e-ISSN 1607-3274 Радіоелектроніка, інформатика, управління. 2018. № 4
p-ISSN 2313-688X Radio Electronics, Computer Science, Control. 2018. № 4

УДК 004.4:004.89:006.72

ИССЛЕДОВАНИЕ МЕТОДОВ ИНТЕЛЛЕКТУАЛЬНОГО АНАЛИЗА БИБЛИОГРАФИЧЕСКИХ ОПИСАНИЙ И РАЗРАБОТКА ПРОГРАММНОЙ СИСТЕМЫ ДЛЯ АНАЛИЗА СПИСКА ЛИТЕРАТУРЫ

Мельников А. Ю. – канд. техн. наук, доцент, доцент кафедры интеллектуальных систем принятия решений, Донбасская государственная машиностроительная академия, Краматорск, Украина.

Комиссаров К. М. – студент, Донбасская государственная машиностроительная академия, Краматорск, Украина.

АННОТАЦИЯ

Актуальность. Рассмотрена задача анализа библиографического списка с целью автоматизации проверки правильности его составления. Объектом исследования являлся процесс формирования библиографических описаний на основе действующих стандартов. Предмет исследования – модели и методы интеллектуального анализа для проверки и исправлений библиографического описания. Цель работы – повышение эффективности научно-педагогической деятельности за счет применения интеллектуальной системы для проверки списка литературных источников в курсовых и дипломных работах студентов.

Метод. Проанализирован стандарт и перечислены элементы, составляющие библиографическое описание. Рассмотрены существующие инструменты для формирования и проверки списка литературных источников, выявлены их недостатки, обоснована необходимость создания программного средства для проверки библиографических описаний. Приведено исследование методов интеллектуального анализа: рассмотрены методы регулярных выражений, нечеткого поиска с применением нечетких регулярных выражений, конечного автомата и нейронной сети Хемминга. Создана ранее не существующая модель проверки библиографического описания на соответствие действующим стандартам, основанная на методах регулярных выражений, нечеткого поиска, конечного автомата и нейронной сети Хемминга. Представлена объектно-ориентированная модель проектируемой компьютерной системы на языке визуального моделирования UML. Описана работа компьютерной реализации программной системы, осуществленной в среде визуального программирования C#.

Результаты. Разработана программная система – интеллектуальная система принятия решений – для проверки библиографического описания и частично автоматизированного исправления с пользовательскими указаниями, сформированными на основе базы знаний.

Выводы. Разработанная интеллектуальная система позволяет выявить и исправить ошибки оформления библиографического описания. Основой программы является база знаний, которую можно дополнить при наличии каких-либо непредусмотренных ситуаций. Кроме этого, данная система дает возможность накапливать базу данных уже проверенными списками литературы. Разработанную систему можно улучшать без вмешательства в исходный код программы. При этом уровень ее возможностей и правильности исправлений, выдачи замечаний напрямую зависит от полноты и правильности составленной базы знаний. К перспективам дальнейших исследований можно отнести использование других стилей оформления списка литературы, отличных от ДСТУ 7.1:2006, таких, как IEEE.

КЛЮЧЕВЫЕ СЛОВА: библиографический список, библиографическое описание, ДСТУ 7.1:2006, ISBD, регулярные выражения, нечеткий поиск, расстояние редактирования, нечеткие регулярные выражения, конечный автомат, нейросеть Хемминга, Unified Modeling Language, C#, WorldAddIn.

ВВЕДЕНИЕ

Обязательной структурной частью любой научной работы является список использованной литературы, который служит источником информации об использованных (рассматриваемых, цитируемых, упоминаемых) в тексте работы документах. Библиографический список составляется из библиографических описаний, которые содержат совокупность сведений о документе, его части, или группе документов, приводимых в определенной последовательности и дающих возможность идентифицировать данный документ.

В нашей стране библиографическое описание необходимо составлять согласно действующему стандарту Украины – ДСТУ 7.1:2006 «Библиографическая запись. Библиографическое описание». Стандарт определяет достаточно много требований к оформлению библиографических описаний для различных видов источников, устанавливает разное количество обязательных элементов, в зависимости от вида источника, наличия сведений в предписанных источниках, специфики материала и т.д., является базовым для системы стандартов, правил, руководств, методических пособий по составлению библиографической записи [1–4]. Очевидно, что применение систем автоматизации и проверки составления библиографического списка позволит повысить эффективность научно-педагогической деятельности

Объектом исследования является процесс формирования библиографических описаний на основе действующих стандартов.

Предметом исследования являются модели и методы интеллектуального анализа для проверки и исправлений библиографического описания.

Цель повышение эффективности научно-педагогической деятельности за счет применения интеллектуальной системы для проверки списка литературных источников в курсовых и дипломных работах студентов.

DOI 10.15588/1607-3274-2018-4-12

1 ПОСТАНОВКА ПРОБЛЕМЫ

Дано описание структуры библиографического описания. На этой основе и результатах анализа часто допускаемых ошибок необходимо сформировать базу знаний для идентификации областей с целью исправления ошибок – приведения списка в соответствие со стандартом ДСТУ 7.1:2006.

База знаний должна быть самообучаемой, реализованной в виде программного приложения – интеллектуальной системы принятия решений.

2 ОБЗОР ЛИТЕРАТУРЫ

Библиографическое описание – совокупность сведений о документе, приведенных по определенным правилам, устанавливающим наполнение и порядок следования областей и элементов, предназначенных для идентификации и общей характеристики документа [1–4].

Согласно ДСТУ 7.1–2006, в состав библиографического описания входит ряд обязательных областей и элементов: область заглавия и сведений об ответственности, область издания, область специфических сведений, область выходных данных, Область физической характеристики, область серии, область примечания, область стандартного номера (или его альтернативы) и условий доступности. Для более четкого разделения областей и элементов, а также для различения предписанной и грамматической пунктуации применяют пробелы в один печатный знак до и после предписанного знака. Исключение составляют точка и запятая – пробелы оставляют только после них [3–4].

Элементы могут быть обязательными и факультативными. Обязательные элементы обеспечивают идентификацию документа и поиск. Факультативные элементы дают дополнительные сведения о документе (параллельное заглавие, сведения, относящиеся к заглавию, сведения о сопроводительном материале, условия доступности, цена и т.п.). Наряду со стандартизацией общей структуры библиографического описания учитывалась специфика документов разных видов. Для каждого из них выявлен и установлен необходимый минимум обязательных элементов для идентификации. Например, для книг – пять обязательных элементов, а для стандартов – два. При необходимости библиографическое описание документа любого вида по ДСТУ 7.1–2006 можно составить только из обязательных элементов [5-6]. Пунктуация в библиографическом описании выполняет две функции – обычных грамматических знаков препинания и знаков предписанных (условно разделяющих области и аналогичные элементы описания) для распознавания отдельных элементов в любом ISBD, на любом языке [6]. В качестве предписанной пунктуации выступают знаки препинания и математические знаки.

В стандарте ГОСТ ДСТУ 7.1-2006 определены три вида библиографического описания: одноуровневое, многоуровневое и аналитическое. Будем рассматривать только одноуровневое и аналитическое библиографические описания, так любое многоуровневое описание можно представить как несколько одноуровневых. На однотомное издание составляют одноуровневую запись, состоящую из одной части (одного уровня). Также одноуровневая запись может быть составлена на многотомное издание в целом, на отдельный том (часть, выпуск, номер и т. п.), на группу томов многотомного издания. Схема одноуровневого описания в наиболее полном варианте описана в стандарте [3].

В ходе анализа не удалось найти универсальных программных решений для проверки правильности оформления списка использованной литературы или библиографического описания как на соответствие стандарту ДСТУ 7.1:2006, так и на соответствие другим стандартам, в том числе международному ISBD. Наиболее распространены следующие программы и инструменты для формирования и проверки библиографического описания:

1. Встроенный инструмент для формирования библиографического списка в Microsoft Word, который появился с 2007 версии офисного пакета. Основными преимуществами данного инструмента являются простота в использовании, интуитивно понятный интерфейс, поддержка нескольких стандартов оформления библиографического описания [7]. Недостатком является несоответствие оформляемых описаний действующим стандартам, как ДСТУ 7.1:2006, так и ГОСТ 7.1:2003.

2. Интернет-ресурс для оформления списка литературных источников с названием vak.in.ua, который предоставляет возможность оформления научных источников в соответствии с требованиями Высшей аттестационной комиссии (ВАК) Украины [8]. Основными преимуществами данного ресурса являются простота, высокая степень соответствия Украинскому стандарту ДСТУ 7.1:2006, большое количество готовых примеров. Несмотря на высокую степень соответствия ДСТУ 7.1:2006, при составлении библиографического описания иногда обнаруживаются грубые ошибки. Например, при составлении аналитического библиографического описания в область сведений об идентифицирующем документе могут попадать как фамилия автора, так и издание. Кроме этого, к недостаткам можно отнести отсутствие возможности формирования списка литературы без подключения к интернету, а так же проверки уже сформированного библиографического описания на его соответствие стандартам.

Рассмотрим несколько методов интеллектуального анализа библиографических описаний.

Так как области описания не могут быть однозначно определены только по их расположению, их можно идентифицировать по формату (предписанным шаблонам) и по ключевым словам, характерными для данной области. Для таких задач используется механизм регулярных выражений.

Регулярные выражения – это формальный язык поиска и осуществления манипуляций со строками, основанный на использовании метасимволов,

DOI 10.15588/1607-3274-2018-4-12

литералов, операторов или конструкций. Механизм регулярных выражений позволяет захватывать целевую строку по заданному шаблону [9-10].

Регулярные выражения могут состоять из обычных слов, символьных классов, множества символов или слов, диапазонов символов. Примеры единичных символов:

[set] – задание множества символов (соответствие символа из заданного набора – «s», «e», «t»).

