров функционирования компонентов систем контроля орфографии и оценка качества ее функционирования.

В настоящей работе представлены результаты исследований, направленных на решение указанных задач.

1. **Модели условной вероятности *n-*граммных искажений.** Определение вероятностей *n-*граммных ошибок связано с обработкой большого объема статистических данных и трудоемкими вычислениями, так как важной особенностью *n-*грамм является то, что их число растет экспоненциально относительно длины n. Следовательно, необходимо специальное моделирование процессов вычисления статистики и вероятностей *n-*граммных ошибок. В работе [4] исследованы закономерности распределения ошибок передачи текстовой информации, предложены способы моделирования и алгоритмы для выявления искаженных элементов (букв, слов) в тексте, кластеризации,

поиска, структуризации; получены частотные характеристики *n-*грамм при большом объеме информации, которые применялись в процессах апробации систем контроля и коррекции орфографических ошибок. Результаты прове- денных экспериментальных исследований использовались при установлении закономерностей появления искаже- ний в информации, определении условных вероятностей *n-*граммных ошибок для решения задач генерации и син- теза текстов из речи.

Заметим, что используемые экспериментальные данные получены на основе теоретических положений при допущении о равновероятности *n-*граммных ошибок, что позволило получить простые математические выражения для проведения аналитических исследований. В связи с этим представим равномерную модель *n-*граммных ошибок.

* 1. *Равномерная модель n-граммных ошибок.* Общая вероятность ошибок, обусловленных ошибками челове- ка-оператора, сканирования и распознавания, искажениями в каналах связи, сбоями электронных средств переда- чи и обработки информации, обозначим через *Р*. Процесс перехода *i* -го сообщения в  *j* -е, как правило, задается

стохастической матрицей переходных вероятностей

*P* , которая считается основным показателем при оценке

*i i*

 

достоверности информации в любой системе передачи и обработки данных.

Общая вероятность ошибок при передаче *i* -го сообщения равна

*P**i*  1 *P**i**i*   *P i j* ,

 

 *j* (*i*  *j* )

где

*P*  − вероятность правильного приема *i* -го сообщения. Средняя вероятность ошибки находится осредне-

нием условных вероятностей ошибки по всему ансамблю сообщений:

*i i*

*P*   *P**i*

*i*

 *P i j* .

 *j* (*i*  *j* )

 

(1)

Формула (1) является двумерной моделью оценки вероятности *Р*, связанной с оценкой монограммной вероят-

ности *P* и диграммной вероятности



*i*

*P* . В случае учета статистики трехграмм необходимо исследовать

*i i*

 

вероятности переходов

*i* *j*   , а при статистике *n-*грамм требуется вычислить вероятности набора

*i* ,*i*'' ,L,*in*   *j* ' , *j* '' ,L, *j n* .

* 1. *Математическая модель условных вероятностей n-грамм.* Пусть задан некоторый язык

*L*(*VT* )

с конеч-

ным алфавитом *VT*

 {*wi*} , где *wi*

– отдельный символ, *VT* – множество цепочек (строк) конечной длины, состоя-

щих из символов алфавита *VT* , *n-*грамма на алфавите *VT*

представляет собой цепочку длиной *n*.

Как правило, *n-*грамма может совпадать с каким-либо высказыванием, быть его подстрокой или вообще не

входить в

*L*(*VT* ) . Например, если алфавит – это буквы ЕЯ плюс дефис, а высказывания – это слова ЕЯ, то

*n-*грамма – это последовательность из *n* символов (букв и дефисов), принадлежащая одному слову; если высказы- вания – это тексты, то *n-*грамма – это последовательность из *N* слов одного текста; если алфавит – это морфологи- ческие описания слов ЕЯ плюс знаки пунктуации, а высказывания – это соответствующие фразам и грамматиче- ски допустимые морфологические описания входящих в них слов, то *n-*грамма – это последовательность грамма- тически допустимых описаний *n* подряд стоящих слов.

Обозначим через *C*(*w*)  *C*(*w*1*w*2 …*wn*1*wn* ) число вхождений строки

*w*  *w*1*w*2 …*wn*1*wn*

в совокупность всех текстов рассматриваемого языка. Предположим, что алфавит рассматриваемого языка содер- жит буквы (без учета регистра) и знаки пунктуации, тогда как пробел, переход на новую строку и начало текста – специальные разделители, не входящие в алфавит. Высказывание в таком языке – это неделимая последователь- ность символов

*p*(*w*)  *C*(*w*) .

*C*(*w*\*)

*w*\*

Вероятность

*p*(*w*)

появления *n-*граммы

*w*  *w*1 …*wn*

равна отношению

*C*(*w*)

к общему числу экземпляров

всех встреченных в совокупности *n-*грамм. В частности, для монограмм, т. е. отдельных символов, имеем

*p*(*wi* ) 

*C*(*wi* )

*C*(*w j* )

*w j*

,

где *wi*

– символ алфавита *V* ; числитель – количество вхождений *wi*

в совокупность всех слов, а сумма в знаме-

нателе − общее число символов в ней.

*T*

Если вероятности появления символов в любой позиции цепочки независимы и одинаково распределены, то вероятность *n-*граммы

*n*

*p*(*w*1...*wn*)   *p*(*wi* ) .

*i*1

Это, в частности, означает, что любые перестановки символов строки ность.

*w*  *w*1 …*wn*

имеют одну и ту же вероят-

Если достоверного априорного знания о равенстве распределений символов в разных позициях строки не су-

*j j*

ществует, следует ввести условные вероятности. Тогда, обозначив через позиции строки стоит символ *w*\* , получим условную вероятность строки

*j*

*p*(*w*  *w*\* )

вероятность того, что в *j*-й

*p*(*w*\* …*w*\* )  *p*(*w*

1

*n j j*

 *w*\* *w*

 *w*\**i*  *j*) *p*(*w*  *w*\**i*  *j*) . (2)

Формула (2) служит также априорной основой при построении алгоритмов автоматической кластеризации слов системы контроля орфографии. В связи с этим ниже рассматриваются решение задач кластеризации слов и специфические подходы для получения эффективных алгоритмов кластеризации слов и просмотра строки текста.

*j j i i*

1. **Математическая модель кластеризации слов.** Можно предложить одностороннюю (например, просмотр строки текста слева или справа) и вместе с тем двухстороннюю модель кластеризации слов, где строка текста по- очередно прослеживается и слева, и справа. Установлено, что алгоритм кластеризации на основе односторонней модели позволяет значительно быстрее, без существенных потерь обеспечить выделение слова и разбиение слов на классы. Рассмотрим кластеризацию на основе односторонней модели при просмотре строки текста с левой стороны.

Корпус слов до некоторой степени редуцируется отображением каждого из *Nv*

слов в *Nc*

классы, где

*Nc*  *Nv* .

При этом основным условием является представление *n-*граммной статистики для полученного корпуса классов слов. Для отображения слова в классы данная модель представляется в виде

*w*  *C*  *C*(*w*) ,

где слово *w* может принадлежать только одному классу. В данной работе кластеризация в классы проведена для слов узбекского языка. При этом в качестве критерия оптимизации кластеризации использована мера наибольше- го подобия, определенная в тренировочном множестве. Заметим, что ключевыми моментами кластеризации слов в классы являются парсинговое моделирование структуры слова на основе словоформ [5], выработка методов поис- ка и оценка их вероятностей при принятых моделях.

* 1. *Расчет компонентов вероятностей односторонней модели.* Компонент вероятности односторонней моде- ли классов представляется в виде

*P*(*wi* )  *P*(*wi* / *C*(*wi**n*1 ),…,*C*(*wi*1 )) . (3)

По модели (3) текущее слово обрабатывается в зависимости от предыдущих слов, отображенных в классы. Следо- вательно, вероятность очередного символа строки также задается в зависимости от предшествующих ему (*n*  1)

символов:

*p*(*wn w*1 …*wn*1 ) . Тогда

*p*(*w*1 …*wn*1*wn* )  *p*(*wn w*1 …*wn*1 ) *p*(*w*1 …*wn*1 ) .

В терминах вероятности "быть справа" для триграмм имеем

*p*(*w*1...*wn*)  *p*(*wn* | *w*1...*wn*  1) *p*(*wn*  1 | *w*1...*wn*  2) *p*(*wn*  2 | *w*1...*wn*  3) *p*(*w*2) ,

в общем случае можно записать

 *n* 

*p*(*w*1...*wn*)   *p*(*wk* | *w*1...*wk*  1)  *p*(*w*1) . (4)

 *k* 2 

Введя фиктивный символ "начало" и приняв, что *p*(*w*1 *w*0 ) есть *p*(*w*1) , выражение (4) представим в виде

*n*

*p*(*w*1...*wn*)   *p*(*wk* | *w*1...*wk*  1)

*k* 1

(5)

Таким образом, марковская цепь (*n* 1) -го порядка оказывается моделью *n-*граммы, а задача оценивания ста- тистических параметров *n-*граммы – хорошо изученной задачей оценивания параметров марковской цепи.

Следует отметить, что вследствие наличия множества возможных типичных строк символов значения вероят-

ностей, вычисленные по формуле (5), очень малы и их использование связано с большими трудностями вычисли- тельного характера. Поэтому для упрощения вычислений выражение (4) целесообразно записать в виде

*n*

log *P*(*w*1*w*2 *w*3...*wn* )  log *P*(*w*1 *w*1*w*2 *w*3...*wn* ),

*k* 1

однако для определения (6) необходимы многократные вычисления:

(6)

*n*

*P*   *pi* .

*i*1

Задавая log(*a*  *b*)  log *a*  log(1 *b* / *a*) , вычисляем log *P* по следующему рекурсивному алгоритму: Начало: log *P*  log *p*1

Рекурсия: *a*  max(log *pn* , log *pn*1 )

*b*  min(log *pn* , log *pn*1 )

log *pn*1  *a*  log(1 exp(*b*  *a*))

Конец: log *P*  log *pn* .

Для проведения аналитических исследований эффективности систем контроля орфографии также представляет интерес получение упрощенных оценок вероятностей *n-*грамм.

1. **Упрощенные оценки условных вероятностей *n-*грамм.** Как правило, оценкой вероятности *n-*граммы слу- жит частота ее встречаемости:

*p*ˆ(*w* | *w*

...*w*

)  *f* (*w* | *w*

...*w*

)  *C*(*wi* *n*...*wi* 1*wi* ) .

*i i*  *n i*  1

*i* *n i* 1

*i*

*i i*  *n i*  1

*C*(*w* ...*w w* )

Поскольку частота появления ошибок в виде *n-*грамм представляет случайную величину, частотные характери- стики можно интерполировать для получения их осредненных оценок.

Общая оценка условных вероятностей *n-*грамм также оценивается с учетом частоты их встречаемости:

*p*ˆ(*w* | *w* ...*w*

)  *f* (*w* | *w* ...*w*

)  *C*(*wi**n* …*wi*1*wi* ) ,

*i i**n i*1

*i i**n i*1

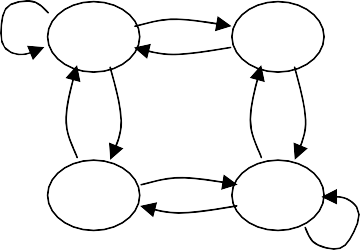
*C*(*wi* )

где *C*(*wi* ) – общее число *n-*грамм, встреченных в последовательности.

В качестве методики получения упрощенной оценки вероятностных переходов предложим упрощенную зна- ково-основанную диграммную модель.

*а б* 4.1. *Диграммная модель ошибок.* Рассмот-

*w* рим диграммную модель, которая требует ве-



*S*1

*S*2

*w*

*S*3

*S*4

роятностей формы *P*(*w w* ) . Обозначим час-

*wi*  *Aw* : *P* (*wi**S*1)

*i j*

*w*44

*w*12

*wi*  *Aw* : *P* (*wi**S*2)

*w*21

*w*22

*Fi j*

*Fi j*

тоты символа или слова через *Fi*, а условные частоты *Fi|j* представим как число следования символа *j* за символом *i*. Тогда оценку макси- мальной вероятности запишем в виде

*P*(*w w* )   .

Рис. 1. Цепь Маркова (*а*) и вероятностные состояния цепи (*б*)

*i j*

 *Fi j Fj*

*i*

Рассмотрим цепь Маркова (рис. 1, *а*), в которой переходы происходят по стрелкам с вероятностями *рij*. На рис. 1, *б* показаны текущие состояния *Si*, выдаваемые символами *i*; причем каждое состояние имеет собственное распределение вероятности.

В данном случае вероятности переходов устанавливаются по формуле

*Fi j* 1

*P*(*wi wj* ) 

.

*Aw*   *Fi j i*

Следует отметить, что вероятности перехода зависят от состояния цепи Маркова, которое является постоян- ным числом. Например, если в момент времени *t* = 0 мы в состоянии *s* с вероятностью перехода *pss*, то вероятность постоянства этого состояния оценивается экспоненциальным разложением

*P*(сост  *s*)  exp(*t* /  )

с характерным временем   1/ log *pss* . Это время прямопропорционально масштабу длины, если модель выдает символы равной длины.

Вероятность переходов между состояниями определим по следующей формуле:

1 *pss*  1 exp(1 /  )  1 /  (  1) .

Большие значения τ исключают переходы в масштабе длины знака и являются желательным поведением системы. Однако если характерное время τ установлено меньшим или равным 1010 знаков, то это не будет подавлять переход.

В случае если известно большее количество данных об индивидуальных частотах символа, то по моделям мо- нограммы лучше определяются вероятности диграмм. Поэтому введем процедуру интерполирования диграммных распределений более простой моделью монограммы:

*P*(*wi wj*

  *Fi*  (1 ) *Fi j* ) ,

*N Fi*

где *N –* общее число символов;  определяется эмпирически.

Модель монограммы с однородным распределением может сглаживать и более сложные модели, например триграммную модель.

4.2. *Триграммная модель ошибок*. С целью упрощения оценки условных вероятностей триграмм будем исполь- зовать линейную интерполяцию

*p*ˆ(*wi* | *wi*2 *wi*1 )  *q*2 *f* (*wi* | *wi*2 *wi*1 )  *q*1 *f* (*wi* | *wi*1 )  *q*0 *f* (*wi* ) ,

где

*f* (*wi* …)

– выборочные оценки, которые определяются следующим образом:

*f* (*w* | *w w*

)  *C*(*wi*2 *wi*1*wi* ) ,

*f* (*w* | *w*

)  *C*(*wi*1*wi* ) ,

*f* (*w* )  *C*(*wi* ) .

*i i*2

*i*1

*C*(*w w* )

*i i*1

*C*(*w* ) *i C*

*i*2 *i*1

*i*1

Здесь *C* – общее число экземпляров всех символов, остальные величины в знаменате- лях – число для соответствующих (*n–*1)-грамм, за которыми следует допустимый в рассмат- риваемом языке символ. В каждом слове это число для (*n–*1)-грамм на единицу меньше, чем для *n-*грамм, в случае если число (*n–*1)- грамм больше нуля, в противном случае это число равно 0.

Для упрощенной вероятностной оценки авторами данной работы предложен метод рекурсивной линейной интерполяции относи- тельных оценок частоты различных порядков

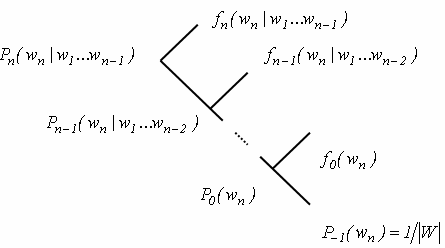


Рис. 2. Рекурсивная линейная интерполяция

*f k* (), *k*  0... *n* . На рис. 2 приведена рекурсивная схема смешивания,

на основе которой запишем выражение для вычисления условных вероятностей

*Pn* (*wn* | *w*1 ,..., *wn*1 )  (*w*1 ,..., *wn* )*Pn*1 (*wn* | *w*1 ,..., *wn*2 )  (1 (*w*1,..., *wn* )) *f n* (*wn* | *w*1 ,..., *wn*1 ) ,

*P*1 (*w*)  uniform(*W* ) ,

где

*w*1 ,…, *wn*1

* контекст порядка *n*, когда предсказано

*wn* ;

*f n* (*wn w*1,…, *wk* )

* относительная частотная оценка

порядка *k* для условной вероятности *Pn* (*wn w*1 ,…, *wk* ) :

*f k* (*wn* | *w*1,…, *wk* )  *C*(*wn* , *w*1,…, *wk* ) *C*(*w*1,…, *wk* ),

*k*  0…*n*,

*C*(*wn* , *w*1,…, *wk* )  

*wk* 1*Wk* 1

… 

*wn* *Wn*

*C*(*wn* , *w*1,…, *wk* , *wk* 1 …*wn* 1 ),

(*w*1,…, *wk* ) [0,1],

*k*  0…*n*

*C*(*w*1,…, *wk* )   *C*(*wn* , *w*1,…, *wk* ),

*w**W*

* коэффициенты интерполяции.

Заметим, что коэффициенты (*w*1 ,…, *wk* )

сгруппированы в эквивалентные классы на основе диапазона, в ко-

торый попадает индекс *C*(*w*1,…, *wk* ) ; для каждого эквивалентного класса диапазоны индекса установлены таким образом, что статистически достаточное число событий (*wn* | *w*1,…, *wk* ) попадает в пределы этого диапазона.

Предложенная выше методика оценки условных вероятностей ошибок в текстах на основе *n-*граммной модели позволяет оценить их значения в виде осредненных характеристик появления однократных, двукратных и трех- кратных ошибок, которые являются важными факторами при оценке качества применения способов контроля дос- товерности текстовой информации.

1. **Оценка достоверности информации.** Поскольку в системах контроля орфографии основным элементом проверки и коррекции является слово текста, при построении таких систем на первый план выдвигаются задачи распознавания слова и его элементов. В [2, 6] разработаны интерполяционные и экстраполяционные алгоритмы распознавания элементов текста, в том числе слова. Ниже рассмотрены методики получения вероятностных моде- лей выделения слов в строке текста в предположении, что распознавание слова осуществляется по указанным ал- горитмам статистического распознавания.
   1. *Вероятностная модель распознавания элементов текста.* Начнем с выделения строки слов

*W*ˆ  arg max *P*( *A* | *W* )*P*(*W* ) ,

&

*W*

где *A* обозначает наблюдаемое слово; *P*( *A* / *W* ) – условная вероятность того, что слово в строке *W* представляется

в виде образа *A* ; *P*(*W* ) – априорная вероятность появления слова в тексте *W*. Исследование заключается в оценке

значения вероятности *P*(*W* ) .

Пусть строка задается набором слов *W*  *w*1, *w*2 ,..., *wn* , тогда по теореме Байеса имеем

*n*

*P*(*W* )   *P*(*wi* | *w*1, *w*2 ,…, *wi*1 ) .

*i*1

Заметим, что пространство параметра

*P*(*wk* | *w*1 , *w*2 ,…, *wk* 1 )

очень широко, причем слова *wi* принадлежат сло-

варю *V* большого размера. Для распознавания представляется предыстория лентного класса, определяемого функцией Ф(*Wk* 1 ) , а также

*Wk*  *w*1 , *w*2 ,…, *wk* 1

в виде эквива-

*n*

*P*(*W* )   *P*(*wk* | Ф(*Wk* 1 )) .

*k* 1

Тогда задача определения вероятности выделения слов сводится к нахождению эквивалентных классификаторов Ф и методов оценки *P*(*wk* | Ф(*Wk* 1 )) .

Поскольку для распознавания слова в тексте предлагается использование *n*-граммной модели языка, функция эквивалентной классификации представляется в виде

Ф(*Wk* 1 )  *wk* *n*1 , *wk* *n*2 ,…, *wk* 1 .

Следует отметить, что определение формы Ф(*Wk* 1 ) предшествует решению задачи оценки *P*(*wk* | Ф(*Wk* 1 )) , яв- ляющейся критерием качества распознавания и соответственно контроля достоверности элементов текста.

* 1. *Оценка качества распознавания слова.* Качество системы контроля орфографии, как правило, определяет-

ся достоверностью распознавания слова на основе словаря словоформ. Поэтому при решении поставленной зада- чи важным моментом является определение показателя ошибки распознавания слова. Для этого находим наиболее

благоприятное слово, произведенное алгоритмом распознавания *W*ˆ

и истинной последовательностью слов. Затем

подсчитывается число неправильных слов *W*ˆ

в общем числе слов в *W* .

Особенность контроля текстовой достоверности заключается в том, что при построении алгоритма распозна- вания и соответственно системы контроля орфографии используется большой объем словарей словоформ и пре- фиксов слов, при этом алгоритм позволяет выделить несоответствующие слова, обеспечить эквивалентную клас- сификацию префикса слова и использовать априорную информацию при предсказании следующего слова.

Как одну из оценок качества распознавания слова можно использовать энтропию основного источника инфор- мации

где *H w*

*N*

*Hw* (*M* )  exp(1 / *N* ln[*PM* (*wk* | *Wk* 1 )]) ,

*k* 1

* энтропия слова в строке; *N* − число слов в общем объеме словаря тестируемого материала.

1. **Парсинговое моделирование структуры слова на основе словоформ.** Аргументы приведенных моделей эквивалентной классификации и оценки качества распознавания определяются на основе изложенного ниже ново- го механизма применения *n-*граммной структурированной модели естественного языка, который включает проце- дуры парсингового кодирования и поиска последовательности контролируемых слов.
   1. *Парсинговое кодирование.* Пусть *W* – предложение длиною *n* слов, к которому добавим в начало  *s*  и в конец  /*s*  , так что получим *w*0  *s*  и *wn*1  *s*  .

Обозначим через *Wk*  *w*0 …*wk*

число *k*-префиксов слова в предложении, тогда *WkTk*

будет *k*-префиксом слова-

парсинга. Для кодирования последовательности слов построим дерево слова-парсинга. Отметим, что *k* -префикс слова-парсинга содержит только те бинарные поддеревья, диапазоны которых полностью включены в *k* -префиксы

слова, за исключением

*w*0  *s*  . Отдельные слова вместе с их позиционными признаками (POS-признак) могут

быть расценены как корневые деревья.

