Программное приложение Programma’s Edit редактирования текстов, интегрированное в программный комплекс PFuzClass нечеткой генеалогической классификации для текстологических задач

**Человек:** Авторы демонстрируют возможности и обсуждают перспективы нового программного приложения для редактирования текстов, имеющих общее происхождение и сохраняющих общую структуру. Это приложение разработано авторами с целью интеграции с программой нечеткой генеалогической классификации. Объектом этого исследования являются тексты средневековых произведений, имеющих длительную историю бытования и сохранившихся в большом количестве вариантов. Целью настоящего исследования является выявление среди вариантов текста не обнаруживаемых экспертами неявных связей с последующей экспертной оценкой их значимости. Суть предложенного подхода состоит в адаптации и развитии аппарата теории нечетких множеств, что позволяет с заданной степенью надежности выделять из сохранившихся вариантов текста близкородственные группы и устанавливать между ними генеалого-преемственные связи. Предлагаемая классификация опирается не только на количественные данные (число разночтений, коэффициенты близости текстов списков), но и на качественные параметры (весовые коэффициенты разночтений), учитывающие уровень их значимости для текстологической классификации, что ранее в формализованных подходах не принималось во внимание. Успешная реализация данного программного приложения предоставит в руки исследователю мощный и удобный инструмент для массового сопоставления и анализа средневековых текстов с помощью алгоритмизированных математических методов.

**Key words:** редактирование текстов, коллация рукописей, лемматизация, токенизация, нечеткая классификация, генеалогические связи, контаминация, типологизация, наибольшая общая подпоследовательность, весовые коэффициенты

=================================

**FastText\_KMeans\_Clean:** 2. Описание приложения Programma’s Edit. -е – единственное число, именительный и винительный падеж; -я – единственное число, родительный падеж и множественное число, именительный и винительный падежи; -ю – единственное число, дательный падеж и двойственной число, родительный и местный падежи и т.д. Приложение Programma’s Edit сделано таким образом, чтобы можно было вводить данные о каждом слове, относящие его к лемме, классу и к категориям слов. Таким образом, лемматизация позволит приложению определять случаи, когда в двух разных вариантах текста стоит одно и то же слово, но в разных грамматических формах, что делает его непохожим на исходное слово. Приложение должно быть в состоянии генерировать входные данные для PFuzClass на основе данных текста и дополнительных данных о взаимосвязи между словами узлов и насколько каждый вариант правилен для каждого узла (см. первую часть настоящей статьи).

**Key words part:** 0.4230769230769231

=================================

**FastText\_KMeans\_Raw/:** Так, среди 20 сохранившихся вариантов текста представлен один, у которого ввиду механической утраты листов отсутствует заглавие (Мазуринский список (РГАДА, ф. 196, оп.I, N 1700)). 2. Описание приложения Programma’s Edit. -е – единственное число, именительный и винительный падеж; -я – единственное число, родительный падеж и множественное число, именительный и винительный падежи; -ю – единственное число, дательный падеж и двойственной число, родительный и местный падежи и т.д. Приложение Programma’s Edit сделано таким образом, чтобы можно было вводить данные о каждом слове, относящие его к лемме, классу и к категориям слов. Таким образом, лемматизация позволит приложению определять случаи, когда в двух разных вариантах текста стоит одно и то же слово, но в разных грамматических формах, что делает его непохожим на исходное слово. Text Node Connection . Создание языка, текста и варианта текста в Programma’sEdit. Приложение должно быть в состоянии генерировать входные данные для PFuzClass на основе данных текста и дополнительных данных о взаимосвязи между словами узлов и насколько каждый вариант правилен для каждого узла (см. первую часть настоящей статьи).

**Key words part:** 0.4230769230769231

=================================

**FastText\_PageRank\_Clean/:** 2. Описание приложения Programma’s Edit. Структура данных приложения Programma’sEdit. Вариант текста (список). Узел, из которого состоит текст. Text Node Connection . Грамматический класс слова. Word Class Category . 3. Описание работы Programma’s Edit.