[^set] – задание отрицания множества символов (соответствие любого символа, который не входит в заданный набор).

Классы символов соответствуют одному набору символов и состоят из языковых элементов, приведенных в таблице символьных классов [9–10].

Для управления элементами регулярных выражений используются квантификаторы. Квантификатор указывает количество вхождений предшествующего элемента (знака, группы или класса знаков), которое должно присутсвовать во входной строке, чтобы было зафиксировано соответствие. Привязки (якоря) являются атомарными утверждениями нулевой ширины, добавляют условия для соответствия в зависимости от текущей позиции в строке, но не предписывают обработчику перемещаться по строке или обрабатывать символы. Элементы регулярных выражений можно группировать. Конструкции группирования отображают части выражений регулярных выражений и обычно захватывают части входной строки. Конструкции изменения позволяют объединять наборы для соответствий (регулярные выражения) в одно регулярное выражение, работают по принципу логического «или». Для выполнения операций поиска и замены с помощью регулярных выражений предусмотрен механизм подстановки. Подстановки – это языковые элементы регулярных выражений, которые поддерживаются в шаблонах замены. То есть группы выражений могут использоваться в качестве элементов подстановки при выполнении операции «поиска-замены».

Таким образом, с помощью регулярных выражений можно находить любые строковые конструкции в тексте, составленные по заранее известному шаблону. Однако они находят исключительно соответствия, которые строго соответствуют этому шаблону. В противном случае механизм регулярных выражений не захватывает целевую подстроку. Кроме того, составление регулярных выражений требует большой внимательности, на практике очень часто бывают ситуации, когда регулярные выражения захватывают строку, которая не предполагалась быть целевой.

Нечеткий поиск – это поиск информации, при котором выполняется сопоставление заданному образцу поиска или близкому к этому образцу значению. Алгоритмы нечеткого поиска используются в большинстве современных поисковых систем, например, для проверки орфографии. В случае нечеткого поиска для оценки сходства используются специальные метрики нечеткого поиска. Метрикой нечеткого поиска называют функцию расстояния между двумя словами, позволяющую оценить степень их сходства в данном контексте. В качестве метрик используют расстояния Хемминга, Левенштейна, Дамерау-Левенштейна [11].

Расстояние Хемминга – это число позиций, в которых соответствующие символы двух слов одинаковой длины различны. Согласно определению, расстояние Хемминга имеет один существенный недостаток – сравнивать можно только слова одинаковой длины. Из-за этого данную метрику практически не применяют на практике.

Расстояние Левенштейна или расстояние редактирования – это минимальное количество операций (вставки одного символа и замены одного символа на другой), необходимых для преобразования одной строки в другую. Расстояние Левенштейна применяют для исправления ошибок в словах, биоинформатике для сравнения хромосом и для полнотекстового поиска. Данная метрика имеет следующие недостатки: при перестановке слов в предложении расстояние принимает большое значение; значительно зависит от длины слова. Для расчета расстояния Левенштейна используют метод Вагнера-Фишера.

Расстояние Дамерау-Левенштейна – это мера разницы двух строк символов, определяемая как минимальное количество операций вставки, удаления, замены и перестановки соседних символов, необходимых для перевода одной строки в другую. Данная метрика отличается от расстояния Левенштейна только добавлением новой операции (перестановки).

Недостатком применения нечеткого поиска является более низкая производительность по сравнению с обычным. Кроме этого, при использовании нечетких регулярных выражений вероятность нахождения неожидаемых соответствий намного выше. Поэтому в нечетких регулярных выражениях очень важно устанавливать минимально возможную дистанцию редактирования для каждой группы.

Для нечеткого поиска могут быть использованы широко известные алгоритмы и коды Хемминга. Коды Хемминга давно и успешно применяются при кодировании и декодировании информации, позволяя успешно восстановить утерянную пре передачи информацию. Алгоритмы Хемминга могут быть применены в вопросах информационного поиска. Особенно эффективно коды Хемминга работают вместе с аппаратом нечеткой логики. Коды Хемминга – простейшие линейные коды с минимальным расстоянием 3, то есть способные исправить одну ошибку. Сети Хэмминга представляют собой одну из разновидностей нейронных сетей. Принцип работы сетей Хэмминга основан на определении расстояния Хэмминга между объектами и нахождении наиболее близкого [12].

DOI 10.15588/1607-3274-2018-4-12

Для обнаружения ошибок в сетях Хемминга используют коды обнаружения ошибок, для исправления – корректирующие коды. Для этого при записи и передачи информации в полезные данные добавляют специальным образом структурированную избыточную информацию, а при чтении (приеме) ее используют для того, чтобы обнаружить или исправить ошибки. Число ошибок, которое можно исправить, ограничено и зависит от конкретного применяемого кода. Таким образом, алгоритмы Хэмминга могут быть использованы для поиска слов, которые в исходном запросе набраны с ошибками.

Однако необходимо учитывать одну особенность сетей Хемминга. Если при написании была опечатка или даже две, то алгоритм работает хорошо, но если был пропущен символ или добавлен лишний, то Хеммингово расстояние может оказаться слишком большим. Для того чтобы устранить этот недостаток, мы будем подавать на вход как само искомое слово, так и это же слово, исключая поочередно один символ в каждой позиции и добавляя одну букву в каждую позицию. Такой подход позволит найти практически все случаи ошибок – опечатка, пропуск символа, лишний символ.

Следует отметить, что, несмотря на большую эффективность кодов Хемминга, они не лишены определенных недостатков. Линейные коды, как правило, хорошо справляются с редкими и большими ошибками и опечатками. Однако их эффективность при сравнительно частых, но небольших ошибках менее высока.

Недостатки применения метрики Хемминга и применения нейронной сети Хемминга: отсутствие возможности поиска слов разной длины; размер нейронной сети Хемминга и производительность поиска зависит от объема словаря, которому обучается нейросеть, и в котором ищется слово; метод применим только для заранее известного алфавита, в противном случае поиск приводит к непредсказуемому результату.

Для выполнения некоторых проверок – таких, как наличие обязательных областей и правильность следования областей – целесообразно рассмотреть систему, поведение которой зависит от текущего состояния, и вариантами перехода из текущего состояния в новое состояние. Примером такой системы является конечный автомат [13].

Конечный автомат является некой абстрактной моделью, которая описывает дискретную систему, поведение которой зависит от текущего состояния и функции перехода в новое состояние. С помощью конечного автомата обычно представляют некий алгоритм, который допускает или не допускает цепочки входных символов, принадлежащих входному алфавиту.

3 МАТЕРИАЛЫ И МЕТОДЫ

Для осуществления проверки библиографического описания необходимо определить каждую его область, после чего возможно будет осуществить ряд таких проверок, как правильность расположения элементов, правильность следования областей, соответствие данных этим областям. Для решения этих задач будут использоваться регулярные выражения, так как они являются мощным инструментом и показывают высокую эффективность в задачах, связанных с обработкой строк.

Как уже упоминалось ранее, с помощью регулярных выражений можно определить область при наличии вхождения искомого шаблона, если это не противоречит ее расположению. Тем не менее, может возникнуть проблема неоднозначности. Например, фамилия и инициалы могут присутствовать в различных областях, таким образом, они не могут быть критериями для идентификации. Поэтому при формировании правил для идентификации следует выделять наиболее характерные признаки областей по их содержимому.

Диаграмма выражения для идентификации области физической характеристики представлена на рис. 1.

Согласно ДСТУ, область выходных данных является обязательной областью. Отсутствие места издания или издательства не могут быть причиной отсутствия приводимых сведений, которые формируются следующим образом: «. – Место издания : Издательство, Год». Для идентификации области выходных данных используется выражение и диаграмма, представленные на рис. 2.

Для идентификации области сведений об издании, используются наиболее характерные для нее словосочетания: «Издание», «Версия», «Ред.» (рис. 3).

Для идентификации области примечания, в качестве шаблона используются наиболее часто встречаемые в ней словосочетания, такие как «Библиогр.:», «экз», «Рез.», а так же возможное указание страниц (рис. 4).

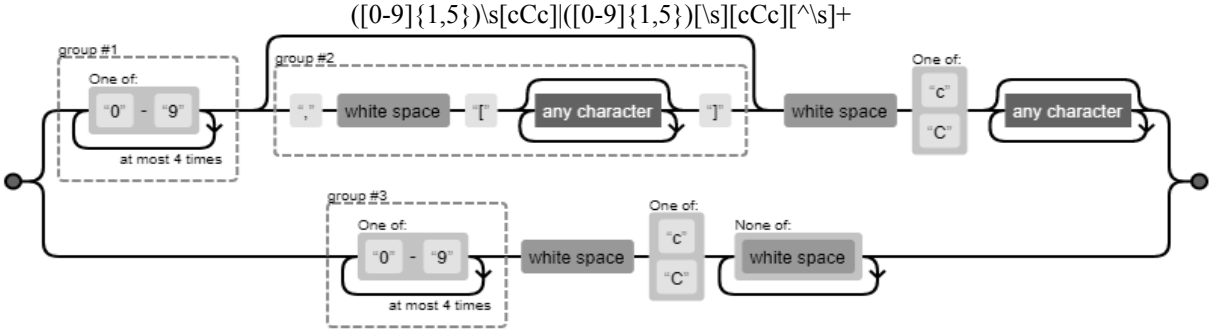

Рисунок 1 – Диаграмма выражения для идентификации области физической характеристики

DOI 10.15588/1607-3274-2018-4-12

[А-Яа-я][.][\s][:][^\n]+|[^\s]+[,][\s][1-2][0-9][0-9][0-9]|[А-Я][а-я]+[\s]:[\s][А-Я]*[а-я]*[\s]*,[\s][0-9]{1,4}

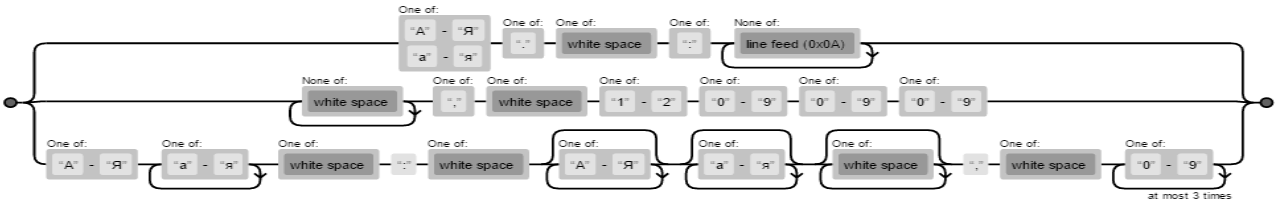

Рисунок 2 – Диаграмма выражения для идентификации области выходных данных

[^\n]+[Ии]зд[^\n]+|[Ии]зд[^\n]+|[^\n]+[Ии]зд|[^\n]+[Вв]ерс[^\n]+|[^\n]+[Вв]ариант|[^\n]ред.

