Правительство Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего профессионального образования
"Национальный исследовательский университет "Высшая школа экономики"
Отделение программной инженерии
Кафедра Управления разработкой программного обеспечения

УТВЕРЖДАЮ
Зав. кафедрой УРПО
_____ С.М. Авдошин
«__» _____ 2014г.

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
по направлению 231000.62 Программная инженерия
подготовки бакалавра

На тему: ***Программа реконструкции слов с запретами по мультимножеству подслов в гипотезе единичного сдвига***

Студентки группы № 472ПИ _____ Стёпушкиной Н. В.

_____
(Дата)

Научный руководитель
д.т.н., профессор _____ Ульянов М. В.

_____
(Дата)

Москва, 2014 г.

Реферат

Сведения о работе:

Работа состоит из 62 страниц. Она содержит три главы: «Обзор решения задач в данной области», «Используемые методы и алгоритмы», «Реализация и анализ результатов». Кроме того, в работе содержится 18 иллюстраций, 8 таблиц, 5 приложений. При работе использовалось 13 источников, приведенных в соответствующем разделе.

Аннотация:

В данной работе исследовались методы реконструкции слов по подсловам в рамках гипотезы единичного сдвига.

Целью работы является описание и разработка эффективного алгоритма построения слов, сохранения и представления данных, реализации визуального представления.

В тексте данной работы приводится описание процесса работы алгоритма. В конце представляется заключение и обзор достигнутых результатов. Результатом является готовый программный продукт на языке C++, позволяющий выполнять поставленные задачи.

Ключевые слова:

Комбинаторика слов, реконструкция слова, гипотеза единичного сдвига, граф де Брейна, Эйлеров путь, матрица смежности, запрещенные слова.

Содержание

Оглавление

Реферат ... 2
Сведения о работе: ... 2
Аннотация: ... 2
Ключевые слова: ... 2
Определения, обозначения и сокращения ... 5
Введение ... 6
Обзор решения задач в выбранной области ... 8
Постановка задачи ... 8
Область применения ... 8
Существующие исследования ... 10
Используемые методы и алгоритмы ... 11
Предпосылки работы алгоритма ... 11
Этапы решения ... 11
Начало работы алгоритма ... 12
Введение запрещенных слов ... 15
Реализация алгоритма и анализ результатов ... 17
Тестирование алгоритма ... 17
Дополнительные функции программы ... 19
Заключение ... 21
Список использованных источников ... 22
Приложения ... 23
Приложение А. ... 24
Техническое задание ... 24
Приложение Б. ... 31
Пояснительная записка ... 31
Приложение В. ... 39
Руководство пользователя ... 39
Приложение Г. ... 53
Программа и методика испытаний ... 53
Приложение Д. ... 61
Текст программы ... 61

Определения, обозначения и сокращения

В дальнейшем тексте работы под термином «слово» будем понимать последовательность букв над некоторым заданным алфавитом. Здесь следует отметить, что в данном случае буква может состоять из любого символа или последовательности символов, заданных этим алфавитом.

Подслово это некоторая строгая последовательность символов слова, начинающаяся в определенной позиции. Для ее получения берется часть слова, представляющая собой последовательность символов определенной длины. Если представить слово набором букв, то подслово представляет собой содержимое некоторого окна заданной длины, наложенного на это слово.

Гипотеза единичного сдвига это гипотеза, по которой строятся подслова. Следуя ей, каждое из подслов является последовательностью символов, полученной путем сдвига окна относительно некоторого реконструируемого слова на одну букву.

Префикс и суффикс слова – подслова без последней и первой буквы соответственно.

Введение

Проблема данной работы относится к одной из многочисленных задач комбинаторики слов. Эта тема достаточно обширна и в ней можно выделить огромное количество более мелких и узкоспециализированных проблем, иногда пересекающихся между собой.

Задача реконструкции слов по некоторым исходным данным относится к вопросу восстановления полной информации по ее частичным или искаженным описаниям. Для простоты восстанавливаемые данные можно обозначить реконструируемым словом над некоторым известным конечным алфавитом. Частичные описания в данной работе представляют собой некоторые наборы слов, которые можно получить из восстанавливаемого слова.

Любую информацию можно представить в виде слова или набора слов, задачи комбинаторики слов включают в себя различные области, тесно связанные с различными аспектами жизни. Проблема реконструкции слов связана с вопросами кодирования, конечными автоматами, теорией графов и множеством других аналитических и практических задач, например, зависимость свойств молекулы ДНК от последовательности нуклеотидов в ней [1, 2, 3]. Соответственно, данные можно представить в качестве последовательности некоторых символов или групп символов и на основании их решать задачу реконструкции наиболее подходящего слова, отвечающего всем установленным правилам.

Формулируя задачу реконструкции слова, представим набор слов, с помощью которых оно восстанавливается, как некоторые последовательности символов, являющиеся частями этого слова. Представим задачу о кратном покрытии [4] как поиск некоторого слова a минимальной длины. При этом каждое из слов a_1, a_2, …, a_n получается путем сдвига некоторого окна заданной длины по исходному слову a на один символ. Соответственно, одним из условий задачи о кратном покрытии является то, что каждое из слов a_i встречается в качестве подслова ровно m_i раз, где m – кратность подслова, и каждое из них уникально при образовании последовательности a_1, a_2, …, a_n.

Таким образом, перед нами встает задача реконструкции конечного слова на основе строго заданной последовательности подслов, имеющихся в этом слове. Отдельной интересной задачей при этом является поиск слова, не содержащего изначально заданных запрещенных слов, определяющих некоторый язык.

Условие о сдвиге окна вдоль слова на один символ является одним из вариантов необходимого для реконструкции слова ограничением. При этом дальнейшее развитие данного вопроса может быть связано с изменением правил построения искомого слова, введением дополнительных ограничений для каждой конкретной задачи или, напротив, изменение начальной гипотезы единичного сдвига.

Обзор решения задач в выбранной области

Постановка задачи

Рассмотрим подробнее существующие наработки, способные помочь в решении задачи реконструкции, возможные алгоритмы и представленные аналоги программных систем.

В пространстве комбинаторики слов постоянно появляется множество разнообразных тем, затрагивающих различные области. При этом старые вопросы непрерывно уточняются и изменяются под определенные задачи, которые, в свою очередь, также меняются со временем, становясь более универсальными или, наоборот, узконаправленными, но это не меняет их сложности и необходимости для дальнейшей разработки.

Условно проблему реконструкции слова, например, можно отнести к проблемам распознавания выводимости в математической логике [5]. Любое слово, являющееся частью восстанавливаемого слова, должно быть возможно выведено из соседнего с ним. По условию постановки задачи, каждое из слов a_i является последовательностью символов, полученной из другого слова a_j путем сдвига окна относительно некоторого реконструируемого слова. Соответственно, совокупность, или последовательность всех слов $a_1, \dots, a_n$, расставленная с соблюдением некоторых правил и в определенном порядке, который при наличии некоторых условий может быть не единственным, образует последовательность символов a, которая и является искомым словом.

Рассмотрим одну из моделей задач восстановления объекта по некоторым обрывочным данным [6]. В ней ставится вопрос о принадлежности некоторого объекта к одному из классов, но сам объект при этом задается с помощью обрывочной информации, заключающейся во множестве его подслов, и рассматриваются возможные случаи и особенности.

Область применения

Вообще наука комбинаторика возникла уже давно: первые упоминания о вопросах, близких к данной тематике, упоминаются еще в XII веке в китайских рукописях, хотя эти труды в основном затрагивали обычные жизненные вопросы.

Задача восстановления слова по заданной частичной информации с учетом некоторых известных правил также возникала не раз. Появление интереса к комбинаторике слов можно отнести еще к началу прошлого века, к работам Акселя Туэ [7].

Различные задачи реконструкции слов, так или иначе связанные между собой, рассматривались многими математиками. Результаты их исследований позволяют существенно продвинуться в решении реальных задач и на основании их строить алгоритмы и готовые программные средства.

В первой постановке проблема реконструкции слова строилась в бинарном виде, но алфавит, из которого строится слово, может представлять собой на самом деле как последовательность различных символов, так и их комбинации, каждую из которых можно условно назвать одной буквой. В таком варианте построения задачи восстанавливаемое слово может представлять собой объект, обладающий любым объемом информации, ограниченным лишь конкретной практической задачей и вычислительной мощностью определенного компьютерного средства. Для пояснения идеи можно привести машинные языки, в которых все данные задаются с помощью последовательностей нулей и единиц различной длины.

Так, например, вопросом восстановления некоторого объекта, описываемого словом над заданным алгоритмом, может являться восстановление расположения нуклеотидов в молекуле ДНК, как приводилось выше, что сильно сказывается на ее структурно-динамических свойствах.

Другой областью применения подобной задачи можно считать теорию автоматов и графов, где с помощью набора слов, являющихся частями слова, и некоторыми условиями их взаимодействия строится конечная модель, реализующая ту или иную функцию.

Теоретически с помощью несложных формул можно вычислить вероятность единичного восстановления слова или вычисления необходимых условий, при которых реконструкция такого слова вообще возможна. Также это дает возможность узнать минимальный размер получаемого слова и некоторые прочие параметры. Эти данные могут значительно уменьшить сложность работы, ее время и затраты ресурсов при программной реализации данной задачи.

Множество сложных задач можно легко разбить на более легкие и универсальные подзадачи и свести к готовым и интересным решениям. Многие из них решаются теоретически, чтобы в дальнейшем дать начало новым задачам, уже, возможно, имеющим

под собой практическую ценность. Другие начинают разрабатываться с реальной задачи, специальной под которую и продумывается алгоритм.

Существующие исследования

Хотя в строгом определении задача реконструкции слов звучит вполне определенно, она может служить некой отправной точкой для дальнейшего изменения в зависимости от различных поставленных вопросов. Поэтому данная область комбинаторики слов продолжает вызывать интерес у ученых и порождать новые задачи и решения.

Чуть позднее задача реконструкции слов по заданным подсловам была подробнее рассмотрена и изучена [8, 9]. Предложенные проблемы и возможные варианты представляют определенный интерес для восстановления слов и поиска всех возможных решений. Авторы старались как можно глубже изучить вопрос восстановления информации из известных подслов при наличии некоторых условно заданных параметров и разработать свои алгоритмы и пути решения проблем, возникающие на каждом этапе теоретической постановки задачи.

Одним из алгоритмов, близких по смыслу к поставленной задаче, можно назвать пакет BruijnViz, разработанный в демонстрационном варианте для Института математики СО РАН [10]. Тем не менее, программа так и не появилась в конечной версии и не была опубликована.

Аналитические наработки авторов различных изысканий в данной области представляют определенную информационную ценность и могут послужить отправной точкой для дальнейших разработок этой темы. Тем не менее, некоторые из вопросов, достаточно простых в теории, представляют собой сложность на практике, так что данную тему нельзя считать полностью закрытой и исследованной.

На основании приведенных исследований предложенных вариантов можно построить алгоритм решения задачи реконструкции набора слов по известным подсловам, заданным над некоторым конечным алфавитом.

Используемые методы и алгоритмы

Предпосылки работы алгоритма

Определим условия поставленной задачи. На основе набора подслов полученного путем сдвига окна заданного размера относительно некоторого искомого слова, необходимо реконструировать это слово. При этом могут существовать запрещенные слова любой длины, которые не должны являться подсловами получаемого слова.

Для удобства обозначим подслова, являющиеся частью искомого слова, как a_1, a_2, …, a_n, а реконструируемое слово как a. Несложно заметить, что подслова имеют длину меньшую, либо равную размеру слова a. При этом длина восстанавливаемого слова равна $k + n - 1$, где k – длина каждой из частей. Во всех прочих случаях задача не имеет решения.

Предложенная последовательность слов может быть представлена как последовательность де Брейна. Соответственно, если на основе этих слов построить ориентированный граф де Брейна, то затем можно реконструировать возможные слова с помощью конкатенации первых букв вершин в порядке прохождения по графу [11]. Очевидно, что последовательность де Брейна может состоять из любых символов заданного алгоритма, а при реконструкции слов необходимо пройти через все вершины, помеченные реконструирующими подсловами.

Задача восстановления слова, поставленная таким образом, сводится к поиску Гамильтоновых путей в графе, который относится к классу NP-трудных задач. В данном случае будет разумным размечать не вершины, а дуги как фрагменты искомого слова. В таком виде задача становится проблемой поиска всех Эйлеровых путей, менее трудоемкой и сложной. Соответственно, если граф в данном случае не содержит ни одного решения, то и задача в целом для данных подслов не может быть решена. Таким образом, задачу реконструкции слов можно разбить на несколько этапов.

Этапы решения

1) Простейшие проверки возможности построения слов. Здесь учитывается принадлежность слов алфавиту, их длина. Также сюда входит проверка, не является ли какое-либо из запрещенных слов частью любого из заданных подслов. Если на любом из этапов проверки произошло нахождение ошибки, слово с такими параметрами невозможно реконструировать и алгоритм прекращает

работу. Также подобные простейшие проверки будут проводиться в процессе работы алгоритма.

2) Построение ориентированного связного графа. Подслова могут повторяться, так что необходимо учитывать кратность всех элементов графа. На этом этапе также может возникнуть ошибка в случае, если один или несколько элементов являются несвязными, но на данном этапе она не учитывается.
3) Поиск всех возможных эйлеровых путей в полученном графе. Иначе это можно назвать составлением различных последовательностей де Брейна минимальной длины. Размер конечного слова, как уже было сказано, при этом зависит от размера каждого из подслов и их количества. На данном этапе обнаруживается проблема несвязности графа или невозможности достижения какой-либо из вершин.
4) Удаление запрещенных слов. Слова, имеющие в себе любое из запрещенных подслов, также являются запрещенными. Благодаря этому условию, даже при верно введенных начальных данных конечная последовательность слов может оказаться пустой.
5) Проверка на наличие реконструированных слов и вывод их последовательности на экран.
6) Дополнительная функция, позволяющая пользователю увидеть на экране иллюстрацию построенного ориентированного графа.

Начало работы алгоритма

Первая часть заключает в себя простейшие формулы и сравнения, не требующие особых затрат. Она всего лишь ограничивает ввод заведомо неправильных данных и сберегает время и ресурсы в простейших случаях. Но последующие части требуют алгоритмов, уменьшающих затраты на вычисления.