**Key words part:** 0.38461538461538464

=================================

**FastText\_PageRank\_Raw/:** 2. Описание приложения Programma’s Edit. Вариант текста (список). Узел, из которого состоит текст. Text Node Connection . Например, “предъсловие”. Грамматический класс слова. Word Class Category . 3. Описание работы Programma’s Edit.

**Key words part:** 0.38461538461538464

=================================

**Mixed\_ML\_TR/:** Говоря о парсиномии, отметим, что на схожих принципах строят свои модели и методы такие текстологи как Деес [18] (1973), Гриффит [19] (1970-е), Ваттель [20] (2004). Под разночтениями понимаются не только отдельные слова, но и сочетания слов, перестановки, даже пропуски или вставки целых глав. Таким образом, лемматизация позволит приложению определять случаи, когда в двух разных вариантах текста стоит одно и то же слово, но в разных грамматических формах, что делает его непохожим на исходное слово. Ниже приведена структура данных Programma’s Edit с кратким описанием основных функций. Вариант текста (список). Так, например, для варианта текста Соф-1285 разделителями, стоящими до слова, являются “[“, “(“. Приложение должно быть в состоянии генерировать входные данные для PFuzClass на основе данных текста и дополнительных данных о взаимосвязи между словами узлов и насколько каждый вариант правилен для каждого узла (см. первую часть настоящей статьи).

**Key words part:** 0.4230769230769231

=================================

**MultiLingual\_KMeans/:** Говоря о парсиномии, отметим, что на схожих принципах строят свои модели и методы такие текстологи как Деес [18] (1973), Гриффит [19] (1970-е), Ваттель [20] (2004). Под разночтениями понимаются не только отдельные слова, но и сочетания слов, перестановки, даже пропуски или вставки целых глав. Ниже приведена структура данных Programma’s Edit с кратким описанием основных функций. Вариант текста (список).

**Key words part:** 0.38461538461538464

=================================

**Multilingual\_PageRank/:** Позже этот метод был развит группой московских ученых во главе с Л.И. Бородкиным и Л.В. Миловым (1977) [7]. Понятно, что подобный исчерпывающий анализ текста мало осуществим. Так, среди 20 сохранившихся вариантов текста представлен один, у которого ввиду механической утраты листов отсутствует заглавие (Мазуринский список (РГАДА, ф. 196, оп.I, N 1700)). Во втором узле мы видим 5 вариантов разночтений. Санкт-Петербургской Духовной Академии, N 129; помечен как СПБДА-129) и Мазуринском списке (Маз). Например, слово “чадо” имеет только одно значение рода – среднее. Визуальный дизайн приложения нагляден, удобен для пользователя и удовлетворяет стандартам Windows. Их можно разбить на четыре группы.

**Key words part:** 0.38461538461538464

=================================

**RuBERT\_KMeans\_Without\_ST/:** Приложение Programma’s Edit является важным элементом разрабатываемого комплекса, призванным максимальным образом облегчить исследователю ввод и обработку исходных текстовых данных и подготовку их для дальнейшей классификации. Среди них второй – это пропуск слова (тип ошибки 5), а четвертый и пятый варианты являются лексическими изменениями (слово “предсловие” копиист заменяет на “православие”) и имеют максимальный вес (тип ошибки 1). Напомним, что под леммой понимается слово (точнее, словоформа), приведенное к нормальной, словарной форме. Напомним, что речь идет о нахождении текстовых инвариантов и связана с применением алгоритма поиска наибольшей общей подпоследовательности (см. также первую часть настоящей статьи).

**Key words part:** 0.5

=================================

**RuBERT\_KMeans\_With\_ST/:** В этой связи необходимо упомянуть пакет программ TuStep, разрабатываемый в Тюбингенском университете с 1960-х гг. под руководством В. Отта [13], и программа COLLATE (1989) П. Робинсона из Оксфордского университета [14]. Приложение Programma’s Edit является важным элементом разрабатываемого комплекса, призванным максимальным образом облегчить исследователю ввод и обработку исходных текстовых данных и подготовку их для дальнейшей классификации. Например, для словоформы “предъсловие” это будет:. Для ее решения в приложении Programma’s Edit предусмотрено применение одного из вариантов алгоритма поиска наибольшей общей подпоследовательности (Longest Common Subsequence) [25]. Редактирование текста в Programma’sEdit. Приложение должно быть в состоянии генерировать входные данные для PFuzClass на основе данных текста и дополнительных данных о взаимосвязи между словами узлов и насколько каждый вариант правилен для каждого узла (см. первую часть настоящей статьи).