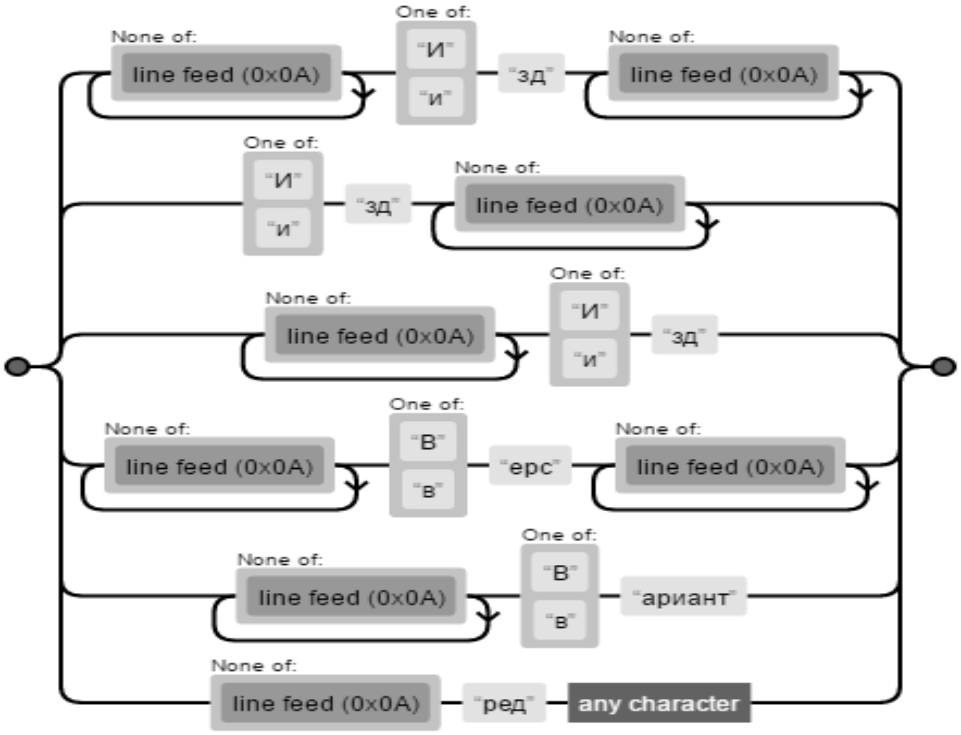

Рисунок 3 – Диаграмма выражения для идентификации области издания

[Бб]иблиогр.:[^\n]+|[0-9]+[\s]экз|Рез.:[^\n]+|[^\n]+с.[\s][0-9]{1,5}[–][0-9]{1,5}.

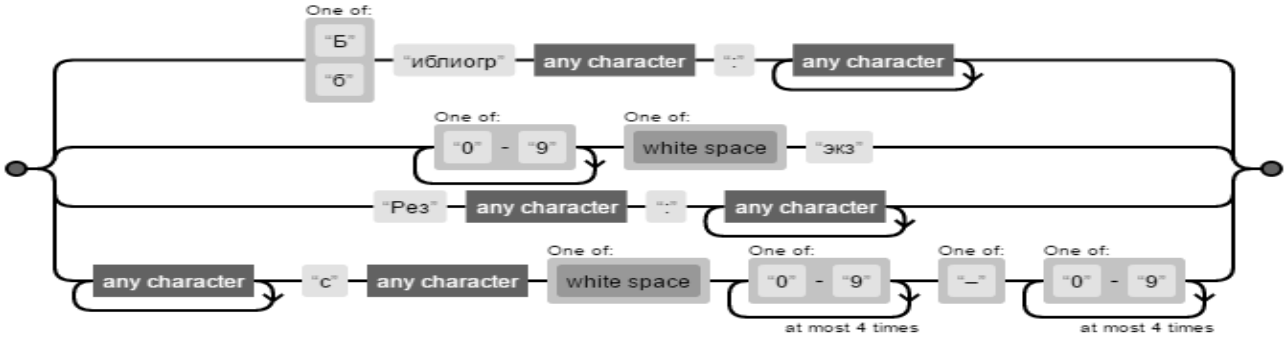

Рисунок 4 – Диаграмма выражения для идентификации области примечания

Характерным признаком для области серии являются скобки, исходя из того что согласно ДСТУ, содержание серии приводится в скобках (диаграммы выражений далее не приводятся):

^\((.*)\)$.

Вполне очевидно, что если область начинается с ISBN/ISSN или содержит словосочетания «Режим доступа», «URL», «http», то эта область является областью стандартного номера и условий доступности. Правило данной области выглядит следующим образом:

IS[SB]N[^\n]+|[^\n]+Режим доступа:[^\n]+|[^\n]+http[^\n]+|[^\n]+URL[^\n]+

DOI 10.15588/1607-3274-2018-4-12

Область заглавия и сведений об ответственностях не нуждаются в идентификации, так как она всегда является началом библиографического описания, и является обязательной областью.

После того, как идентифицирован определенный набор областей, можно перейти к этапу исправления в областях. Правила для исправлений сформированы на основе анализа часто допускаемых ошибок при оформлении списка литературы в курсовых и дипломных работах. Для выполнения исправлений используются правила, хранящиеся в базе данных. Как уже было отмечено ранее, в базе данных составляются цепочки для исправления (или выдачи на них замечания) уже известных ошибок. Такая база данных называется базой знаний, так как содержит информацию о человеческом опыте и знаниях в некоторой предметной области.

Разрабатываемая система не является самообучающейся. Знания для такой системы формируются экспертом или инженером по знаниям, то есть человеком, который владеет знаниями об особенностях и требованиях составления списка литературы, и часто допускаемых ошибок на практике. Из этих знаний и формируются шаблоны для поиска и замены. Таким образом, необходимо составить правила для уже известных ошибок.

В сведениях об ответственности при наличии предписанного сокращения [и др.], должен быть указан только один автор, фамилии и инициалы всех остальных авторов опускаются. Для исправления таких случаев сформировано следующее правило для замены:

([А-Я][.][\s][А-Я][.][\s][А-Я][а-я]+)[,][\s]([А-Я][.][\s][А-Я][.][\s][А-Я][а-я]+([,][\s])*)+([\s]\[и др\.\]) → $1$4

Далее приведено нечеткое регулярное выражение для общего обозначения материала:

\[({~1}Текст|{~2}(Видеозапись|Звукозапись|Изоматериал|Комплект|Кинофильм|Микроформа|Мультимедиа|Предмет|Рукопись)|{~2}(Электронный\sресурс))]

Операторы нечеткого выражения {~1} и {~2} задают допустимое число ошибок – 1 и 2 сответственно, допустимое для захвата подстроки. Например, в качестве шаблона «Электронный ресурс» будут также захватываться «Электроный ресурс», «электронный ресурс», «Електронний ресурс»:

([А-Яа-я][.][\s][:]\s.+[,][\s][1-2][0-9][0-9][0-9])|([А-Я][а-я]+[\s]:\s[А-Я]*[а-я]*\s*,[\s][09]{1,4})|(\[[Бб]\.\s[Мм]\.\][\s][:]\s.+[,][\s][1-2][0-9][0-9][0-9])|(([А-Яа-я][.][\s][:]|[А-Я][а-я]+[\s]:)\[[Бб]\.\s[Ии]\.\]\s)|(\[[Бб]\.\s[Мм]\.\s:\s[Бб]\.\s[Ии]\.])

Если в области примечания следует указание страниц, при этом оно находится не в начале области, сокращение «с.» следует приводить с маленькой буквы:

([^\n]:[\s]{0,1})С.([\s][0-9]{1,4}–[0-9]{1,4}) → $1с.$2.

Часто приводят несколько обозначений материала. Согласно ДСТУ необходимо общее обозначение материала приводится только один раз, и если их несколько, следует приводить наиболее общий:

(\[(.*)\])([\s]{0,1}[+]{0,1}[\s]{0,1})(\[(.*)\]) → $1.

В некоторых случаях потребуется выполнять исправления в несколько этапов из-за невозможности выполнить ряд исправляющих действий одним правилом. Например, если в области заглавия приводят несколько авторов, что не допускает стандарт. В этом случае необходимо выполнить всех приводимых авторов, кроме первого, в сведения об ответственности:

([А-Я][а-я]+[\s][А-Я][.][\s][А-Я][.])[,][\s]((([,][\s]){0,1}[А-Я][а-я]+[\s][А-Я][.][\s][А-Я][.])+)([\s][А-Я][^\n]+?)[\s]{0,1}(([\/]{2})|[.][\s][–]) → $1$5 / $1, $2 $6.