На рис. 3 показан полный парсинг некоторого слова. Схема определяет бинарный парсинг

( *s*  *SB*)(*w*1,*t*1)…(*wntn* )( /*s* , *SE*) , где последовательность *SB/SE* − отличительный POS-признак для

 *s*  /  /*s*  соответственно с ограничениями, что ( /*s* ,*TOP*)

* единственно дозволенный заголовок;

(*w*1,*t*1 )…(*wntn* )( /*s* , *SE*) формирует элемент, возглавляемый ( /*s* ,*TOP*) .

Парсинги определяются, когда ( /*s* ,*TOP*) – заголовок любого элемента, который доминирует (над  /*s*  ),

но не  *s*  .

На рис. 4 представлена схема взаимодействия модулей системы кодирования для построения алгоритма распо- знавания элементов на основе парсингового дерева. Система кодирования состоит из трех модулей:

1. "Предсказатель слова" предсказывает следующее слово *wk* 1 , данное *k*-префиксом слова-парсинга, затем пе- редает управление на "Tаггер";
2. "Taггер" предсказывает POS-признак

*tk* 1

следующего слова, данного *k*-префиксом слова-парсинга, и по-

следнего предсказанного слова *wk* 1 , затем передает управление модулю "Конструктор";

1. "Конструктор" наращивает существующую двоичную расширенную структуру, повторно генерируя перехо- ды, до тех пор пока управление не перейдет к модулю "Предсказатель" по достижении пустого перехода.

Теперь рассмотрим получение оценки вероятностей обмена информацией между модулями парсинговой модели.

* 1. *Вероятностные оценки парсинговой модели.* Обозначим вероятность распознавания последовательности

слов *W* в парсинговой модели через *P*(*W* ,*T* ) , где *T* − дерево полного парсинга. Вероятностная модель должна

быть способной различить желательные и менее желательные парсинги. Для того чтобы получить правильное на- значение вероятности *P*(*W* ,*T* ) , необходимо определить надлежащие условные вероятности каждому переходу.

Вероятность *P*(*W* ,*T* ) последовательности слов *W* и полного парсинга *T* рассчитывается следующим образом:

*n*1

*P*(*W* ,*T* )  [*P*(*w* | *W T* )*P*(*t* | *W T* , *w* )*P*(*T k* | *W T* , *w* ,*t* )] .

*k* 1

*Nk*

*k k* 1 *k* 1

*k k* 1 *k* 1

*k k* 1

*k* 1 *k* 1 *k k*

Здесь

*P*(*T k*

| *W T*

, *w* ,*t* )   *P*( *pk* | *W T* , *w* ,*t* , *pk* … *pk* ) ; *W T*

− (*k* 1) -й префикс слова-парсинга;

*k* 1

*k* 1 *k* 1

*k k i k* 1 *k* 1 *k k* 1

*i*1

*i*1

*k* 1 *k* 1

*w* – слово, предсказанное "Словопредсказателем"; *t* – признак, назначенный для *w* "Таггером"; *T k* − пошаговая

*k k k*

*k* 1

парсинговая структура, которая генерирует *T*  *T*

|| *T k*

, когда парсинговая структура построена на вершине *T*

*k k* 1

*k* 1

*k* 1

и вновь предсказанного слова

*wk* ; запись  обозначает конкатенацию;

*Nk* 1 − число операций, выполняемых

"Конструктором" на позиции *k* входной строки перед передачей управления "Словопредсказателю" ( *Nk* -я опера-

*p*

ция на позиции *k* – нулевой переход, причем

*Nk* представляет собой функцию от *T* );

*k* обозначает *i*-е действие

"Конструктора", выполненное в позиции *k* строки слова, и представляется следующим образом:

*i*

( /*s* ,*TOP* ')

( /*s* ,*TOP*)

Предсказывать слова

( *s* , *SB*)

(*w* \_1,*t* \_1) (*w* \_ *n*,*t* \_ *n*)( /*s* , *SE*)

Примыкать\_(налево, направо)



ПРЕДИКТОР

ТАГГЕР

Нуль

Тег слова

КОНСТРУКТОР

Рис. 3. Полный парсинг Рис. 4. Взаимодействия модулей системы парсингового кодирования

*pk* {(*adjoin*  *left*, *NTag*), (*adjoin*  *right*, *NTag*),(*uniray*, *NTag*)} ,1  *i*  *N* , *pk*  *null*, *i*  *N* .

*i k i k*

Заметим, что каждое (*W T* , *w* ,*t* , *pk* … *pk* ) , *i*  1,...*N* , определяет значащий *k*-префикс слова-парсинга

*WkTk*

*k* 1 *k* 1 *k k* 1

в позиции *k* в предложении.

*i*1 *k*

1. **Алгоритм оптимизации компонентов модели распознавания, контроля достоверности и поиска сло- воформ.** Для гарантирования надлежащей вероятностной модели по набору полных парсингов для любого пред- ложения *W*, вероятностям "Конструктора" и "Предсказателя слова" необходимо задать определенные значения. Набор ограничений на значения вероятностей компонентов различных моделей совместим со следующим алго- ритмом:
2. *P*(*null* | *WkTk* )  1, *ifh* \_{1}.*word*  *s*  и *h* \_{0}  ( /*s* ,*TOP*) , т. е. перед предсказанием  /*s*  гарантиру-

ется, что (  *s*  , *SB*) примыкает к последнему (прошлому) шагу процесса парсинга;

1. *P*((*adjoin*  *right*,*TOP*) | *WkTk* )  1, если *h* \_ 0  ( /*s* ,*TOP*) и *h* \_{1}.*word*  *s*  ;
2. *P*((*adjoin*  *right*,*TOP*) | *WkTk* )  1, если *h* \_ 0  ( /*s* ,*TOP*) и *h* \_{1}.*word*  *s*  .

Шаги 2, 3 гарантируют, что парсинг, произведенный моделью, совместим с определением полного парсинга;

1.  0*s*.*t*.*Wk* 1*Tk* 1 , *P*(*wk*  /*s* | *Wk* 1*Tk* 1 )  . На этом шаге обеспечивается остановка модели. Как только ко- нец символа предложения  /*s*  сгенерирован, модель заканчивает парсинг с вероятностью, равной единице.
   1. *Оптимизация работы "предсказателя"*. Рассмотрим иерархическую схему и алгоритм построения стеков для нахождения нового слова – объекта контроля. Предположим, что каждый стек содержит частичные парсинги – гипотезы, которые были построены одним и тем же числом операций "Предсказателя" и "Конструктора". Частич-

ный парсинг в каждом стеке оценивается согласно принятому критерию ln(*P*(*W* ,*T* ))

вершины.

начиная с самой высокой

На рис. 5 показана схема действий алгоритма, связанных с просмотром нового слова *Wk* 1 . (Здесь *Pk* − макси-

мальное число операций примыкания для *k*-кратного префикса слова; так как дерево двоично, *Pk*  *k* 1 .)

Процедура поиска строится на основе двух параметров:

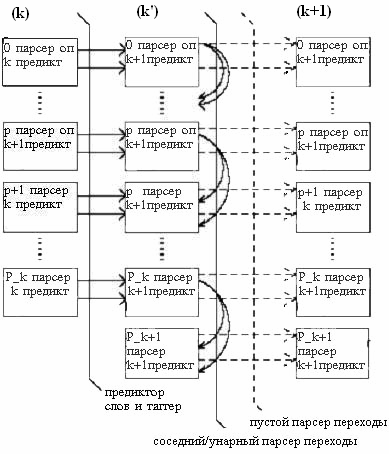


Рис. 5. Цикл расширения поиска

− максимальная глубина стека − максимальное число гипотез, кото- рые стек может содержать в любое данное время;

− порог лог-вероятности − разли- чие между оценками лог-вероятнос- ти наиболее вероятной и наименее вероятной гипотез в любом данном состоянии стека, причем порог лог- вероятности не может быть больше заданного значения.

**Заключение.** Таким образом, теоретические и практические ис- следования проблемы построения компьютерной системы текстовой информации, проведенные с целью разработки методов и алгоритмов контроля и коррекции орфографии на основе *n-*граммной модели есте- ственного языка позволили опреде- лить закономерности распределения *n-*граммных ошибок; оценить досто-

верность информации при равномерных и неравномерных гипотезах *n-*кратных искажений; провести парсинговое кодирование и моделирование структуры слова на основе словоформ; оценить качество распознавания, кластери- зации, поиска элемента текста; моделировать процессы реализации алгоритмов эквивалентной классификации. Полученные вероятностные модели парсингового представления слов, кодирования и поиска позволяют оценить качество распознавания, эффективно моделировать процессы реализации алгоритмов эквивалентной классифика- ции в системах контроля и коррекции орфографических ошибок.

Предложены методы и алгоритмы оптимизации параметров функционирования компонентов системы контро- ля орфографии, которые реализованы в виде самостоятельных программных модулей, соответствующих требова- ниям разработки пакетов прикладных программ. Полученные теоретические положения исследований позволили построить программную систему контроля и коррекции орфографии узбекского языка на основе *n-*граммной мо- дели, которая показала высокое качество функционирования в системах электронного документооборота пред- приятий различных форм собственности.

**Роль BIM-технологий**

**в организации и технологии строительства**

**Аннотация.** Строительная отрасль не стоит на месте, она постоянно развивается, появляются новые технологии, инновационные материалы, совершенствуются машины и механизмы. Важным инструментом строительства и проектирования являются BIM-технологии (Building Information Modeling). BIM — это процесс, в результате которого формируется информационная модель здания или сооружения. На каждой стадии есть модель, которая даёт информацию об объеме обработанной на данный момент информации (архитектурной, конструкторской, технологической, экономический). Развитие BIM так же не останавливается, а наоборот, возрастает довольно быстрыми темпами. Уже имеющиеся программные комплексы постоянно совершенствуются, разработчиками выпускаются частые обновления. В настоящее время BIM применяется не только для сложных уникальных сооружений (заводы, уникальные здания), его так же активно внедряют и использую для проектирования многоэтажной застройки, что в свою очередь имеет массу преимуществ.

С точки зрения экономики — это возможность создания цифровых моделей реального мира экономики. Цифровые модели на основе современных технологий измерений позволят

обеспечить учёт различных ресурсов в экономике и процессов, которые происходят с этими ресурсами. Информационные технологии помогают экономике преодолеть извечный недостаток — это высокий уровень издержек. Для некоторых отраслей на рынке этот фактор является критичным. На сегодняшний момент для некоторых компаний нет другого способа снижения издержек, нежели внедрение IT-решений. Преимущества цифровой экономики это

— увеличение производительности труда; простота централизованного управления, налогообложения и контроля; глобальная автоматизация и стандартизация всех хозяйственных процессов: производственных, образовательных, медицинских, социальных и т. д.; снижение бюрократии и коррупции; развитие цифровых денег и пр.

**Ключевые слова:** технологии; сооружения; BIM; среда; модель; моделирование

### Введение

BIM-технологии (Building Information Model) и BIM-проектирование призваны помочь проведению масштабных и объемных работ по проектированию зданий и сооружений.

Внедрение информационного моделирования в России происходит постепенно, так 5 марта 2021 года было принято постановление № 331, обязывающее с первого января 2022 года применять BIM на объектах госзаказов.

С точки зрения организации, технологии и экономики строительства информационное моделирование позволяет упорядочивать процессы строительства, уменьшать производственные расходы и увеличивать прибыль. Кроме того, использование BIM позволяет сокращать сроки выполнения работ и повышать их качество.

Цель исследования — это определение значимости BIM-технологий, их влияния на организацию, технологию и экономику строительства, а также изучение основных программных обеспечений, повышающих работоспособность и продуктивность организаций.

### Основные направления информационного моделирования

В настоящее время всё больше Российских крупных компаний внедряют BIM технологии в проектирование. Современный этап развития информационных технологий в отечественном строительном комплексе характеризуется переходом от разработки чертежей в двухмерном пространстве к информационному моделированию всего процесса строительства, что актуализирует использование BIM-технологий [1].

Переход на BIM компаний средних и малых может оказаться болезненным ввиду необходимости закупки необходимого ПО (программного обеспечения), обучения персонала, разработки необходимых шаблонов, перевода проектов в новый формат. Производительность труда сотрудников при таком переходе может снизиться, но постепенно возрастет, достигая более высокого уровня [2].

При внедрении BIM, компании могут получить следующие улучшения в организационном и экономическом плане:

* создание единой платформы для взаимодействия вне зависимости от месторасположения клиентов, подрядчиков, поставщиков на базе информационных моделей;
* ориентация на заданные KPI, введение расчётов CapEX и OpEX;
* адаптивный мониторинг;
* повышение эффективности на всех стадиях жизненного цикла;
* внедрение информационных систем на базе моделей для автоматизации основных бизнес-процессов1.

Главное достоинство BIM, которое неоднократно подчеркивалось теми, кто с ним работал — умение данного программного комплекса устранять строительные нестыковки и ошибки еще на этапе проектирования. Это, безусловно, экономит время и силы проектировщиков 2 . Удобство трехмерной BIM-модели еще и в том, что из нее можно автоматически получать 2D-чертежи, ведомости, спецификации, графики проектных работ. BIM-модель описывает весь жизненный цикл сооружения, а значит, может применяться и на этапе его эксплуатации [3].

В настоящее время, в условиях отмены ДДУ (договоров долевого участия) применение BIM может стать еще и конкурентным преимуществом. Сметная прозрачность крайне важна для банков, которые теперь будут контролировать и финансировать строительство [4]. А предельная честность застройщика перед банком может повлиять на кредитную ставку — естественно, в меньшую сторону3.

Далее авторы рассмотрели программы наиболее широкого функционала, которые активно применяются при проектировании и строительстве.

Программный пакет ARCHICAD позволяет наполнять модель информацией и затем использовать эту информацию при помощи функции автоматического построения фасадов, разрезов, деталей и фрагментов проекта, формировать спецификации и экспликации, оформлять документацию и прочее. Одновременно работать над проектом, используя общий файл, может вся команда специалистов, что существенно уменьшает время проектирования.

При передаче информации используется облачное хранение на серверах компаний с возможностью доступа внешних проектировщиков. Благодаря универсальному IFC — формату архитекторы и проектировщики, работавшие над созданием 3D — модели в ARCHICAD, могли без искажений передавать информацию и ставить задачи смежникам, выполнявшим свою часть работы в Revit и Civil 3D [5]. Таким же образом, только в обратном порядке, выполненные задачи передается архитекторам для добавления в BIM-модель сооружения — при помощи связанных модулей IFC (Industry Foundation Classes). IFC универсальный формат данных, позволяющий вести обмен информацией между программами, поддерживающими BIM-процесс. Так же широко используются в проектировании продукты AutoDesk.

BIM помогает минимизировать количество ошибок уже на ранних этапах реализации проекта. Убрать все коллизии, «неудобные» пересечения инженерных коммуникаций, которые при обычном проектировании зачастую не видны. Из виртуального проекта можно получить

1 [Электронный ресурс] — Электронное сетевое издание всероссийский отраслевой интернет-журнал

«Строительство.RU». Российский BIM: свой среди чужих, чужой среди своих.

2 [Электронный ресурс] — Зиганшин А.М., Зиганшин М.Г. Учебно-методическое пособие для учебной и научной работы студентов направления «Строительство» (квалификация «магистр»). Изд. 2-е, перераб. и дополн.

/ А.М. Зиганшин, М.Г. Зиганшин. — Казань: Изд-во Казанск. гос. архитект.-строит. ун-та, 2019. — 349 с.

3 [Электронный ресурс] — [https://stroi.mos.ru/builder\_science/tiekhnologhiia-bim-iedinaia-modiel-i-](https://stroi.mos.ru/builder_science/tiekhnologhiia-bim-iedinaia-modiel-i-sviazannyie-s-etim-zabluzhdieniia?from=cl) [sviazannyie-s-etim-zabluzhdieniia?from=cl.](https://stroi.mos.ru/builder_science/tiekhnologhiia-bim-iedinaia-modiel-i-sviazannyie-s-etim-zabluzhdieniia?from=cl) Талапов В.В. «Технология BIM: суть и особенности внедрения информационного моделирования зданий». М., 2015. (дата обращения 09.03.2016).

детальные данные об объемах строительства. В итоге, BIM сэкономит застройщику не только время, но и деньги, оптимизировав себестоимость стройки4.

Для достижения максимального эффекта от использования BIM-технологий необходимо внедрять их на всех этапах реализации проекта, быть компанией полного цикла. В этом случае модель живет от концепции до ввода объекта в эксплуатацию и в дальнейшем как основа для создания эксплуатационной системы [6].

Модель должна быть связана с различными справочными и информационными системами. Это даст возможность увязывать информационную модель с планами строительства, финансирования, поставками материалов.

Данную связь можно обеспечить за счет правильной классификации элементов модели и создания баз данных материалов, строительных процессов и трудовых ресурсов [7]. Такая система позволит быстро анализировать данные, оперативно вносить изменения при необходимости, просчитывать сроки возведения как всего объекта, так и отдельных конструктивных элементов5.

Авторы считают, что без активного внедрения технологий BIM Россия может отстать от мировой строительной индустрии. Отсутствие объёмных моделей и работа с двухмерными чертежами серьёзно затрудняет приход зарубежных инвесторов на российский рынок недвижимости. И наоборот, российским компаниям необходимо осваивать новые технологии, для того чтобы строить за рубежом. Сейчас экономическая ситуация такова, что девелоперы должны полностью пересмотреть свой подход к работе [8]. Высокая конкуренция диктует свои правила. Снизить себестоимость строительства, но не за счет качества, как это обычно практикуется, а за счет высокой технологичности одна из важнейших задач [9].

Примером может служить зарубежный опыт: Сингапур стал одной из первых стран, где стали использовать преимущества BIM. Дорожная карта внедрения информационного моделирования в строительстве, разработанная правительством Сингапура с привлечением ведущих мировых экспертов, предполагает создание и поддержку BIM-модели всего островного государства. Уже создано единое государственное облачное информационное пространство для хранения, анализа и применения информационных моделей. Теперь экспертиза Сингапура принимает проекты площадью более 5 тыс. кв. м исключительно в виде BIM-модели. В Сингапуре с применением BIM построены целые города, модель используется на всех стадиях жизненного цикла объекта, от проектирования до эксплуатации. В России же применение BIM модели на стадии эксплуатации можно встретить очень редко. Кроме того, благодаря модели все заинтересованные лица могут получить доступ к информации об инфраструктуре и инженерных сетях окружающих участков и в соответствии с этим принимать какие-либо проектные и управленческие решения [10].

Зарубежный опыт в проектировании таких сложных проектов может играть большую роль. В каждой стране существуют свои технологические требования к объекту и человеческие потребности, которые объект должен будет удовлетворять, после воплощения в жизнь. Из этого следует, что на этапе проектирования генерируются идеи, решения для успешной реализации проекта, поэтому необходим обмен знаниями. К примеру, это может быть информация о применяемой технике при производстве строительных работ, технологических решениях, способах проектирования и используемых программных обеспечениях.

4 [Электронный ресурс] — [https://rcmm.ru/tehnika-i-tehnologii/47484-bim-tehnologii-dlja-aura-apart-teper-](https://rcmm.ru/tehnika-i-tehnologii/47484-bim-tehnologii-dlja-aura-apart-teper-my-prosto-ne-mozhem-a-tochnee-ne-hotim-rabotat-inache.html) [my-prosto-ne-mozhem-a-tochnee-ne-hotim-rabotat-inache.html.](https://rcmm.ru/tehnika-i-tehnologii/47484-bim-tehnologii-dlja-aura-apart-teper-my-prosto-ne-mozhem-a-tochnee-ne-hotim-rabotat-inache.html)

5 [Электронный ресурс] — [https://stroi.mos.ru/builder\_science/tiekhnolo-ghiia-bim-iedinaia-modiel-i-](https://stroi.mos.ru/builder_science/tiekhnolo-ghiia-bim-iedinaia-modiel-i-sviazannyie-s-etim-zabluzhdieniia?from=cl) [sviazannyie-s-etim-zabluzhdieniia?from=cl.](https://stroi.mos.ru/builder_science/tiekhnolo-ghiia-bim-iedinaia-modiel-i-sviazannyie-s-etim-zabluzhdieniia?from=cl)

Для проектирования сложных проектов активно используются такие ПО (программное обеспечение) как Tekla, Aveva.

Tekla Structures — это мощное программное обеспечение для информационного моделирования и проектирования строительных конструкций. Tekla Structures применяется на протяжении всего проекта от концепта до производства. Данное ПО локализовано в соответствии со строительными нормами и ГОСТами.

Модели Tekla способны обрабатывать самый высокий уровень проработки LOD 500. Уровень проработки модели (LOD) это справочная информация из архитектуры, инжиниринга, строительства (AEC) для спецификации и присвоения артикулов с уровнем ясности данных, надежности информационной модели зданий (BIM), что делает их действительно технологичными.