**Key words part:** 0.6153846153846154

=================================

**RUBERT\_page\_rank\_Without\_ST/:** Объектом ее изучения являются тексты средневековых произведений. Этому варианту также ставится в соответствие тип 8. Необходим, поскольку у разных классов разные категории. Word Class Category . При запуске программы пользователю предлагается создать новый файл или открыть уже существующий.

**Key words part:** 0.38461538461538464

=================================

**RUBERT\_page\_rank\_With\_ST/:** Применяя этот метод, удается не только построить стемму, но и восстановить утраченные звенья. 1. Экспорт в программу нечеткой классификации. Этому варианту также ставится в соответствие тип 8. Word Class Category . Необходимость в нем возникает в том случае, когда пользователю нужно видеть больше текста из каждого варианта.

**Key words part:** 0.4615384615384616

=================================

**RUSBERT\_KMeans\_Without\_ST/:** Важнейшим особенностью приложения Programma’s Edit является его интегрированность с программой PFuzClass. Каждый ее элемент есть число, процент унаследованных ошибочных разночтений из одного списка в другой. Так, в первом узле располагаются три варианта названия произведения, встретившиеся у 20 его списков. Узел, из которого состоит текст.

**Key words part:** 0.38461538461538464

=================================

**RUSBERT\_KMeans\_With\_ST/:** Разрабатываемый авторами настоящей статьи программный комплекс PFuzClass предназначен для решения задач, связанных с нечеткой формализованной классификацией текстов, и опирается на иные подходы, чем вышеописанные. Важнейшим особенностью приложения Programma’s Edit является его интегрированность с программой PFuzClass. Если ошибочные варианты, соответствующие одному узлу, помечены одной цифрой, то для программы нечеткой классификации они неразличимы (например, два варианта “православие” и “православию”). Один Text Item может содержать несколько частей слова.

**Key words part:** 0.4615384615384616

=================================

**RUSBERT\_page\_rank\_Without\_ST/:** Каждый узел содержит несколько разночтений. Text Node Connection . Поле, содержащее один вариант слова. Необходим, поскольку у разных классов разные категории. Word Class Category .

**Key words part:** 0.3461538461538461

=================================

**RUSBERT\_page\_rank\_With\_ST/:** Каждый узел содержит несколько разночтений. Вариант текста (список). Часть слова в тексте. Части слова объединяются в слова. Необходим, поскольку у разных классов разные категории.

**Key words part:** 0.38461538461538464

=================================

**Simple\_PageRank/:** Идея этого подхода заключается в отыскании среди множества списков линейных структур (“характерных нулей”), формирующих отдельные ветви стеммы. Новый этап в области компьютерной текстологии начинается с конца 1980-х гг., с начала эры персональных компьютеров, и связан с появлением многофункциональных интерактивных программ, доведенных до стадии коммерческих продуктов. Таким образом, лемматизация позволит приложению определять случаи, когда в двух разных вариантах текста стоит одно и то же слово, но в разных грамматических формах, что делает его непохожим на исходное слово. Например, для той же словоформы “предъсловие” равноправным вариантом является “предсловье” и т.д. Эта задача особенно важно при импорте больших кусков текста, в том числе если уже имеется вариант текста. Такие разделители позволяют автоматически определять какие символы относятся к словам, а какие – нет (например, знаки препинания, скобки и т.д.). Приложение Programma’s Edit (сокращенно PEdit) разработано на языке C, являющимся одним из безусловных лидеров среди систем современного программирования, позволяющим создавать приложения для работы под управлением операционных систем Microsoft Windows и Linux и, в частности, обеспечивающим интеграцию с визуальной средой обработки Delphi, в которой реализована основная часть программного комплекса.