Представим дуги в ориентированном графе как подслова, с помощью которых реконструируется граф. При этом каждая дуга имеет начало и конец и является ориентированной.

Поскольку задача сводится к реконструкции в гипотезе единичного сдвига, представим каждое из подслов как два элемента: последовательность букв без первой буквы и без последней. Для простоты назовем их префиксом и суффиксом подслов [9]. В сущности,

каждая такая пара является реконструирующим набором слов для своего подслова. Тогда каждая из дуг ведет от своего префикса к своему же суффиксу, и их мы можем обозначить в качестве вершин. При этом, если начальные данные заданы верно, суффикс каждого из подслов является префиксом одного из других подслов. В таком случае можно построить граф, в котором неповторяющиеся последовательности без одной буквы являются вершинами, а сами подслова означают дуги, ведущие от префикса к суффиксу.

Таким образом, объединив все вершины и дуги в одну картинку, мы получаем ориентированный граф, реконструирующий набор возможных слов. Так как вершины могут повторяться, то и дуги могут несколько раз вести из одной вершины в другую и обратно. Для построения графа просто запишем для каждой дуги ее кратность, то есть, так называемая степень вершины. Поскольку нас интересуют именно дуги, являющиеся подсловами, кратность вершин нас при этом не интересует.

Теперь перейдем к поиску всех возможных эйлеровых путей, воспользовавшись для этого идеей возведения в степень матрицы смежности графа. Как известно, матрица смежности, возведенная в некоторую степень m, обладает следующим свойством: элемент в -й строке, j-м столбце равен числу путей из i-й вершины в j-ю, состоящих из ровно m ребер [12].

Взяв эту идею за основу, мы можем представить полученный граф как матрицу смежности. Для этого обозначим каждый из элементов набора префиксов и суффиксов всех подслов своим собственным номером. Теперь каждая дуга a_i ориентированного графа ведет из вершины t_k в вершину t_l, где числа k и l обозначают соответствующие номера элементов из набора префиксов и суффиксов. Соответственно, мы получаем матрицу смежности ориентированного графа, демонстрирующую все возможные переходы.

Для поиска всех возможных переходов введем дополнительные символы: «+» и «,», используемые в алгоритме. Для удобства в процессе умножения мы оперируем не самими подсловами искомого слова, а их символическими обозначениями, которые будут переводиться в слова на пятом этапе. Таким образом, одно символическое обозначение подслова от другого в конечной последовательности будет отделяться символом «,». Соответственно, одна последовательность перечисленных в строгом порядке частей отделяется от другой с помощью символа «+», означающего переход к следующей последовательности подслов, образующих реконструируемое слово.

В процессе возведения матрицы в степень мы будем удалять из набора подслов те последовательности, в которых количество одинаковых подслов превышает их кратность.

Соответственно, элемент матрицы, умноженный на пустой элемент, также будет давать пустой элемент.

При умножении матрицы каждый новый элемент, удовлетворяющий всем условиям, записываем в конец набора подслов через символ «,». Тем не менее, благодаря тому, что каждому из подслов, префиксов и суффиксов в ходе работы алгоритма присваивается индивидуальный номер, это не создает дополнительные ограничения на использование знаков «,» и «+» и избавляет от путаницы.

Также во время умножения матриц могут появляться новые наборы последовательностей, которые, соответственно, отделяются знаком «+». При умножении всей полученной последовательности на новый элемент на него умножается поочередно каждый из наборов подслов, отделенных этим знаком. Некоторые, а то и все наборы в таком случае могут превратиться в пустой элемент при превышении кратности какого-либо из подслов.

Умножение получающейся на каждом шаге матрицы проводится столько раз, сколько всего исходных частей, по которым реконструируется слово. При этом, отметим еще раз, количество дуг совпадает с количеством подслов только с учетом их кратностей.

Стоит заметить, что предложенный алгоритм также не универсален, хотя является красивым решением задачи для большого набора входных данных. Так, например, одним из исключений является такой набор входных слов, где каждое из них состоит всего из одного символа. Очевидно, что в этом случае реконструируемые слова получаются путем всех возможных перестановок букв. Решение этой задачи с помощью графов является, во-первых, более сложным, чем простая перестановка каждой буквы, а, во-вторых, не может решаться с помощью описанного выше алгоритма: в данном случае нарисованный граф будет несвязным и, следовательно, не даст набора возможных слов.

В случае таких входных данных единственным ограничивающим условием являются запреты по подсловам. Наиболее простым и эффективным решением в данном случае будет рекурсивный алгоритм [13].

Таким образом, программа выбирает подходящий алгоритм в зависимости от входных данных. В простейшем случае, когда слова состоят из одной буквы и, соответственно, сдвиг на один символ дает абсолютно новое слово, используется обычный алгоритм поиска всех возможных перестановок. Для более сложных наборов используются свойства графа де Брёйна и Эйлерова пути. Финальный набор элементов представляет собой все возможные реконструируемые слова без учета запретов.

Введение запрещенных слов

Формально эту задачу можно представить в виде того же ориентированного графа, из которого удаляются все пути, составляющие собой запрещенные слова. Задача несколько усложняется тем, что размер этих слов различен и может быть как меньше любого из подслов, так и больше восстанавливаемого слова в целом.

Самые простейшие варианты, такие как маленький или слишком большой размер запрещенного слова, находятся уже на первом этапе, и на основании их уже ставится вопрос о дальнейшей обработке данных. Так что на четвертом этапе нас интересуют лишь те запрещенные слова, которые имеют размер больше, чем у любого подслова, но меньше либо равный, чем у реконструируемого слова.

На этом этапе возникает та проблема, что длина запрещенного слова так относится к длинам подслов, что это слово невозможно исключить из рассмотрения путем удаления из графа соответствующих переходов. Мы снова возвращаемся к разбиванию подслов и их последовательностей на буквы и дальнейшую работу с ними. Соответственно, дальнейшие варианты работы алгоритма могут представлять собой два случая: удаление из списка готовых слов всех, включающих в себя запрещенные, или на каждом этапе умножения матриц проверка вхождения запрещенных слов и удаление из графа соответствующих путей с учетом их кратностей.

Сложность первого варианта ограничивается количеством и длиной запрещенных слов, а также количеством и длиной полученных слов. Он является простейшим и самым, пожалуй, надежным, но, конечно, далеко не самым удобным в плане затрат времени и ресурсов.

Последний вариант облегчается тем, что на каждом из этапов умножения матрицы смежности рассматриваются все возможные переходы из одной вершины в другую и задействуются все дуги и вершины, пока не объединенные в один путь. Соответственно, как только длина умножаемой последовательности символов начинает превышать длину одного из запрещенных слов, нам больше нет необходимости дальше проверять это слово. Это вводит дополнительную сложность с проверками длин последовательностей и слов и постоянным переведением символического обозначения подслов в последовательность символов. Если представлять подслово не в символическом виде, может возникнуть проблема с проверками и переводом: алфавит задается извне и может содержать группы

символов, похожие друг на друга. Так, например, в этом алфавите могут быть символы типа «0» и «00», являющиеся двумя разными буквами.

Тем не менее, во втором случае путь, содержащий в себе запрещенное слово, сразу удаляется из графа. В дальнейшем это также может привести к серии проверок, что усложняет работу алгоритма.

На последнем, пятом этапе производится перевод полученных последовательностей, из которых уже удалены все запрещенные слова, в конечные слова. Используя символическое представление каждого из подслов, мы переводим их следующим способом: первое подслово пишется полностью, затем дописывается по последней букве из следующих за ним в строгом соответствии с их порядком в последовательности. Заметим, что при желании, напротив, можно записывать первую букву каждого подслово и последнее подслово целиком, что дает такой же результат.

На этом алгоритм завершает свою работу и производится вывод данных на экран. В качестве дополнительной функции пользователь может выбрать иллюстрацию построенного по входным данным графа в дополнительной панели на экран.

Реализация алгоритма и анализ результатов

В качестве языка программирования был выбран C++ в среде разработки Microsoft Visual Studio 2013. Он считается одним из самых эффективных и кроссплатформенных языков и используется повсеместно.

Тестирование алгоритма

Помимо обычного тестирования работоспособности программы на предмет выявления ошибок, проблем или возможных методов упрощения работы приложения, был проведен анализ эффективности алгоритма. На основе ввода различных данных несколькими людьми были получены результаты, представленные на графиках внизу.

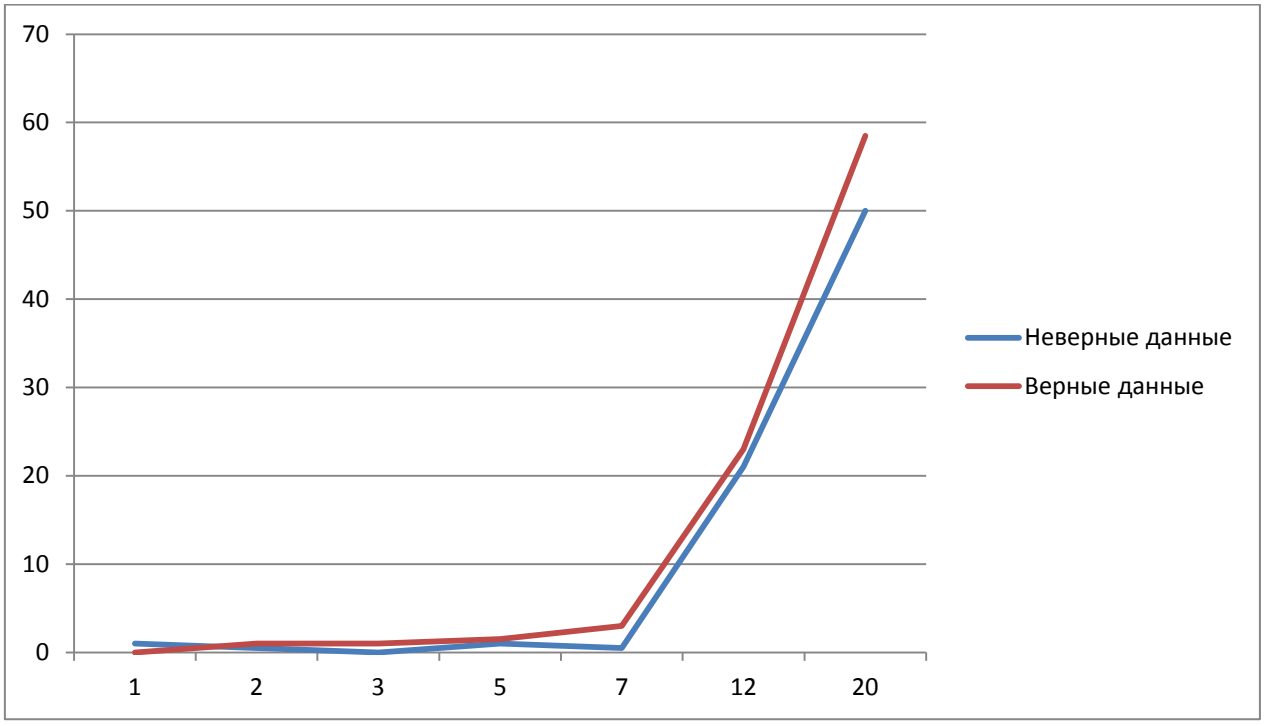

Рисунок 1

На рисунке 1 представлено среднее время работы приложения. По вертикали расположено время в тиках, а по горизонтали – количество подслов. В среднем количество букв в подслове также равнялось количеству подслов. Так, например, для 12 подслов и 12 букв в каждом из них количество тиков, за которое алгоритм полностью выполнялся,

равнялось 20. Как видно из графика, при вводе неверных данных время работы алгоритма существенно сокращалось.

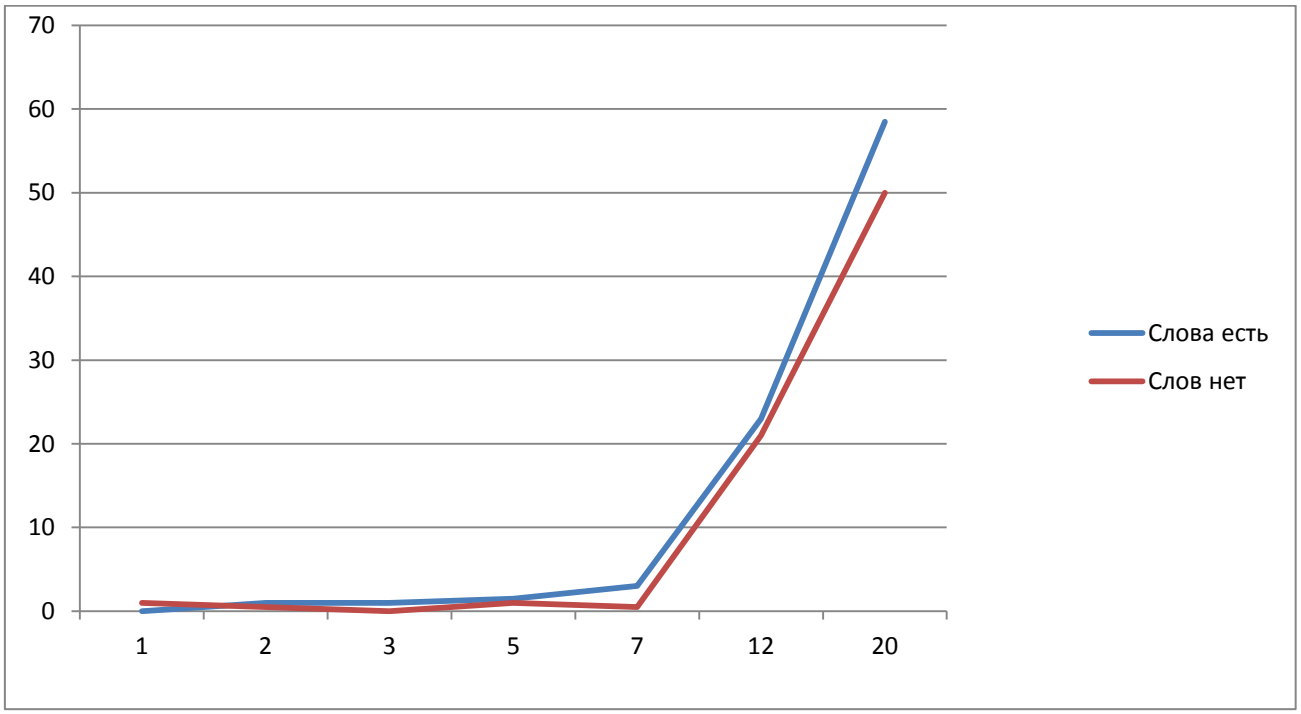

Рисунок 2

Рисунок 2 иллюстрирует также время работы в тиках для тех же значений подслов, но с использованием запрещенных слов. Функция, помеченная красным цветом на графике, показывает изменение времени работы алгоритма при наличии запрещенных слов, таких, что данный набор не реконструирует ни одно слово. Время работы алгоритма для такого набора данных, что программа возвращает одно или более слово, помечено синим цветом.