Также следует учесть, что в сведениях об ответственности инициалы идут в обратном порядке:

([/][\s]|[,][\s])+([А-Я][а-я]+)[\s]([А-Я][.][\s][А-Я][.])([\s]([^\/]))){0,1} → $1$3 $2$5.

Бабенко Л. П. Метод нормализации знаний об инфраструктуре разработки программ / Поляничко С. Л., Мясницов В. Н. // Кибернетика и системный анализ. — 2006. — № 1. — 167-174 с.

Бабенко Л. П. Метод нормализации знаний об инфраструктуре разработки программ / С. Л. Поляничко, В. Н. Мясницов // Кибернетика и системный анализ. — 2006. — № 1. — 167-174 с.

Рисунок 5 – Перенос инициалов в области сведений об ответственности

Таким образом, создавая правила для исправлений с помощью регулярных выражений, можно выполнить любые исправления в оформлении элементов областей и даже переносить элементы в другую область. Так как исправления по правилам будут выполняться последовательно, большое внимание необходимо уделить порядку построению правил, которое должно соответствовать порядку их выполнения.

В качестве механизма проверки правильности следования областей и присутствия обязательных областей мы используем модель конечного автомата. В библиографическом описании мы имеем 8 областей, которые идут в строго установленной последовательности:

S1 – Область заглавия и сведений об ответственности;
S2 – Область издания;
S3 – Область специфических сведений;
S4 – Область выходных данных;
S5 – Область физической характеристики;
S6 – Область серии;
S7 – Область примечания;
S8 – Область стандартного номера и условий доступности.

DOI 10.15588/1607-3274-2018-4-12

В итоге мы имеем 8 состояний, из них 3 являются обязательными практически для всех видов документов – «Область заглавия (S1)», «Область выходных данных (S4)», «Область физической характеристики (S5)». Таким образом, можно допустить цепочку (набор областей в библиографическом описании) при достижении состояния S5. Модель конечного автомата в виде графа изображена на рисунке:

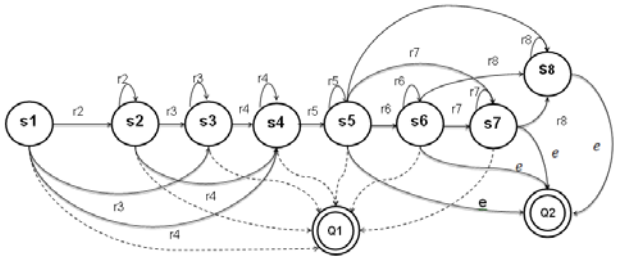

Рисунок 6 – Конечный автомат проверки в виде графа

На основе представленного графа получаем таблицу переходов данного конечного автомата. Строки соответствуют текущему состоянию, столбцы – выражению для перехода. На их пересечении – новое состояние, в которое попадает конечный автомат при соответствующем текущем состоянии и входному выражению.

Таблица 1 – Таблица переходов конечного автомата проверки

	R2	R3	R4	R5	R6	R7	R8	E
S1	S2	S3	S4	Q1	Q1	Q1	Q1	Q1
S2	S2	S3	S4	Q1	Q1	Q1	Q1	Q1
S3	Q1	S3	S4	Q1	Q1	Q1	Q1	Q1
S4	Q1	Q1	S4	S5	Q1	Q1	Q1	Q1
S5	Q1	Q1	Q1	S5	S6	S7	S8	Q2
S6	Q1	Q1	Q1	Q1	S6	S7	S8	Q2
S7	Q1	Q1	Q1	Q1	Q1	S7	S8	Q2
S8	Q1	Q1	Q1	Q1	Q1	Q1	S8	Q2

Таким образом, модель конечного автомата реализует последовательную проверку областей с помощью регулярных выражений. В данном конечном автомате обязательными областями являются состояния S1, S4и S5. Остальные области могут являться обязательными только при наличии таких сведениях в предписанных источниках информации.

При допустимом порядке и наборе областей конечный автомат попадает в терминальное состояние Q2, которое говорит о том, что цепочка допущена автоматом. В противном случае, мы попадаем в терминальное состояние Q1, которое не допускает заданную последовательность.

Для выполнения орфографических исправлений в названиях городов поиска будем использовать нейросеть Хемминга. На вход системы подается слово S, которое будем искать, и таблица базы данных со списком названий t, в которой будем осуществлять поиск. На выходе получаем – номер n(t, S) слова в списке t, которое наиболее близко к исходному слову S.

```
                    . . .
'ф'  "00001"   'с'  "01000"   'л'  "10110"
'ы'  "00011"   'а'  "01001"   'щ'  "10111"
                    . . .
```

Рисунок 7 – Коды символов

Входное слово из букв русского алфавита преобразуется в слово в алфавите 01, которое затем подается на вход нейронной сети, т.е. каждой букве в соответствие ставится слово из символов 0 и 1 длины 5. Кодирование строится таким образом, чтобы стоящие рядом на компьютерной клавиатуре символы имели близкие по Хеммингу коды. Таким образом, должно достигаться наиболее эффективное исправление опечаток.

тест	выход
Акмьлу	Акмола
Алмуты	Алматы
Архунгельск	Архангельск
. . .	. . .
Тьльятти	Тольятти
Тклу	Тула
Черкуссы	Черкассы
Ярьслувль	Ярославль

Рисунок 8 – Пример исправлений

Информационная модель системы создавалась на унифицированном языке моделирования UML (Unified Modeling Language). Визуальное моделирование с помощью UML представляет собой поэтапный спуск от наиболее общей концептуальной модели системы к логической, а затем и к физической модели, причем модель представляет собой совокупность так называемых диаграмм [14].

Возможности системы представлены на диаграмме вариантов использования (use case diagram, диаграмма прецедентов), которая отображает концептуальную модель системы (рис. 9).

Программа предполагает два режима работы – работу с базой знаний и работу с программой (обычный режим работы). При работе с базой знаний предполагается, что в качестве пользователя будет выступать эксперт в данной предметной области или инженер по знаниям. Работа с базой знаний включает в себя добавление ключей для идентификации, с помощью которых будут идентифицироваться области, и добавление ключей для исправления, которые представляют собой правила для поиска и замены фрагментов в нужных областях библиографического описания.

DOI 10.15588/1607-3274-2018-4-12

Структура системы показана на диаграмме классов (рис. 10), типовая последовательность действий пользователя – на диаграмме состояний (рис. 11), а сущности программной системы – на диаграмме компонентов (рис. 12).