Обновленное ПО 2021 года доступно в трех независимых от материала конфигурациях, что позволяет подобрать идеальный вариант под самые разные задачи:

* Tekla Structures Carbon — позволяет планировать и отслеживать всю деятельность по проектированию, деталировке, производству и монтажу строительных конструкций. Можно открывать все типы моделей и чертежей Tekla Structures, а также генерировать отчеты. Возможно отслеживать и контролировать ход выполнения проекта, добавлять информацию в модель, определять последовательности монтажа, управлять согласованиями, просматривать статус производства и календарное планирование и т. д.
* Tekla Structures Graphite позволяет создавать многоматериальные 3D-модели для проектирования, управления ими и использования их совместно с другими пользователями. Можно создавать общую проектную информацию, такую как чертежи общего вида, спецификации, отчеты и 3D-модели. Кроме того, Graphite можно использовать для создания схем расположения арматуры на стройплощадке и передавать информацию на станки для гибки и обработки арматуры. Здесь интересным и перспективным направлением можно считать 3D печать. Имея модель появляется возможность осуществлять заводское изготовление не несущих конструкций, не требующих армирования. Это, в свою очередь, дает полную свободу действий дизайнерам и архитекторам, проектирующим здания, а также позволяет осуществить экономию средств финансовых и материальных за счёт снижения затрат на оплату труда и энергоресурсы [11].
* Tekla Structures Diamond включает в себя все возможности Tekla Structures. Можно создавать полностью деталированные 3D-модели вне зависимости от типа строительных материалов, управлять ими и использовать их совместно с другими пользователями.

Tekla предоставила программное обеспечение BIM для помощи в планировании и строительстве архитектурно разнообразного и структурно сложного проекта мусоросжигательно завода в Копенгагене Амагер Бакке (рис. 1).6

Ключевое понятие системы программных продуктов AVEVA — цифровая 3D-модель объекта. В основу разработки всех информационных систем компании AVEVA положена концепция создания и управления всей технической информацией об объекте на протяжении всего его жизненного цикла. Данное ПО — это полностью интегрированная технология ведения объекта — от разработки обоснования инвестиций, технологической части проекта, детального проекта, выпуска проектной документации, управления логистикой, контроля

6 [Электронный ресурс] — Электронное сетевое издание всероссийский отраслевой интернет-журнал

«Строительство.RU». BIM-технологии для AURA Apart: «Теперь мы просто не можем, а точнее не хотим, работать иначе».

закупок, поставок и складирования, контроля за монтажными и пусконаладочными работами до обслуживания при эксплуатации7.

***Рисунок 1.*** *Мусоросжигательный завод, спроектированный с помощью Tekla (фото заимствовано: электронный ресурс* [*https://novate.ru/blogs/091120/56667/*](https://novate.ru/blogs/091120/56667/)*)*

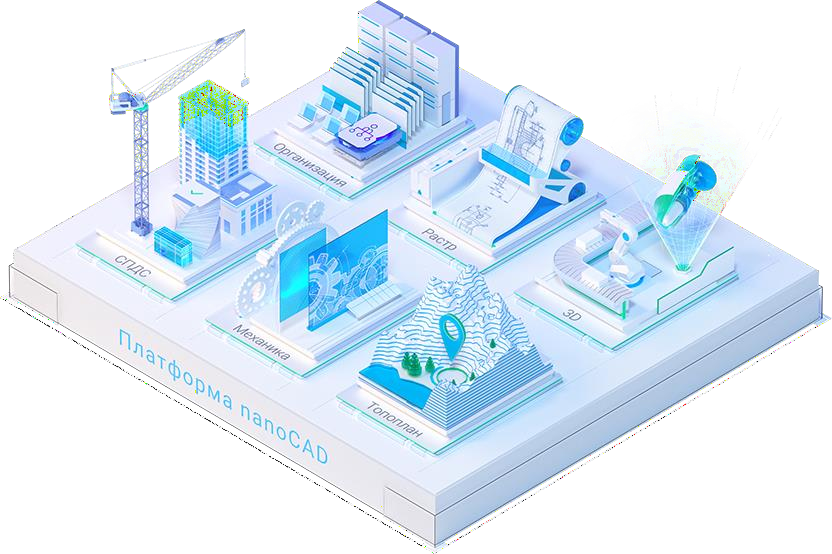
Сегодня всё больше представителей бизнес-сферы рассматривает применение информационных технологий как возможность повышения эффективности производства и оказания услуг. Это является отражением определённого этапа развития экономики в целом: растёт значимость конкуренции, компании ищут дополнительные средства повышения рентабельности бизнеса. А информационные технологии — это своего рода «тюнинг» для бизнеса предприятий, точная настройка ряда параметров для достижения максимальной эффективности работы [12].

В противовес зарубежным программам российскими специалистами разработана

«NanoCAD Plus — отечественная классическая универсальная САПР-платформа». Это единая платформа, которая даёт возможность оперативно обмениваться информацией между участниками проекта. Платформа полностью адаптирована к Российским стандарт. NanoCAD имеет большое количество приложений к платформе, необходимых для процессов проектирования и строительства, основные модули отражены на рисунке 2.

С точки зрения организации и технологии строительства большой интерес представляет приложение «Стройплощадка». В его функции входит менеджер проектов, с помощью него задаются необходимые здания, объемы работ, техника, материалы, указывается количество рабочих и число смен. Состав работ возможно формировать на основе ГЭСН и ЕНиР. Приложение производит расчёт потребности в материалах, кадрах, воде, электроэнергии и прочего. Так же рассчитываются и отображаются опасные зоны действия грузоподъёмных механизмов, расчет формируется на основе характеристик механизмов, и изменяется автоматически с изменением их параметров. Программа также отрисовывает временные здания, производит расчёт: календарных планов, графиков, ведомостей, графиков движения рабочей силы. Инструменты позволяют отрисовывать: временные дороги с учётом уширений, мест разворота и пешеходных переходов, сложные откосы, кроме того позволят рассчитывать

7 [Электронный ресурс] — [https://www.aveva.com/ru-ru/solutions/engineering/.](https://www.aveva.com/ru-ru/solutions/engineering/)

количество дорожных плит. Большим преимуществом является база строительной техники, при выборе объекта, к примеру, крана, его сразу можно сразу разместить на плане8.

***Рисунок 2.*** *Модули платформы nanoCAD (фото заимствовано: электронный ресурс* [*https://www.nanocad.ru/products/platform/*](https://www.nanocad.ru/products/platform/)*)*

Кроме представленных программ существуют вспомогательные ПО способные упростить и упорядочить процесс строительной деятельности. Далее авторы приводят описание данных ПО.

ProjectWise — это ПО компании Bentley Systems, Inc. Предназначено для комплексного обмена рабочей информацией и сотрудничества. Это комплексная система управления проектными данными. ProjectWise позволяет реализовать полноценный документооборот, управление данными и электронный архив масштаба предприятия, при этом обеспечивая надежное хранение данных, контроль доступа к документам и аудит работы с документами (учет всех действий сотрудников при работе с документами), быстрый атрибутивный поиск документов, подготовку и реализацию выпуска проектной документации на бумажных и электронных носителях, формирование наименований и кодов документов в автоматическом режиме, а также создание отчетов в соответствии со стандартами предприятия и многое другое.9

SAP (Systeme, Anwendungen und Produkte in der Datenverarbeitung, что в переводе с немецкого: системы, приложения и продукты в обработке данных) — это ПО, которое всё чаще внедряют в работу крупных предприятий. Основанная задача данного обеспечения — это предоставление широты возможностей для ведения гибкого бухгалтерского учета. Это комплекс решений для выстраивания общего информационного пространства на базе предприятия и эффективного планирования ресурсов и рабочих процессов. Данное ПО базируется на трёх звеньях: клиент, сервер, СУБД (система управления базами данных). Такая система позволяет объединять решения в двух основных сферах: бухучет и составление отчетности (фиксация производственных затрат, управление средствами и заказами), а также логистика (планирование, сбыт, оформление счетов, осуществление прямых продаж и регулярные отгрузки товаров, материально-техническое снабжение, с проведением закупок и

8 [Электронный ресурс] — <https://www.nanocad.ru/products/ppr/>.

9 [Электронный ресурс] — [https://www.bentley.com/ru/products/brands-/projectwise.](https://www.bentley.com/ru/products/brands-/projectwise)

контролем запасов). Таким образом, SAP — это целый комплекс решений, обладающий следующими функциями:

* автоматизация труда бухгалтера;
* упрощение торговых и складских операций;
* облегчение учета кадров, финансов, акций и других активов;
* модернизация логистики;
* составление максимально наглядных зарплатных графиков.

Primavera ещё одно программное обеспечение компании Oracle для управления проектами. Портфель Primavera включает в себя несколько продуктов. Данное ПО позволяет вести учёт и осуществлять управление массивами данных по проекту и финансовым возможностям организации.10 Пользование данным обеспечением способствует повышению показателей эффективности управления проектами за счёт автоматизации задач по планированию и контролю выполнения. Primavera позволяет осуществлять ряд следующих функций:

* оценка рисков, прогнозирование и анализ;
* расчёт влияния текущего процесса на степень загрузки ресурсов компании;
* создание среды для возможности взаимодействия всех лиц, которые участвуют в процессе.

К наиболее распространённым продуктам системы Primavera относятся:

* Primavera Risk Analysis — позволяет анализировать риски, которые связаны с издержками и сроками реализации проекта на протяжении всего жизненного цикла. Позволяет определить будет ли успешен проект;
* Primavera Portfolio Management — инструменты для своевременной реализации проектов и программ, повышения дохода и поддержания конкурентоспособности;
* Oracle Primavera Prime — данный инструмент предназначен для долгосрочного планирования инвестиций и управления портфелем капитальных вложений. С его помощью возможно составлять бизнес планы долгосрочных капитальных проектов и управлять денежными потоками, добиваясь максимальной доходности;
* Primavera Unifier — лучшее в своем классе решение, оно позволяет управлять капитальными проектами и объектами любого размера на любом вертикальном рынке. С его помощью можно эффективно управлять затратами, документооборотом, графиками, ресурсами, денежными средствами.

### Вывод

Внедрение BIM, с точки зрения организации и технологии строительства, позволяет упорядочить все производственные процессы, объединить информацию по проекту в единой системе с возможностью доступа к информации вне зависимости от месторасположения специалистов, упростить и автоматизировать процесс создания сопутствующей документации

10 [Электронный ресурс] — [https://gantbpm.ru/oracle-primavera-programma/.](https://gantbpm.ru/oracle-primavera-programma/)

(ведомости, спецификации, табличные расчёты, графики, производственные планы), сократить сроки на поиск технических характеристик, благодаря подгруженным в ПО базам данных. Использование BIM — это возможность избежать ошибок на этапе проектирования, быстро реагировать на изменяющуюся ситуацию, оперативно вносить изменения, и как следствие исключить непредвиденные затраты.

По результатам исследования, авторами выведены следующие преимущества от применения программ BIM-технологий:

* оптимизация проекта;
* снижение затрат;
* увеличение скорости проектирования;
* повышение качества принятых решений;
* взаимодействие в режиме реального времени;
* эффективный процесс строительства;
* лучшая организация строительного процесса;
* увеличение скорости строительств;
* исключение, уменьшение возможных ошибок;
* сокращение издержек.

**РАЗРАБОТКА СИСТЕМЫ РАСПОЗНАВАНИЯ РЕЧИ НА ОСНОВЕ СКРЫТЫХ МАРКОВСКИХ МОДЕЛЕЙ ОТДЕЛЬНЫХ СЛОВ**

# ВВЕДЕНИЕ

Распознавание голоса в компьютерных системах весьма распространено. Распо- знавание речи и, как следствие, голосовая идентификация нашли свое применение

◯*c Савин А. Н., Тимофеева Н. Е., Гераськин А. С., Мавлютова Ю. А., 2017*

во всех сферах человеческой деятельности. Благодаря системам распознавания ре- чи обеспечивается безопасность от несанкционированного проникновения в защи- щенную зону. Такие системы содержат базу данных голосов сотрудников, имеющих доступ к защищаемой зоне, и предотвращают допуск людей, чьих голосов в ней нет [1, 2].

В настоящее время широко разрабатываются и внедряются интеллектуальные системы управления различными объектами, которые позволяют осуществлять кон- троль за объектами в реальном времени. Управление такими системами можно осу- ществлять различными способами, одним из них является метод голосовых команд. При этом защиту объекта от несанкционированного доступа можно решить, исполь- зуя индивидуальные особенности голоса каждого человека.

Уровень развития современной микропроцессорной техники (например, мобиль- ные устройства связи) позволяет использовать сложные вычислительные алгоритмы, основанные на цифровой потоковой обработке статистических данных в реальном времени. Поэтому разработка таких алгоритмов является весьма актуальной.

Одним из путей решения вышеуказанных задач является использование для рас- познавания фрагментов речи математического аппарата скрытых марковских моде- лей (СММ) [3]. Данная работа посвящена разработке алгоритма и соответствующего программного модуля, осуществляющего формирование СММ для отдельных слов требуемого словаря команд системы управления объекта, на основе кодирования признаков звукового сигнала, использующего линейные предсказания.

# СТРУКТУРА СИСТЕМЫ РАСПОЗНАВАНИЯ РЕЧИ НА ОСНОВЕ СММ

Рассмотрим дискретную систему, имеющую конечное множество из *N* состоя- ний — *S* = {*s*1*, . . . , sN* }, в каждом из которых она может принимать одно из *M* значений из набора наблюдаемых параметров *V* = {*v*1*, . . . , vM* } — алфавита. Состоя- ние системы *qt* в момент времени *t*, принимающее одно из *N* значений множества *S*, зависит только от её состояния *qt*−1 в момент времени *t* −1, а значение наблюдаемого параметра *ot* в момент времени *t* зависит только от состояния *qt*, т. е. не зависит от времени.

Вероятности переходов между состояниями системы задаются матрицей *A*. Ве- роятности выпадения каждого из *M* значений наблюдаемого параметра системы в каждом из *N* состояний системы задаются набором векторов *B*. Вероятность по- явления некоторого начального состояния системы задаётся вектором *π*. При этом последовательность состояний, в которых пребывает система *Q* = *q*1*, . . . , qT* , внеш- нему наблюдателю не видна, а видит он только последовательность наблюдений *O* = *o*1*, . . . , oT* (здесь *T* — длина последовательности), т. е. система ведёт себя как

«чёрный ящик». Модель такой системы получила название СММ и в компактной записи обозначается *λ* = (*A, B, π*) [2].

Для моделирования отдельного слова может быть выбрана лево- правая СММ (рис. 1) на основе предположения о том, что в каждый момент времени система переходит в

новое состояние [4]. Соответственно

неизвестное число скрытых состоя- ний *N* в этом случае определяется

Рис. 1. Структура лево-правой СММ

Fig. 1. The structure of the left-right hidden Markov models (HMM)

длиной и количеством сегментов, на которые слово разбивается при анализе его признаков. Процесс распознавания с использованием СММ предполагает два этапа (рис. 2).



Построение и обучение СММ по речевой базе, содержащей записи слов диктора

Распознавание команд в звуковом потоке

Распознавание слова

Анализ признаков

Выделение слова

Звуковой поток/файл

База СММ слов

Обучение СММ

Анализ признаков

Выделение последовательности одинаковых слов

Речевая база

Код слова

Рис. 2. Структура системы распознавания речи на основе использования СММ Fig. 2. Structure of the speech recognition system based on the use of HMM

В режиме обучения элементы системы имеют следующее функциональное назна- чение:

•

речевая база содержит записи слов, повторяющихся несколько раз для обеспе-

чения адекватности получаемых СММ, которые будут доступны для распозна- вания;

выделение последовательности одинаковых слов из файла речевой базы с по- мощью предварительной обработки (подавление шума, фильтрация и т. д.); анализ признаков и определение алфавита слова *V* , по которому формируется последовательность наблюдений *O*;

•

•

обучение СММ — подбор параметров СММ, чтобы она как можно лучше опи- сывала реальную наблюдаемую последовательность *O* символов алфавита *V* анализируемого слова;

•

сохранение СММ в базе — словаре.

•

В режиме распознавания:

выделение слова из входного звукового потока с помощью предварительной

•

обработки;

анализ признаков распознаваемого слова и формирование соответствующей по- следовательности наблюдений *O*;

•

распознавание слова с использованием базы СММ и генерация кода распозна- ваемого слова.

•

Таким образом, для реализации данной структуры необходимо всего 4 модуля:

модуль выделения слов из звукового потока, модуль анализа признаков слова, модуль обучения СММ с базой моделей, модуль распознавания слов.

# АЛГОРИТМ ВЫДЕЛЕНИЯ ОТДЕЛЬНЫХ СЛОВ ИЗ ЕДИНОГО ЗВУКОВОГО ФАЙЛА

В режиме обучения файл должен содержать несколько раз произнесенное одним диктором требуемое слово. Это необходимо для получения достоверной последо- вательности наблюдений *O*, соответствующей данному слову. На рис. 3 приведена блок-схема алгоритма предварительной обработки звукового файла, основанного на

Начало процедуры выделения команд из звукового файла

Вычисление среднего значения длительности команд и округление его до ближайшего числа, кратного «Шагу сегментов (*MА* )»(см. далее «Анализ признаков»)

Завершение процедуры выделения команд из звукового файла

Вырезка из входного файла отрезков одинаковой длины,

соответствующих командам, и сборка их в массив для дальнейшей обработки

Загрузка файла с повторяющимся словом

записанного ранее одним диктором

Удаление коротких «пауз» (меньше минимальной длины команды), возникающих внутри команды, например, между слогами слова

Удаление коротких «команд» (меньше минимальной длины команды), возникающих внутри паузы, например, при наличии импульсных шумов в паузах между командами

Вычисление индексов, в которых огибающая начинает превышать заданный уровень шума

(начало «команды»), а затем становится ниже уровня шума (конец «команды» - начало «паузы»)

Нормировка входного звукового файла дана таким образом, чтобы максимальная амплитуда была равна единице.

Выделение огибающей входного сигнала:

1. «выпрямление» сигнала с помощью операции «Модуль»;
2. фильтрация «выпрямленного» сигнала с помощью НЧ FIR фильтра с частотой среза примерно 10 Гц

Удаление первых 0.1 с из записи для исключения переходных процессов записи и фильтрации

Изменение частоты дискретизации звукового файла, так как для потокового распознавания необходимо, чтобы она была примерно равна 7 Ks/с, т.е. не очень большая, но достаточная, чтобы оцифровать сигналы с частотой до 3.5 КГц

Ограничение спектра сигнала звуковым диапазоном голоса с помощью полосно-

пропускающего фильтра Батерворта. При этом убирается постоянная составляющая из сигнала

Рис. 3. Блок-схема алгоритма выделения отдельных повторяющихся слов из звукового файла

Fig. 3. A flowchart of an algorithm for selecting separate repetitive words from a sound file

вычислении огибающей и выделении на ее основе участков файла соответствующих повторяющимся словам.

При этом на выходе формируется массив отрезков звукового файла одинаковой длины, соответствующих повторяемому слову, что позволяет использовать усреднен- ные входные данные при обучении СММ слова, делая её тем самым более адекват- ной.

Алгоритм выделения отдельных слов из звукового потока встроен в модуль построения СММ слов, реализованный в среде графического программирования LabVIEW компании National Instruments [5]. На рис. 4 показан процесс выделе- ния команды из звукового файла, содержащего десять раз повторяющееся слово

«Вперёд».

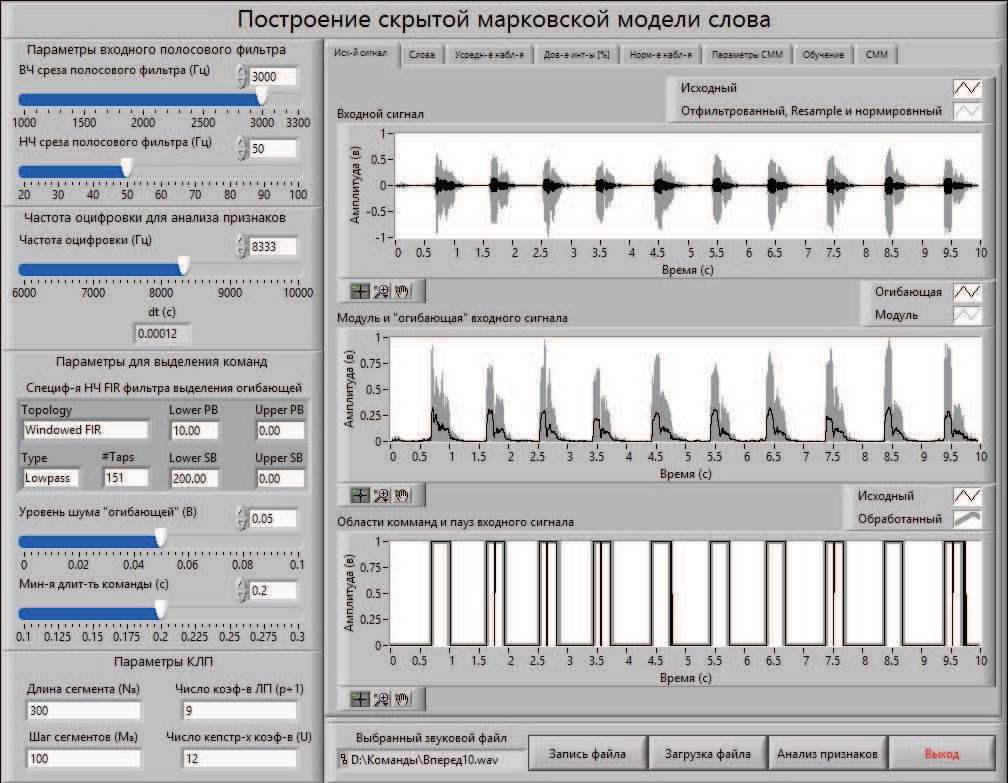


Рис. 4. Выделение повторяющихся слов из звукового файла Fig. 4. Selecting duplicate words from a sound file

На первом графике рис. 4 чёрным изображен исходный сигнал, серым — от- фильтрованный, с измененной частотой дискретизации и нормированный. Из запи- си убираются первые 0.1 с, соответствующие переходным процессам при включении микрофона и предварительной фильтрации.

Выделение слова осуществляется путем анализа огибающей сигнала. Индексы, в которых огибающая начинает превышать заданный в начале уровень шума, соот- ветствуют началу команды. Индексы, в которых огибающая становится ниже уровня шума, — концу команды. Паузы внутри команды отсеиваются с помощью задан- ной заранее минимальной длительности команд. Таким же образом отсеиваются и лишние шумы.