**Key words part:** 0.4230769230769231

=================================

**TextRank/:** Таким образом, лемматизация позволит приложению определять случаи, когда в двух разных вариантах текста стоит одно и то же слово, но в разных грамматических формах, что делает его непохожим на исходное слово. Например, для той же словоформы “предъсловие” равноправным вариантом является “предсловье” и т.д. Эта задача особенно важно при импорте больших кусков текста, в том числе если уже имеется вариант текста. Если у нескольких вариантов текста один вариант данного слова, то это слово не разделяется на отдельные Text Item. Так, например, для варианта текста Соф-1285 разделителями, стоящими до слова, являются “[“, “(“. Как видно на рис.5, каждая строка таблицы редактора соответствует своему варианту текста, а столбец – узлу разночтения. Приложение должно быть в состоянии генерировать входные данные для PFuzClass на основе данных текста и дополнительных данных о взаимосвязи между словами узлов и насколько каждый вариант правилен для каждого узла (см. первую часть настоящей статьи).

**Key words part:** 0.4230769230769231

=================================

**TF-IDF\_KMeans/:** Если ошибочные варианты, соответствующие одному узлу, помечены одной цифрой, то для программы нечеткой классификации они неразличимы (например, два варианта “православие” и “православию”). 2. Описание приложения Programma’s Edit. Вариант текста (список). Все Text Item, принадлежащие Text Node соответствуют всем вариантам, встретившимся на данном месте. Например, таковыми в древнерусском языке для существительных являются категории падежа, числа. Привязка значений категорий конкретному слову. Необходимость в нем возникает в том случае, когда пользователю нужно видеть больше текста из каждого варианта.