Как видно из предложенных графиков, время работы программы возрастает в зависимости от количества введенных данных. При этом алгоритм не зависит от последовательности расположения подслов, приводя к одинаковым результатам.

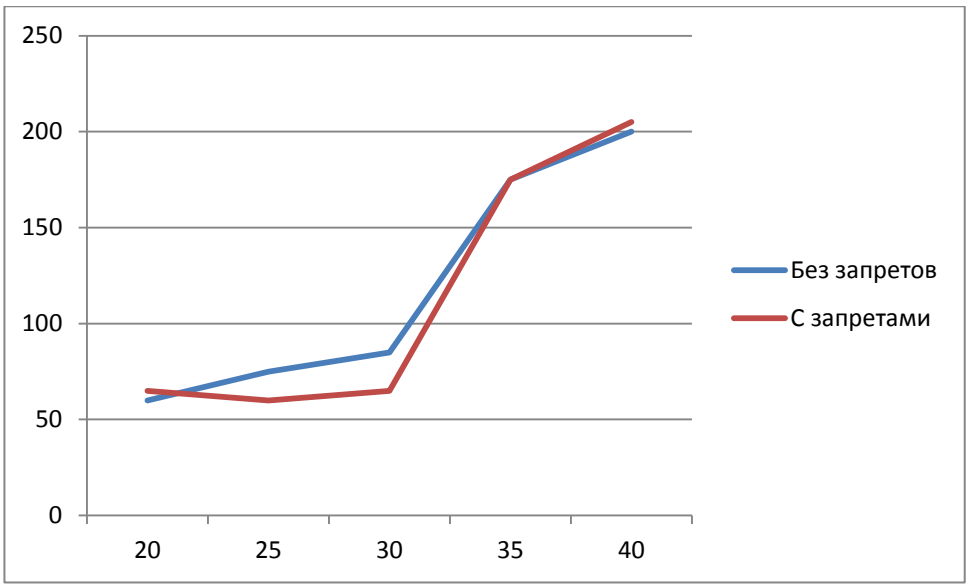

Рисунок 3

На рисунке 3 изображено время работы алгоритма в тиках для двух вариантов значений. Функция, обозначенная синим цветом, отражает работу при заданных исходных данных без ввода запрещенных слов, функция, обозначенная красным цветом – для данных с учетом запрещенных слов, таких, что часть реконструируемых слов при этом становилась неверной. По горизонтали отражается количество введенных подслов. По вертикали показывается среднее время работы алгоритма для обоих случаев.

Как видно из представленного графика, для достаточно больших вводимых данных (более 20 подслов), количество затраченного времени существенно увеличилось, но при этом разница во времени между случаем с вводом запрещенных слов и случаем, когда запрещенных слов не было, незначительна.

Дополнительные функции программы

Основная дополнительная функция программы состоит в иллюстрации построенного в процессе работы алгоритма графа. При построении графа нет существенной разницы в затратах по времени, но от количества вводимых данных зависит размер дуг и вершин. Таким образом, при введении слишком больших слов или большого количества вершин, изучение графа становится неудобным из-за размера текста.

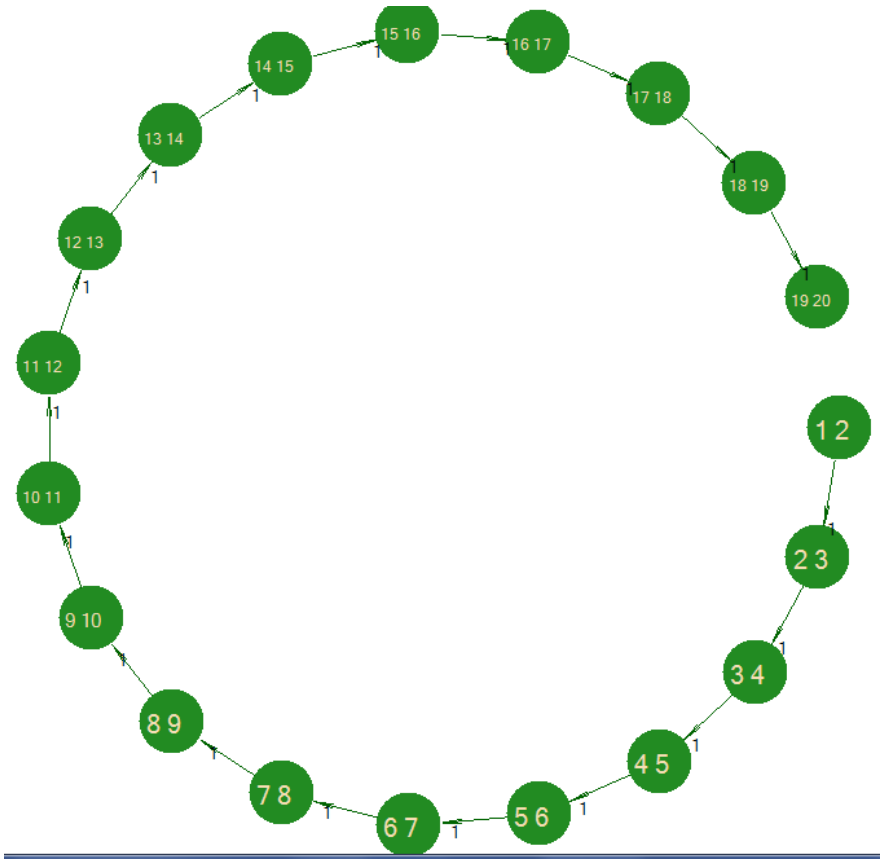

Рисунок 4

Оптимальным и наиболее наглядным может являться относительно небольшой граф, состоящий не более, чем из двадцати вершин. На рисунке 4 изображен один из критических вариантов такого графа.

Заключение

В рамках выпускной квалификационной работы требовалось создать программу, которая позволяла бы реконструировать слова на основе набора слов в гипотезе единичного сдвига. Для разработки программы были изучены существующие подходы и решения и проработан алгоритм, позволяющий эффективно решить поставленную задачу.

Разработанная программа позволяет реконструировать слова с учетом запрещенных подслов, сохранять, изменять и загружать данные. Кроме того, программа иллюстрирует созданный в процессе работы алгоритма ориентированный граф.

В целом, программа показала достаточно хорошие результаты. Время работы существенно менялось в зависимости от вводимых данных, но при сравнительно малом количестве вводимых слов (около 20), тем не менее оставалось достаточно малым, чтобы не быть замеченным пользователем.

Для использования программы при решении реальных задач могут быть проведены дополнительные доработки. Например, автоматизация работы системы для ввода данных без участия пользователя, автоматическое считывание или запись данных для удобства работы и увеличения производительности, или дальнейшая доработка под конкретно поставленные задачи.

Список использованных источников

1. **Красиков** И. В. Алгоритмы. Просто как дважды два / И. В. Красиков, И. Е. Красикова. – М. : Эксмо, 2007. – 256 с. – (Просто как дважды два).
2. **Головкин** М. В. Математические методы анализа электрофоретических картин расщепления ДНК / М. В. Головкин и др. – М. : Ки&М 2009 Т. 1 №3 с. 287-295 (анализ и моделирование сложных живых систем).
3. **Хопкрофт**, Джон, Э., **Мотвани**, Раджив, **Ульман**, Джеффри, Д. Введение в теорию автоматов, языков и вычислений / Хопкрофт Д. Э., Мотвани Р., Ульман Д. Д., 2-е изд. :Пер. с англ. – М. : Издательский дом «Вильямс», 2002. – 528 с. : ил. – Парал. тит. англ.
4. **Леонтьев** В. К. Задачи восстановления слов по фрагментам и их приложения / Леонтьев В. К. – М : Апрель-июнь 1995, Том 2, №2, с. 26-48 (Дискретный анализ и исследование операций).
5. **Шапорев** С. Д. Математическая логика. Курс лекций и практических занятий / Шапорев С. Д. – СПб. : БХВ-Петербург, 2005. – 416 с:ил.
6. **Зенкин** А. И., **Леонтьев** В. К. Об одной неклассической задаче распознавания / Зенкин А. И., Леонтьев В. К. – М. : Журнал вычислительной математики и математической физики, 1984, Т. 24, №6, с. 925-931.
7. **Пащенко** Р. Э., **Пащенко** Э. И. Формирование кодофазоманипулируемых фрактальных сигналов на основе последовательности Морса-Туэ / Пащенко Р. Э., Пащенко Э. И.; рецензент: д.т.н., проф. Карлов В. Д. – М. : Собрание научных трудов Харьковского университета Воздушных Сил, 2010, выпуск 3(25).
8. **Сметанин** Ю. Г. Реконструкция по частичным представлениям в комбинаторике слов / Сметанин Ю. Г. – М. : Дис. … д-ра физ-мат. Наук: Москва, 2003 184с. РГБ ОД
9. **Сметанин** Ю. Г., **Ульянов** М. В. Реконструкция слов по конечному мультимножеству подслов в гипотезе сдвига 1. / Сметанин Ю. Г., Ульянов М. В. – Томск : Кибернетика и системный анализ №1 2014 (в печати).
10. **Евдокимов А.А., Левин А. А.** Инструментарий графического исследования символьных последовательностей. / Евдокимов А. А., Левин А. А. – Новосибирск: Прикладная дискретная математика, 2008, выпуск №1 (Прикладная теория графов).
11. **Свами** М., **Тхуласираман** К. Графы, сети и алгоритмы / Свами М., Тхуласираман К.; под редакцией д-ра техн. наук Горбатова В. В. – М: «Мир», 1984г. - 443с
12. **Седжвик Р.** Фундаментальные алгоритмы на C++. Алгоритмы на графах: Пер. с англ. / Седжвик Р. – СПб: ООО «ДиаСофтЮП», 2002. – 496с.
13. **Швец** А. Н., Perl. Примеры программ / Швец А. Н.- М: Методическое пособие, 2008-2014. (школа №54 при механико-математическом факультете МГУ им. М. В. Ломоносова.

Приложения

Приложение А.

Техническое задание
RU.17701729.507600-01 ТЗ №37

Правительство Российской Федерации

Федеральное государственное автономное образовательное

учреждение высшего профессионального образования

"Национальный исследовательский университет

"Высшая школа экономики"

Отделение программной инженерии

Кафедра Управления разработкой программного обеспечения

«Программа реконструкции слов с запретами по мультимножеству подслов в гипотезе единичного сдвига»

Техническое задание
RU.17701729.507600-01 ТЗ №37

Листов 6

Руководитель:
Д.т.н., профессор
_____Ульянов М.В.

Исполнитель:
Студентка группы 472 ПИ
_____Стёпушкина Наталия

Москва 2014 г

УТВЕРЖДЕН

RU.17701729.507600-01 ТЗ №37

Программа реконструкции слов с запретами по мультимножеству подслов в гипотезе единичного сдвига

Техническое задание

RU.17701729.507600-01 ТЗ №37

Инв. № подл.	Подп. и	Взам. инв. №	Инв. № дубл.	Подп. и дата

Листов

2014

Введение

Наименование приложения:

Программа называется в данном документе «Программа реконструкции слов с запретами по мультимножеству подслов в гипотезе единичного сдвига» или упрощенно «WR». Аббревиатура расшифровывается как «Words Reconstruction».

Основания для разработки:

Приказ № 6.18.1-02/2911-13 от 29.11.13 «Об утверждении тем и руководителей выпускных квалификационных работ студентов отделения программной инженерии факультета бизнес-информатики».

Организация, утвердившая этот документ – НИУ ВШЭ, факультет бизнес-информатики, отделение программной инженерии.

Краткое описание приложения:

Программа позволяет пользователю реконструировать набор слов по задаваемым данным, представляющим собой неполную информацию о реконструируемом слове. Словом в данной программе называется любой набор символов над заданным алфавитом, образующих подслова, построенные в соответствии с определенными известными правилами:

- Все подслова, на основе которых реконструируется слово, принадлежат определенному алфавиту;
- Длина всех задаваемых пользователем подслова равна;
- Любое подслово встречается в реконструированном слове ровно один раз;
- Каждое из подслов получается из реконструируемого слова путем сдвига относительно него на один символ заданного алфавита;
- Реконструируемые слова не содержат запрещенных слов, также задаваемых пользователем.

Пользователь задает набор реконструирующих и набор запрещенных слов, на основе которых получает множество слов, удовлетворяющих приведенным условиям. Полученные результаты можно сохранять и загружать в программу по желанию пользователя.

Плановые сроки работы:

Начало работы – 15 ноября 2013, окончание работы – 30 мая 2014.