Элементы управления модуля (граничные частоты среза входного полосового фильтра, частота дискретизации сигнала для анализа признаков, параметры НЧ

фильтра огибающей, уровень шума огибающей, минимальная длительность команды) позволяют подбирать требуемые параметры на этапе выделения команд для обеспе- чения построения адекватных СММ слов.

# АЛГОРИТМ АНАЛИЗА ПРИЗНАКОВ СЛОВ

Для системы распознавания речи каждому слову необходимо сопоставить набор признаков. Этот процесс в [4] предложено осуществлять на основе анализа периодич- ности спектра фрагментов звукового сигнала (кепстральный анализ), предварительно обработанного с помощью алгоритмов линейного предсказания. Такой процесс назы- вается кодированием на основе линейного предсказания (КЛП). Алгоритм анализа признаков на основе КЛП, используемый при распознавании отдельных слов, приве- ден на рис. 5, 6.

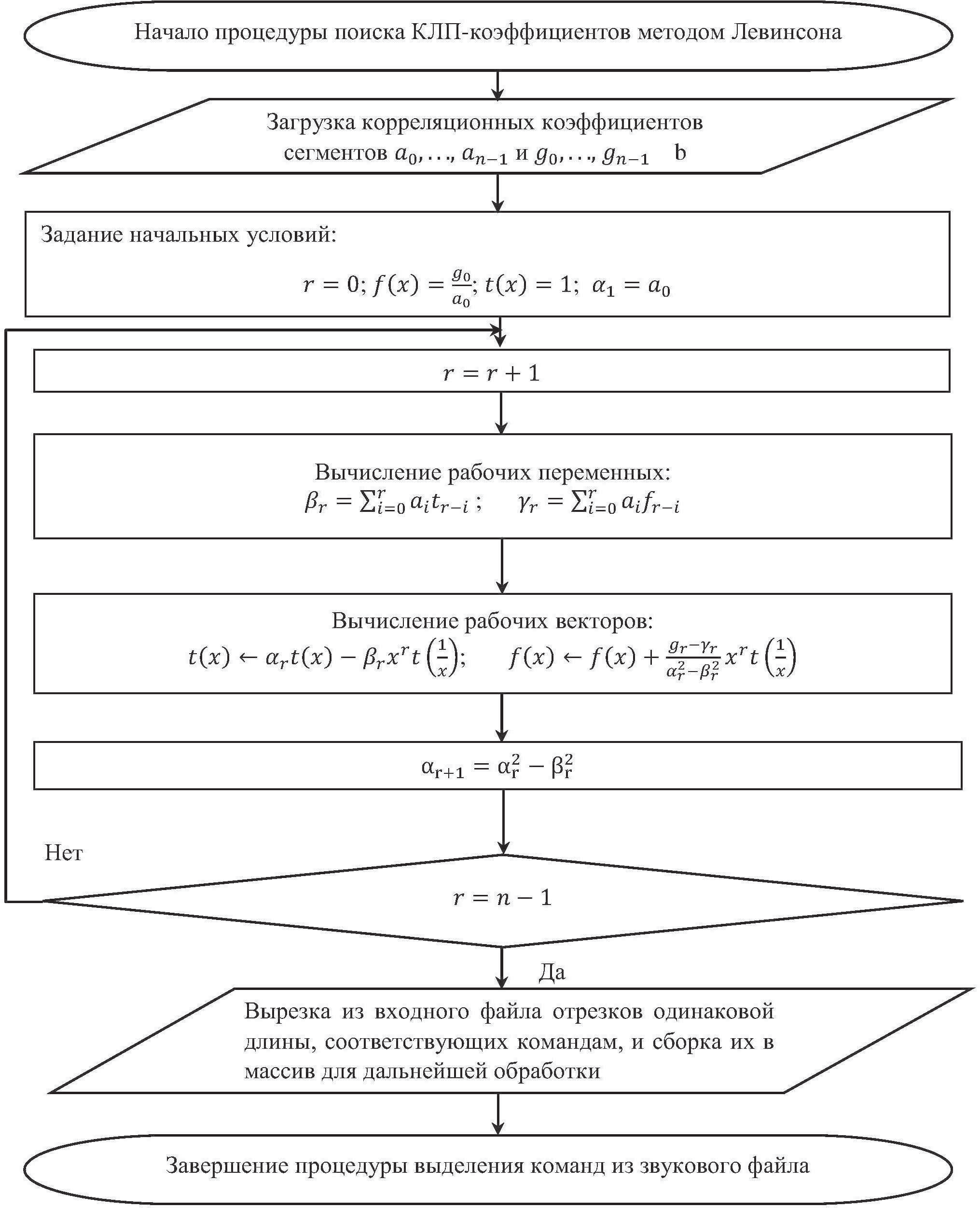


Рис. 5. Блок-схема процедуры алгоритма Левинсона вычисления КЛП-коэффициентов Fig. 5. Block diagram of the procedure of Levinson’s algorithm for calculating linear prediction

coding (LPC) coefficients

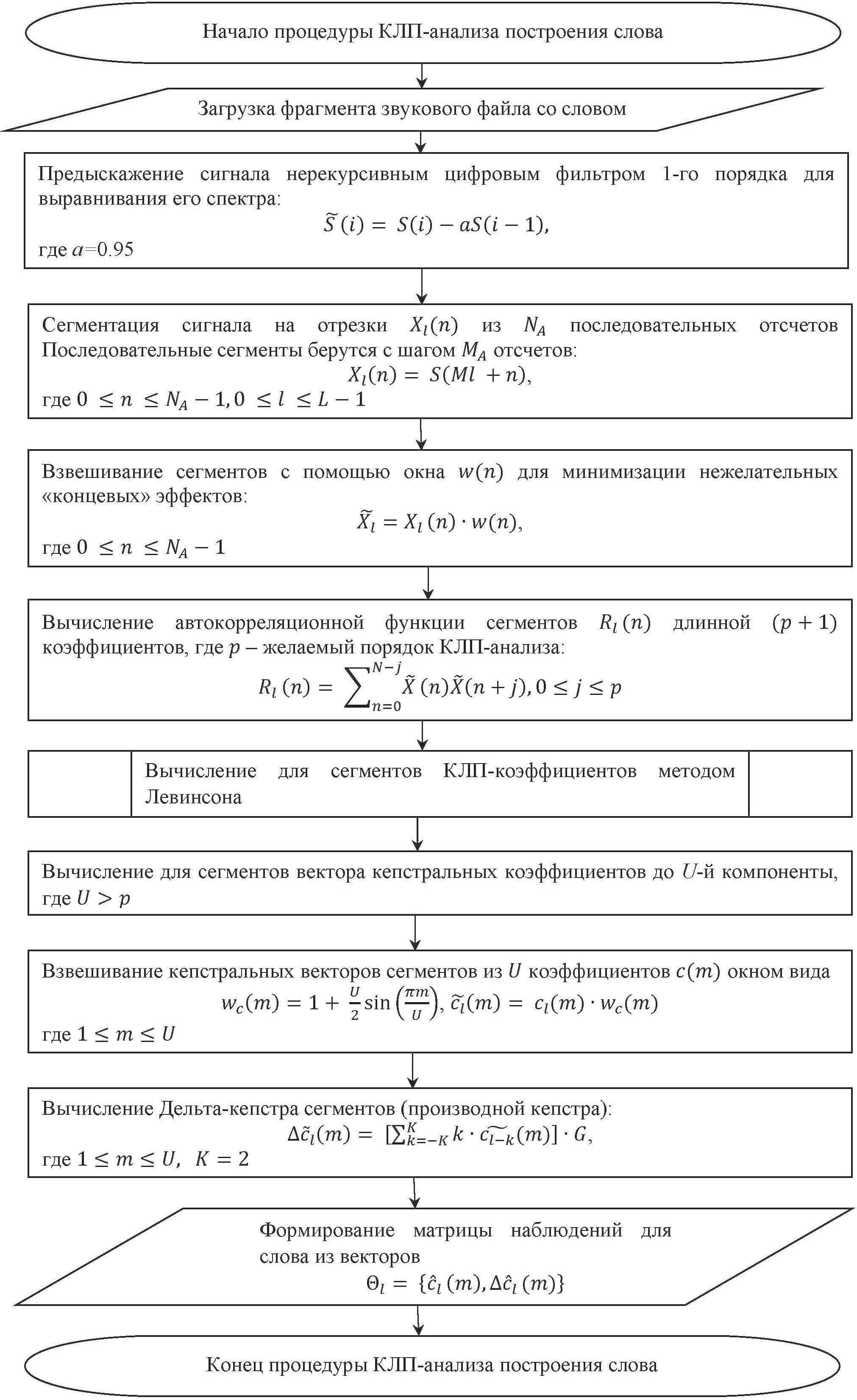


Рис. 6. Блок-схема алгоритма анализа признаков слова на основе КЛП Fig. 6. Flowchart of word analysis algorithm based on LPC

Достоинством спектральной обработки звуковых сигналов является то, что при переходе из временной области в частотную представление информации становится

более наглядным, компактным. Причем, чем более «простым» является сигнал во временной области, тем в большей степени происходит сжатие информации.

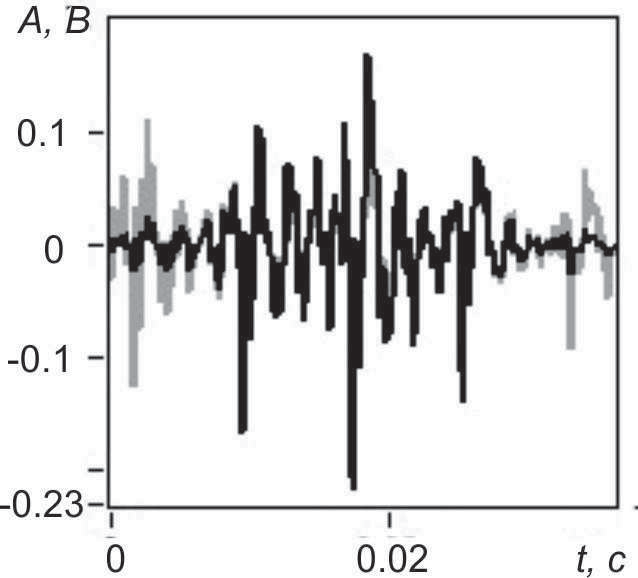
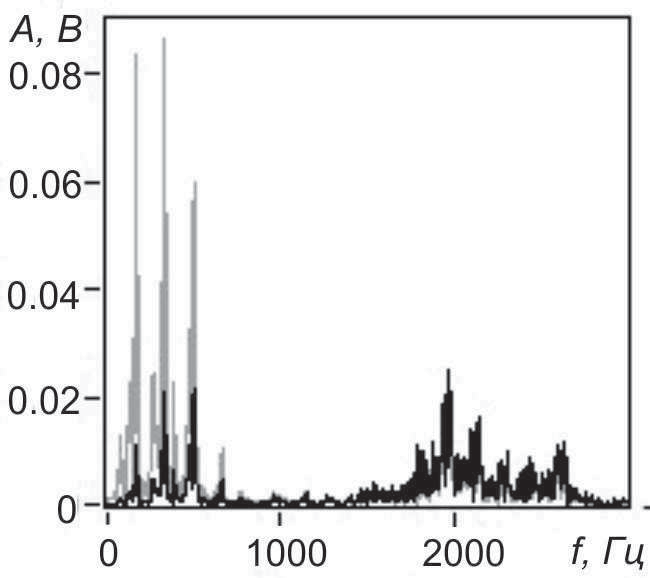
Выявление периодичности в спектре (кепстральный анализ) позволяет более до- стоверно и точно охарактеризовать особенности произношения дикторов. При этом спектральная информация представляется еще более компактно. Каждый гармони- ческий ряд исходного спектра представляется в идеале всего одной составляющей в кепстре [4].

Использование линейного предсказания, основанного на автокорреляционной фильтрации, должно улучшать отношение сигнал – шум исходного сигнала и убирать из него случайные артефакты. Вычисление коэффициентов линейного предсказания осуществляется с помощью алгоритма Левинсона (см. рис. 5) [6].

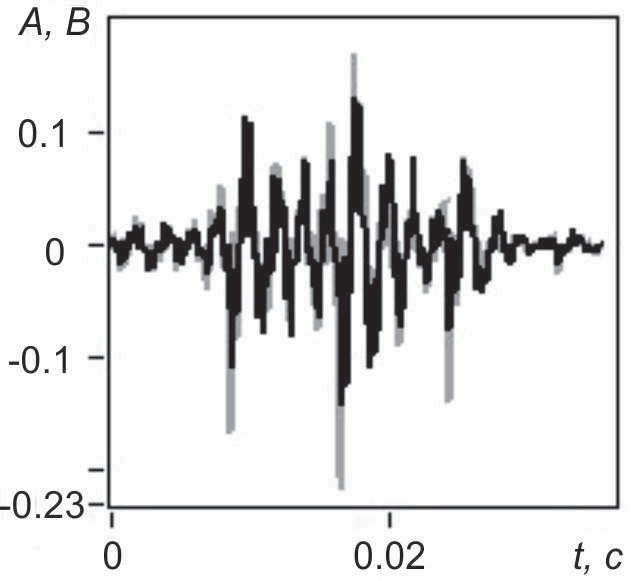
В процессе анализа признаков слова каждый участок, выделенный ранее из файла и соответствующий повторяющемуся слову, разбивается на небольшие перекрываю- щиеся отрезки – сегменты и затем обрабатывается согласно алгоритму, приведенному на рис. 5, 6. Как видно (рис. 7, а), в результате предискажения сигнала происходит выравнивание спектра, что обеспечивает равноценность спектральных компонент при анализе признаков.

Оконное взвешивание (рис. 7, *б*) уменьшает сигнал на концах сегментов и уве- личивает в центре, минимизируя нежелательные концевые эффекты.

Линейное предсказание (рис. 7, *в*) на основе алгоритма Левинсона (см. рис. 5) убирает сглаживает выбросы и случайные артефакты в анализируемом сигнале.



а / *a б / b*



*в / с*

Рис. 7. Спектр исходного (**–**) и предискаженного (–) сигналов (а), исходный (–) и взвешенный (**–**) сегменты сигнала (*б*), исходный взвешенный (–) и предсказанный (**–**) сегменты сигнала (*в*)

Fig. 7. The spectrum of the initial (**–**) and pre-faded (–) signals (*a*), the initial (–) and weighted (**–**) signal segments (*b*), the initial weighted (–) and predicted (**–**) signal segments (*c*)

На выходе алгоритма формируется необходимая для распознавания слова матри- ца, строки которой образуются конкатенацией взвешенного кепстрального и соот- ветствующего взвешенного дельта-кепстрального векторов сегментов. Каждая такая строка является набором признаков сегмента — вектором наблюдений и соответству- ет одному символу из алфавита *V* СММ слова в последовательности наблюдений *O*. Количество строк определяет число состояний *N* , в которых находилась лево-правая СММ слова.

Настройка параметров КЛП (число отсчётов в сегменте *NA*, число отсчётов в смещении сегментов *MA*, порядок КЛП-анализа *p*, число кепстральных коэффици- ентов *U* ) осуществляется посредством соответсвующих элементов (см. рис. 5).

# ПОСТРОЕНИЕ СММ СЛОВА

В процессе построения СММ слова для повышения её адекватности необходимо использовать матрицу векторов наблюдений, полученную статистическим усреднени- ем матриц векторов наблюдений повторяющихся слов. При этом усредняются векто- ра наблюдений, соответствующие одним и тем же моментам времени повторяющихся слов.

Оценка достоверности выборочных средних значений признаков сегментов (эле- ментов векторов наблюдений) повторяющихся слов проводится с помощью довери- тельных интервалов, вычисляемых при уровне статистической значимости *α* = 0*,* 05. Сравнение степени разброса, т. е. оценка однородности выборочных дисперсий значений элементов векторов наблюдений, вычисленных по результатам анализа при- знаков повторяющихся слов, осуществляется с использованием критерия Кохрена [7] Обеспечение статистически значимых выборочных средних значений элементов векторов наблюдений и однородности их дисперсий, т. е. получение достоверной по- следовательности наблюдений для анализируемого слова, достигается подбором па-

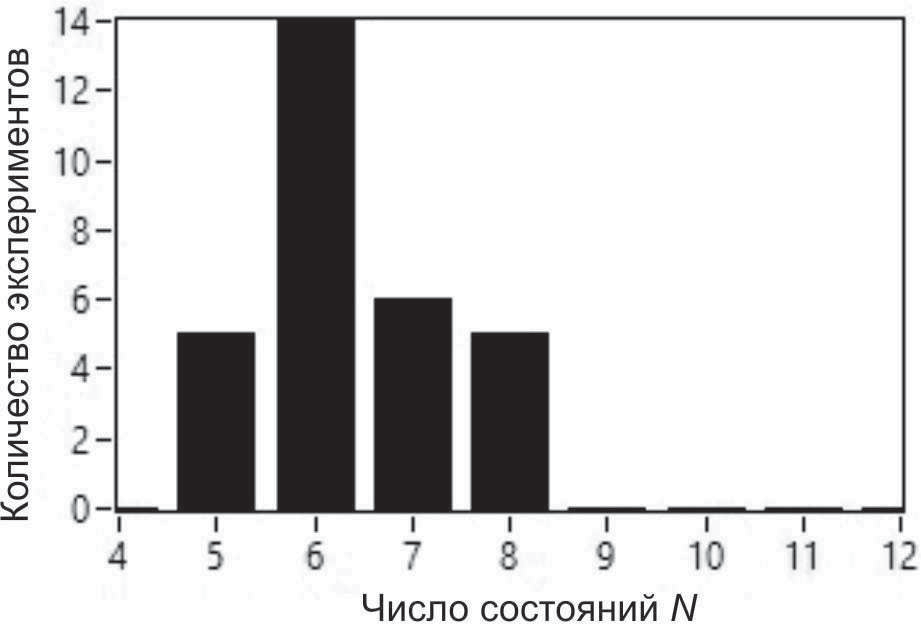
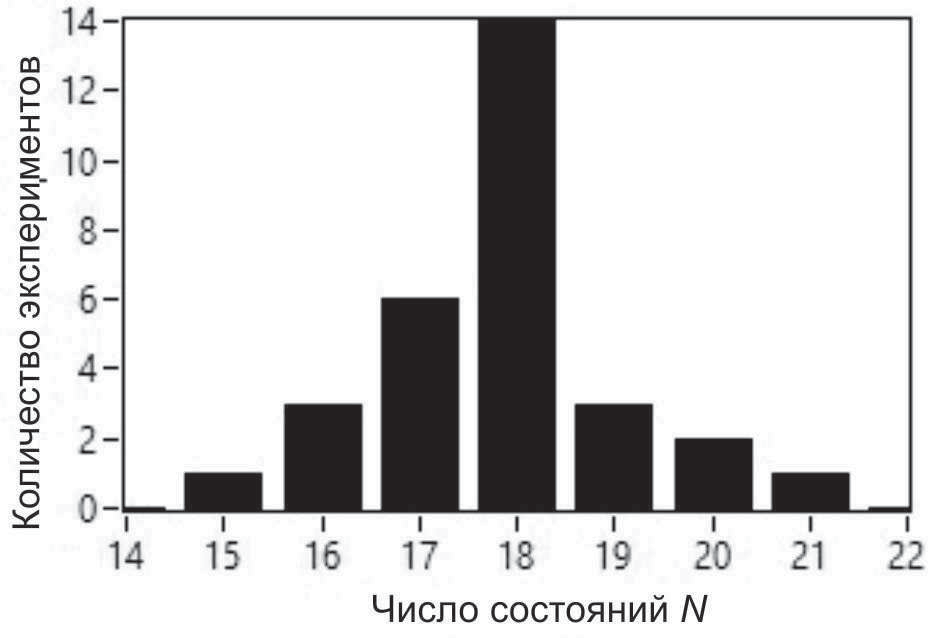
раметров обработки входного сигнала и параметров КЛП (см. п. 2, 3).

Оценку расстояния между символами алфавита *V* слова — усреднёнными век- торами наблюдений — было предложено делать с помощью евклидовой нормы. При этом для обеспечения равнозначности признаков при вычислении расстояния про- водилась их нормировка. В качестве нормирующего для каждого элемента вектора наблюдений использовался диапазон его изменения в матрице, расширенный с учё- том доверительного интервала.

Формирование алфавита СММ слова осуществляется удалением повторяющихся строк из нормированной матрицы средних значений наблюдений, если такие имеют- ся. При этом сравнивается расстояние между текущей строкой и остальными. Если оно меньше некоторой заданной величины, то строка с большим индексом удаля- ется, так как считаем, что эти строки соответствуют одному и тому же символу. Соответственно число строк получившейся прореженной матрицы определяет коли- чество символов *M* алфавита *V* , индексы строк являются значениями алфавита, а сами строки — признаками символов.

Последовательность наблюдений слова определяется сравнением строк матрицы алфавита *V* (прореженной) с исходной нормированной матрицей средних значений наблюдений. Если расстояние между строками меньше некоторой заданной величи- ны, использованной при построении алфавита, то индекс строки матрицы алфави- та *V* записывается в последовательность наблюдений *O*. Длина последовательности наблюдений *T* равна числу строк исходной матрицы, а число состояний *N* равно *T* в случае лево-правой СММ слова.

На рис. 8 приведены результаты экспериментов по определению числа состоя- ний *N* для слов «Вперёд» и «Стоп». На вход модуля построения СММ для каждого из этих слов подавалось тридцать звуковых файлов, содержащих по пятнадцать повторений, произнесённых одним диктором. Во всех случаях число состояний *N* совпадало с количеством символов *M* алфавита *V* , что соответствует предположе- нию о лево-правой структуре СММ (см. рис. 1) для этих слов из-за отсутствия в них повторяющихся звуков.