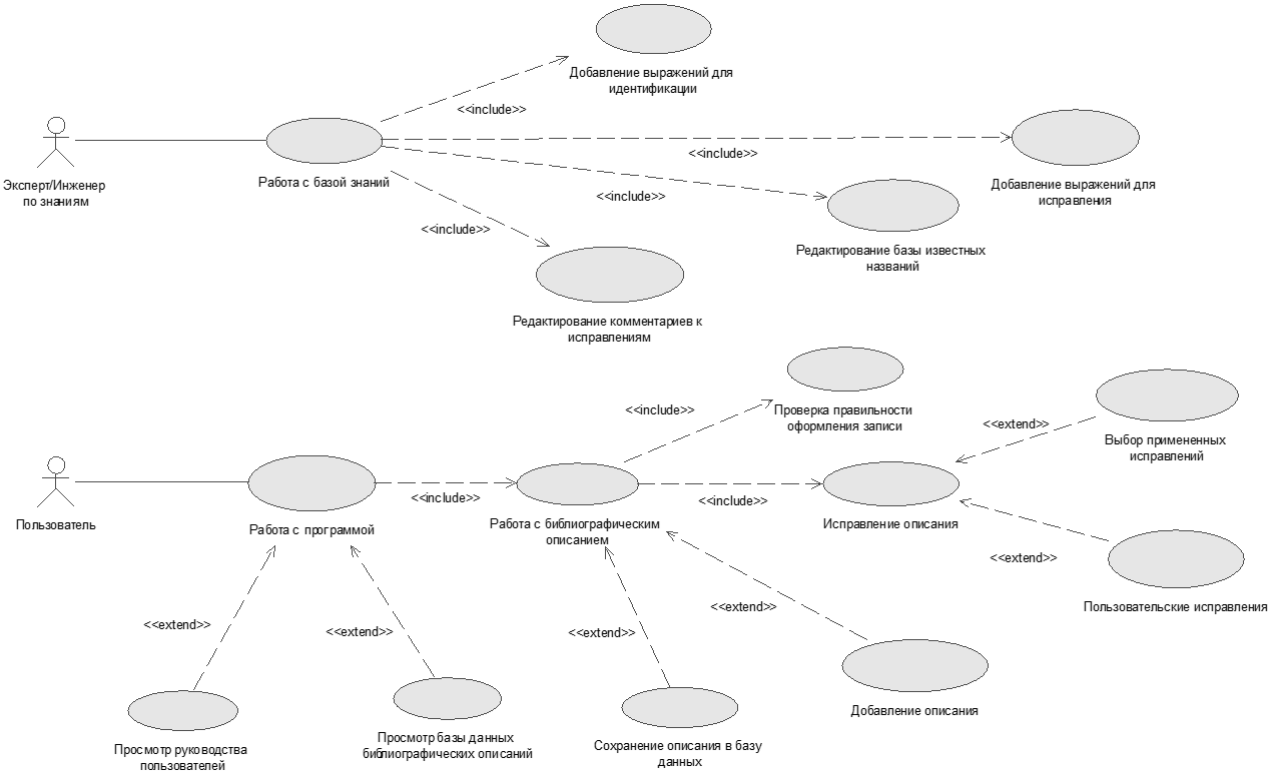

Рисунок 9 – Диаграмма вариантов использования

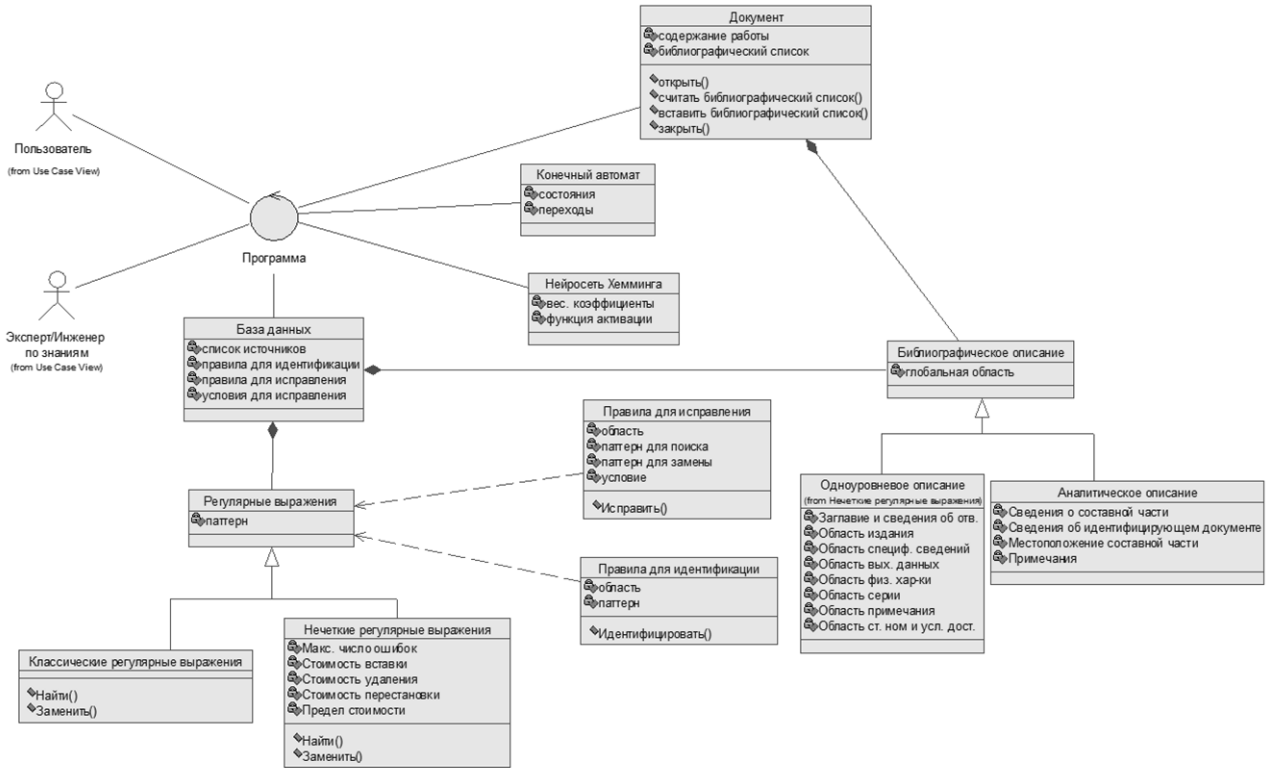

Рисунок 10 – Диаграмма классов

DOI 10.15588/1607-3274-2018-4-12

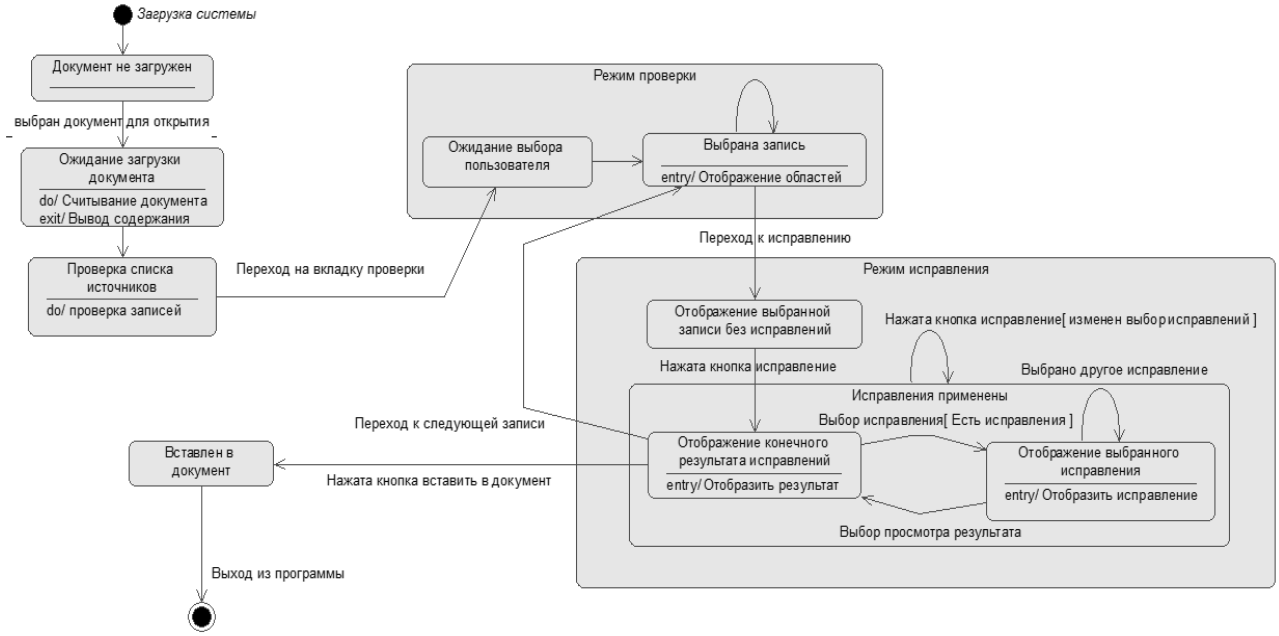

Рисунок 11 – Диаграмма состояний

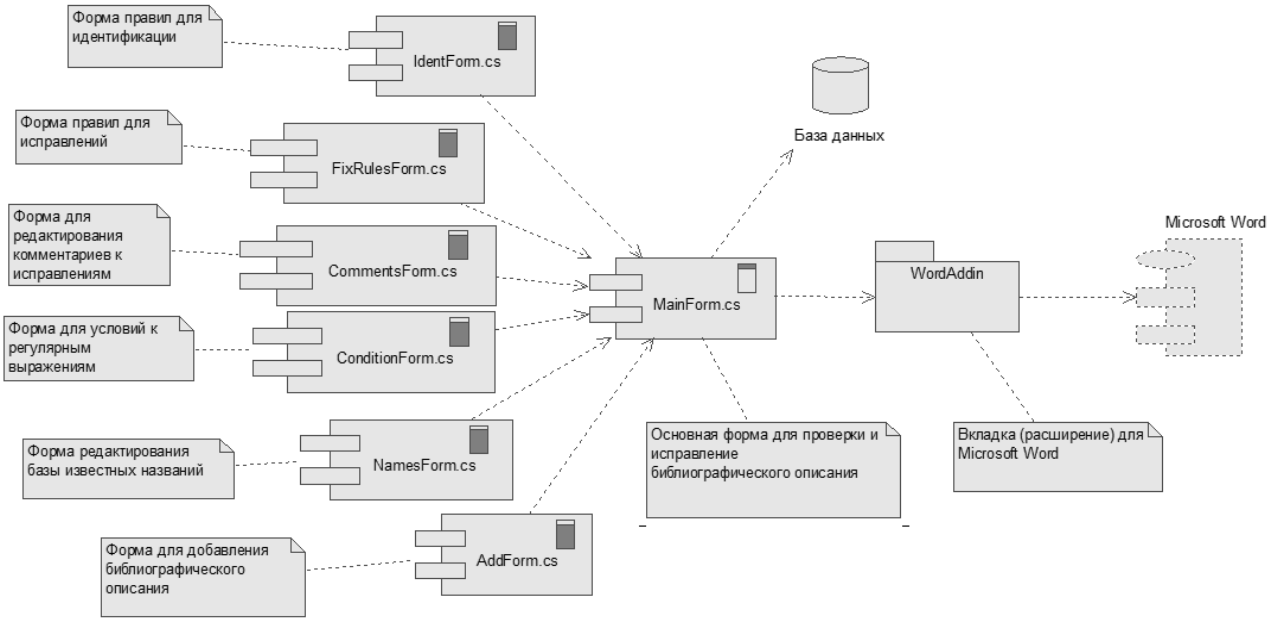

Рисунок 12 – Диаграмма компонентов

4 ЭКСПЕРИМЕНТЫ

Разработанная информационная модель интеллектуальной системы для анализа библиографического описания была реализована в среде визуального программирования Visual Studio 2017 C# на платформе Microsoft .NET Framework [15-18].

Для реализации программной системы используются такие встроенные компоненты, как Regex – библиотека регулярных выражений, которая содержит методы Match, IsMatch и Replace.

Для работы с документами пакета Microsoft Office используется внешний пакет Interrop.Office. Данный пакет включен в .NET Framework CLR (Common Language Runtime) и включает в себя использование COM (Component Object Model) для взаимодействия с объектами .NET, в данном случае с объектами приложения Office.Application и Office.Document.

Для работы с базой данных используется встроенный компонент OLE DB, который поставляется платформой .NET Framework, и также использует объектную модель компонентов COM.

Для осуществления нечеткого поиска и нечетких регулярных выражений используется пакет TRE. Данный пакет является реализацией нечетких регулярных выражений на языке C, который использует метрики расстояния Левенштейна.

DOI 10.15588/1607-3274-2018-4-12

Внешний вид программы состоит из пяти вкладок, которые включают разные этапы проверки и исправления списка литературных источников:

– Содержание – при работе здесь видно все загруженное в программу содержание области списка литературы;

– Проверка – здесь можно будет проверить список литературы в целом, при этом выделив нужное описание, и просмотреть его области;

– Исправление ошибок – здесь осуществляется исправление библиографического описания, которое было предварительно выбрано на вкладке проверки;

– Управление списком – здесь формируется список всех имеющихся (загруженных в программу) источников для обратной вставки в документ;

– База данных – взаимодействие с базой данных, то есть загрузка и сохранение библиографических описаний.