Предполагаемый дизайн программы:

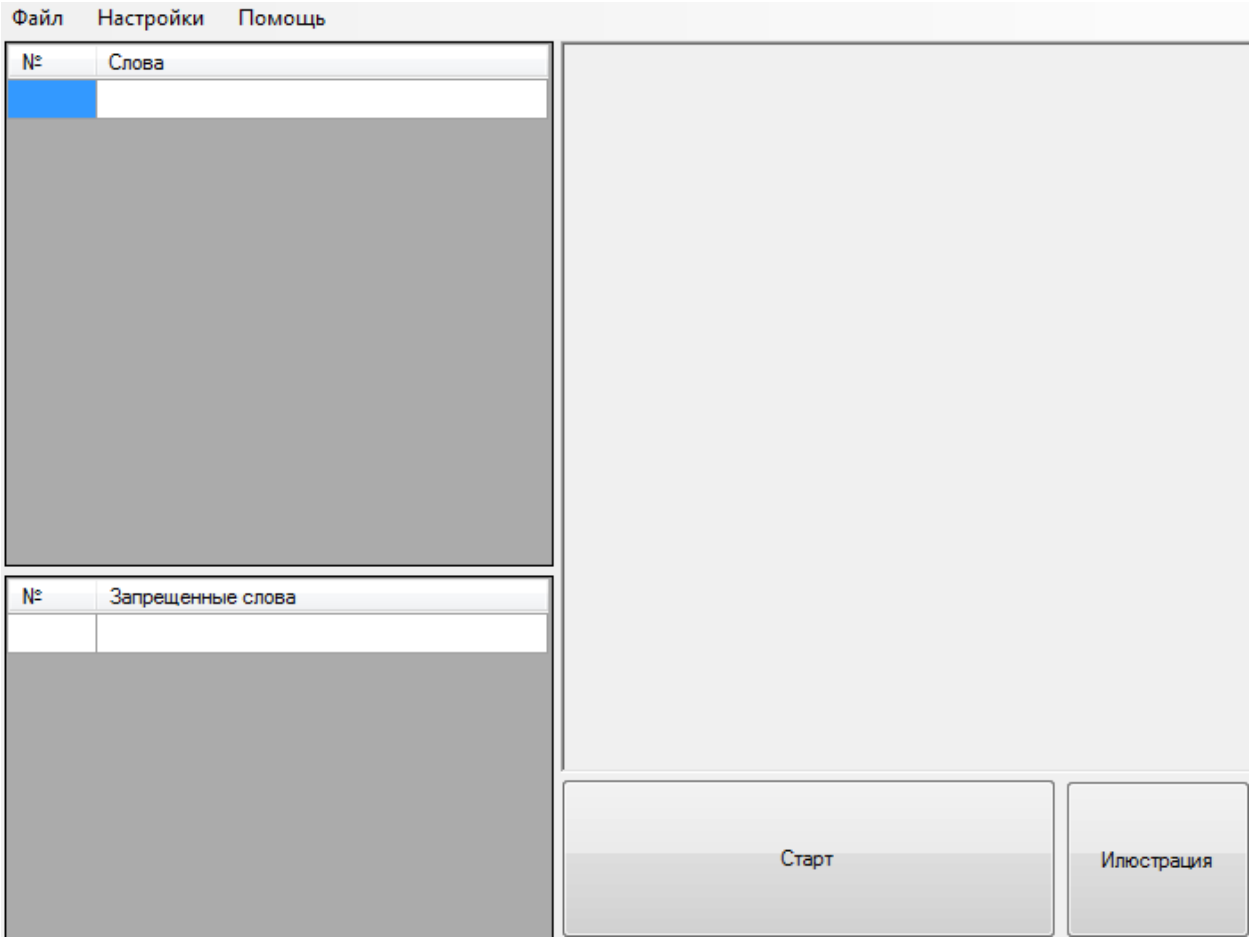

Рисунок 5

Назначение разработки

Функциональное назначение.

Программа предназначена для реконструкции слов на основе задаваемых пользователем данных. Основной целью является упрощение нахождения всех удовлетворяющих условиям слов.

Области применения программы:

Приложение возможно использовать по следующим направлениям:

- Как система поиска реконструируемых слов;

- В качестве прототипа для решения смежных с данной задач;
- Для изучения ориентированных графов, в которых ребра представлены заданными словами.

Требования к программе

Реализуемые функции:

- Создание и редактирование подслов;
- Создание и редактирование запрещенных слов;
- Реконструкция набора всех слов, удовлетворяющих условиям;
- Возможность сохранения и загрузки данных в программу;

Требование к входным данным:

- Ввод и редактирование подслов и запрещенных слов;
- Все подслова, на основе которых реконструируется слово, принадлежат определенному алфавиту;
- Длина всех задаваемых пользователем подслова равна;
- Запрещенные слова могут состоять из любого количества букв заданного пользователем алфавита а также посторонних символов;
- Любое подслово встречается в реконструированном слове ровно один раз;
- Каждое из подслов получается из реконструируемого слова путем сдвига относительно него на один символ заданного алфавита;
- Реконструируемые слова не содержат запрещенных слов, также задаваемых пользователем.
- Знак пустого символа считается переходом к новой букве слова над некоторым заданным алфавитом;
- Входные данные могут вводиться путем загрузки из выбранного пользователем файла.

Требования к выходным данным:

- Возможность сохранения данных;
- Вывод списка реконструированных слов после нажатия на соответствующую кнопку;

- В случае неверно введенных данных пользователю выдается соответствующее сообщение;
- Возможность просмотра краткой справки о программе с объяснением ее работы.

Требования к информационной и программной совместимости

На системе должен быть установлен .NET Framework 4.0, который поддерживается следующими платформами:

Microsoft® Windows® XP и более поздними версиями.

Требования к составу и параметрам технических средств

Таблица 1

Необходимый процессор	Рекомендуемый процессор	Необходимое ОЗУ	Рекомендуемое ОЗУ	Прочие требования
Pentium 90 MHz*	Pentium 90 MHz или с более высоким быстродействием	32 MB*	96 MB или больше	Видеокарта VGA, минимум 4 MB CD-ROM 4x, мышь, клавиатура, USB-порт.

*Или минимум, требуемый операционной системой, какой бы она ни была.

Стадии и этапы разработки

Таблица 2

1.	Изучение и утверждение темы выпускной квалификационной работы
2.	Разработка технического задания
3.	Разработка прототипа программы, выполняющего основные функции
4.	

	Разработка документации по выпускной квалификационной работе
5.	Согласование и утверждение прототипа для дальнейшей разработки
6.	Создание готовой и оформленной выпускной квалификационной работы

Приложение Б.

Пояснительная записка
RU.17701729.502150-01 81 №37

Правительство Российской Федерации

Федеральное государственное автономное образовательное

учреждение высшего профессионального образования

"Национальный исследовательский университет

"Высшая школа экономики"

Отделение программной инженерии

Кафедра Управления разработкой программного обеспечения

«Программа реконструкции слов с запретами по мультимножеству подслов в гипотезе единичного сдвига»

Пояснительная записка
RU.17701729.502150-01 81 №37

Листов 7

Руководитель:
Д.т.н., профессор
_____Ульянов М.В.

Исполнитель:
Студентка группы 472 ПИ
_____Стёпушкина Наталия

Москва 2014 г

УТВЕРЖДЕН
RU.17701729.502150-01 81 №37

Программа реконструкции слов с запретами по мультимножеству подслов в гипотезе единичного сдвига

Пояснительная записка

RU.17701729.502150-01 81 №37

Инв. № подл.	Подп. и	Взам. инв. №	Инв. № дубл.	Подп. и дата

Листов 7

2014

Общие сведения

Наименование программы

Полное наименование: Программа реконструкции слов с запретами по мультимножеству подслов в гипотезе единичного сдвига

Краткое наименование: WR

Основания для проведения работ

Приказ № 6.18.1-02/2911-13 от 29.11.13 «Об утверждении тем и руководителей выпускных квалификационных работ студентов отделения программной инженерии факультета бизнес-информатики».

Организация, утвердившая этот документ – НИУ ВШЭ, факультет бизнес-информатики, отделение программной инженерии.

Цели, назначение и область использования приложения

Программа предназначена для реконструкции слов. При этом приложение проверяет входные данные и на их основании получает набор выходных значений, которые выводит на экран.

Программа может использоваться в учебных целях или в качестве прототипа для решения конкретных задач, связанных с комбинаторикой и воссозданием данных по обрывочной информации.

Основные технические решения

Функции, выполняемые программой

- позволяет вводить, сохранять, изменять и загружать входные подслова и запрещенные слова;
- реконструирует все возможные по условию задачи слова и реализует их вывод на экран в специальную панель;
- позволяет получить иллюстрацию в виде ориентированного графа с маркированными вершинами.

.

Функциональная структура

На рисунке 6 представлена функциональная структура приложения.

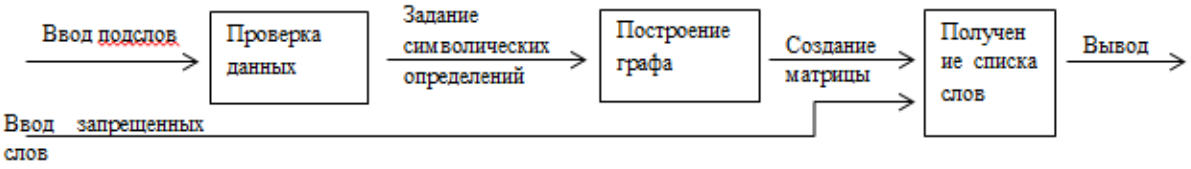

Рисунок 6

Алгоритм работы программы

Программа получает на вход набор подслов и запрещенных слов. Сначала она выполняет несколько простых проверок, как, например, проверка на количество букв во вводимом слове.

Далее, если введенные данные верны, программа преобразует каждое из слов в символический номер, что позволяет избежать случайных пересечений имен. Затем программа по полученному набору строит матрицу смежности графа, которую также использует при построении иллюстрации.

Затем происходит возведение матрицы смежности графа в некоторую степень, что позволяет получить последовательность символических номеров, описывающих конечное слово. На этом этапе производится проверка запрещенных слов и удаление слов, содержащих в себе запрещенные слова.

В конце происходит обратное преобразование символических имен в последовательность букв, образующих реконструируемое слово.

Описание и функциональное назначение классов и методов

Описание классов

Таблица 3

Название класса	**Функции**
Calculating	Выполнение основного алгоритма, подсчет и получение результата
Graphs	Рисование графа, расчет сопутствующих данных
MainForm	Получение входных данных, вызов функций, вывод на экран

Описание методов класса Calculating

Таблица 4

Название	Тип	Аргументы	Функции
Answer	Array<String^>^	Array<String^>^	Основная функция, производящая последовательный вызов остальных функций
Calc	Array<String^>^	Array<String^,2> Array<String^,2>	Умножение двух матриц
Collect	Array<String^>^	Array<String^>^	Проверка входных слов
Control	Array<String^>^	Array<String^>^	Создание матрицы смежности графа
Forbiddens	Array<String^>^	Array<String^>^ Array<String^>^	Работа с запрещенными словами
One_letter_word	Array<String^>^	Array<String^>^	Функция поиска слов для случая, когда слова состоят из 1 буквы
Recursive	void	Int Array<String^>^	Рекурсивная функция для поиска слов из 1 буквы
vertex	Array<String^>^	Array<String^>^	Присвоение вершин и переходов графу

Описание методов класса Graphs

Таблица 5

Название	Тип	Аргументы	Функции

DrawArrow	void	Graphics^ Pen^ Float Float Float Float Float	Рисовка стрелок – дуг графа
Graphs		void	Инициализация компонентов
Graphs_Activated		System::Object^ System::EventArgs^	Перерисовка на панели
Graphs_Paint	void	System::Object^ System::Windows::Forms::PaintEventArgs^	Рисовка графа
InitializeComponent	void	void	Инициализация компонентов
Panel1_Paint	void	System::Object^ System::Windows::Forms::PaintEventArgs^	Перерисовка на панели

Описание методов класса MainForm

Таблица 6

Название	Тип	Аргументы	Функции
Button1_Click	void	System::Object^ System::EventArgs^	Кнопка «Старт» - вызов основных функций алгоритма
Button2_Click	void	System::Object^ System::EventArgs^	Вызов окна, в котором будет рисоваться граф
dataGridView1_CellBeginEdit	void	System::Object^ DataGridViewCellCancelEventArgs^	Нумеровка первой ячейки каждой строки таблицы запрещенных

			слов
dataGridView1_CellValueChanged	void	System::Object^ DataGridViewCellEventArgs^	Нумеровка первой ячейки каждой строки таблицы запрещенных слов при изменении ячейки
dataGridView3_CellBeginEdit	void	System::Object^ DataGridViewCellCancelEventArgs^	Нумеровка первой ячейки каждой строки таблицы Подслов
dataGridView3_CellValueChanged	void	System::Object^ DataGridViewCellEventArgs^	Нумеровка первой ячейки каждой строки таблицы подслов при изменении ячейки
InitializeComponent	void	Void	Инициализация компонентов
MainForm		Void	Инициализация компонентов
справкаToolStripMenuItem_Click	void	System::Object^ System::EventArgs^	Вызов справки
таблицаЗапрещенныхСловToolStripMenuItem_Click	void	System::Object^ System::EventArgs^	Сохранение данных из таблицы запрещенных слов в файл
таблицаЗапрещенныхСловToolStripMenuItem1_Click	void	System::Object^ System::EventArgs^	Удаление элементов из таблицы

			запрещенных слов
таблицаЗапрещенныхСловToolStripMenuItem2_Click	void	System::Object^ System::EventArgs^	Загрузка данных из файла в таблицу запрещенных слов
таблицаСловToolStripMenuItem_Click	void	System::Object^ System::EventArgs^	Сохранение данных из таблицы подслов в файл
таблицаСловToolStripMenuItem1_Click	void	System::Object^ System::EventArgs^	Удаление элементов из таблицы запрещенных слов
таблицаСловToolStripMenuItem2_Click	void	System::Object^ System::EventArgs^	Загрузка данных из файла в таблицу слов

Описание классов и методов дано в таблицах 3, 4, 5 и 6.

Методы и средства разработки

Для создания приложения используется программное обеспечение Visual C++ в среде разработки Microsoft Visual Studio 2013.

Приложение В.

Руководство пользователя
RU.17701729.502150-01 34 №37

Правительство Российской Федерации

Федеральное государственное автономное образовательное

учреждение высшего профессионального образования

"Национальный исследовательский университет

"Высшая школа экономики"

Отделение программной инженерии

Кафедра Управления разработкой программного обеспечения

«Программа реконструкции слов с запретами по мультимножеству подслов в гипотезе единичного сдвига»

Руководство пользователя
RU.17701729.502150-01 №37

Листов 12

Руководитель:
Д.т.н., профессор
_____Ульянов М.В.