а / *a б / b*

Рис. 8. Количество полученных в экспериментах состояний для слов «Вперёд» (а) и «Стоп» (*б*)

Fig. 8. The number of states obtained in the experiments for the words „Forward“ (*a*) and „Stop“ (*b*)

Как видно из рис. 8, закон распределения отклонений числа состояний *N* от средних значений близок к нормальному. Соответственно для слова «Вперёд» сред- нее значение *N* составило 18 2*.*6% при 95% -й доверительной вероятности, а для слова «Стоп» — 6 5*.*7%.

±

±

Следовательно, данные, полученные с помощью предложенного алгоритма пред- варительной обработки звукового файла в совокупности с КЛП-анализом, являются достаточно надёжными и их можно использовать для построения СММ слова.

Процесс построения начинается генерированием по известному числу состояний *N* и количеству символов *M* алфавита *V* исходной СММ *λ* = (*A, B, π*), имеющей случайные параметры. При этом матрица вероятностей переходов между состояни- ями A, матрица вероятностей каждого наблюдения в каждом состоянии B, а также вектор вероятностей начального состояния *π* должны удовлетворять стохастическим ограничениям [3].

Далее необходимо так подобрать параметры исходной СММ, чтобы вероят- ность соответствия последовательности наблюдений, сгенерированной этой СММ *λ*∗ = (*A*∗*, B*∗*, π*∗) и полученной ранее последовательности наблюдений *O* = *o*1*, . . . , oT* слова, была максимально возможной. То есть исходную СММ *λ* = (*A, B, π*), име- ющую вероятность *p*(*O*|*λ*) генерирования заданной последовательности наблюде- ния *O* = *o*1*, . . . , oT* слова, надо обучить по этой последовательности наблюде- ния *O* = *o*1*, . . . , oT* , чтобы вероятность *p*(*O λ*∗) генерирования последовательности

|

*O* = *o*1*, . . . , oT* , обученной СММ *λ*∗ = (*A*∗*, B*∗*, π*∗), была максимально возможной.

Одним из вариантов обучения СММ *λ* = (*A, B, π*) по заданной последовательно-

сти наблюдений *O* = *o*1*, . . . , oT* является применение алгоритма Баума – Велша [3].

Алгоритм позволяет уточнять параметры исходной СММ *λ* = (*A, B, π*) таким образом, чтобы у уточнённой СММ *λ*∗ = (*A*∗*, B*∗*, π*∗) вероятность *p*(*O λ*∗) увеличи- валась. Итеративное применении алгоритма до схождения в одной точке позволяет максимизировать *p*(*O*|*λ*∗), т. е. настроить СММ *λ*∗ = (*A*∗*, B*∗*, π*∗) на заданную после- довательность наблюдений *O* = *o*1*, . . . , oT* слова. На рис. 9 приведены зависимости изменений вероятности *p*(*O λ*∗) и её приращения ∆*p*(*O λ*∗) на каждом итерационном шаге при настройке методом Баума – Велша СММ на слово «Стоп». Эти зависимости имеют характерный для метода Баума – Велша вид.

|

| |

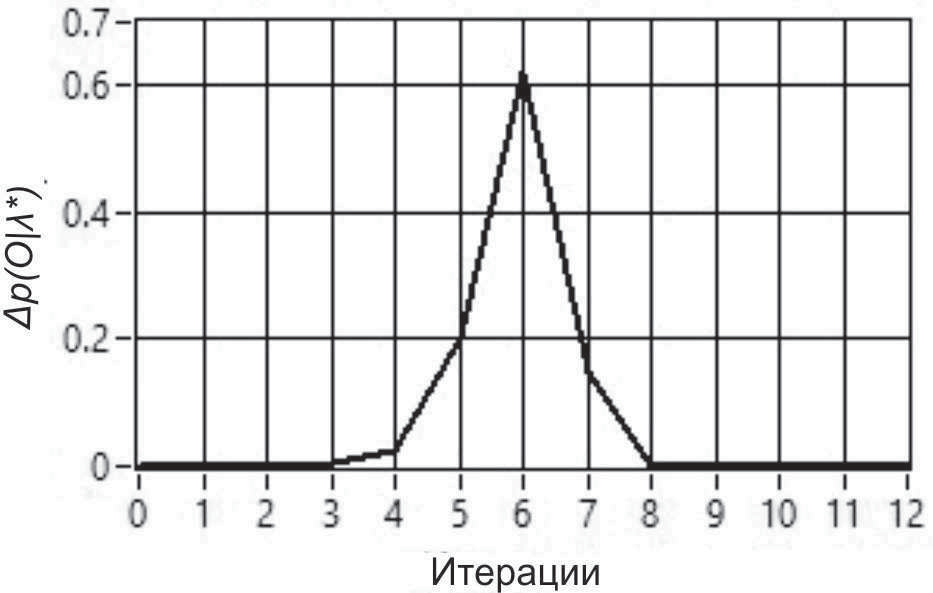
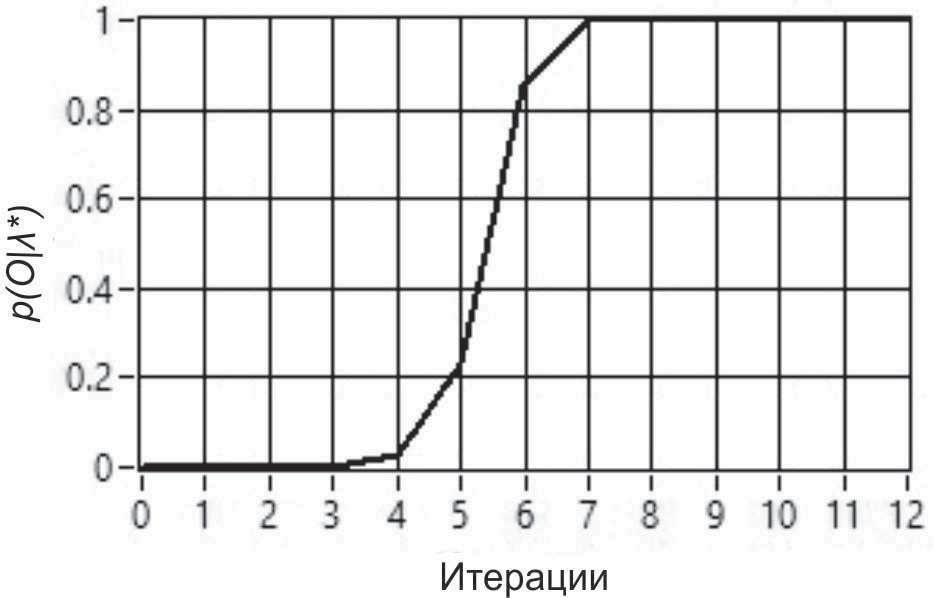
В начале обучения значения вероятности *p*(*O λ*∗) имеют, как правило, величи- ны меньшие или сравнимые с используемой для оценки сходимости положительной величиной *ε* (см. рис. 9, а), но разность значений *p*(*O*|*λ*∗) на каждом шаге увеличи- вается, т. е. приращение ∆*p*(*O*|*λ*∗) *>* 0 растет (см. рис. 9, *б*).

|

При завершении обучения значения вероятности *p*(*O λ*∗), как правило, сходятся к некоторой величине, при этом приращение ∆*p*(*O λ*∗) *>* 0, но оно начинает умень- шаться и стремиться к 0.

|

|



а / *a б / b*

Рис. 9. Изменения вероятности СММ *p*(*O*|*λ*∗): а — и её приращения ∆*p*(*O*|*λ*∗); *б* — при обучении методом Баума – Велша

Fig. 9. Changes in the probability of the HMM *p*(*O*|*λ*∗): *a* — and its increment ∆*p*(*O*|*λ*∗);

*b* — when learning by the Baum – Welsh method

Следовательно, для корректной оценки сходимости итерационного процесса обу- чения необходимо контролировать не только величину приращения ∆*p*(*O λ*∗), но и знак его изменения, т. е. для завершения обучения должно выполняться условие

|

∆*p*(*O*|*λ*∗) ≤ *ε* при уменьшении ∆*p*(*O*|*λ*∗).

В разработанном модуле построения СММ для оценки сходимости процесса обу- чения использовано значение *ε* = 1*.*1 ∗ 10−19, соответствующее машинной точности. Недостатком алгоритма Баума – Велша при обучении СММ является поиск ло- кального максимума *p*(*O λ*∗), а не глобального. Поэтому для достижения хорошего результата требуется, как правило, несколько запусков при различных начальных

|

условиях.

Таким образом, используя последовательности наблюдений и алфавит моделиру- емых слов, получаемые на первых этапах обработки, с помощью алгоритма Баума – Велша можно строить соответствующие адекватные СММ для систем распознавания речи.

# ЗАКЛЮЧЕНИЕ

Разработанный программный модуль позволяет эффективно подготавливать необ- ходимые исходные данные на основе кодирования признаков звукового сигнала, ис- пользующего линейные предсказания, строить СММ отдельных слов и проводить их обучение с помощью алгоритма Баума – Велша. Построенные СММ слов предполага- ется использовать в интеллектуальных системах управления различными объектами.

# РАСПОЗНАВАНИЕ СПАМ-СООБЩЕНИЙ С ИСПОЛЬЗОВАНИЕМ МЕТОДОВ МАШИННОГО ОБУЧЕНИЯ

# Введение

По данным Лаборатории Касперского [1] каждый год спам занимает примерно половину объема всего мирового почтового трафика. Сначала спам рассылался напрямую на единичные адреса пользователей, и его было легко блокировать. Со временем появились сложные системы мас- совой рассылки и высокоскоростные интернет-каналы, которые позволи- ли быстро и дешево осуществлять рассылку спама.

В связи с этим были разработаны различные методы распознавания спам-сообщений. Обнаружение спама в электронной почте может быть выполнено как с использованием методов машинного обучения, так и с помощью других средств [2]. В представляемой работе используется подход, основанный на использовании алгоритмов машинного обучения для классификации текстов.

# Постановка задачи и используемые алгоритмы

Пусть L={L1,…,Lm} — множество писем электронной почты с мет- ками (обычное письмо или спам), а U={U1,…,Un} — неразмеченное множество писем, где Ui соответствует *i*-му письму. Предполагается, что элементы L и U отличны друг от друга. Письмо представляется в виде вектора признаков, размерность которого равна размеру словаря V, xi=(xi1,xi2,…,xi|v|), где xij=1, если *i*-е письмо содержит *j*-е слово и xij=0 – иначе. На этих данных строится фильтр F: U→{1,0}, который классифи- цирует письмо как спам или реальное письмо.

Для классификации текстов в работе использовались следующие ал- горитмы:

-упрощенный алгоритм Байеса [4] — вероятностный классификатор, основанный на теории Байеса с допущением о независимости признаков;

* метод опорных векторов (англ. support vector machine, SVM) [5], который строит в *n*-мерном пространстве признаков такую гиперплос- кость, чтобы она разделяла объекты выборки наиболее точно;
* метод *k* ближайших соседей (англ. k-nearest neighbors, kNN) [6], суть которого заключается в том, что находятся *k* соседей, которые наиболее близко расположены к рассматриваемому объекту с неизвест- ной меткой. Далее новый объект относят тому классу, который является наиболее распространённым среди *k* соседей;
* многослойный персептрон (англ. multilayer perceptron, MLP) [7] — один из вариантов нейронной сети прямого распространения. Под про- цессом обучения нейронной сети понимается поиск таких значений ве- сов и порогов сети, которые минимизирую ошибку. На основе собран- ных исторических данных веса и пороговые значения корректируются автоматически. В процессе корректировки происходит расчет ошибки путем вычисления выходных сигналов и сравнения их с целевыми.

# Обзор данных и предобработка текста

В работе использовался набор данных Spam Email, опубликованный на платформе Kaggle [3]. В нем содержится 86,6 % реальных сообщений (4825 строк) и 13,4 % спама (747 строк), эти два вида сообщений пере- мешаны между собой. Для работы алгоритмов классификации требуется произвести нормализацию и векторизацию исходных сообщений, для чего использовался язык программирования Python и программная библиотека NLTK.

Нормализация текста включает в себя следующие действия:

1. приведение всех документов к нижнему регистру;
2. удаление слов, не содержащих смысловой информации;
3. удаление знаков пунктуации;
4. разбиение документов на токены; в работе для сравнения ис- пользуются униграммы (токен состоит из одного слова) и триграммы (токен состоит из трех слов);
5. лемматизация текста, иными словами, приведение слова к начальной форме, учитывая морфологию слова.

Алгоритмы машинного обучения не имеют функционала для работы с текстом на естественных языках. По этой причине текст должен быть преобразован в числовые векторы. Распространенным методом извлече- ния признаков из текста является формирование множества, элементами которого являются отдельные слова, встречающиеся в тексте.

В качестве входных данных модели векторизации принимают токе- низированные текстовые данные, над ними могут производиться не все этапы нормализации. В случае использования «мешка» слов [9] вместо

токена обозначена его частота использования в отдельном документе. В случае TF-IDF — подсчитывается важность каждого токена в докумен- тах [9]. В работе используются следующие вариации векторизации:

1. модель «мешка слов», документы разбиты на униграммы (первый вариант);
2. модель «мешка слов», документы разбиты на триграммы (второй вариант);
3. модель TF-IDF, документы разбиты на униграммы (третий вариант);
4. модель TF-IDF, документы разбиты на триграммы (четвер- тый вариант).

# Полученные результаты

Обучение проводилось на 70 % данных для всех алгоритмов клас- сификации. Распределение происходило следующим образом: трениро- вочному набору соответствует первые 70 % документов, остальные 30 % документов относятся к тестовому набору.

Подбор параметров алгоритмов осуществлялся методом перебора.

В итоге были использованы следующие параметры:

* в качестве классификатора для реализации алгоритма Байеса взят

«MultinomialNB»;

* для алгоритма SVM использовалась линейная функция ядра;
* в алгоритме KNN для обучения было выбрано 3 «соседа»;
* в MLP было 2 скрытых слоя по 2 нейрона, функция активации скрытого слоя — «logistic», функция оптимизации весов «adam», посто- янная скорость обучения.

В таблицах 1 и 2 представлены лучшие результаты для каждого ме- тода, полученные с использованием Python с библиотекой Scikit-learn и WEKA соответственно. Для оценки моделей использовались следующие показатели (используются общепринятые англоязычные названия, т. к. перевод на русский может внести неоднозначность) [9]: accuracy, recall, precision, F1-score, specificity, time.

*Таблица* 1

**Результаты тестирования моделей, созданных на Python**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Метод и харак-  теристика | Accuracy | Recall | Precision | F1\_score | Specificity | Time |
| Метод Байеса,  «мешок» слов, униграммы | 0.967 | 0.973 | **0.988** | 0.98 | **0.93** | **3.1s** |
| SVM, «мешок» слов, униграммы | **0.972** | **0.992** | 0.976 | **0.984** | 0.847 | 17.2s |
| KNN, TF-IDF,  униграммы | 0.949 | 0.983 | 0.959 | 0.971 | 0.738 | 5s |
| MLP, «мешок» слов, униграммы | 0.971 | 0.987 | 0.978 | 0.982 | 0.856 | 67.29s |

*Таблица* 2

**Результаты тестирования моделей, Weka**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Метод и харак- теристика | Accuracy | Recall | Precision | F1\_score | Specificity | Time |
| Метод Байеса,  «мешок» слов, униграммы | 0.964 | 0.97 | **0.985** | 0.979 | **0.985** | **0.65s** |
| SVM, «мешок»  слов, униграммы | **0.982** | **0.996** | 0.984 | **0.99** | 0.984 | 1.3s |
| KNN, «мешок» слов, униграммы | 0.951 | 0.987 | 0.947 | 0.967 | 0.947 | 83.27s |
| MLP, «мешок» слов,  униграммы | 0.963 | 0.994 | 0.981 | 0.987 | **0.985** | 23.46s |

В таблицах 1 и 2 в столбце “time” указано суммарное время обуче- ния и тестирования модели. Из полученных результатов следует, что мо- дели, обученные на данных с унарной токенизацией, производят класси- фикацию точнее, а обучение производится быстрее. Скорость работы связана с тем, что в наборах с унарной токенизацией меньше элементов, поэтому параметр времени не учитывался при сравнении моделей, по- строенных на данных с разным содержанием токенов.

# Тестирование на новых данных и анализ результатов

Дополнительно было произведено тестирование моделей классифи- кации с добавлением новых данных, не входящих в первоначальный набор текстовых сообщений. Данный набор взят с платформы Kaggle [8]. Из набора были выбраны первые 2000 строк, документы были нормали- зованы и преобразованы в векторы признаков аналогично основному набору. После добавления новых документов общее их количество со- ставило 7555. После нормализации и удаления документов, полностью состоящих из стоп-слов, количество документов составило 6833.

Тестирование проводилось для тех сочетаний алгоритмов, парамет- ров и методов предобработки, которые показали лучшие результаты на предыдущей проверке для языка Python. В таблице 3 отображены значе- ния метрик для каждого случая.

Тестирование на новых данных показало эффективность рассмот- ренных моделей классификации. Метод опорных векторов показал наилучшие результаты. При этом скорость работы у него не такая высо- кая, как у методов Байеса или KNN.

*Таблица* 3

**Результаты классификации на новых данных**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Метод и харак- теристика | Accuracy | Recall | Precision | F1\_score | Specificity | Time |
| Метод Байеса,  «мешок» слов, униграммы | 0.985 | 0.989 | 0.993 | 0.99 | 0.958 | **4.4s** |
| SVM, «мешок» слов, униграммы | **0.996** | **0.999** | **0.996** | **0.998** | **0.974** | 28.8s |
| KNN, TF-IDF,  униграммы | 0.994 | 0.998 | 0.995 | 0.996 | 0.968 | 6.4s |
| MLP, «мешок» слов,  униграммы | 0.989 | 0.993 | 0.992 | 0.993 | 0.951 | 84.69s |

Таким образом, если требуется высокая скорость обучения, стоит выбирать один из следующих методов: полиномиальный метод Байеса или KNN. Если обучение моделей производится редко или есть потреб- ность в очень высокой точности классификации, стоит использовать ме- тод опорных векторов.

В исследовании Н. Сутта, З. Лю, Х. Чжан [10] было проведено срав- нение методов SVM, KNN, Байеса и других методов для выявления наиболее точных классификаторов. В указанной публикации для анализа тоже использовались два набора данных: первый состоял из одного набора сообщений, второй — из двух разных. Данные были представле- ны в виде векторов признаков по схеме TF-IDF, использовались уни- граммы, биграммы и триграммы. Наибольшая точность была достигнута при обучении модели SVM c линейной функцией ядра на втором виде набора данных, с токенизацией при *n* = 1 и *n* = 2. Точность (англ. accura- cy) модели SVM ≈ 0.99. KNN показал точность меньше, при этом макси- мальная точность достигается при использовании биграмм. Классифика- тор, построенный по модели Байеса, показал худший результат по всем проверенным n-граммам практически среди всех моделей, проверенных в работе.

Результаты анализа, произведенного в исследовании [10], близки к тем, что были получены в представляемой работе: высокие точности до- стигаются при обучении на данных, состоящих из разных наборов сооб- щений, в обоих случаях SVM показывает более высокую точность по сравнению с другими алгоритмами. В работе Н. Сутта, З. Лю, Х. Чжан модели, обученные на n-граммах (при *n* > 1), показывали высокие ре- зультаты, что может быть объяснено большим объемом использованного набора данных (более 100 тысяч сообщений).

# Заключение

В работе были построены классификаторы для определения спама в текстовых сообщениях. В качестве предварительной обработки текстов были проведены нормализация данных и представление полученных до- кументов в векторы признаков по схемам «мешка» слов и TF-IDF.

В работе была выявлена неэффективность обучения на данных с триграммной токенизацией. Это связано с относительно небольшим набором данных для обучения. Словосочетания, состоящие из несколь- ких слов, встречаются реже в предложениях, чем отдельно взятые слова.

Ко второй проблеме можно отнести скорость обучения и тестирова- ния моделей. При увеличении объема данных время, затрачиваемое на обучение модели классификации, будет увеличиваться. В таком случае требуется использовать программные реализации на языках с высокой скоростью работы. В работе были рассмотрены реализации на двух язы- ках: Python и Java (Weka). Скорость выполнения классификации на Java выше, чем на Python. Следовательно, если требуется высокая скорость обучения и классификации, лучше использовать Java.

Высокой точности удалось достичь при обучении модели SVM на документах, разбитых на униграммы и с векторизацией «мешка слов» в Weka и Python: значения меры F1 равны 0.984 и 0.99 соответственно, что является неплохим результатом. После добавления новых докумен- тов для тестирования, значения точностей и других метрик моделей воз- росли. Мера F1 модели SVM стала равна 0.998.

**Программная инструментальная система создания адаптивных пользовательских интерфейсов**

## Введение

Пользовательский интерфейс — средство, обеспе- чивающее взаимопонимание человека и компьютера. Потому очень важно сделать такую связь дружествен- ной и интуитивно-понятной.

Проблемам разработки пользовательских интерфей- сов посвящено значительное количество отечественных и зарубежных научных трудов, которые используются в различных предметных областях [1–5]. При их проек- тировании современными авторами предлагаются раз- личные подходы. Часто при разработке интерфейсной части программного обеспечения применяются методы искусственного интеллекта: построение адаптивных интерфейсов на основе генетических алгоритмов [6], экспертные системы, теории нечетких множеств [7–9] и др. Вместе с тем, с целью облегчения работы поль- зователя с программным средством, задействованы методы системного анализа, синтеза, абстрагирования и построения онтологической модели [10]. Часть уче- ных при решении проблемы построения адаптивных пользовательских интерфейсов предлагают использо- вание принципа разделения декларативного описания математических моделей и их процедурную интерпре- тацию [11]. Другими авторами применяются известные модели качества программного обеспечения [12], также используются метафорические или идиоматические подходы при создании или усовершенствовании интер- фейсной части программного средства [13].