**Key words part:** 0.4615384615384616

=================================

**Текст:** Одновременно с коллацией текстологам приходилось решать следующую задачу – группирование вариантов текста (списков) и установление между ними генеалого-преемственных связей. Для решения этой задачи французским ученым Фроже был предложен метод групп (1968), основанный на применении математической теории множеств [6]. Позже этот метод был развит группой московских ученых во главе с Л.И. Бородкиным и Л.В. Миловым (1977) [7]. Применяя этот метод, удается не только построить стемму, но и восстановить утраченные звенья. Отметим, что для успешного применения метода групп необходимо выполнение довольно жестких требований к модели, в частности отсутствие контаминации (правки текста по нескольким спискам).. Другой подход к реконструкции генеалогического дерева был предложен Ж. Зарри (1973) [8], в основе которого лежит развитие идей французского текстолога Кантена [9]. Идея этого подхода заключается в отыскании среди множества списков линейных структур (“характерных нулей”), формирующих отдельные ветви стеммы. Разработанная на основе предложенного подхода программа Quentin/80 генерирует всевозможные варианты стеммы, учитывая при этом и ситуации контаминации [10].. Иной подход к классификации текстов с большой рукописной традицией связан с привлечением кластерных методов. Первооткрывателями этого направления можно считать американских текстологов Колвелла и Тьюна (1963) [11]. В его основе лежит преобразование т.н. матрицы близости, полученной в результате попарного сравнения текстов списков. В конце 1980-х гг. метод Колвелла-Тьюна был развит в работах советского текстолога А.А. Алексеева [12].. Новый этап в области компьютерной текстологии начинается с конца 1980-х гг., с начала эры персональных компьютеров, и связан с появлением многофункциональных интерактивных программ, доведенных до стадии коммерческих продуктов. В этой связи необходимо упомянуть пакет программ TuStep, разрабатываемый в Тюбингенском университете с 1960-х гг. под руководством В. Отта [13], и программа COLLATE (1989) П. Робинсона из Оксфордского университета [14]. Оба программных продукта охватывают весь “производственный цикл” от ввода текста в электронный формат до его критического издания.. Отметим, что для решения задачи классификации в новой версии программы COLLATE применяется кладистический анализ [15]. Основой его является принцип экономии (парсиномии), согласно которому в процессе эволюции переход из одного состояния в другое должно произойти за минимальное число шагов [16]. Для проведения подобной классификации текстологи могут использовать программу PAUP [17], разработанную специально для биологов. Говоря о парсиномии, отметим, что на схожих принципах строят свои модели и методы такие текстологи как Деес [18] (1973), Гриффит [19] (1970-е), Ваттель [20] (2004).. Разрабатываемый авторами настоящей статьи программный комплекс PFuzClass предназначен для решения задач, связанных с нечеткой формализованной классификацией текстов, и опирается на иные подходы, чем вышеописанные. Суть этого подхода заключается в адаптации и развитии аппарата теории нечетких множеств для задачи массового сопоставления текстов с целью выявления не обнаруживаемых экспертами неявных связей с последующей экспертной оценкой их значимости. Приложение Programma’s Edit является важным элементом разрабатываемого комплекса, призванным максимальным образом облегчить исследователю ввод и обработку исходных текстовых данных и подготовку их для дальнейшей классификации.. При конструировании Programma’s Edit авторы придерживаются следующих базовых принципов: 1) “пошаговость”, переход от “простого” к “сложному”; 2) модульность; 3) наглядность и универсальность; 4) интегрированность; 5) алгоритмичность.. . 1. Экспорт в программу нечеткой классификации. Важнейшим особенностью приложения Programma’s Edit является его интегрированность с программой PFuzClass. Здесь уместно сказать несколько слов о самой нечеткой классификации. Объектом ее изучения являются тексты средневековых произведений. Такие тексты в процессе своего исторического бытования подвергались многочисленным изменениям, как бессознательного, так и сознательного характера. Это приводило к появлению групп близкородственных вариантов текста (списков). Ситуацию осложняет тот факт, что копиисты при переписывании могли править текст по одному или нескольким спискам, принадлежавшим к разным группам (редакциям). В этом случае говорить об однозначной принадлежности списка той или иной редакции вряд ли приходится.. Все эти соображения, по мнению авторов, весьма естественным образом вписываются и учитываются в рамках теории нечетких множеств, появившейся и интенсивно развивающейся с середины 1960-х гг. [21] На основе идеи привлечения этой теории одним из авторов настоящей статьи была предложена модель и построен формализованный алгоритм [22]. Разработанная на основе этого алгоритма программа нечеткой генеалогической классификации позволяет с заданной степенью надежности выделять из