Исполнитель:
Студентка группы 472 ПИ
_____Стёпушкина Наталия

Москва 2014 г

УТВЕРЖДЕН
RU.17701729.502150-01 34 №37

Программа реконструкции слов с запретами по мультимножеству подслов в гипотезе единичного сдвига

Руководство пользователя

RU.17701729.502150-01 34 №37

Листов 12

Инв. № подл.	Подп. и	Взам. инв. №	Инв. № дубл.	Подп. и дата

2014

Введение

Область применения

Программа может использоваться как при решении прикладных задач, так и в качестве учебного пособия для изучения теории графов или как прототип для разработки более узкоспециализированных задач.

Краткое описание возможностей

Программа представляет собой приложение, осуществляющее реконструкцию слов по введенным подсловам. Помимо основной задачи, система позволяет проверять наличие запрещенных подслов, строить граф по заданным начальным данным и отображать его на экране. Для удобства пользователя, также в приложение встроены функции загрузки, сохранения и изменения данных.

Уровень подготовки пользователя

Для работы с программой допускается пользователь с уровнем квалификации «уверенный пользователь ПК» или выше. Для комфортного взаимодействия с системой необходимо знать доступные команды приложения.

Подготовка к работе

Программное обеспечение, необходимое для работы

Для правильного функционирования приложения необходимо установленное программное обеспечение Microsoft Visual Studio 2013 для операционной системы Windows.

Порядок загрузки программы

Приложение запускается с помощью файла «WR.exe», расположенного в основной папке. Для комфортной работы возможна необходимость извлечь папку «WR» из архива или перенести ее на рабочий компьютер. В результате должно появиться окно, изображенное на рисунке 7.

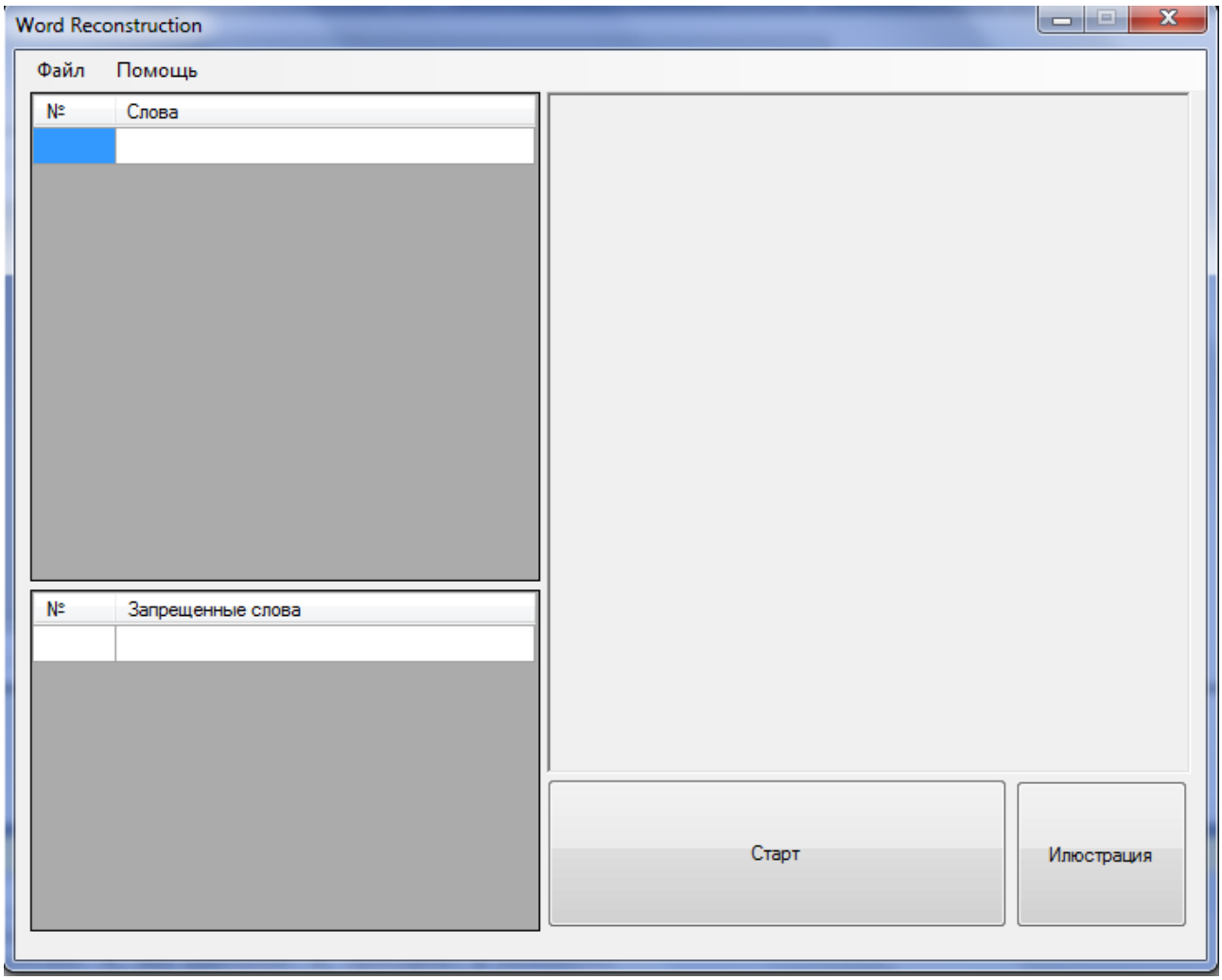

Рисунок 7

Описание операций

Ввод и изменение данных

Для работы приложения необходимо подавать на вход данные.

Для ввода или изменения подслов необходимо нажать мышкой в соответствующую ячейку в правом столбике таблицы, расположенной в верхнем левом углу. На рисунке 8 показан внешний вид программы с несколькими введенными значениями подслов.

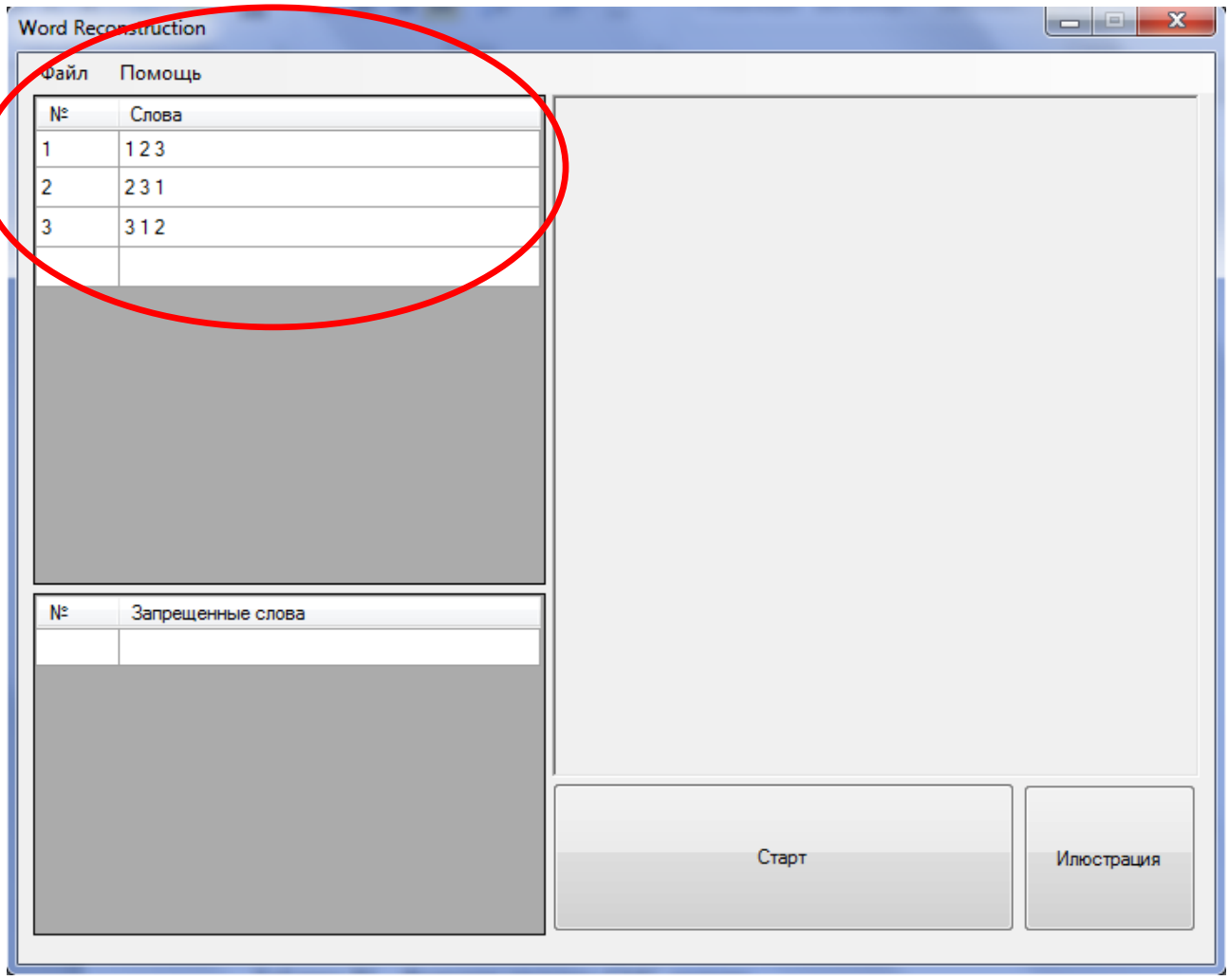

Рисунок 8

Каждое новое подслово располагается на новой строке, появляющейся автоматически. Индексы подслов также задаются автоматически. Буквы в подсловах разделяются пробелами, ввод производится пользователем с клавиатуры или с помощью загрузки из файла.

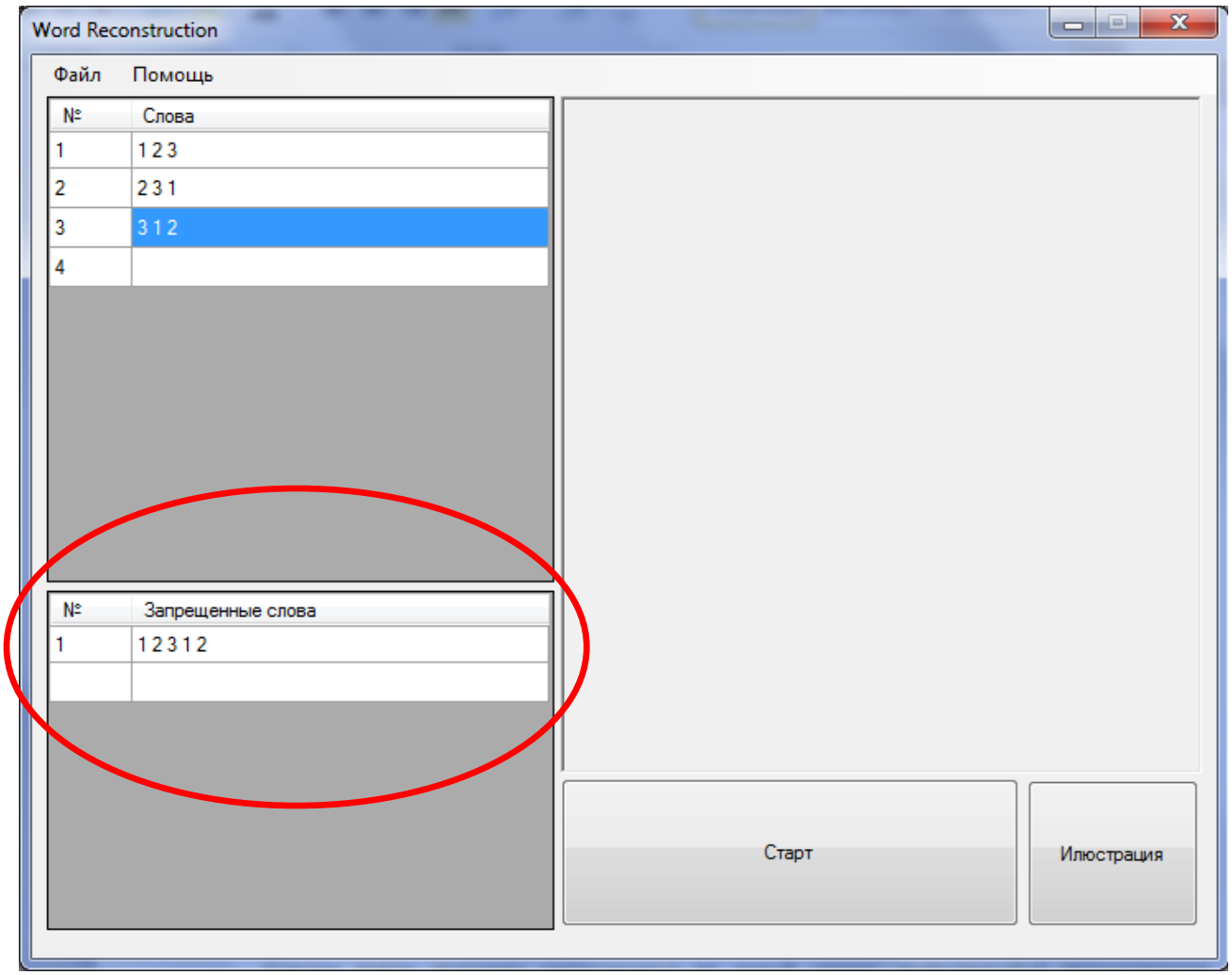

Рисунок 9

На рисунке 9 показан внешний вид программы после ввода одного запрещенного слова. Они располагаются в нижней таблице в правом столбце и вводятся так же, как и подслова. При этом буквы в запрещенных словах отделяются друг от друга пробелами, а каждое новое слово вписывается в новую ячейку, появляющуюся автоматически.

Для изменения данных в обеих таблицах необходимо нажать мышкой на той ячейке, которую требуется изменить, после чего просто ввести с клавиатуры новые данные.

Загрузка и сохранение данных

Для загрузки, сохранения или обновления списка подслов и запрещенных слов необходимо выбрать пункт «Файл» в верхней части панели, как изображено на рисунке 10.