Также осуществляется проектирование прототипов интерфейса с недетерминированным конечным авто- матом [14], и разрабатываются специализированные системы [15, 16]. Имеется опыт непосредственного привлечения потенциальных пользователей к созданию интерфейсной части программного средства [17].

В системах автоматизации проектирования (САПР) пользовательский интерфейс — важная часть системы. Интерфейс входит в состав лингвистического обеспе- чения и представлен диалоговыми языками. Однако

функциональность систем не стоит на месте и посто- янно расширяет свои возможности, это в свою очередь отражается на интерфейсе пользователя. Он становится более сложным, непонятным для новичков и не эр- гономичным, а это сказывается на производительно- сти труда инженера-конструктора. Проблема создания адаптивных интерфейсов для прикладных программ в САПР также актуальна [2, 18].

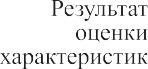
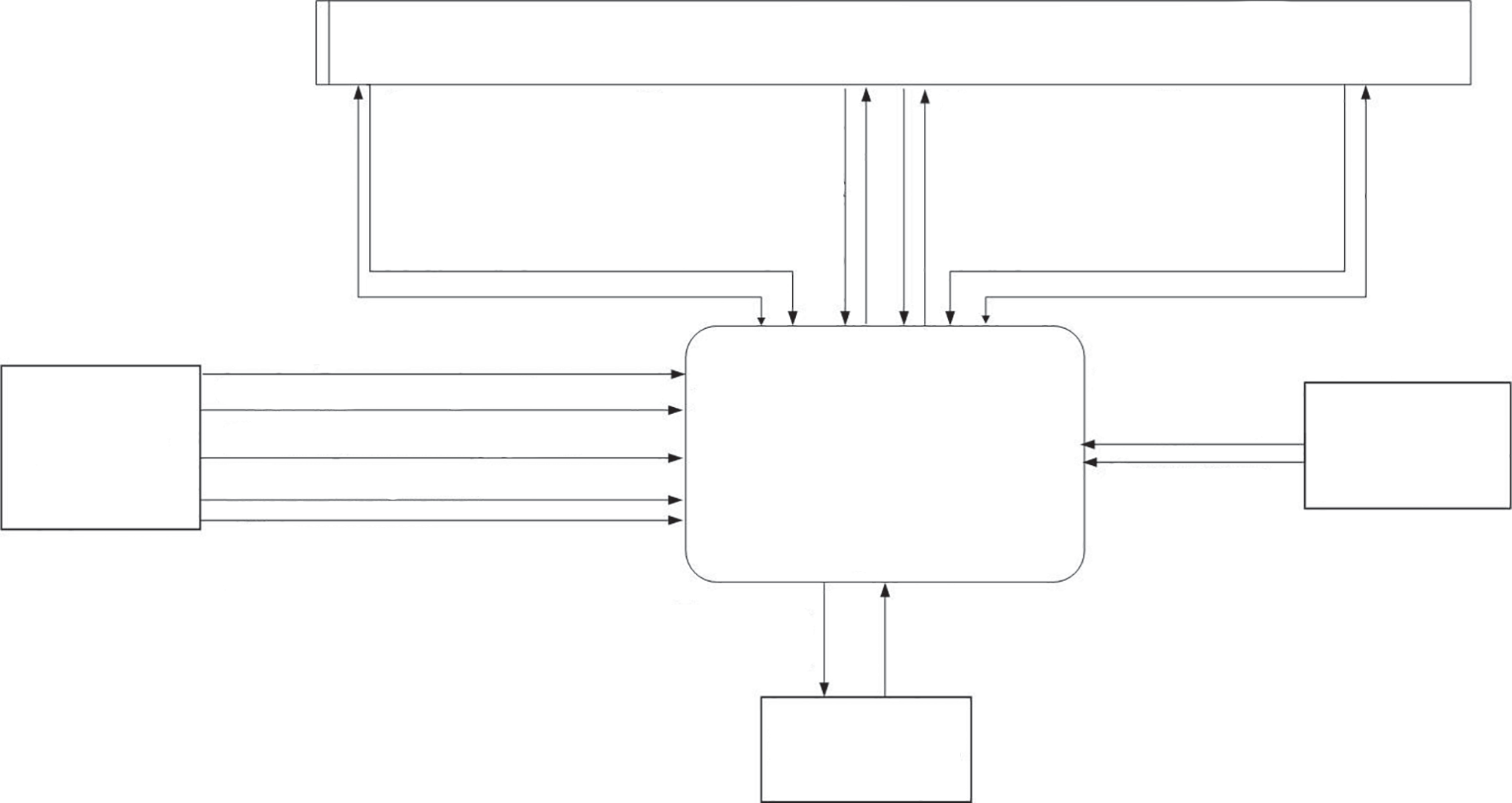
В отличие от существующих аналогов, предлага- емая программная инструментальная система (ПИС) позволяет подбирать не шаблон целиком, а каждый ком- понент пользовательского интерфейса (размер шрифта, кнопок, расстояние между кнопками, цветовая гамма, звуковое сопровождение, наличие подсказок и команд- ной строки). Таким образом, шаблон интерфейса уни- версален для каждого пользователя.

## Постановка задачи

В настоящей работе реализовано решение проблемы адаптации интерфейсов к особенностям пользователя на примере инженера-конструктора и его автоматизиро- ванного рабочего места. Рабочее место включает в себя прикладные программы для проектирования машино- строительных изделий. Для решения данной задачи вы- полнена разработка специальной программной системы для создания адаптивных прототипов интерфейсов на основе характеристик пользователя.

Разработанная ПИС имеет возможность создания прототипа интерфейса, адаптированного под характе- ристики пользователя. ПИС реализована в три этапа: оценка характеристик пользователя; реализация под- бора компонентов интерфейса; применение подобран- ного набора компонентов к интерфейсу прикладной программы.

Для наглядности представления движения информа- ционных потоков при проектировании ПИС построена потоковая модель Data Flow Diagram (DFD) с помощью Case-средства автоматизированного проектирования



*Рис. 1*. Движение информационных потоков при работе программной инструментальной системы

*Fig. 1*. The movement of information flows during the operation of the software development system

BP-win. На рис. 1 представлена схема DFD с точки зрения программной системы.

Интерфейсная часть программного обеспечения сформирована на основе оценки характеристик поль- зователя. Для каждого пользователя подобраны ком- поненты интерфейса: размер шрифта, размер кнопок, расстояние между кнопками, цветовая гамма, наличие командной строки, наличие звукового сопровожде- ния и наличие подсказок. После оценки характеристик пользователя результаты сохраняются в базе данных программной системы.

Для проведения тестирования пользователей в ПИС разработчиком вносятся контрольно-измерительные материалы, направленные на диагностику сформиро- ванности каждой его характеристики.

При подборе компонентов интерфейса под опреде- ленного пользователя использован метод экспертного оценивания. Для этого на начальном этапе работы экс- перт формирует базу правил на основе продукционной модели знаний.

В ходе работы ПИС выполнено сравнение резуль- татов оценки характеристик пользователя с правилами нечеткой экспертной системы, которая является ядром программного средства. В итоге формируется набор компонентов интерфейса и генерируется прототип ин- терфейса, который соответствует данному пользова- телю.

Декомпозиция контекстной диаграммы представ- лена на рис. 2.

Процесс разработки программного средства вклю- чает в себя восемь этапов (рис. 2). На начальном этапе формируется и сохраняется типовая информация. Затем формируется база оценочных материалов, по которым

будет производиться оценка характеристик пользова- теля.

Эксперт формирует структуру экспертной систе- мы, создавая лингвистические переменные и термы. Входные переменные — оцениваемые характеристики пользователя, а выходные — компоненты интерфейса, на основе которых будет формироваться прототип при- кладной программы.

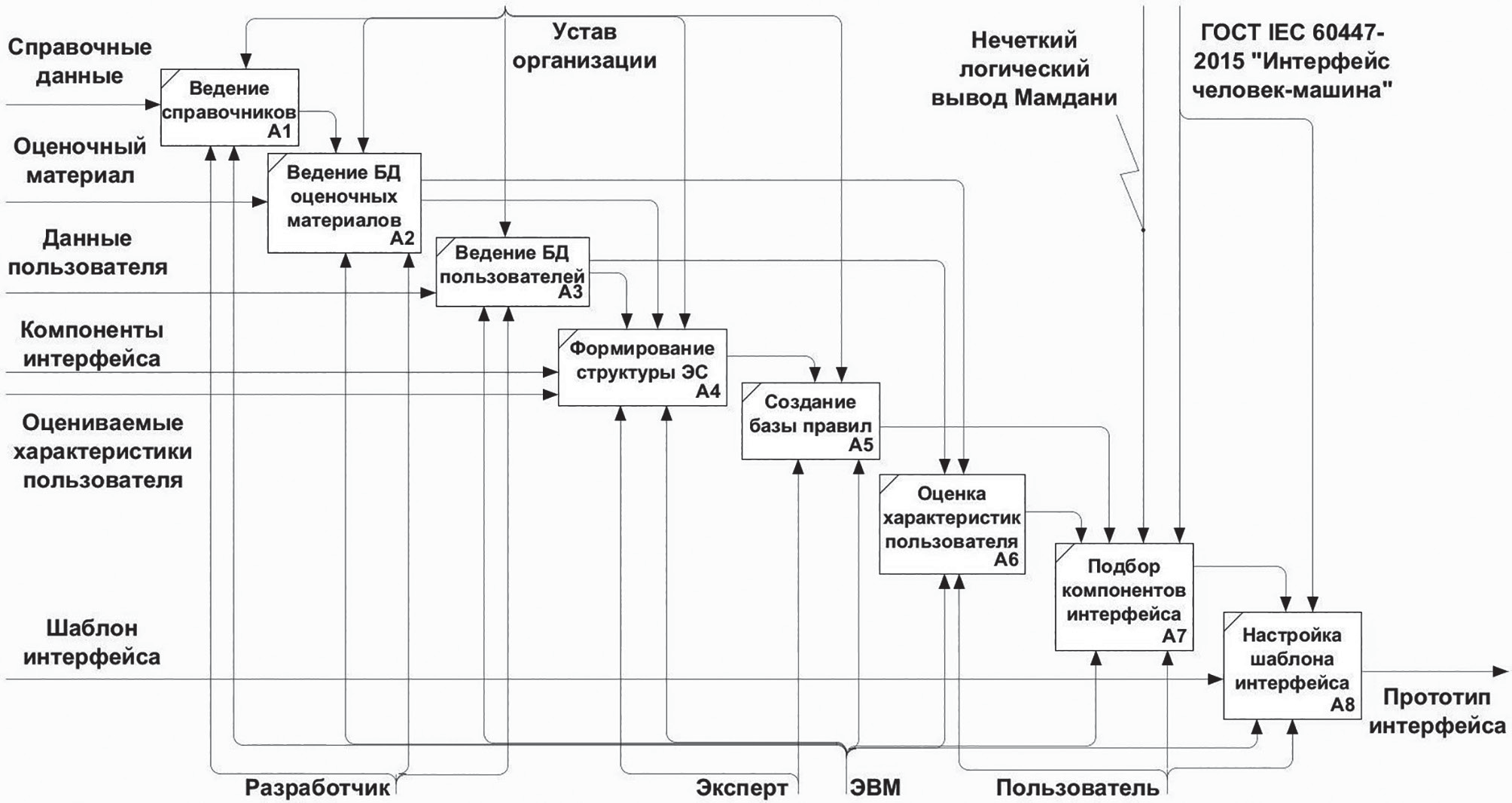
Следующий этап — оценка характеристик пользо- вателя, при котором требуется выбрать характеристику и оценить ее с помощью разработанных оценочных ма- териалов. Основной этап — подбор компонентов интер- фейса, где загружаются результаты оценки пользователя и база правил нечеткой экспертной системы. В ходе рабо- ты экспертной системы вычисляется подходящий набор компонентов интерфейса для конкретного пользователя. На заключительном этапе происходит непосред- ственное создание прототипа адаптивного интерфейса прикладной программы, который предоставляется для

работы в системе САПР.

## Математическая модель экспертной системы

Инструментом для определения компонентов интер- фейса в ПИС служит нечеткая ЭС. Основные входные данные — сведения о пользователе. Данные харак- теризуются различной степенью неопределенности, неоднозначности, внутренней противоречивостью, неполнотой, а также представляют количественные и качественные оценки параметров [19].

Так как данные являются трудно формализованны- ми и специфическими, то при выборе метода искус- ственного интеллекта использована интеллектуальная система, базирующаяся на нечеткой логике и нечетких множествах.



*Рис. 2*. Декомпозиция контекстной диаграммы.

БД — база данных; ЭВМ — электронно-вычислительная машина; ЭС — экспертная система

*Fig. 2*. Context diagram decomposition

БД — database; ЭВМ — electronic computer; ЭС — expert system

Из анализа нечеткого понятия «Системный опыт» было сформировано базовое терм-множество, со- стоящее из трех нечетких переменных: «Низкий»,

«Средний» и «Высокий», и установлена область рас- суждений в виде *X* = [0; 100] (баллов). Далее была по- строена функция принадлежности для каждого лингви- стического терма из базового терм-множества *T*.

Существует большое количество стандартных форм кривых для задания функций принадлежности. Самыми распространенными считаются: треугольная, трапецеи- дальная и гауссова функции принадлежности.

Совокупность функций принадлежности для каж- дого терма из базового терм-множества *T* обычно изо- бражается на одном графике. Пример лингвистической переменной «Системный опыт» представлен на рис. 3 в виде трапециевидной функции принадлежности.

Наиболее распространенный способ логического вывода в нечетких системах — механизм Мамдани. Механизм использует минимаксную композицию не-



*Рис. 3*. Графики функций принадлежности значений лингвистической переменной «Системный опыт»

*Fig. 3*. Graphs of the values belonging functions of the linguistic variable “System experience”

четких множеств и включает в себя последовательность действий [19].

*Фаззификация* или приведение к нечеткости. Определяются степени истинности, т. е. значения функций принадлежности для левых частей каждого правила (предпосылок или антецедентов). Для базы правил с *m* правилами обозначим степени истинности как *Aik*(*xk*), *i* = 1..*m*, *k* = 1..*n*.

*Нечеткий вывод*. Определяются уровни «отсечения»

для левой части каждого из правил: α*i* = min*i*(*Aik*(*xk*)), и находятся «усеченные» функции принадлежности: *Bi*\*(*y*) = min*i*(α*i*, *Bi*(*y*)).

*Композиция* или объединение полученных усе-

ченных функций. Используется максимальная ком- позиция нечетких множеств: μ(*y*) = max*i*(*Bi*\*(*y*)), где μ(*y*) — функция принадлежности итогового нечеткого множества.

*Дефаззификация* или приведение к четкости. Суще- ствует несколько методов дефаззификации. Например, центроидный метод или метод среднего центра [19].

С учетом введенных понятий построена нечеткая модель, основанная на бинарном нечетком отноше- нии *S*, которая строится на двух базисных множествах *X* и *Y*.

*X* = {*x*1{*z*1}, *x*2{*z*2}, *x*3{*z*3}, …, *x*7{*z*7}} описывает множество компонент интерфейса, где *z* — множество, характеризующее каждое *x*.

*Y* = {*y*1{*k*1}, *y*2{*k*2}, *y*3{*k*3}, …, *y*7{*k*7}} — множество характеристик пользователя, где *k* — множество, харак- теризующее каждое *y*.

Элементы универсумов имеют следующий содер- жательный смысл:

1. *x*1 — цветовая гамма (*z*1 — черно-белый, *z*2 — не- сколько цветов, *z*3 — любые цвета), *x*2 — размер шрифта (*z*1 — крупный, *z*2 — средний, *z*3 — мел- кий), *x*3 — размер кнопок (*z*1 — крупные, *z*2 — сред- ние, *z*3 — мелкие), *x*4 — расстояние между кноп- ками (*z*1 — большое, *z*2 — среднее, *z*3 — малое), *x*5 — звуковое сопровождение (*z*1 — наличие, *z*2 — отсутствие), *x*6 — наличие подсказок (*z*1 — наличие, *z*2 — отсутствие), *x*7 — наличие командной строки (*z*1 — наличие, *z*2 — отсутствие).
2. *y*1 — системный опыт (*k*1 — высокий, *k*2 — сред- ний, *k*3 — низкий), *y*2 — компьютерная грамотность (*k*1 — высокая, *k*2 — средняя, *k*3 — низкая), *y*3 — опыт работы с подобными программами (*k*1 — высо- кий, *k*2 — средний, *k*3 — низкий), *y*4 — машинопись (*k*1 — высокая, *k*2 — средняя, *k*3 — низкая), *y*5 — дальтонизм (*k*1 — есть, *k*2 — нет), *y*6 — моторика рук (*k*1 — высокая, *k*2 — средняя, *k*3 — низкая), *y*7 — память (*k*1 — высокая, *k*2 — средняя, *k*3 — низкая).

Входными данными являются характеристики пользователей, которые задаются лингвистическими переменными. В табл. 1 представлены все входные лингвистические переменные. Для каждой характе- ристики определено множество, которое измеряется в баллах. Для образования новых термов использо- ваны процедуры: синтактическая, представляющая собой логическую связку AND (И), и семантическая — min(μ*A*(*x*). μ*B*(*x*)).

На примере лингвистической переменной

«Системный опыт» описано присвоение значений тер-

мов и построение графиков функций принадлежности нечеткого множества (рис. 3). Для остальных входных лингвистических переменных произведена аналогичная процедура.

Далее добавлены лингвистические переменные вы- ходных данных. Они представлены в табл. 2. Для ка- ждой характеристики определено множество из разных единиц измерения.

Для работы механизма нечеткого вывода сформи- рованы продукционные правила. Фрагмент базы пра- вил представлен в табл. 3. Приведено формирования правил для лингвистической переменной «Цветовая гамма», для остальных переменных: «Размер кнопок»,

«Расстояние между кнопками», «Размер шрифта»,

«Звуковое сопровождение», «Подсказки», «Командная строка» терм-множества сформированы аналогично.

Экспертная система позволила получить решение на основе описанных правил базы знаний.

## Проектирование программной инструментальной системы

Для визуализации работы ПИС использована диа- грамма вариантов использования — визуальная модель, отражающая спецификацию программного средства с точки зрения ее функциональности (рис. 4).

Работать с программной системой могут эксперт и пользователь, которые входят в систему с разными правами доступа. Для пользователя подбирается ин- терфейс.

*Таблица 1*. Входные лингвистические переменные

*Table 1.* Input linguistic variables

|  |  |  |
| --- | --- | --- |
| Название | Терм-множество (*T*) | Множество-область (*X*), баллы |
| Системный опыт | Высокий | 65–100 |
| Средний | 35–70 |
| Низкий | 0–45 |
| Компьютерная грамотность | Высокий | 70–100 |
| Средний | 35–75 |
| Низкий | 0–40 |
| Опыт работы с подобными программами | Есть | 50–100 |
| Частично | 25–60 |
| Нет | 0–30 |
| Машинопись | Быстро | 75–150 |
| Нормально | 30–80 |
| Медленно | 0–40 |
| Дальтонизм | Есть | 0–1 |
| Нет | 0,9–2 |
| Моторика рук | Высокая | 65–100 |
| Средняя | 30–70 |
| Низкая | 0–35 |
| Память | Отличная | 75–100 |
| Умеренная | 40–80 |
| Плохая | 0–45 |

*Таблица 2*. Выходные лингвистические переменные

*Table 2.* Output linguistic variables

|  |  |  |
| --- | --- | --- |
| Название, единица измерения | Терм-множество (*T*) | Множество-область (*X*) |
| Размер шрифта, пиксел | Крупный | 14–18 |
| Средний | 11–15 |
| Мелкий | 8–12 |
| Размер кнопок, коэффициент | Крупный | 2–4 |
| Средний | 1,5–2,5 |
| Мелкий | 0–2 |
| Расстояние между кнопками, коэффициент | Большое | 2–4 |
| Среднее | 1,5–2,5 |
| Малое | 0–2 |
| Цветовая гамма, коэффициент | Черно-белый | 0–1,5 |
| Несколько цветов | 1,4–2,5 |
| Любые цвета | 2,4–4 |
| Звуковое сопровождение, коэффициент | Наличие | 0,85–2 |
| Отсутствие | 0–0,9 |
| Наличие подсказок, коэффициент | Наличие | 0,85–2 |
| Отсутствие | 0–0,9 |
| Наличие командной строки, коэффициент | Наличие | 0,85–2 |
| Отсутствие | 0–0,9 |

*Таблица 3*. Нечеткие продукционные правила

*Table 3.* Fuzzy production rules

|  |  |  |  |
| --- | --- | --- | --- |
| Если | Условие | то | Терм-множество |

|  |  |  |  |
| --- | --- | --- | --- |
| Если | Д = Есть | то | Черно-белый |
| Если | Д = Нет И СО = Низкий | то | Любые цвета |
| Если | Д = Нет И СО = Средний | то | Несколько цветов |
| Если | Д = Нет И СО = Высокий | то | Несколько цветов |

Входная лингвистическая переменная Цветовая гамма

Примечание: Д — дальтонизм; СО — системный опыт

Эксперт занимается настройкой экспертной систе- мы. В настройку входят: формирование структуры; для каждой входной переменной подбор оценочного мате- риала; создание базы правил и тестирование созданной экспертной системы с возможностью подробной трас- сировки расчетов.

Пользователь после авторизации может оценить свои характеристики с помощью оценочного материала, назначенного экспертом, и приступить к формированию личного адаптивного интерфейса прикладной програм- мы, который предоставляется для работы в САПР.

В проектной части ПИС можно выделить три этапа. На первом — производится оценка характеристик поль- зователя. На втором — независимо от выбора условия формирования прототипа (создать новый или заменить старый прототип), происходит подбор компонентов интерфейса. На третьем — на основе сформированного набора компонентов создается прототип адаптивного интерфейса прикладной программы. Каждый из этих

этапов непосредственно взаимодействует с базой дан- ных программной системы.