Для формирования базы знаний регулярных выражений выбрано хранилище в виде реляционной базы данных Microsoft Access. Структура разрабатываемой базы данных выглядит следующим образом:

– main – главная таблица, в которой предполагается хранение списка библиографии;

– ident – таблица, содержащая регулярные выражения для идентификации областей;

– rules – таблица, содержащая регулярные выражения в виде правил для исправлений (замены) фрагментов описания, как в целом, так и отдельных его областей;

– conditions – таблица, содержащая условия, необходимые для выполнения исправлений из набора правил rules;

– comments – таблица, содержащая комментарии, которые будут использоваться подсистемой объяснений при выполнении исправлений (замен) из набора правил rules.

Имея построенные правила для идентификации, сформируем базу знаний, добавляя для каждой области соответствующее регулярное выражение для идентификации. Для одной области может быть несколько выражений идентификации. Окно редактора выражений для идентификации представлено на рис. 13.

Аналогичным образом формируются правила для исправлений, как показано на рис. 14.

Для сформированных правил необходимы комментарии, которые будут выводиться пользователю при наличии исправлений.

Для случаев, когда какие-либо фрагменты библиографического описания исправить невозможно, предусмотрена подсистема замечаний, которая вместо осуществления исправления выведет определенное замечание, и по возможности выделит фрагмент, к которому оно относится.

После того, как сформирована база знаний, можно перейти к работе программы в обычном режиме. Необходимо выбрать проверяемый документ, после чего он будет загружен в программу и отобразится во вкладке «Содержание». Перейдя на вкладку проверки, можно видеть, какие библиографические описания нуждаются в исправлениях. Зеленым цветом помечены описания, которые не нуждаются в исправлениях. Желтым цветом – описания, в котором предлагается не более одного исправления. Красным цветом подсвечены описания, для которых предлагается более одного исправления.

В случае аналитического библиографического описания разделы областей делятся на составную часть и идентифицирующий документ (рис. 15).

После нажатия кнопки «Исправить» в нижней таблице появится список выполненных исправлений с объяснениями к исправлениям, если они имеются в базе знаний. Номер исправления соответствует номеру правила, которое выполнило это исправление (рис. 16).

На вкладке исправления ошибок также предусмотрена возможность редактирования. Есть возможность отредактировать сам список, который показывается в верхней части программы, после чего программа попросит подтверждение выполненных изменений. Кроме этого, предусмотрена возможность редактировать содержимое прямо в таблице. Для перехода в режим редактирования в таблице необходимо нажать кнопку «Редактировать».