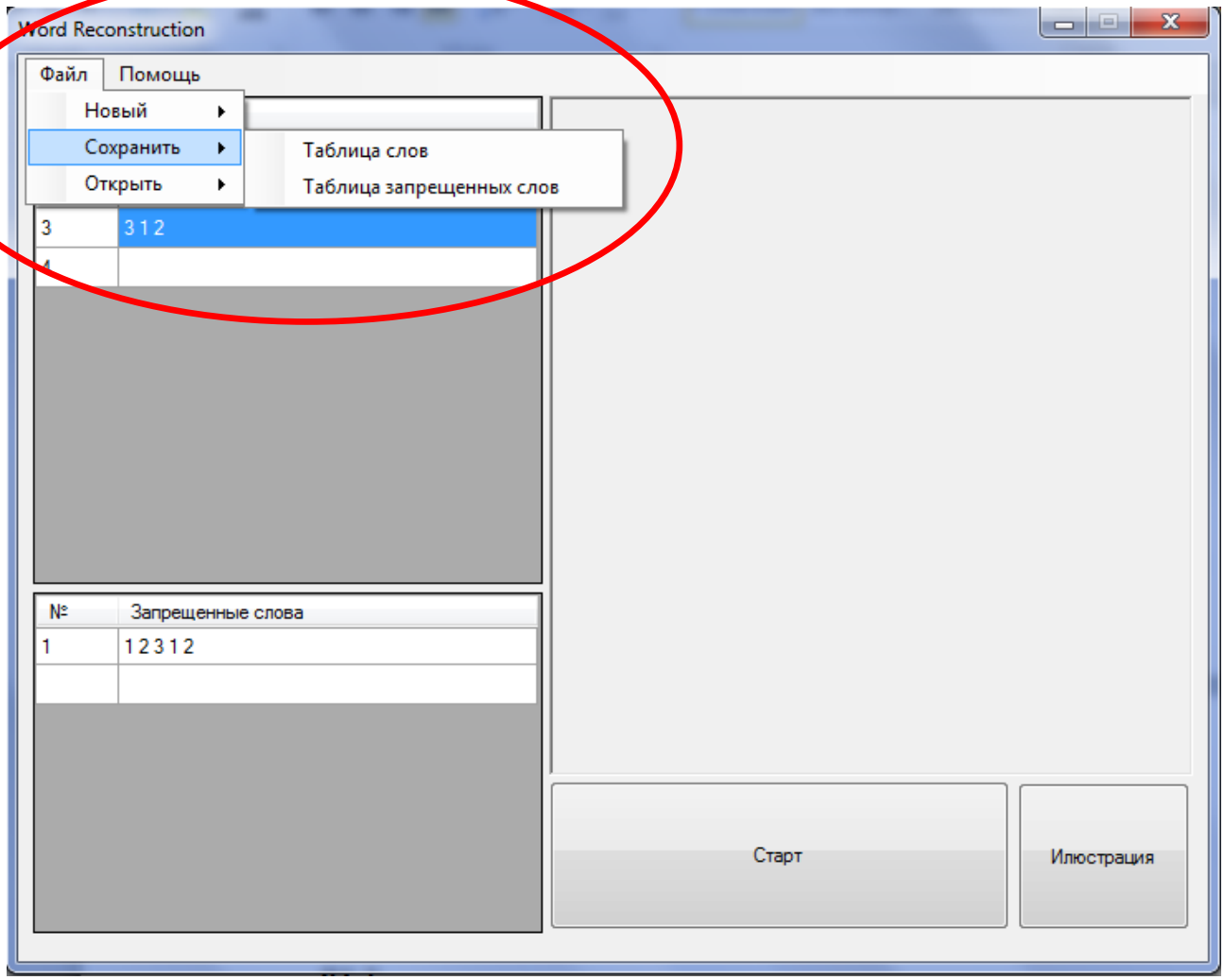

Рисунок 10

Появляется окно с тремя вариантами: «Новый», «Сохранить» и «Открыть». В каждом из пунктов есть два подпункта – «Таблица слов» и «Таблица запрещенных слов». Пункт меню «Новый» обнуляет значения соответствующей таблицы, выбранной пользователем с помощью подпунктов, «Сохранить» - производит сохранение данных из выбранной таблицы в указанный пользователем файл, а «Открыть» загружает данные из файла в соответствующую таблицу.

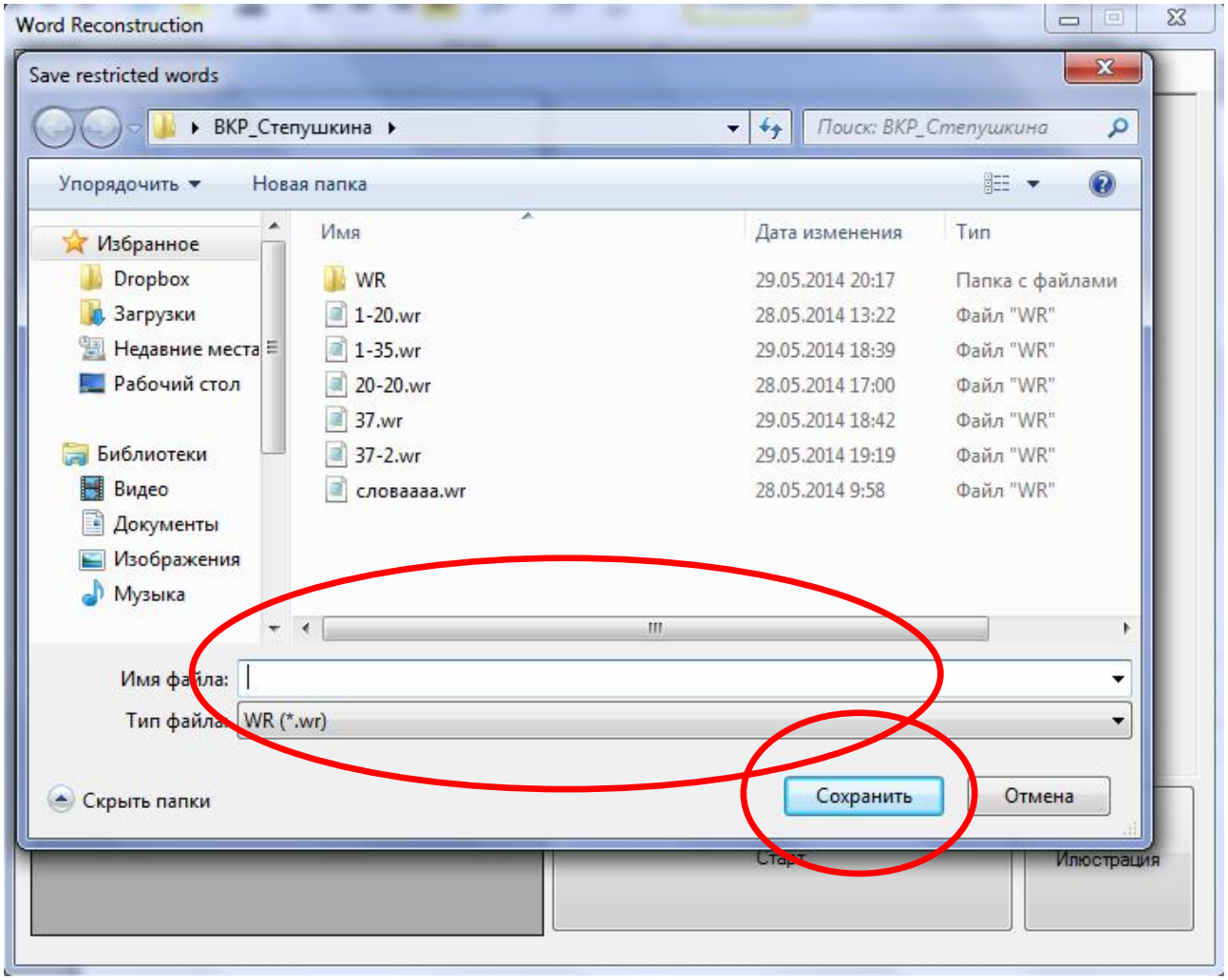

Рисунок 11

При выборе пункта меню «Сохранить» открывается форма, как показано на рисунке 11. Для сохранения данных достаточно написать желаемое имя файла или выбрать файл для замены из предложенного списка и нажать кнопку «Сохранить».

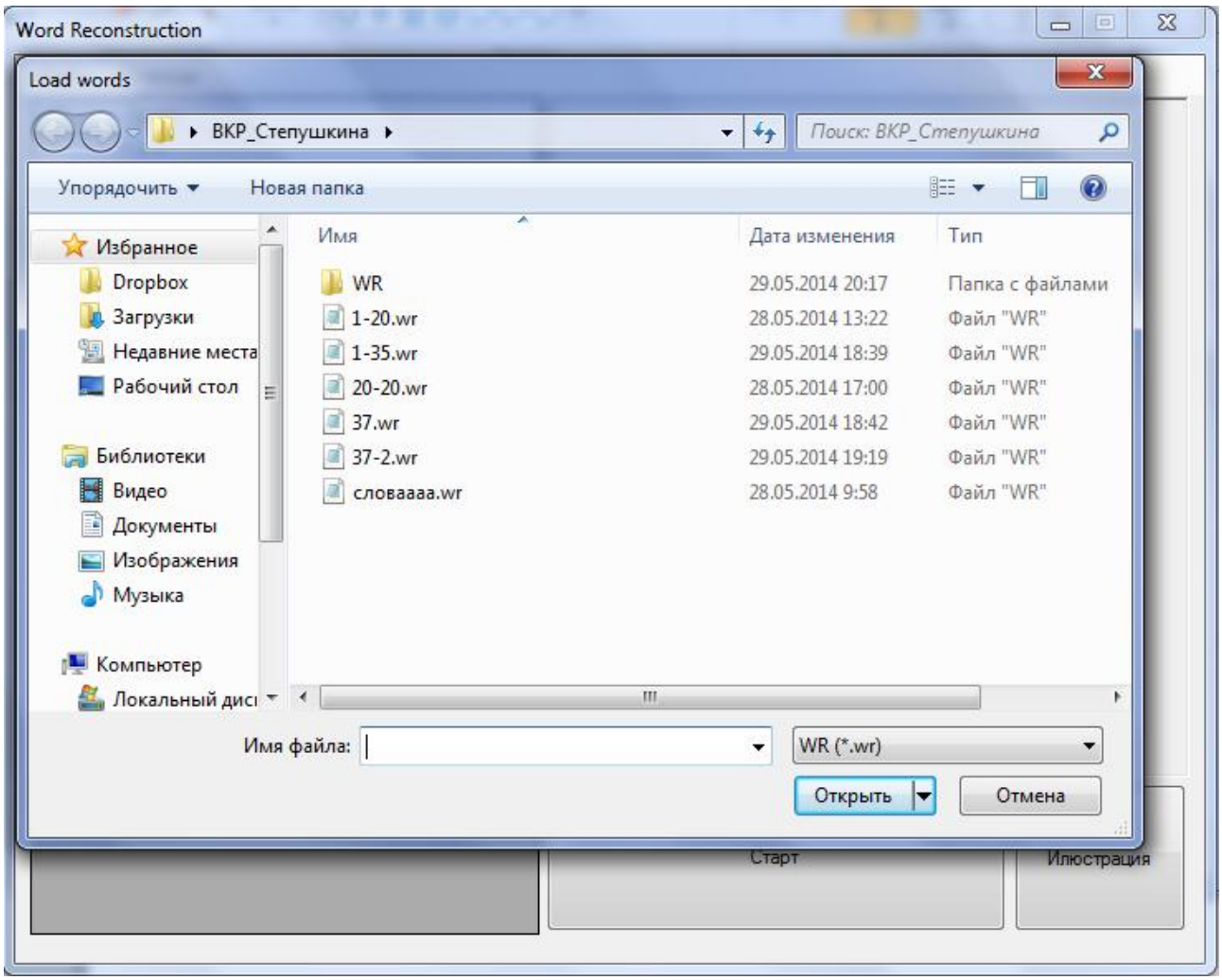

Рисунок 12

При выборе пункта меню «Открыть» открывается форма, изображенная на рисунке 12. Пользователь выбирает необходимый файл и нажимает кнопку «открыть», после чего данные выгружаются в соответствующую таблицу в программе.

Работа с приложением

Работа алгоритма

Для работы с программой необходимо сначала ввести данные в таблицу подслов и таблицу запрещенных слов. После этого пользователь нажимает на кнопку «Старт», расположенную в нижней части приложения и помеченную на рисунке 13.