## Практическая реализация программной инструментальной системы

Перед тем как пользователь будет подбирать компо- ненты интерфейса, эксперт создает структуру эксперт- ной системы (рис. 5). Выбрав пункт меню «Эксперт», задаются лингвистические переменные и термы [20].

Далее эксперт может приступить к формированию базы правил (рис. 6).

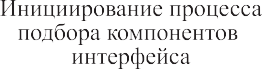
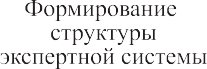
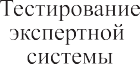
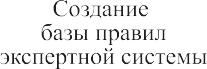
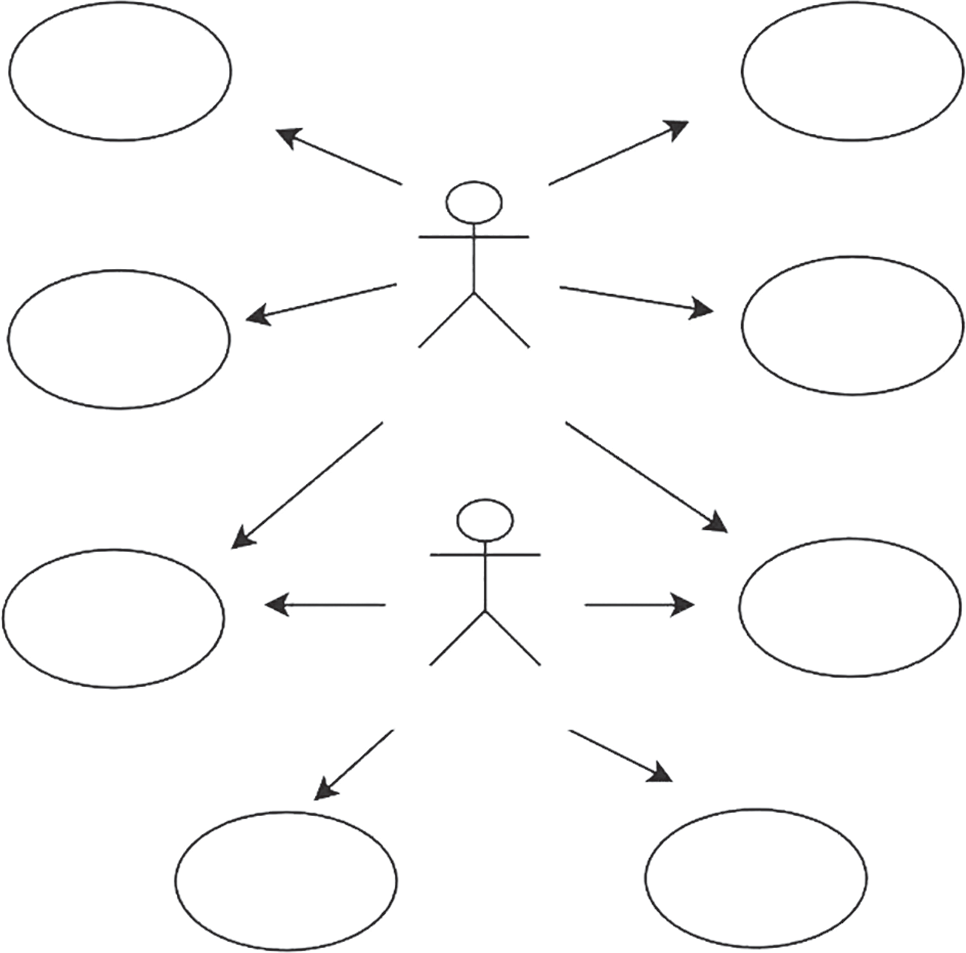
После формирования структуры и создания базы правил проводится тестирование экспертной системы (рис. 7). Эксперт, выбрав пункт меню «Подбор компо- нентов», может установить входные значения. Нажав на кнопку «Рассчитать», ПИС выведет результат подбора компонентов интерфейса в качественных и количе- ственных значениях. Подробные расчеты, выполняе-

*Рис. 4*. Диаграмма вариантов использования

*Fig. 4*. Diagram of use cases

мые экспертной системой, можно просмотреть, нажав на соответствующую кнопку и сохранить в файл (по желанию).

При входе в ПИС пользователю требуется оце- нить свои характеристики с помощью различных видов диагностики. Например, для оценки качества

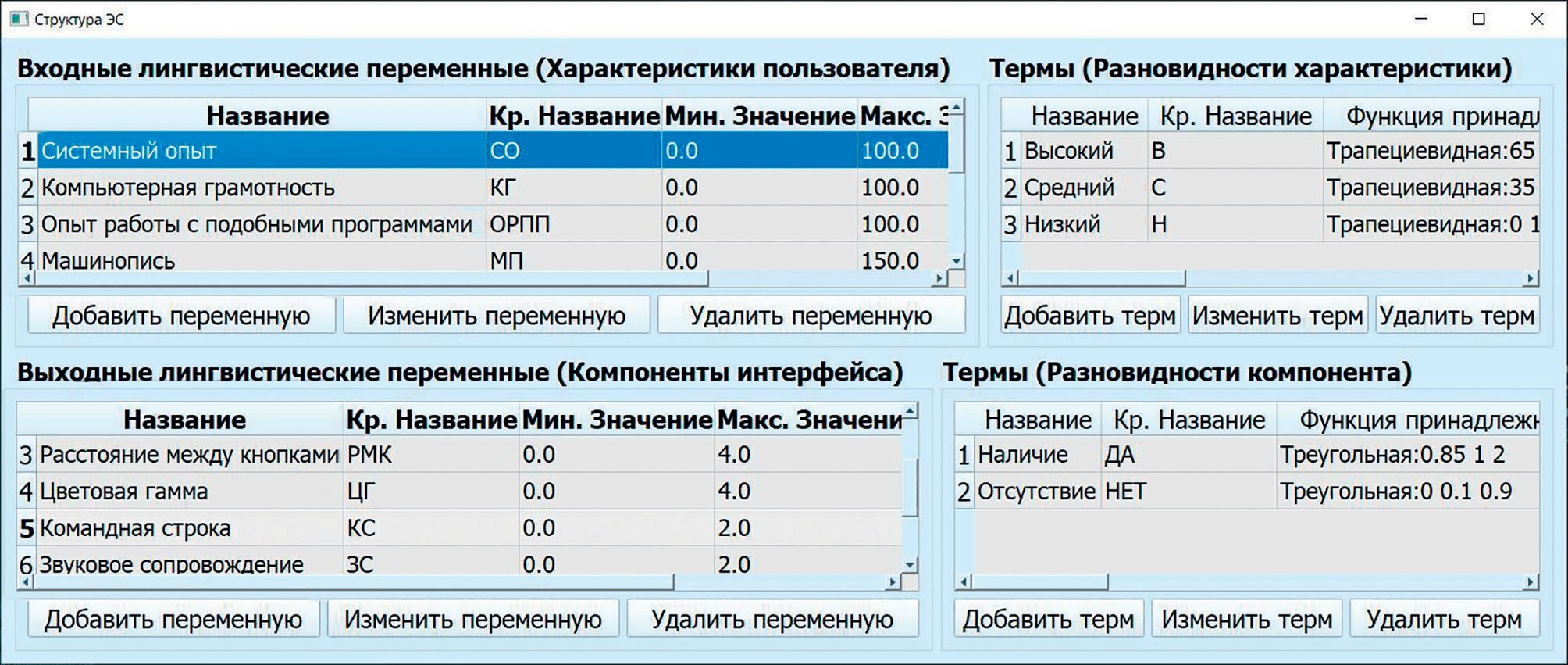


«Компьютерная грамотность» пользователю предлага- ется пройти тестирование.

После оценки характеристик пользователь может перейти к подбору компонентов интерфейса и созда- нию адаптированного прототипа, выбрав пункт меню

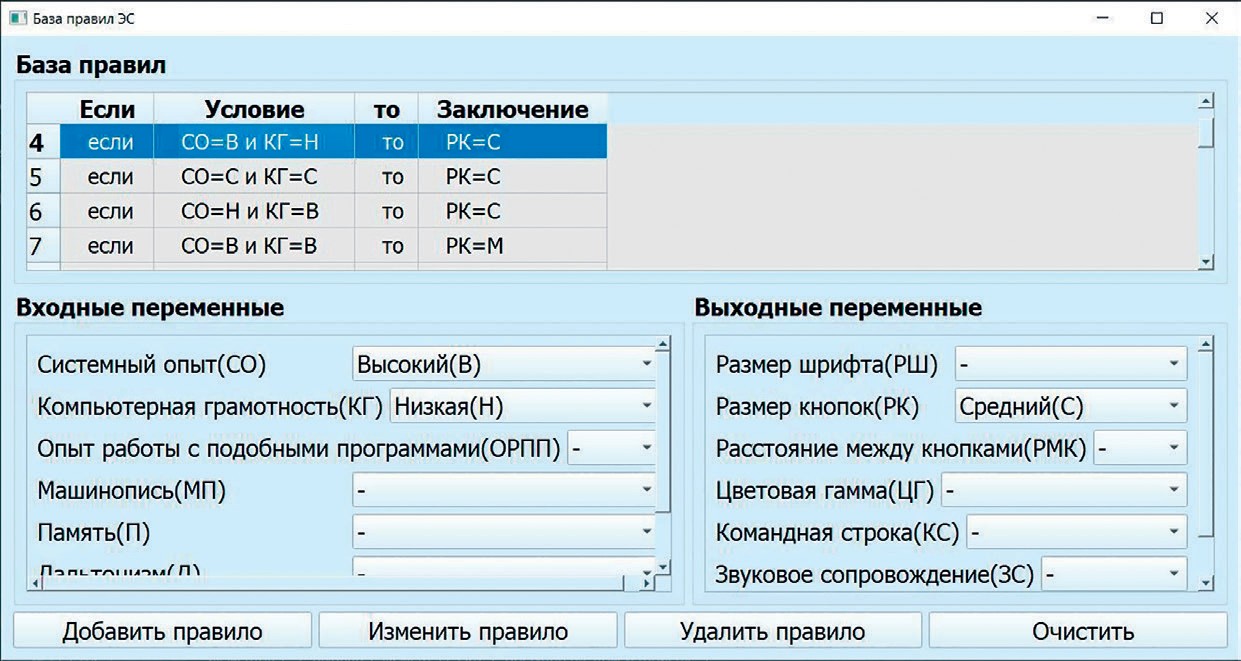
«Подбор компонентов интерфейса». Для этого тре- буется выбрать режим создания прототипа и увидеть созданный адаптированный прототип интерфейса при- кладной программы (рис. 8).

В приведеном примере (пользователь неопытный) получен прототип, который состоит из следующих ком- понентов: размер шрифта — средний; размер кнопок — средний; расстояние между кнопками — большое; цветовая гамма — несколько цветов; наличие подска- зок — есть; наличие звукового сопровождения — есть, наличие командной строки — нет. Проведенное юза- билити тестирование подтвердило адаптированность интерфейсной части программного обеспечения для определенной аудитории пользователей.



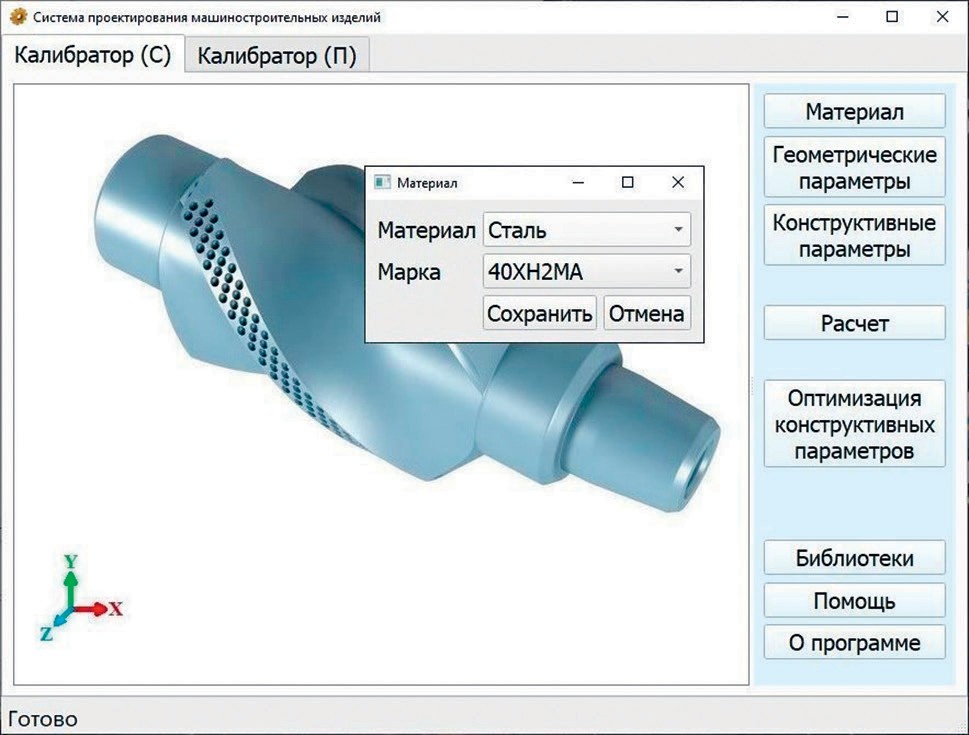
*Рис. 5*. Структура экспертной системы

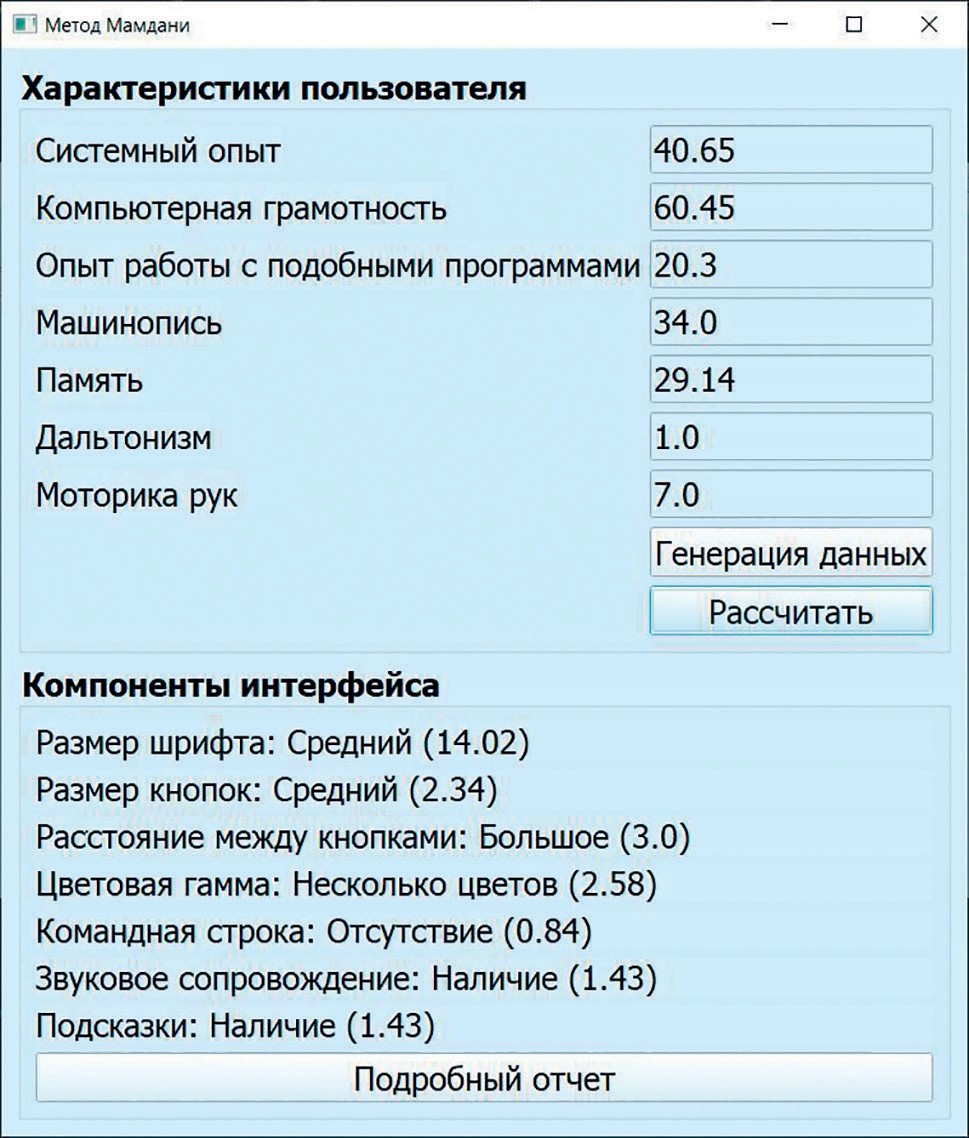
*Fig. 5*. Structure of the expert system



*Рис. 6*. База правил экспертной системы

*Fig. 6*. The base of the rules of the expert system



*Рис. 7*. Тестирование экспертной системы

*Fig. 7*. Expert system testing

## Обсуждение результатов

Разработана ПИС, которая учитывает характе- ристики и возможности пользователя. Для подбора компонентов интерфейса под определенного пользо- вателя использован метод экспертного оценивания. Сформирована база правил на основе продукционной модели знаний. Выполнена оценка результатов полу- ченных характеристик пользователя и правил эксперт- ной системы. В результате создан набор компонентов интерфейса и сгенерирован прототип интерфейса, кото- рый соответствует конкретному пользователю.

Так как характеристики пользователя имеют раз- личные степени неопределенности, неоднозначности,

*Рис. 8*. Фрагмент адаптивного интерфейса

*Fig. 8*. Fragment of the adaptive interface

внутреннюю противоречивость и др., и являются труд- но формализованными и специфическими, то целе- сообразно использовать интеллектуальные системы, базирующиеся на нечеткой логике и нечетких множе- ствах. Наиболее приемлемым в данном случае является метод Мамдани.

## Заключение

Внедрение разработанной экспертной системы по- зволит просто подобрать набор элементов интерфейса под каждого инженера-конструктора и сформировать адаптивный прототип интерфейса прикладной про- граммы. Данный результат улучшит автоматизиро- ванное рабочее место специалиста, а взаимодействие человека и компьютера станет более комфортным и эргономичным.

**Проектирование быстрой программной реализации специализированной нейросетевой архитектуры**

**с разреженными связями**

Постоянно растущий объем данных требует создания алгоритмов для их обработки. По этой причине в ряде областей задачи машин- ного обучения и интеллектуального анализа данных становятся все более распространен- ными и актуальными. Важное место в них за- нимает извлечение признаков, поскольку ис- пользование слабо релевантных признаков резко снижает эффективность любых алгорит- мов. И если еще 10–20 лет назад был распро- странен ручной подбор признаков [1], то сей- час наиболее перспективными являются нейросетевые алгоритмы, осуществляющие ав- томатический подбор признаков в процессе ре- шения задачи. Это позволяет сократить ручной объем работы для исследователей, а в ряде слу- чаев и повысить качество работы [2]. Однако глубокие нейронные сети сложны с вычисли-

тельной точки зрения, что затрудняет их при- менение в высоконагруженных системах с жесткими требованиями по времени ответа.

Кроме того, глубокие нейронные сети плохо подходят для решения задач онлайнового (ди- намического) обучения, поскольку они ста- тичны и не могут обучаться в режиме реаль- ного времени. По этой причине в подобных за- дачах легкие модели (например, логистическая регрессия) зачастую выигрывают у тяжеловес- ных, поскольку позволяют обновлять пара- метры в режиме реального времени. Однако столь простые модели требуют ручного под- бора признаков, что отнимает много времени у исследователей, и порой необходимо привле- чение к работе экспертов заданной предметной области. В работе [3] авторами была предло- жена нейронная сеть, которая осуществляет ав-

томатизированный подбор признаков, поддер- живая при этом обновление параметров в ре- жиме реального времени. Данная статья посвя- щена вопросам программной реализации этой архитектуры.

# Описание архитектуры

Рассматриваемая модель представляет со- бой нейронную сеть с одним скрытым слоем. Отличие от традиционного персептрона заклю- чается в разреженности архитектуры: нейроны скрытого слоя связаны только с частью нейро- нов входного слоя. Выходной слой нейронной сети состоит из одного нейрона с сигмоидаль- ной функцией активации, которая выдает веро- ятность для заданного набора входных пара- метров.

На вход модель принимает заданное коли- чество признаков. Сами признаки категориаль- ные, и для их подачи на вход применяется уни- тарное кодирование [4]. Каждому признаку от- водится одинаковый диапазон значений во входном векторе. Схема архитектуры пред- ставлена на рисунке 1.

Специфичность данной архитектуры заклю- чается в том, что она имеет разреженные связи с определенной структурой. И это дает ей ряд преимуществ.

# Особенности разреженной нейросетевой архитектуры

Исследования работы головного мозга по- казывают, что биологические нейроны коди- руют информацию в разреженном и распреде- ленном видах. Согласно оценкам, процент ак- тивных нейронов в один и тот же момент времени колеблется от 1 до 4. Это соответ- ствует балансу между разнообразием возмож- ных представлений и небольшим потребле- нием энергии. Традиционные сети прямого распространения без использования *L*1 регуля- ризации не обладают таким свойством. Напри- мер, при использовании сигмоидальной функ- ции активации нейроны после начальной ини- циализации имеют устойчивое состояние на середине между режимами насыщения. Это выглядит неестественно с биологической точки зрения и вредит оптимизации на основе градиентного спуска.