сохранившихся вариантов текста близкородственные группы и устанавливать между ними генеалого-преемственные связи.. В своих вычислениях эта программа использует матрицу нечеткого отношения, построенную в результате попарного сличения списков. Эта матрица представляет собой таблицу, число строк и столбцов которой равно количеству списков. Каждый ее элемент есть число, процент унаследованных ошибочных разночтений из одного списка в другой. Считается, что чем больше это число, тем достовернее выглядит гипотеза о том, что первый текст хронологически предшествует второму. И, наоборот, чем ближе это число к нулю, тем более независимы тексты друг от друга (располагаются на разных ветвях стеммы).. Под разночтениями понимаются не только отдельные слова, но и сочетания слов, перестановки, даже пропуски или вставки целых глав. Понятно, что все они играют неодинаковую роль для классификации: одни разночтения должны учитываться с большим весом, другие – с меньшим. Для решения этой задачи нами (Баранкова Г.С., Шпирко С.В.) была разработана типологизация разночтений [23]. Все разночтения были разбиты на 14 типов и для каждого из них одним формализованным методом вычислен вес.. В качестве ошибочных разночтений мы понимаем как непосредственные ошибки, вызванные описками, пропусками, орфографией, так и сознательное редактирование текста стилистического, идеологического, лексического характера. Таким образом, предварительный этап классификации включает “ручную” работу текстолога. В идеале каждое разночтение должно быть расценено с позиций ошибочности/нормы и отнесено к своему типу. При этом не стоит упускать из вида тот факт, что понятие уклонения от нормы могло меняться в зависимости от времени, места создания рукописи. Например, в каждом скриптории могли быть свои понятия об орфографической норме правописания. Понятно, что подобный исчерпывающий анализ текста мало осуществим. Но следует помнить, что чем детальнее будет анализ, тем актуальнее и “рельефнее” получится дальнейшая классификация.. Возвращаясь обратно к Programma’s Edit, выясним какие элементы должны обязательно присутствовать в этом приложении. Это удобно продемонстировать на примере классификации текстов древнерусского произведения “Предсловие покаянию” (20 списков) [24]. Ниже приведен фрагмент Excel-таблицы, служащей входными данными для программы нечеткой классификации.. Рис. 1. Таблица разночтений списков “Предсловия покаянию” (фрагмент). . В столбце A указаны номера узлов разночтений (около 1000), в столбце B – номера разночтений (более 4500), в столбце C – их типы, а в столбце D – собственно разночтения. Каждый узел содержит несколько разночтений. Так, в первом узле располагаются три варианта названия произведения, встретившиеся у 20 его списков. Мы видим, что второй встретившийся вариант – усеченный (в нем нет начального словосочетания “предсловие покаянию”). Этому варианту присвоен восьмой тип ошибки (пропуск/изменение названия главы), имеющий высокий вес для дальнейшей классификации. Квадратные скобки третьего варианта означают пропуск, отсутствие разночтения (заглавия в данном случае) в этом узле. Так, среди 20 сохранившихся вариантов текста представлен один, у которого ввиду механической утраты листов отсутствует заглавие (Мазуринский список (РГАДА, ф. 196, оп.I, N 1700)). Этому варианту также ставится в соответствие тип 8.. Во втором узле мы видим 5 вариантов разночтений. Среди них второй – это пропуск слова (тип ошибки 5), а четвертый и пятый варианты являются лексическими изменениями (слово “предсловие” копиист заменяет на “православие”) и имеют максимальный вес (тип ошибки 1).. Каждый из последующих столбцов, начиная с E, соответствует своему списку (Соф-1262, Соф-1285, СПБДА-129 и т.д.). В этих столбцах проставляются коды ошибок: нормальные чтения для соответствующего списка помечены нулем, а ошибочные (уклонения от нормы) – любым натуральным числом, отличным от нуля. Если ошибочные варианты, соответствующие одному узлу, помечены одной цифрой, то для программы нечеткой классификации они неразличимы (например, два варианта “православие” и “православию”). Если по каким-либо причинам мы не хотим, чтобы разночтение участвовало в расчете, то оно помечается знаком тире. Такую ситуацию мы видим в Дубенском сборнике (РНБ, собр. Санкт-Петербургской Духовной Академии, N 129; помечен как СПБДА-129) и Мазуринском списке (Маз).. . 