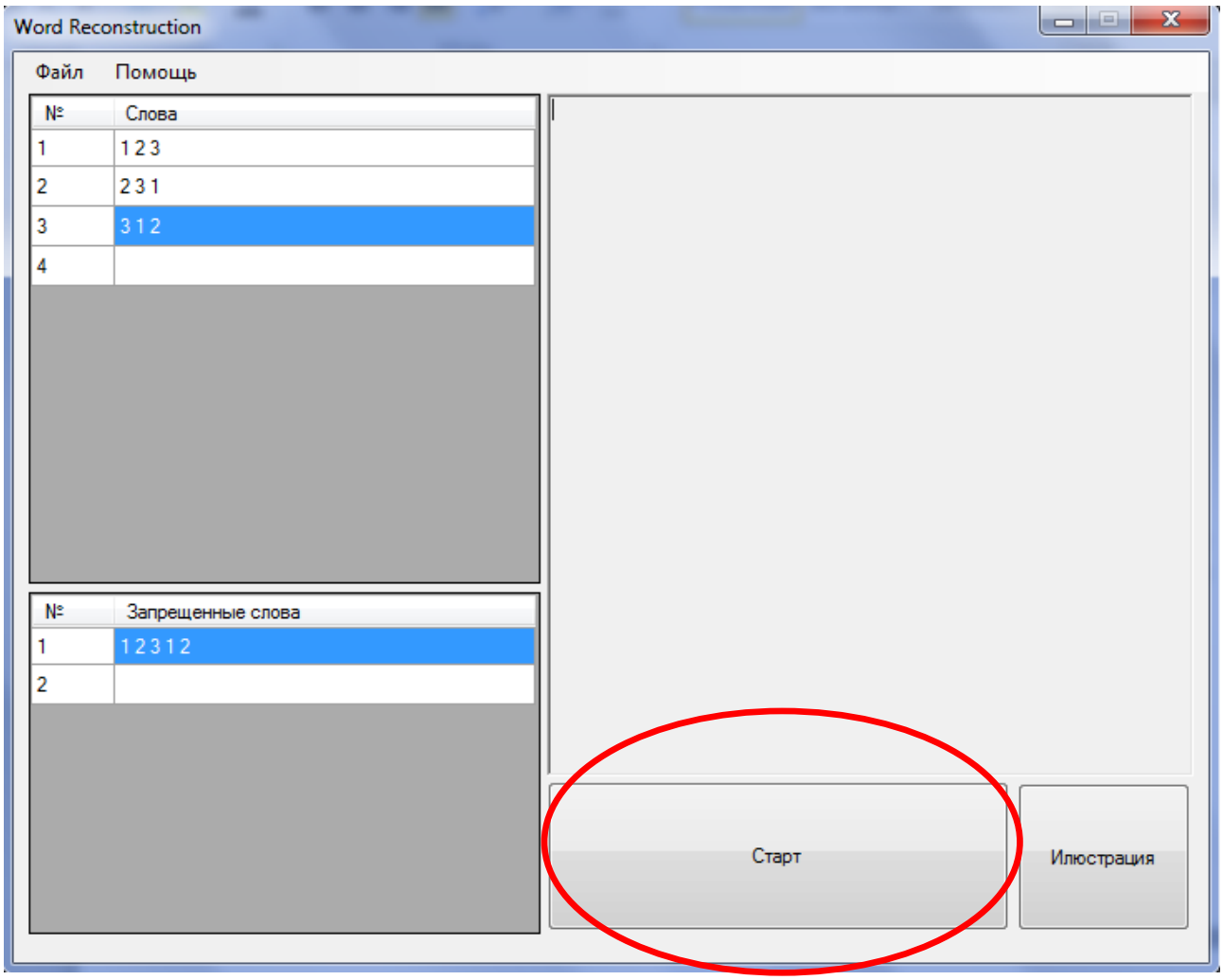

Рисунок 13

После выполнения алгоритма программа записывает данные на панель в верхнем правом углу (рисунок 14).

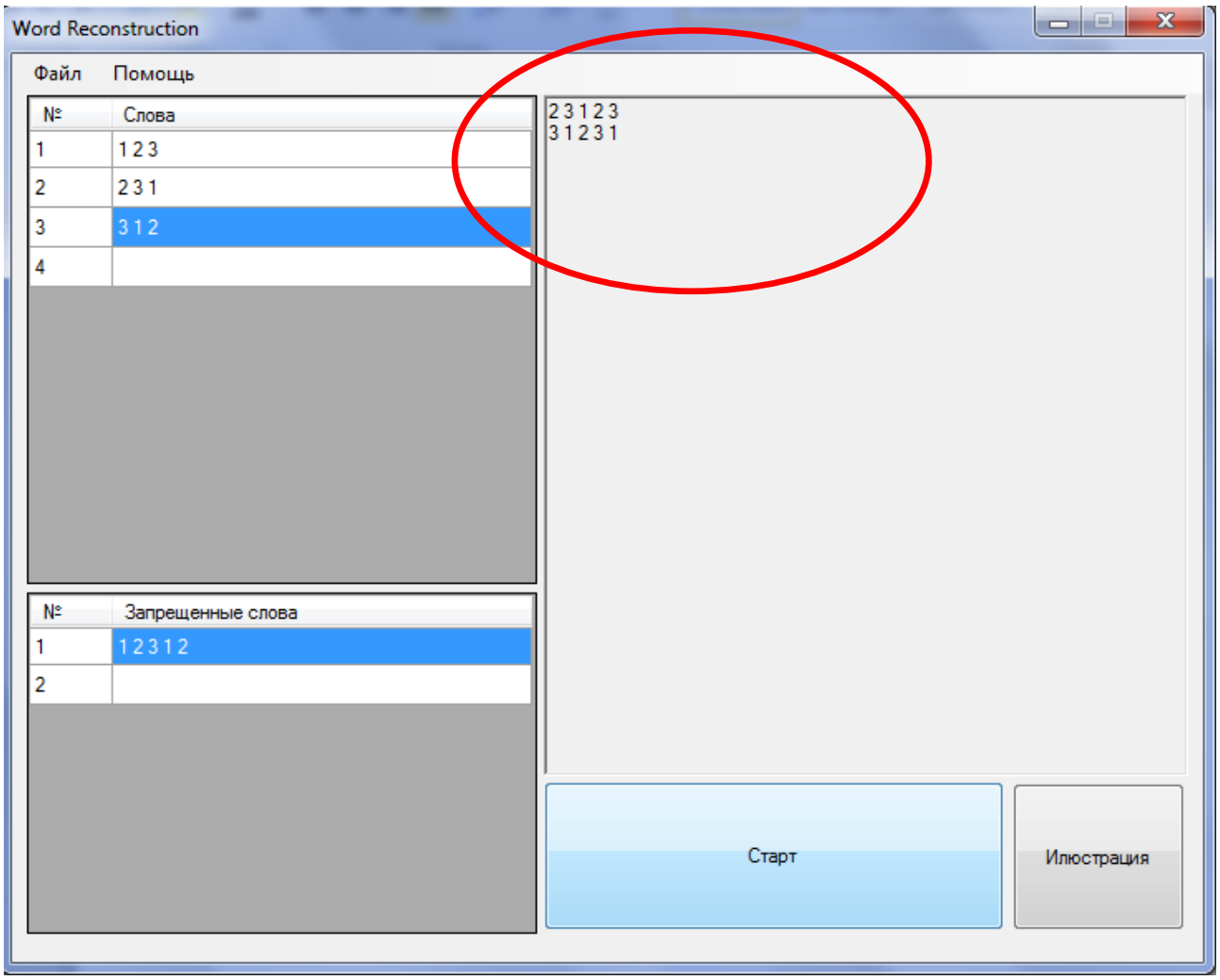

Рисунок 14

В случае, если данные были введены неверно, или список запрещенных слов составлен так, что невозможно реконструировать ни одно слово, программа выдает соответствующее сообщение (рисунок 15).

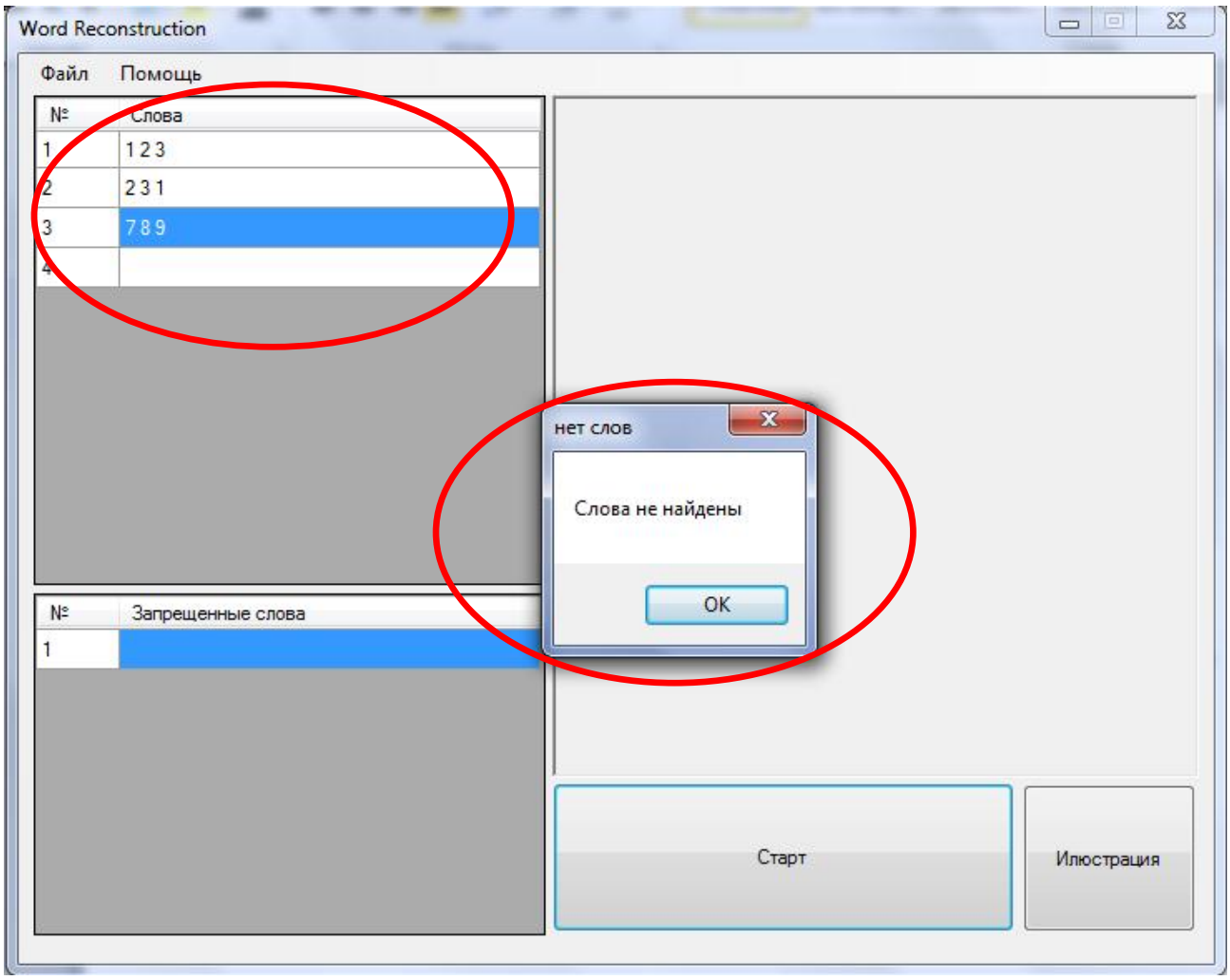

Рисунок 15

Иллюстрация графа

Дополнительной функцией программы для комфортной работы пользователя является построение графа по исходным данным для большей наглядности и удобства. Для построения графа необходимо нажать на кнопку «Иллюстрация» в правом нижнем углу приложения. Пример работы изображен на рисунке 16.

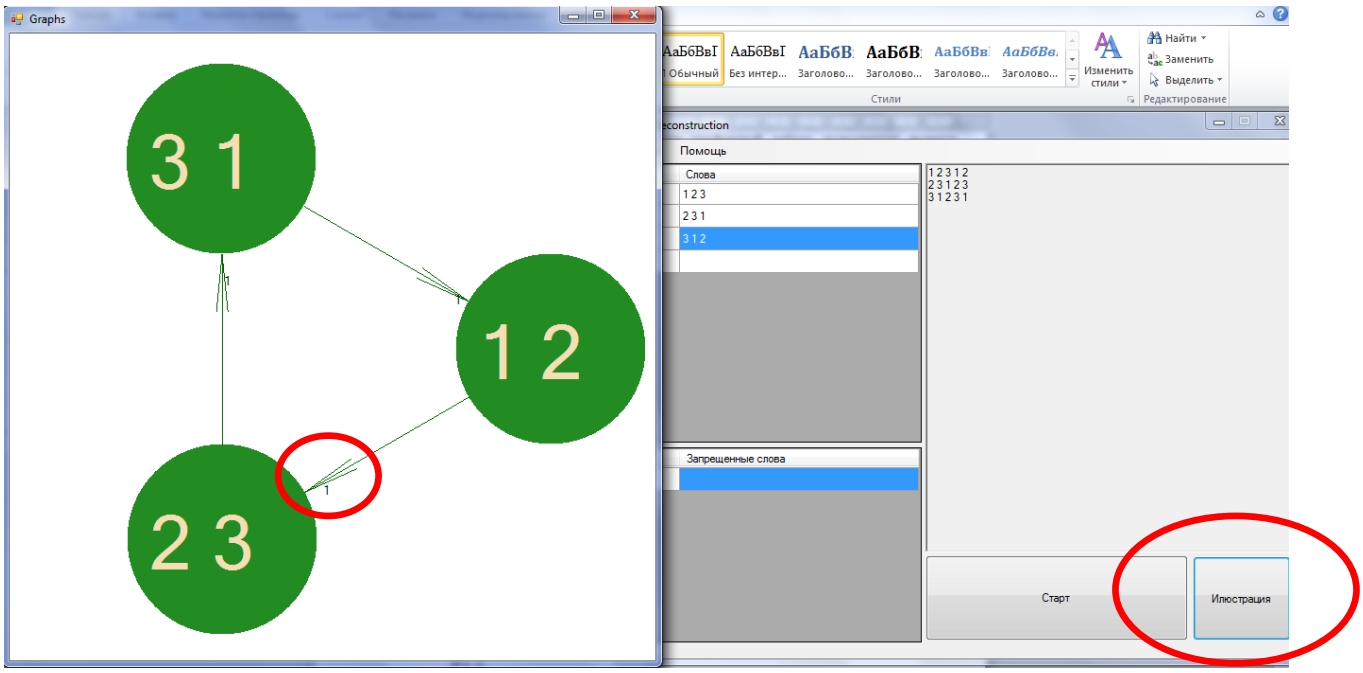

Рисунок 16

Иллюстрация показывает вершины графа и переходы между ними. Стрелки, изображенные на рисунке, демонстрируют направление дуг графа, а цифры возле каждой стрелки – кратность дуги.

Ошибки

При вводе неверных данных программа может выдать одно из сообщений, означающих ошибку. Возможные сообщения и ситуации, в которых они выводятся, приведены в таблице 7.

Таблица 7

Сообщение	**Ситуация**
Все слова запрещены	Невозможно реконструировать слово с заданными запрещенными словами
Слова не найдены	Заданные подслова не реконструируют ни одно слово
Неверные данные	Длина различных подслов не совпадает
Данные не введены	Нажата кнопка «Старт», когда не введено ни одно подслово

Приложение Г.

Программа и методика испытаний
RU.17701729.502150-01 51 №37

Правительство Российской Федерации

Федеральное государственное автономное образовательное

учреждение высшего профессионального образования

"Национальный исследовательский университет

"Высшая школа экономики"

Отделение программной инженерии

Кафедра Управления разработкой программного обеспечения

«Программа реконструкции слов с запретами по мультимножеству подслов в гипотезе единичного сдвига»

Программа и методика испытаний

RU.17701729.502150-01 51 №37

Листов 6

Руководитель:
Д.т.н., профессор
_____Ульянов М.В.