Разреженные представления имеют не- сколько преимуществ [5]. В контексте рассмат- риваемой задачи стоит выделить эффективное представление данных переменного размера. Различные входы могут содержать разное ко- личество информации, и потому их более удобно представлять в виде структур с пере- менным размером. Стоит отметить, что слиш-



**relu12**

feature\_1

**relu1N**

Нейроны

скрытого слоя, соединенные с нейронами, отвечающими за 2 входных признака

feature\_2

**relu2N**

σ

pr

**relu1**

feature\_N

**relu2**

Нейроны

скрытого слоя, соединенные с нейронами, отвечающими за 1 входной признак

**reluN**

*Рис. 1. Схема предложенной нейросетевой архитектуры*

*Fig. 1. The proposed neural network architecture*

ком высокая степень разреженности может приводить к деградации модели, поскольку она сокращает ее емкость. Однако на сегодняшний день известно, что глубокие нейронные сети часто содержат избыточное число параметров (что приводит к усложнению вычислений и ро- сту потребления ресурсов), поэтому их можно значительно упростить без существенной по- тери качества [6]. Помимо вычислительной из- быточности, большое количество параметров зачастую ухудшают обобщающую способ- ность моделей, делая их более подверженными переобучению.

Следовательно, можно удалить много пара- метров нейронной сети без существенного ухудшения (а порой и с улучшением) произво- дительности. Разумеется, такие изменения при- водят к возникновению разреженных архитек- тур. Помимо пониженных требований к памяти (нужно хранить меньшее число параметров), сокращение числа параметров позволяет упро- стить итоговые вычисления и уменьшить время предсказания, что играет роль в высоко- нагруженных системах или в системах, работа- ющих со слабым аппаратным обеспечением.

Есть различные стратегии сокращения па- раметров нейронных сетей. Стратегии усече- ния весов были предложены еще Лекуном в ра- боте [7], и они же остаются наиболее популяр- ными до настоящего времени. Относительно недавно в работе [8] был предложен алгоритм сокращения количества связей, основанный на похожести нейронов. Стратегия прореживания нейронной сети может быть также встроена в обучение модели. Еще один подход заключа- ется в обучении маленькой модели, которая по своему поведению будет имитировать боль- шую модель [9]. Кроме того, в некоторых рабо- тах предлагается обучать глубокие модели, но с меньшим числом параметров. Оставшиеся параметры при этом должны предсказываться на основании уже обученных.

В предложенной нейросети разреженность уже заложена в саму архитектуру с учетом ре- шаемой задачи (предсказание поведения поль- зователей в Интернете). Это осложняет ее реа- лизацию на базе таких нейросетевых фрейм- ворков, как Pytorch, Tensorflow, Theano и т.д. Разреженность связей можно реализовать при помощи обнуления некоторых весовых коэф- фициентов или путем использования модулей Sparse. Однако полученная реализация получа- ется в несколько раз медленнее, чем простая логистическая регрессия, что осложняет пере- ход на нее в высоконагруженных системах. По

этой причине для применения данной архитек- туры в задаче предсказания поведения пользо- вателей в Интернете было принято решение разработать самостоятельную программную реализацию, оптимизированную непосред- ственно под данную архитектуру. Далее рассматривается концепция программной реа- лизации и описываются разработанные алго- ритмы, позволяющие обеспечить быстродей- ствие предложенной нейронной сети.

# Диаграмма классов системы

Диаграмма основных классов реализован- ной библиотеки представлена на рисунке 2. Класс feature\_t используется для описания под- держиваемых признаков в модели, level\_t – для описания уровня модели, который может со- стоять из нескольких признаков (в таком слу- чае применяется класс hashing trick [10]), а levels\_holder\_t описывает конфигурацию мо- дели целиком (модель состоит из набора уров- ней). Помимо конфигурации, у модели есть набор настроек (к примеру, количество элемен- тов, отводимых на представление каждого признака), который описывается классом lr\_settings\_t. В результате класс самой нейросе- тевой модели наследует от класса настроек и класс конфигурации модели. Сама нейросете- вая модель описывается двумя классами: класс neural\_net\_t позволяет только использовать мо- дель (загрузив ее предварительно из дампа), а класс neural\_net\_fitter\_t – обучать. Такое разде- ление дает возможность программным компо- нентам, которые только используют обучен- ную модель, не хранить множество дополни- тельных параметров, связанных с настройками обучения. Из рисунка 2 видно, что класс lr\_settings\_t оказывается включенным дважды. Для разрешения этой ситуации используется виртуальное наследование. Также с классом neural\_net\_fitter\_t жестко связан объект опти- мизатора (композиция), то есть он создается в конструкторе класса neural\_net\_fitter\_t. Разде- ление оптимизаторов и самих моделей дает по- тенциальную возможность использовать одни и те же оптимизаторы для обучения разных мо- делей.

# Структуры данных и алгоритмы создания нейросети

Для реализации выбран язык программиро- вания C++, так как он включает в себя множе- ство современных инструментов программиро-

**Logger**

-m\_filename: string

-m\_log\_level: int

+Logger()

+log(log\_level: int, msg: string)

+setLogLevel(log\_level: int)

**lr\_settings\_t**

**levels\_holder\_t**

-m\_levels: vector<level\_t>

-m\_is\_valid: bool

#logger: Logger\*

+reset(levels: vector<vector<feature\_t>>)

+contains(feature: EFeature): bool

#unique\_key(event: vector<vector<int>>): vector<int>

-reset\_meta\_level()

**level\_t**

-m\_features: vector<feature\_t>

UNREGISTERED UNREGISTERED UNREGISTERED UNREGISTERED UNREGISTERED

-m\_typed: bool

+level\_t()

+reset(features: feature\_t, typed: bool): bool

+features(): vector<features>

+for\_each\_key(e: vector<EFeature>, level\_num: int)

+feature\_from\_event(e: vector<EFeature>): feature\_t

#m\_I: int

#m\_level\_I: vector<int>

+lr\_settings\_t(num\_levels: int)

+N(): int

**feature\_t**

-m\_feature: EFeature

+feature\_t(f: EFeature)

+feature(): EFeature

+create\_by\_name(name: string): feature\_t

**lr\_fitter\_settings\_t**

#m\_a: double #m\_lambda: double #m\_epoch\_count: int #...

+lr\_fitter\_settings\_t(num\_levels: int)

+update(rhs: lr\_fitter\_settings\_t)

**neural\_net\_t**

#m\_num\_hidden: int

UNREGISTERED UNREGISTERED UNREGISTERED UNREGISTERED UNREGISTERED

#m\_hidden\_weights: vector<vector<float>> #m\_output\_weights: vector<float> #m\_structure: vector<vector<int>>

+predict(e: vector<int>): float

+load()

+size()

«enumeration»

**gd\_algorithm\_t**

Vanilla Adam RmsProp

...

«enumeration»

**EFeature**

eAge eGeo eBanner eContents

eUserTopics

...

**neural\_net\_fitter\_t**

-m\_processed\_files: int

-m\_gd\_optimizer: gd\_optimizer\_t

+m\_logloss: float

+neural\_net\_fitter\_t(f: vector<vector<feature\_t>>, name: string)

-N: int

-b1: float

-u: vector<float>

-v: vector<float>

**gd\_optimizer\_t**

+createNNStructure()

+update(e: vector<int>, y: int): float

+fit(): bool

+ready\_to\_fit(): bool

+gd\_optimizer\_t(N: int, gd\_algo: gd\_algorithm\_t, b1: float)

+step(g: double, lr: double, i: int): double

+addEventsCount()

-countCorrection(i: int)

*Рис. 2. Диаграмма классов реализованной библиотеки Fig. 2. The implemented library class diagram*

UNREGISTERED UNREGISTERED UNREGISTERED UNREGISTERED UNREGISTERED

вания, таких как лямбда-функции, move-семан- тика, умные указатели и т.д., сохраняя при этом хорошую производительность [11].

Архитектура нейросети задается статически и хранится в двумерном массиве. Входные дан- ные поступают в виде набора из *L* признаков, каждый из которых имеет *N* различных значе- ний. Таким образом, входное пространство значений разбито на *L* диапазонов, каждый из которых соответствует одному признаку. С учетом архитектуры сети количество нейро-

по номеру набора признаков *li* быстро получать затронутые номера нейронов скрытого слоя, была создана матрица соединений нейронной сети. В этой матрице каждому номеру признака на входном слое соответствует список затрону- тых номеров нейронов скрытого слоя. Для нейронной сети, где *L* = 5 (5 различных катего- риальных признаков), данная матрица выгля- дит так, как показано в таблице 1.

*Таблица 1*

**Пример матрицы соединений**

нов скрытого слоя равняется

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Признак** | **Нейрон** | | | | |
| **1** | **2** | **3** | **4** | **5** |
| 0 | 0 | 5 | 6 | 7 | 8 |
| 1 | 1 | 5 | 9 | 10 | 11 |
| 2 | 2 | 6 | 9 | 12 | 13 |
| 3 | 3 | 7 | 10 | 12 | 14 |
| 4 | 4 | 8 | 11 | 13 | 14 |

*L*  ( *L*  1)  *L* 

2

**для пяти признаков**

*Table 1*

 *L*  (*L*  1) . Поскольку сеть не является полно-



2

связной, хранить полную матрицу весов (с ну- левыми весами в позициях, где нет связей) расточительно, поэтому в разработанной реа- лизации хранятся только ненулевые коэффици- енты. Благодаря фиксированной архитектуре можно заранее рассчитать размер такой мат- рицы с ненулевыми коэффициентами. Чтобы

**The example of a matrix of coupling for 5 features**

Для формирования такой матрицы сначала заполняется вспомогательный двумерный мас- сив соединений для нейронов скрытого слоя (для каждого нейрона скрытого слоя содер- жится массив с номерами связанных с ним нейронов входного слоя). Количество строк равняется количеству нейронов скрытого слоя, а количество столбцов – числу связей каждого нейрона (для данной архитектуры оно равно 1 или 2). Для нейронной сети из пяти признаков такая вспомогательная структура имеет вид, представленный в таблице 2.

*Таблица 2*

**Вспомогательная матрица**

**для нейронной сети с пятью признаками**

*Table 2*

**A utility matrix for a neural network with 5 features**

Такая структура данных заполняется на ос- нове модульной арифметики. Приведем алго- ритм ее заполнения (рис. 3).

Сложность такого алгоритма составляет *O*(*n*2), где *n* – количество уровней в модели. Учитывая, что *n* не превышает нескольких де- сятков, это не является проблемой. Кроме того, создание архитектуры нейросети делается только один раз за всю работу сети, поэтому доля затраченного на это времени крайне не- значительна. Чтобы из такой вспомогательной матрицы получить основную матрицу связей, достаточно однократного прохода по циклу. Алгоритм 2 описывает формирование матрицы соединений нейронной сети (рис. 4).

Сложность данного алгоритма составляет *O*(*n*3), где *n* – количество уровней модели (по- скольку количество скрытых нейронов сети пропорционально квадрату числа уровней в модели). Однако и это не является проблемой по тем же причинам, что и в алгоритме 1. В итоге для каждого значения входного при- знака можно получить номера связанных с ним нейронов скрытого слоя. Поскольку каждый нейрон входного слоя связан ровно с *L* нейро- нами скрытого слоя, веса нейронов скрытого слоя можно хранить в матрице размерности *N**L*×*L*. В случае полносвязной сети аналогич- ного размера для хранения весов скрытого слоя потребовалась бы матрица размера

*L*  (*L*  1)

*N*  *L* 

|  |  |  |
| --- | --- | --- |
| **№ нейрона скрытого**  **слоя** | **№ нейрона 1 входного**  **слоя** | **№ нейрона 2 входного**  **слоя** |
| 0 | 0 | - |
| 1 | 1 | - |
| 2 | 2 | - |
| 3 | 3 | - |
| 4 | 4 | - |
| 5 | 0 | 1 |
| 6 | 0 | 2 |
| 7 | 0 | 3 |
| 8 | 0 | 4 |
| 9 | 1 | 2 |
| 10 | 1 | 3 |
| 11 | 1 | 4 |
| 12 | 2 | 3 |
| 13 | 2 | 4 |
| 14 | 3 | 4 |

, то есть в *L*/2 раз больше. При

2

*L* = 20 признакам получается экономия памяти в 10 раз.

Порядок создания нейронной сети с учетом вспомогательных объектов изображен на диа-

**Алгоритм 1**. Заполнение вспомогательной матрицы для нейронной сети **Исходные параметры**: количество входных уровней levels\_num **Результат**: вспомогательная матрица соединений

count = 0;

i = 0 **до тех пор, пока** i < levels\_num **выполнять**

добавить в матрицу строку номер count с текущим номером уровня i;

инкрементировать счетчик count; перейти к следующему уровню;

**конец**

i = 0 **до тех пор, пока** i < levels\_num **выполнять**

j = i + 1 **до тех пор, пока** j < levels\_num **выполнять**

добавить в матрицу строку номер count, содержащую 2 текущих номера уровня i и j; инкрементировать счетчик count;

перейти к следующему уровню внутреннего обхода;

**конец**

перейти к следующему уровню внешнего обхода;

**конец**

*Рис. 3. Алгоритм 1*

*Fig. 3. Algorithm 1*

**gd\_optimizer\_t**

|  |  |
| --- | --- |
| **Алгоритм 2**. Заполнение основной матрицы соединений нейронной сети  **Исходные параметры**: Вспомогательная матрица соединений connections, Количество нейронов скрытого слоя num\_hidden  **Результат**: основная матрица связей  i = 0 **до тех пор, пока** i < num\_hidden **выполнять**  j = 0 **до тех пор, пока** j < числа нейронов входного слоя, связанных с i-м нейроном скрытого слоя (размер connections[i]) **выполнять**  получить номер нейрона входного слоя level = connections[i][j];  к строке level (соответствует связям набора входных признаков с номером  level) матрицы связей добавить номер нейрона скрытого слоя i;  перейти к следующему нейрону входного слоя, с которым связан i-й нейрон скрытого слоя;  **конец**  перейти к следующему нейрону скрытого слоя;  **конец**  *Рис. 4. Алгоритм 2*  *Fig. 4. Algorithm 2* | |
| грамме последовательности (рис. 5), на кото- рой видно, что при создании объекта нейросети сначала инициализируется объект настроек, за- тем создается конфигурация уровней, после чего инициализируются параметры обучения модели. Затем по вышеописанному алгоритму формируется нейронная сеть с заданной струк- турой, после чего создается объект оптимиза- тора. При создании объекта оптимизатора ис-  пользуются параметры обучения модели. | **Последовательность обработки примера нейросетью**  Обработка примеров нейронной сетью про- изводится в следующей последовательности:   * расчет номеров затронутых индексов (в силу разреженности данных для одного при- мера затрагиваются лишь несколько индексов); * получение номера уровня (это делается по номеру индекса, так как на каждый признак |



**neural\_net\_t**

**lr\_settings\_t levels\_holder\_t**

**lr\_fitter\_settings\_t**

1 : Инициализировать 2 : Настройки модели

3 : Задать уровни согласно

4 : Конфигурация уровней модели

5 : Инициализировать

6 : Настройки процедуры обучения

7 : Задать структуру модели 8 : Получить размер уровня

9 : Размер уровня

«create» 10 : Создать

**gd\_optimizer\_t**

11 : Инициализировать

12 : Запросить настройки обучения

13 : Настройки обучения

14 : Готовый оптимизатор

*Рис. 5. Диаграмма последовательности создания объекта нейронной сети*

*Fig. 5. The sequence diagram of neural network object creation*

отводится одинаковое количество значений);

* получение номеров нейронов скрытого слоя, с которыми связан данный уровень (с ис- пользованием матрицы, изображенной в таб- лице 1);
* расчет взвешенной суммы для нейрона выходного слоя;
* переиспользование рассчитанного набо- ра индексов для предсказания и обновления.

Процесс суммирования нейронов скрытого слоя изображен на рисунке 6. Основная осо- бенность суммирования заключается в том, что оно производится не по нейронам скрытого слоя, а по весам. Сначала рассчитывается вклад во все нейроны скрытого слоя от первого при- знака, затем – от второго и т.д. Такой способ суммирования позволяет получить ускорение за счет локального расположения в памяти сла- гаемых (весов).

При предсказании сначала производится расчет затронутых индексов и значений нейро- нов скрытого слоя. Эти значения используются при получении предсказания. Во время обуче- ния сначала предсказывается вероятность для данного примера, затем рассчитывается функ- ция потерь, на основании которой вычисля- ются градиенты и обновляются коэффициенты. При этом переиспользуются индексы и значе- ния коэффициентов на скрытом слое, получен-



**F1**

**0 5 6 7 8**

**1 5 9 10 11**

**F2**

**0**

**6 7 8**

**F3**

**2 6 9 12 13**

**F4**

**3 7 10 12 14**

**Структура нейросети**

**4 8 11 13 14**

**F5**

**Входной слой**

**Матрица весов скрытого**

**слоя**

*Рис. 6. Организация суммирования для нейронов скрытого слоя Fig. 6. The summation process for hidden layer neurons*

**0**

**1**

**2**

**3**

**4**

**5**

**6**

**7**

**8**

**9**

**10**

**11**

**12**

**13**

**14**

**Скрытый слой**

ные при обучении модели. Последнее позво- ляет значительно сократить время, затрачивае- мое моделью на обработку одного примера.

Основные элементы процедуры обучения изображены на диаграмме активности (рис. 7). Отметим, что некоторые этапы расчета гради- ентов выполняются параллельно. После обуче- ния модели на каждом примере производится расчет метрики кросс-энтропия. В данной мет- рике суммарная ошибка на выборке складыва- ется из ошибок по отдельным примерам, по- этому ее удобно использовать в задачах онлай- нового обучения для контроля качества работы модели. Если по какой-либо причине не уда- лось получить предсказание для данного при- мера, то и обучение также не производится.

# Сравнение скорости обработки примеров с другими реализациями

После разработки вышеописанной архитек- туры были проведены измерения для сравне- ния времени ее работы с реализациями, сделан- ными на базе уже имеющихся библиотек. В сравнениях участвовали четыре реализации.

1. Собственная реализация логистической регрессии с хешированием составных призна- ков [10]. Использовались 15 комбинаций при- знаков.

|  |
| --- |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |

|  |
| --- |
| **F1** |
| **F2** |
| **F3** |
| **F4** |
| **F5** |



|  |  |
| --- | --- |
| **Получить предсказание модели**  Успешно  **Взять список выставленных индексов из predict**  **Рассчитать значение функции потерь**  **Выставить нулевое значение вероятности**  UNREGISTERED  **Обновить веса скрытого слоя**  **Обновить пороги скрытого слоя Обновить веса выходного слоя**  **Рассчитать значение метрик**  **Вернуть значение вероятности**  *Рис. 7. Диаграмма активности процедуры обучения модели по одному примеру Fig. 7. The activity diagram of model training using one sample* | |
| 1. Вышеописанная реализация нейронной сети с использованием простых признаков на входе. 2. Реализация предложенной архитектуры нейронной сети на базе библиотеки Lasagne с использованием разреженных матриц [12]. 3. Реализация предложенной архитектуры на базе Pytorch [13]. На момент проведения экс- периментов была доступна версия pytorch 0.4.1 (дата релиза – июль 2018 года) без поддержки автоматического расчета градиентов для разре- женных матриц. Поэтому реализация данной архитектуры была произведена на базе плот-   ных матриц. | В эксперименте производились измерения для разной размерности, отводимой на каждый признак. Количество коэффициентов, отводи- мых на признак, изменялось от 28 до 216. Всего использовалось 15 признаков. Таким образом, общий размер входного вектора изменялся от 4 тысяч до 1 млн признаков. Замерялось время получения предсказаний для 100 тысяч приме- ров и обновления модели на 100 тысяч при- меров (1 итерация). Замеры производились на сервере с процессором Intel Xeon CPU E5-2667  3.30 GHz и оперативной памятью Micron 36KSF2G72PZ-1 1333 MHz (0.8ns) объемом  256 Гб. |

**predict 1x2, 100K примеров update 1x2, 100K примеров**



100

90

80

70

60

**время, сек**

50

40

30

20

10

0

8 9 10 11 12 13 14 15 16

**Размерность признака, 2^x**

Реализация лог. регрессии x 10 Разработанная реализация x 10 Реализация на базе Lasagne

Реализация на базе Pytorch

100

90

80

70

60

**время, сек**

50

40

30

20

10

0

8 9 10 11 12 13 14 15 16

**Размерность признака, 2^x**

Реализация лог. регрессии x 10 Разработанная реализация x 10 Реализация на базе Lasagne

Реализация на базе Pytorch

*Рис. 8. Сравнение времени предсказания для 100 тысяч примеров*

*Fig. 8. Prediction time comparison for 100 thousands samples*

*Рис. 9. Сравнение времени обучения моделей для 100 тысяч примеров*

*Fig. 9. Training time comparison for 100 thousands samples*

Результаты измерений для предложенной архитектуры приведены на рисунках 8 и 9.

На графиках видно, что разработанная реа- лизация нейронной сети работает практически так же, как логистическая регрессия, и на поря- док быстрее реализации на базе Lasagne. Реа- лизация на базе Pytorch при увеличении раз- мера вектора начинает работать значительно медленнее, поскольку она основана на плот- ных, а не на разреженных матрицах. Разрабо- танная архитектура легко расширяема до под- держки не только пар, но и троек, четверок признаков и т.д. Однако в нейронной сети, в ко- торой каждый нейрон скрытого слоя связан с тремя нейронами на входном слое, время пред- сказания и обучения значительно возрастает.

# Заключение

В целом можно сделать вывод, что благо- даря описанным оптимизациям разработанная реализация позволяет получить ускорение при- мерно в 10 раз по сравнению с реализацией, написанной на уже готовых нейросетевых фреймворках. Таким образом, использование разработанной реализации целесообразно на практике, особенно в высоконагруженных си- стемах, работающих в режиме реального вре- мени, и только в том случае, когда каждый нейрон скрытого слоя связан с небольшим ко- личеством нейронов на входе. В противном случае предпочтительно использовать тради- ционные плотные матрицы.