2. Описание приложения Programma’s Edit. Для построения классификации текстов, которые имеют общее происхождение, и которые сохраняют общую структуру, имеет смысл их хранить таким образом, чтобы связь этих текстов была наглядной. Programma’s Edit решает эту задачу, позволяя параллельно редактировать тексты, создавать новый вариант текста, исходя из уже имеющегося текста, и редактировать уже его, с возможностью сравнивать с другими вариантами текста.. На очереди стоит задача разработки новой программной функции и аналитических средств, призванных обеспечить автоматическую обработку и сравнение (сличение) вариантов текста. Эту задачу можно решать двумя способами: с привлечением заранее созданного словаря лемм или без него.. Напомним, что под леммой понимается слово (точнее, словоформа), приведенное к нормальной, словарной форме. Так, для класса существительных нормальными являются морфологические формы именительного падежа и единственного числа. В словаре лемм должны храниться все встречающиеся в тексте леммы с указанием вариантов ее окончаний и значений категорий. Например, для словоформы “предъсловие” это будет:. -е – единственное число, именительный и винительный падеж; -я – единственное число, родительный падеж и множественное число, именительный и винительный падежи; -ю – единственное число, дательный падеж и двойственной число, родительный и местный падежи и т.д.. Приложение Programma’s Edit сделано таким образом, чтобы можно было вводить данные о каждом слове, относящие его к лемме, классу и к категориям слов. Таким образом, лемматизация позволит приложению определять случаи, когда в двух разных вариантах текста стоит одно и то же слово, но в разных грамматических формах, что делает его непохожим на исходное слово.. Второй вариант сличения текстов предусмотрен в том случае, когда по каким-либо причинам нет возможности поддерживать словарь лемм. В этом случае задача заключается в том, чтобы в разумное время определить какие слова нового текста соответствуют каким словам старого текста. Например, для той же словоформы “предъсловие” равноправным вариантом является “предсловье” и т.д. Эта задача особенно важно при импорте больших кусков текста, в том числе если уже имеется вариант текста. Для ее решения в приложении Programma’s Edit предусмотрено применение одного из вариантов алгоритма поиска наибольшей общей подпоследовательности (Longest Common Subsequence) [25].. Покажем теперь, как с “программистской” точки зрения в приложении реализованы хранение и организация данных. Ниже приведена структура данных Programma’s Edit с кратким описанием основных функций.. Рис. 2. Структура данных приложения Programma’sEdit. . Language . Язык текста (например, древнерусский, старославянский и т.д.). Этот параметр нужен, чтобы для каждого элемента текста имелась информация о его языке. Таким образом, приложение сможет использовать данные даже других текстов для анализа, а также, если два исследователя работают над одним текстом, то есть возможность объединить их словари или тексты.. Text . Текст, то есть множество вариантов одного текста. Каждый текст имеет название (например, «Предсловие покаянию») и привязку к языку.. Text Version . Вариант текста (список). Также имеет название и привязан к тексту. У каждого варианта есть свойства разделения текста на отдельные узлы (“разделители”). Такие разделители позволяют автоматически определять какие символы относятся к словам, а какие – нет (например, знаки препинания, скобки и т.д.). Text Node . Узел, из которого состоит текст.. Text Node Connection . Содержит данные о порядке узлов.. Text Item . Поле, содержащее один вариант слова. Все Text Item, принадлежащие Text Node соответствуют всем вариантам, встретившимся на данном месте. Если у нескольких вариантов текста один вариант данного слова, то это слово не разделяется на отдельные Text Item. Например, из рис.1 следует, что второму узлу (Text Node) соответствуют пять разночтений (Text Item): “предсловие”, “[предсловие]”, “предъсловие”, “православие” и “православию”.. Text Item Version . Обеспечивает связь между Text Item и Text Version, отношение между которыми “многие-ко-многим”: каждому экземпляру Text Item соответствует несколько экземпляров Text Version и наоборот. Другими словами, каждое разночтение может встречаться в разных списках, а каждый список содержит несколько разночтений.. Word Part . Часть слова в тексте. Один Text Item может содержать несколько частей слова. Дело в том, что не всегда слово стоит слитно в тексте. Например, частица “ся” может стоять перед своим глаголом, как в списке Соф-1262 (РНБ, Софийское собрание, N 1262): “пияньства ся отстати”, “аще ся и каяти”. Возможны еще более трудные случаи, когда части одного слова отделены друг от друга другими словами: “ни ся можеть оукоренити”, “ся ныне проспить” (РНБ, Софийское собрание, N 1285, обозначено как Соф-1285).. Word . Слово в тексте. Части слова объединяются в слова.. Далее в списке данных следует блок, связанный с запланированной в приложении функцией лемматизации.. Word Lemma . Привязка конкретного слова к своей лемме. Например, слово “предъсловию” относится к лемме “предъсловие”.. Lemma . Словоформа в нормальной, словарной форме. Например, “предъсловие”.. Category . Соответствует отдельной грамматической категории для данного языка. Например, таковыми в древнерусском языке для существительных являются категории падежа, числа.. Category Value . Значение категории. Обычно у категории может быть несколько значений. Например, значениями категории числа в древнерусском являются единственное, множественное и двойственное.. Word Class . Грамматический класс слова. Необходим, поскольку у разных классов разные категории.. Word Class Category . Привязка категорий к классу слова.. Word Moving Value . Привязка значений категорий конкретному слову.. Lemma Fixed Value . Привязка к конкретной лемме неизменяющихся значений категорий. Например, слово “чадо” имеет только одно значение рода – среднее.. . 