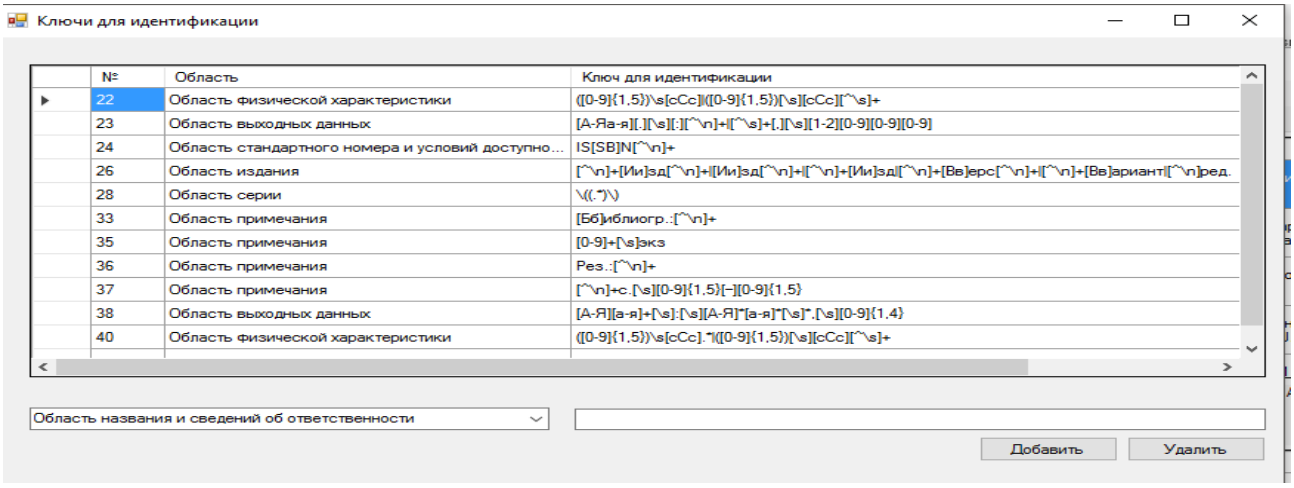

Рисунок 13 – Окно редактора регулярных выражений для идентификации областей

DOI 10.15588/1607-3274-2018-4-12

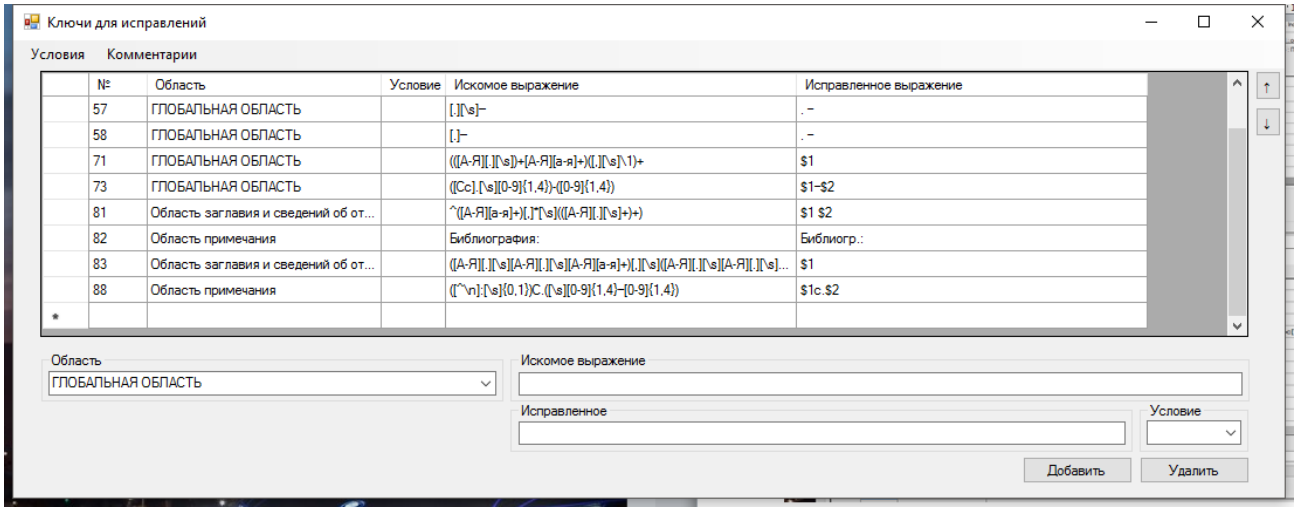

Рисунок 14 – Окно редактора с регулярными выражениями для исправлений

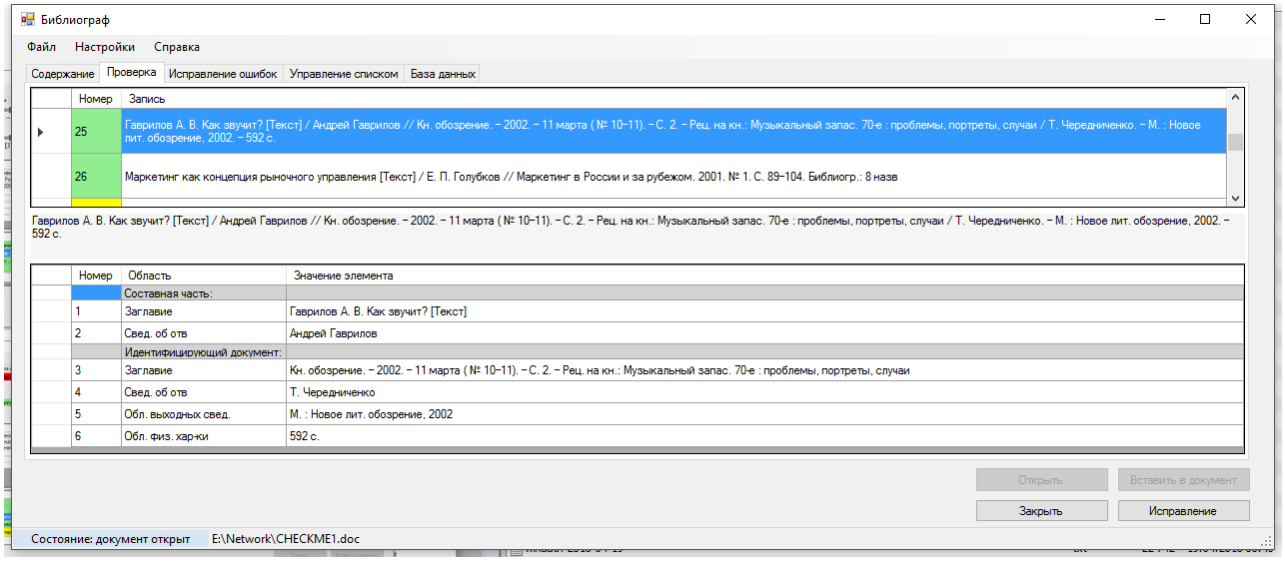

Рисунок 15 – Раздел проверки списка литературных источников

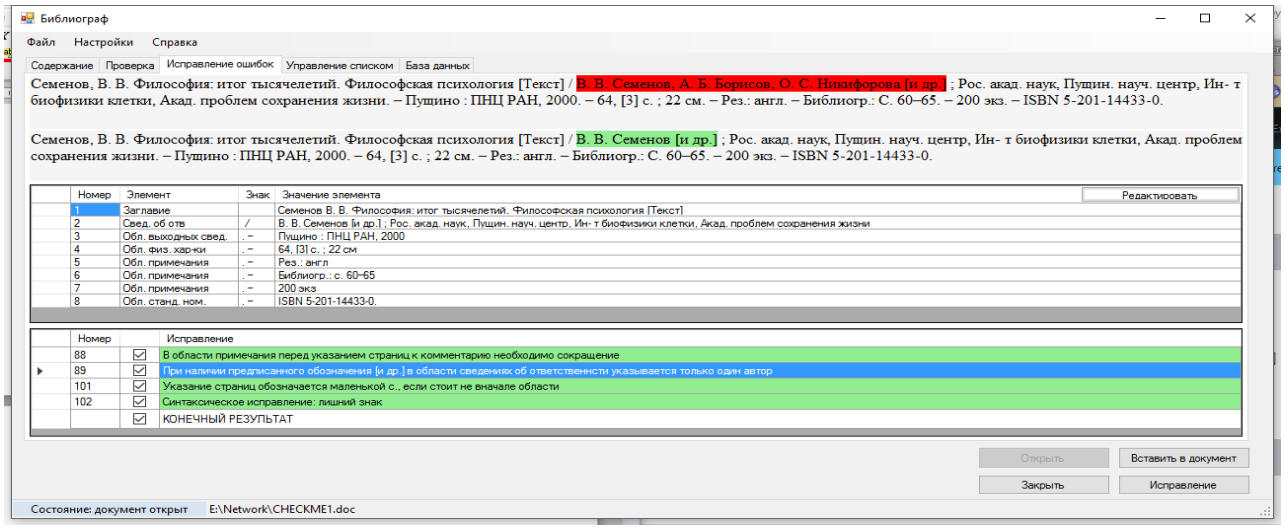

Рисунок 16 – Раздел исправления ошибок библиографического описания

DOI 10.15588/1607-3274-2018-4-12

В разделе «База данных» можно найти библиографическое описание по ключевым словам, а так же сохранить уже имеющиеся исправленные варианты библиографических описаний

После того как библиографические описания проверены и исправлены, их можно вставить обратно в документ. Для этого следует перейти в раздел «Управление списком». В этом разделе отобразятся готовые к вставке записи библиографических описаний. Выделив нужные (по умолчанию выделены все), нажать кнопку вставить, сформированный список вставится обратно в документ.

5 РЕЗУЛЬТАТЫ

Были изучены требования к составлению библиографического описания, проанализированы существующие методы формирования списка литературных источников, выявлены их недостатки, обоснована необходимость создания программного средства для проверки библиографических описаний. Проведено исследование методов интеллектуального анализа, рассмотрены методы регулярных выражений, нечеткого поиска с применением нечетких регулярных выражений, конечного автомата и нейронной сети Хемминга.

В процессе проектирования информационной системы составлена модель в виде UML-диаграмм, которая описывает варианты использования системы, классы и объекты, с которыми работает система, ее основные состояния и алгоритм действий при выполнении исправления.

ВЫВОДЫ

Разработанная интеллектуальная система позволяет выявить и исправить ошибки оформления библиографического описания. Основой программы является база знаний, которую можно дополнить при наличии каких-либо непредусмотренных ситуаций. Кроме этого, данная система дает возможность накапливать базу данных уже проверенными списками литературы.

Таким образом, разработанную систему можно улучшать без вмешательства в исходный код программы. При этом уровень ее возможностей и правильности исправлений, выдачи замечаний напрямую зависит от полноты и правильности составленной базы знаний.

Научная новизна заключается в создании ранее несуществующей модели проверки библиографического описания на соответствие действующим стандартам библиографического описания, основанной на методах регулярных выражений, нечеткого поиска, конечного автомата и нейронной сети Хемминга.

Практическая ценность заключается в разработки интеллектуальной для проверки библиографического описания и частично автоматизированного исправления с пользовательскими указаниями, сформированными на основе базы знаний.

К перспективам дальнейших исследований можно отнести использование других стилей оформления списка литературы, отличных от ДСТУ 7.1:2006, таких, как IEEE.

СПИСОК ЛИТЕРАТУРЫ

1. Стрельникова А. Г. Дипломная работа: подготовка и оформление : пособие для студентов / А. Г. Стрельникова. – СПб. : СпецЛит, 2010. – 96 с.
2. Научные работы : методика подготовки и оформления / [сост. И. Н. Кузнецов]. – Минск : Амалфея, 1998. – 272 с.
3. Система стандартів з інформації та видавничої справи. Бібліографічний запис. Бібліографічний опис. Загальні вимоги та правила складання : ДСТУ ГОСТ 7.1:2006. – [Чинний від 2007-07-01]. – К. : Держспоживстандарт України, 2007. – 52 с. – (Національний стандарт України).
4. Библиографическая запись. Библиографическое описание. Общие требования и правила составления : ГОСТ 7.1–2003. – Взамен ГОСТ 7.1–84. – [Введен 01.07.2004]. – М. : Изд-во стандартов, 2004. – 47 с. – (Система стандартов по информации, библиотечному и издательскому делу).
5. Калинина Г. П. Комментарии к ГОСТу 7.1–2003 / Г. П. Калинина // Библиография. – 2004. – № 4. – С. 65–66.
6. Бахтурина Т. А. Новый стандарт по библиографическому описанию: к внедрению ГОСТ 7.1–2003 / Т. А. Бахтурина // Библиография. – 2004. – № 1. – С. 23–26.
7. Добавление ссылки и создание списка литературы [Электронный ресурс]. – Режим доступа: https://support.office.com/ru-ru/article/Добавление-ссылки-и-создание-списка-литературы
8. Автоматичне оформлення джерел по ВАК України [Электронный ресурс]. – Режим доступа: http://vak.in.ua/
9. Фридл Дж. Регулярные выражения : пер. с англ. / Дж. Фридл. – 3-е изд. – СПб. : Символ-Плюс, 2008. – 608 с.
10. Форта Бен. Освой самостоятельно регулярные выражения : пер. с англ. / Бен Форта. – М. : Вильямс, 2005. – 184 с.
11. Мосалев П. М. Обзор методов нечеткого поиска текстовой информации [Электронный ресурс] / П. М. Мосалев. – Режим доступа: https://cyberleninka.ru/article/n/obzor-metodov-nechetkogo-poiska-tekstovoy-informatsii
12. Харитоненков А. В. Поиск на неточное соответствие: коды Хемминга [Электронный ресурс] / А. В. Харитоненков, Л. П. Вершинина. – Режим доступа: http://www.jurnal.org/articles/ 2009/inf32.htm
13. Белевцов Л. В. Введение в дискретную математику : учебное пособие / Л. В. Белевцов, Е. Ю. Гудкова. – Краматорск : ДГМА, 2013. – 144 с.
14. Мельников А. Ю. Объектно-ориентированный анализ и проектирование информационных систем : учебное пособие / А. Ю. Мельников. – 2-е изд., перераб. и доп. – Краматорск : ДГМА, 2013. – 172 с.
15. Марченко А. Л. Введение в программирование C# / А. Л. Марченко. – М. : МГУ, 2005. – 216 с.
16. Стиллмен Э. Изучаем C# / Э. Стиллмен, Дж. Грин. – СПб. : Питер, 2012. – 256 с.
17. Гуннерсон Э. Введение в C# : Библиотека программиста / Э. Гуннерсон. – СПб., М., Харьков, Минск : Питер, 2001. – 304 с.

DOI 10.15588/1607-3274-2018-4-12

18. Шилдт Г. C# 4.0. Полное руководство / Г. Шилдт. – М. : Вильямс, 2011. – 1056 с.
19. Комиссаров К. М. Разработка системы проверки правильности оформления списка литературных источников в курсовых и дипломных работах / К. М. Комиссаров, А. Ю. Мельников // Молодежь в науке: Новые аргументы: Сборник научных работ III-го Международного молодежного конкурса (Россия, г. Липецк, 29 февраля 2016 г.). Часть II / Отв. ред. А. В. Горбенко. – Липецк : Научное партнерство «Аргумент», 2016. – С. 75–80.
20. Мельников А.Ю. Разработка системы проверки списка литературных источников на основе базы знаний в виде правил на языке регулярных выражений / А. Ю. Мельников, К. М. Комиссаров // Автоматизація та комп'ютерно-інтегровані технології у виробництві та освіті: стан, досягнення, перспективи розвитку : Матеріали Всеукраїнської науково-практичної Internet-конференції. – Черкаси, 2017. – С. 208–210.

Received 27.05.2018.
Accepted 11.06.2018.

УДК 004.4:004.89:006.72

ДОСЛІДЖЕННЯ МЕТОДІВ ІНТЕЛЕКТУАЛЬНОГО АНАЛІЗУ БІБЛІОГРАФІЧНИХ ОПИСІВ ТА РОЗРОБКА ПРОГРАМНОЇ СИСТЕМИ ДЛЯ АНАЛІЗУ СПИСКУ ЛІТЕРАТУРИ

Мельников О. Ю. – канд. техн. наук, доцент, доцент кафедри інтелектуальних систем прийняття рішень, Донбаська державна машинобудівна академія, Краматорськ, Україна.
Комісаров К. М. – магістрант спеціальності «Системний аналіз», Донбаська державна машинобудівна академія, Краматорськ, Україна.