Исполнитель:
Студентка группы 472 ПИ
_____Стёпушкина Наталия

Москва 2014 г

УТВЕРЖДЕН
RU.17701729.502150-01 51 №37

Программа реконструкции слов с запретами по мультимножеству подслов в гипотезе единичного сдвига

Программа и методика испытаний

RU.17701729.502150-01 51 №37

Инв. № подл.	Подп. и	Взам. инв. №	Инв. № дубл.	Подп. и дата

Листов 6

2014

Объект испытаний

Полное наименование приложения

Программа реконструкции слов с запретами по мультимножеству подслов в гипотезе единичного сдвига.

Обозначение: WR

Цель испытаний

Целью проведения испытаний является:

- проверка работоспособности системы;
- соответствие выполняемых функций поставленным задачам;
- устойчивость к возможным ошибкам.

Объем испытаний

В таблице 8 указаны проводимые необходимые испытания.

Таблица 8

Требование	Наименование испытания
Ввод и изменение данных	Проверка ввода данных в таблицу подслов и таблицу запрещенных слов, проверка изменения данных в таблице подслов и таблице запрещенных слов, удаление данных
Иллюстрация	Проверка отображения графа по введенным данным
Работа с файлами	Проверка сохранения данных в файл из таблицы запрещенных слов и таблицы подслов, загрузки данных из таблиц в файл
Работа программы	Проверка верности выводимых на

	экран данных, устойчивость к ошибкам

Условия и порядок проведения испытаний

При проведения испытаний соблюдается следующий порядок действий:

1) Проведение испытаний в соответствии с предполагаемыми к функциям требованиями
2) Оценка степени соответствия требованиям
3) Устранение неполадок

Методы испытаний

Проверка ввода и изменения данных

Открыть программу, вводить и изменять данные в таблице запрещенных слов и в таблице подслов (рис 17). Программа должна изменять данные в соответствии с введенными с клавиатуры данными без появления каких-либо сообщений.

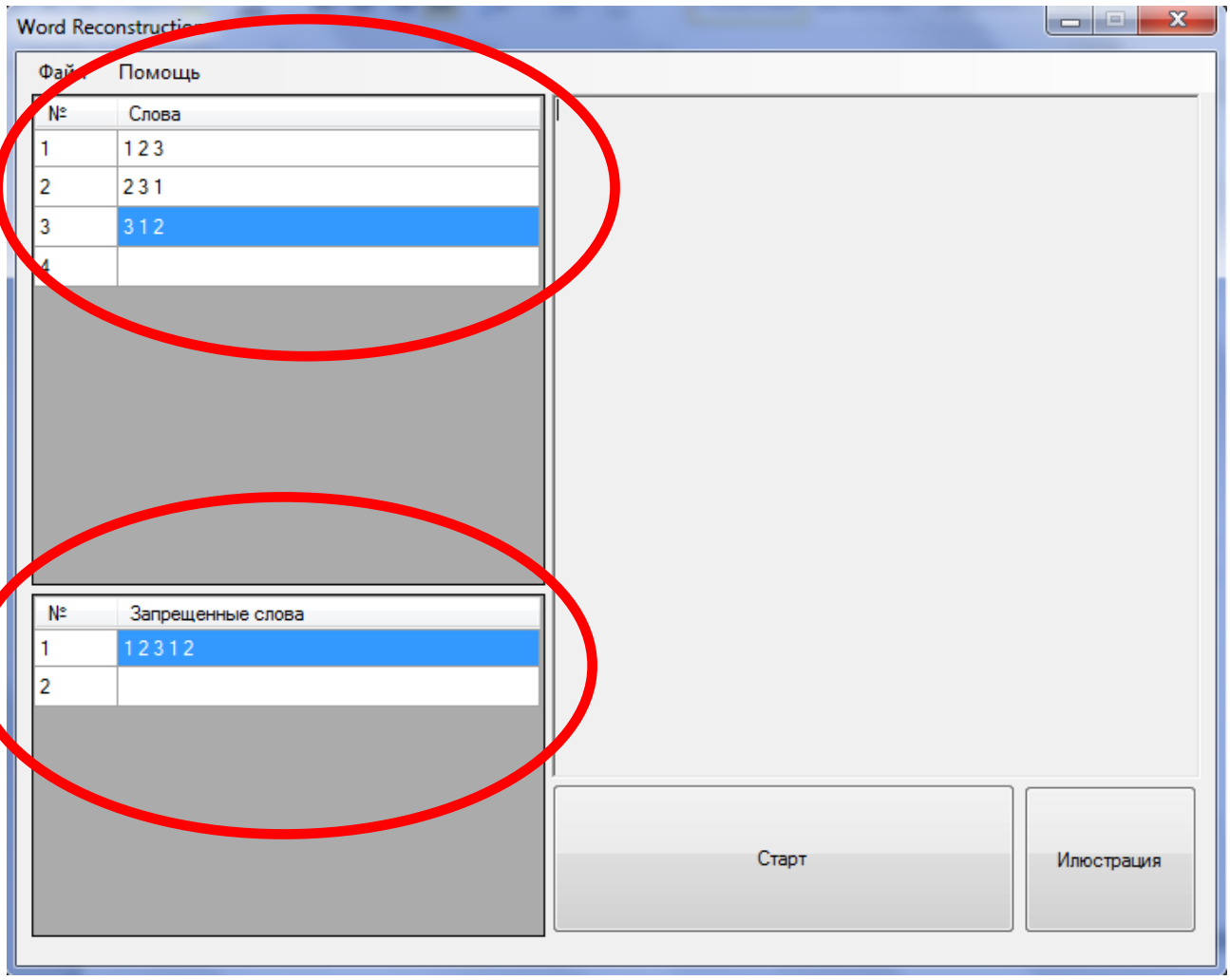

Рисунок 17

Проверка сохранения данных

Для проверки сохранения данных выполняются следующие действия:

1) Вводится текст в таблицу подслов
2) Выбирается пункт «Файл» - «Сохранить», вводится новое имя файла и нажимается кнопка «сохранить»
3) Проверяется содержимое файла: оно должно соответствовать данным, находящимся в таблице подслов

Далее такая же проверка производится для таблицы запрещенных слов.

Такая же проверка проводится для обеих таблиц в случае, когда не введены никакие данные. В этом случае записанный файл должен быть пустым.

1) Проверка перезаписи данных из таблицы:
2) Вводится текст в таблицу подслов
3) Выбирается пункт «Файл» - «Сохранить», выбирается уже существующий файл и нажимается кнопка «сохранить»
4) Проверяется содержимое файла: оно должно соответствовать данным, находящимся в таблице подслов

Такие же действия производятся для таблицы запрещенных слов. Затем такая же проверка проводится в случае с пустыми таблицами.

Проверка загрузки данных

Для проверки загрузки данных выполняются следующие действия:

1) Выбирается пункт «Файл» - «Открыть» - «Таблица слов», выбирается существующий файл и нажимается кнопка «открыть»
2) Проверяется содержимое таблицы подслов: оно должно соответствовать данным, находящимся в этом файле.
3) В таблицу подслов вводятся данные, после чего повторяется пункт 1)
4) Проверяется содержимое таблицы подслов: данные должны были измениться в соответствии с выбранным файлом.

Такая же операция проводится для таблицы запрещенных слов.

Проверка обновления таблиц

Для проверки обновления таблицы подслов выполняются следующие действия:

1) Выбирается пункт «Файл» - «Новый» - «Таблица слов»
2) Проверяется содержимое таблицы подслов: оно должно содержать единственную ячейку. Эта ячейка должна быть пустой.
3) В таблицу подслов вводятся данные, после чего повторяется пункт 1)
4) Проверяется содержимое таблицы подслов: данные должны были обнулиться, оставляется единственная пустая ячейка
5) Производится загрузка файла в таблицу подслов, после чего повторяется пункт 1)

6) Данные в таблице подслов должны обнулиться, в таблице остается единственная пустая ячейка.

Те же действия производятся для таблицы запрещенных слов.

Проверка иллюстрации

Для проверки работы иллюстраций производятся следующие действия:

1) Вводятся верные данные и нажимается кнопка «иллюстрация»
2) Вводятся неверные данные и нажимается кнопка «иллюстрация»
3) В обоих случаях должен выводиться верный граф или несколько графов

Проверка работы программы

Для проверки работы программы выполняются следующие действия:

1) Без ввода данных нажимается кнопка «Старт». Программа должна выдать одно из сообщений, описанных в таблице 7.
2) В таблицу слов вводятся верные данные (см. Рисунок 18), нажимается кнопка «Старт». Программа должна вывести на экран верный ответ.

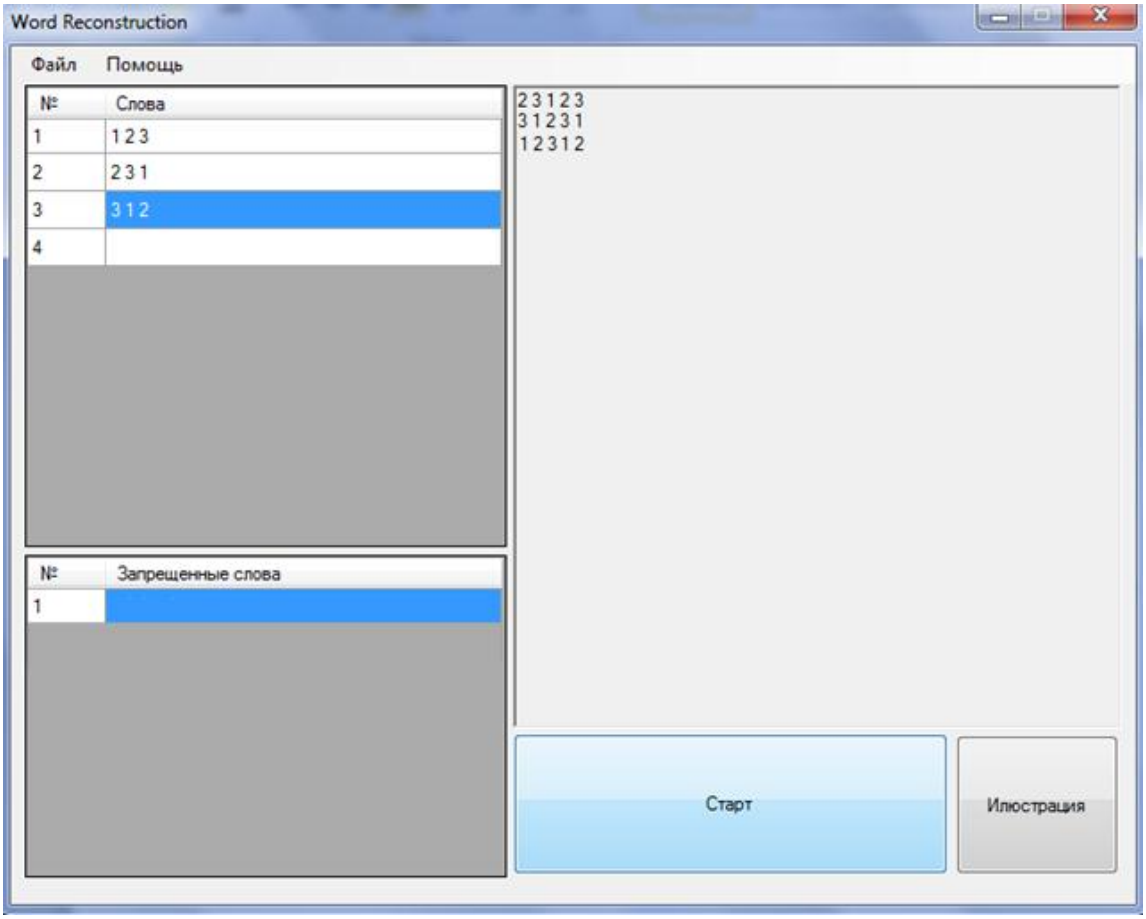

Рисунок 18

3) В таблицу подслов вводятся неверные данные. Программа должна выдать верное окно сообщения, как написано в таблице 7.
4) В таблицу подслов вводится набор различных верных значений, начиная от одного слова и выше. Программа должна каждый раз выдавать верные значения.
5) В таблицу подслов вводятся верные значения, в таблицу запрещенных слов вводятся такие значения, чтобы реконструировались все слова из таблицы подслов. Программа должна выдавать верное значение.
6) В таблицу подслов вводятся верные значения, в таблицу запрещенных слов вводятся такие значения, чтобы реконструировалась только часть слов. Программа должна выдавать верный набор слов.
7) В таблицу подслов вводятся верные значения, в таблицу запрещенных слов вводятся такие значения, чтобы не реконструировалось ни одно слово. Программа должна выдавать сообщение из таблицы 7.
8) В таблицу подслов вводятся неверные значения, в таблицу запрещенных слов вводятся любые значения. Программа должна выдавать сообщение из таблицы 7.
9) В таблицу подслов вводятся слова, состоящие из одной буквы. Программа должна выдавать верный набор слов.

Вывод

Программа прошла запланированные испытания, показав ожидаемые результаты.

Приложение Д.

Текст программы
RU.17701729.502150-01 12 №37

Правительство Российской Федерации

Федеральное государственное автономное образовательное

учреждение высшего профессионального образования

"Национальный исследовательский университет

"Высшая школа экономики"

Отделение программной инженерии

Кафедра Управления разработкой программного обеспечения

«Программа реконструкции слов с запретами по мультимножеству подслов в гипотезе единичного сдвига»

Текст программы

RU.17701729.502150-01 12 №37

Листов 2

Руководитель:
Д.т.н., профессор
_____Ульянов М.В.

Исполнитель:
Студентка группы 472 ПИ
_____Стёпушкина Наталия

Москва 2014 г

УТВЕРЖДЕН
RU.17701729.502150-01 12 №37

Программа реконструкции слов с запретами по мультимножеству подслов в гипотезе единичного сдвига

Текст программы

RU.17701729.502150-01 12 №37

Листов 2

Инв. № подл.	Подп. и	Взам. инв. №	Инв. № дубл.	Подп. и дата

2014

Текст программы находится в Приложении Д к отчетной документации в файле «Текст программы_Стёпушкина» на сайте lms.