3. Описание работы Programma’s Edit. Приложение Programma’s Edit (сокращенно PEdit) разработано на языке C, являющимся одним из безусловных лидеров среди систем современного программирования, позволяющим создавать приложения для работы под управлением операционных систем Microsoft Windows и Linux и, в частности, обеспечивающим интеграцию с визуальной средой обработки Delphi, в которой реализована основная часть программного комплекса. Визуальный дизайн приложения нагляден, удобен для пользователя и удовлетворяет стандартам Windows.. При запуске программы пользователю предлагается создать новый файл или открыть уже существующий.. Рис. 3. Создание файла в Programma’sEdit. . В следующем окне предлагается добавить и редактировать языки и тексты (Language и Text). В этом же окне выводятся в виде таблицы все имеющиеся варианты текста (Text Version). Для каждого из вариантов текста предусмотрена настройка собственных символов-разделителей.. Рис. 4. Создание языка, текста и варианта текста в Programma’sEdit. . Как видно из рис. 4 такие разделители делятся на три группы: до слова (PreChrs), после слова (PostChrs) и остальные, которые могут быть либо до, либо после слова (SepChrs). Так, например, для варианта текста Соф-1285 разделителями, стоящими до слова, являются “[“, “(“. Напомним, что эти символы используются программой при импорте текста с целью разбиения его на узлы.. Следующий экран предназначен для вывода и редактирования структуры варианта текста.. . Рис. 5. Редактирование текста в Programma’sEdit. . . Как видно на рис.5, каждая строка таблицы редактора соответствует своему варианту текста, а столбец – узлу разночтения. Такой “горизонтальный” дизайн удобен при сравнении вариантов текста. Находясь на каком-либо узле (Node), пользователь видит соответствующие разночтения для всех вариантов текста. Например, в третьем узле в Соф-1285 стоит разночтение “[чстаго]”. С помощью кнопок, располагающихся в правой части экрана, пользователь получает возможность осуществлять стандартные операции над узлами: добавлять новые в конец (Append Node); вставлять новые перед существующим (Insert Node); удалять старые (Remove Node); менять местами (Add Permutation).. Наряду с горизонтальным в будущем запланирована разработка и вертикального редактора текстов. Необходимость в нем возникает в том случае, когда пользователю нужно видеть больше текста из каждого варианта. Эта функция будет выполнена в виде менеджера последовательного размещения (Flow Layout). Этот менеджер размещает добавляемые компоненты на экран строго по очереди, строка за строкой. Если какой-либо из элементов не умещается в текущей строке, то он переносится на следующую. При этом будет использоваться способ оперирования элементами интерфейса drag-and-drop, когда слова меняются местами путем захвата и перемещения отображаемого объекта интерфейса манипулятором “мыши”.. Опишем теперь планы на будущее, связанные с разработкой приложения Programma’s Edit. Их можно разбить на четыре группы. Во-первых, необходимо продолжать работу по доработке редактора текста: -добавлять, удалять, менять слова; -предусмотреть метки, чтобы можно было переходить на другое место текста; -вертикальный редактор.. Вторая задача состоит в разработке функции экспорта в программу нечеткой генеалогической классификации. Приложение должно быть в состоянии генерировать входные данные для PFuzClass на основе данных текста и дополнительных данных о взаимосвязи между словами узлов и насколько каждый вариант правилен для каждого узла (см. первую часть настоящей статьи).. Третья задача связана с алгоритмизацией приложения и является на наш взгляд наиболее сложной. Напомним, что речь идет о нахождении текстовых инвариантов и связана с применением алгоритма поиска наибольшей общей подпоследовательности (см. также первую часть настоящей статьи).. Наконец, с третьей тесно связана задача лемматизации, включающая разработку -редактора языков (возможность определять и менять категории, их значения и классы слов); -редактора словаря (добавление лемм и определение их значения категорий); -анализа текста (присвоение словам значения категорий и леммы).. Успешное решение поставленных задач позволит существенно продвинуться не только в направлении по разработке удобного, универсального инструментария по применению алгоритмических методов в текстологии, но и, например, создании качественного электронного корпуса исследуемых текстов, что откроет новые перспективы развития для междисциплинарного подхода в гуманитарной науке.. В заключение авторы приносит большую благодарность за внимание и ценные советы Л.И. Бородкину, чл.-корр., зав. кафедрой исторической информатики исторического факультета МГУ имени М.В. Ломоносова, и Г.С. Баранковой, ведущему научному сотруднику Института русского языка имени В. В. Виноградова..