АНОТАЦІЯ

Актуальність. Розглянуто завдання аналізу бібліографічного списку з метою автоматизації перевірки правильності його складання. Об'єктом дослідження був процес формування бібліографічних описів на основі діючих стандартів. Предмет дослідження – моделі й методи інтелектуального аналізу для перевірки й виправлень бібліографічного опису. Ціль роботи – підвищення ефективності науково-педагогічної діяльності за рахунок застосування інтелектуальної системи для перевірки списку літературних джерел у курсових і дипломних роботах студентів.
Метод. Проаналізовано стандарт і наведено елементи, що становлять бібліографічний опис. Розглянуто існуючі інструменти для формування й перевірки списку літературних джерел, виявлені їхні недоліки, обґрунтована необхідність створення програмного засобу для перевірки бібліографічних описів. Наведено дослідження методів інтелектуального аналізу: розглянуті методи регулярних виражень, нечіткого пошуку із застосуванням нечітких регулярних виражень, кінцевого автомата й нейронної мережі Хемінга. Створено раніше не існуюча модель перевірки бібліографічного опису на відповідність діючим стандартам, заснована на методах регулярних виражень, нечіткого пошуку, кінцевого автомата й нейронної мережі Хемінга. Представлено об'єктно-орієнтовану модель проектованої комп'ютерної системи мовою візуального моделювання UML. Описано роботу комп'ютерної реалізації програмної системи, здійсненої в середовищі візуального програмування C#.
Результати. Розроблено програмну систему – інтелектуальну систему прийняття рішень – для перевірки бібліографічного опису й частково автоматизованого виправлення з користувальницькими вказівками, сформованими на основі бази знань.
Висновки. Розроблена інтелектуальна система дозволяє виявити й виправити помилки оформлення бібліографічного опису. Основою програми є база знань, яку можна доповнити при наявності яких-небудь непередбачуваних ситуацій. Крім цього, дана система дає можливість накопичувати базу даних вже перевіреними списками літератури. Розроблену систему можна поліпшувати без втручання у код програми. При цьому рівень її можливостей і правильності виправлень, видачі зауважень прямо залежить від повноти й правильності складеної бази знань. До перспектив подальших досліджень можна віднести використання інших стилів оформлення списку літератури, відмінних від ДСТУ 7.1:2006, таких, як IEEE.
КЛЮЧОВІ СЛОВА: бібліографічний список, бібліографічний опис, ДСТУ 7.1:2006, ISBD, регулярні вираження, нечіткий пошук, відстань редагування, нечіткі регулярні вираження, кінцевий автомат, нейросеть Хемінга, Unified Modeling Language, C#, WorldAddIn.

UDC 004.4:004.89:006.72

INVESTIGATION OF METHODS OF INTELLECTUAL ANALYSIS OF BIBLIOGRAPHIC DESCRIPTIONS AND DEVELOPMENT OF PROGRAM SYSTEM FOR ANALYSIS OF LITERATURE LIST

Melnykov O. Y. – PhD, Department of Intelligence Systems of Decision Making, Donbass State Engineering Academy, Kramatorsk, Ukraine.
Komissarov K. M. – Student, Donbass State Engineering Academy, Kramatorsk, Ukraine.

ABSTRACT

Context. The problem of analyzing a bibliographic list is examined with the purpose of automating the verification of the correctness of its compilation. The object of the study was the process of creating bibliographic descriptions based on existing standards. The subject of the study is models and methods of intellectual analysis for checking and correcting bibliographic descriptions.
Objective. The goal of the work is to increase the effectiveness of scientific and pedagogical activity through the use of an intellectual system to check the list of literary sources in the students' course and diploma papers.
Method. The standard is analyzed and the elements composing the bibliographic description are listed. Existing tools for forming and checking the list of literature sources are considered, their shortcomings are revealed, the necessity of creating a software tool for checking bibliographic descriptions is grounded. The research of methods of intellectual analysis is given: methods of regular expressions, fuzzy search with application of fuzzy regular expressions, finite automaton and neural network of Hamming are considered. A

DOI 10.15588/1607-3274-2018-4-12

previously existing model for checking the bibliographic description for compliance with existing standards was created, based on the methods of regular expressions, fuzzy search, the finite automaton and the neural network of Hamming. An object-oriented model of the projected computer system in the language of visual modeling of UML is presented. The work of computer implementation of the software system implemented in the C # visual programming environment is described.

Results. A software system was developed – an intelligent decision-making system – to check the bibliographic description and partially automated correction with user instructions generated based on the knowledge base.

Conclusions. The developed intellectual system allows to reveal and correct mistakes of registration of the bibliographic description. The basis of the program is the knowledge base, which can be supplemented if there are any unforeseen situations. In addition, this system makes it possible to accumulate a database of already verified lists of literature. The developed system can be improved without interfering with the source code of the program. At the same time, the level of its capabilities and correctness of corrections, the issuance of comments directly depends on the completeness and correctness of the compiled knowledge base. The prospects for further research include the use of other styles for the design of a list of literature other than DSTU 7.1: 2006, such as the IEEE.

KEYWORDS: bibliographic list, DSTU 7.1: 2006, ISBD, regular expressions, fuzzy search, editing distance, fuzzy regular expressions, finite state machine, Hamming neural network, Unified Modeling Language, C #, WorldAddIn.

REFERENCES

1. Strel'nikova A. G. Diplomnaja rabota: podgotovka i oformlenie, posobie dlja studentov. Sankt-Peterburg, SpecLit, 2010, 96 p.
2. Nauchnye raboty : metodika podgotovki i oformlenija [sost. I. N. Kuznecov]. Minsk, Amalfeja, 1998, 272 p.
3. DSTU GOST 7.1:2006. Sistema standartiv z informaciï ta vidavnichoï spravi. Bibliografichnij zapis. Bibliografichnij opis. Zagal'ni vimogi ta pravila skladannja. Vved. 2007-07-01. Kyiv, Derzhspozhivstandart Ukraïni, 2007, 52 p.
4. Bibliograficheskaja zapis'. Bibliograficheskoe opisanie. Obshhie trebovanija i pravila sostavlenija, GOST 7.1–2003. – Vzamen GOST 7.1–84 ; vved. 01.07.2004. Moscow, Izd-vo standartov, 2004, 47 p.
5. Kalinina G. P. Kommentarii k GOSTu 7.1–2003, *Bibliografija*, 2004, No. 4, pp. 65–66.
6. Bahturina T. A. Novyj standart po bibliograficheskomu opisaniju: k vnedreniju GOST 7.1–2003, *Bibliografija,* 2004, No. 1, pp. 23–26.
7. Dobavlenie ssylki i sozdanie spiska literatury [Jelektronnyj resurs]. Rezhim dostupa: https://support.office.com/ru-ru/article/Dobavlenie-ssylki-i-sozdanie-spiska-literatury
8. Avtomatichne oformlennja dzherel po VAK Ukraïni [Jelektronnyj resurs]. Rezhim dostupa: http://vak.in.ua/
9. Fridl Dzh. Reguljarnye vyrazhenija : per. s angl. 3-e izd., Sankt-Peterburg, Simvol-Pljus, 2008, 608 p.
10. Forta Ben. Osvoj samostojatel'no reguljarnye vyrazhenija : per. s angl. Moscow, Vil'jams, 2005, 184 p.
11. Mosalev P. M. Obzor metodov nechetkogo poiska tekstovoj informacii [Jelektronnyj resurs]/ Rezhim dostupa: https://cyberleninka.ru/ article/n/obzor-metodov-nechetkogo-poiska-tekstovoy-informatsii
12. Haritonenkov A. V., Vershinina L. P. Poisk na netochnoe sootvetstvie: kody Hemminga [Jelektronnyj resurs]. Rezhim dostupa: http://www.jurnal.org/articles/2009/inf32.htm
13. Belevcov L. V., Gudkova E. Ju. Vvedenie v diskretnuju matematiku: uchebnoe posobie. Kramatorsk, DGMA, 2013, 144 p.
14. Mel'nikov A. Ju. Ob#ektno-orientirovannyj analiz i proektirovanie informacionnyh sistem: uchebnoe posobie. 2-e izd., pererab. i dop. Kramatorsk, DGMA, 2013, 172 p.
15. Marchenko A. L. Vvedenie v programmirovanie C#. Moscow, MGU, 2005, 216 p.
16. Stillmen Je., Grin Dzh Izuchaem C#. Sankt-Peterburg, Piter, 2012, 256 p.
17. Gunnerson Je. Vvedenie v C# : Biblioteka programmista / Je. Gunnerson. Sankt-Peterburg, Moscow, Har'kov, Minsk, Piter, 2001, 304 p.
18. Shildt G. S# 4.0. Polnoe rukovodstvo. Moscow, Vil'jams, 2011, 1056 p.
19. Komissarov K. M., Mel'nikov A. Y. Razrabotka sistemy proverki pravil'nosti oformlenija spiska literaturnyh istochnikov v kursovyh i diplomnyh rabotah, *Molodezh' v nauke: Novye argumenty: Sbornik nauchnyh rabot III-go Mezhdunarodnogo molodezhnogo konkursa (Rossija, g. Lipeck, 29 fevralja 2016 g.*). Chast' II. Otv. red. A. V. Gorbenko. Lipeck, Nauchnoe partnerstvo «Argument», 2016. – S. 75–80.
20. Mel'nikov A. Ju., Komissarov K. M. Razrabotka sistemy proverki spiska literaturnyh istochnikov na osnove bazy znanij v vide pravil na jazyke reguljarnyh vyrazhenij, *Avtomatizacija ta komp'juterno-integrovani tehnologiï u virobnictvi ta osviti: stan, dosjagnennja, perspektivi rozvitku : materiali Vseukraïns'koï naukovo-praktichnoï Internet-konferenciï.* Cherkasi, 2017, pp. 208–210.

DOI 10.15588/1607-3274-2018-4-12