МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
«МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ
(национальный исследовательский университет)» (МАИ)

Факультет №8
Прикладная математика и физика

Распространяется: *на правах рукописи.*

ОТЧЕТ
о дипломной работе
по теме:

Распределенное программно-информационное обеспечение статистической модели перевода естественных языков

Руководитель работы: *Е. С. Гаврилов*
Консультант по специальной части: *О. И. Денисова*
Исполнитель студент: *И. К. Никитин*

Москва 2012 г.

ОТ АВТОРА

Текущая сборка документа: 12 января 2012 г. 11:10

Техника в жизни человека играет значительную роль и машинный перевод не стал исключением. для облегчения и упрощения перевода была сделана попытка разработать распределенную статистическую систему машинного перевода.

Работа сверстана с использованием XeLaTeX. Документ оформлен по требованиям Московского Авиационного Института, не всегда эти требования соответствуют принятым государственным или мировым стандартам, да и здравому смыслу. Так что, уважаемые читатели, — не взыщите.

РЕФЕРАТ

Дипломная работа содержит 104 страницы, 13 рисунков, 21 таблицу, 4 приложения. Список использованных источников содержит 63 позиции.

Ключевые слова:
N-ГРАММЫ, ДЕКОДИРОВАНИЕ, ДЕРЕВО СУПЕРВИЗИИ, ЖАДНЫЙ ИНКРЕМЕНТНЫЙ ПОИСК, МАШИННОЕ ОБУЧЕНИЕ, МОДЕЛЬ ПЕРЕВОДА, МОДЕЛЬ ЯЗЫКА, НАУЧНЫЙ ТЕКСТ, ОСОБЕННОСТИ НАУЧНО-ТЕХНИЧЕСКОЙ ЛИТЕРАТУРЫ, ОТКРЫТАЯ ТЕЛЕКОММУНИКАЦИОННАЯ ПЛАТФОРМА, ПАРАЛЛЕЛЬНЫЙ КОРПУС, ПЕРЕВОД, ПОДХОДЫ К МАШИННОМУ ПЕРЕВОДУ, ПОИСК ПО ПЕРВОМУ НАИЛУЧШЕМУ СОВПАДЕНИЮ, РАСПРЕДЕЛЕННЫЕ ВЫЧИСЛЕНИЯ, СИСТЕМА МАШИННОГО ПЕРЕВОДА, СИСТЕМА СТАТИСТИЧЕСКОГО МАШИННОГО ПЕРЕВОДА, СТАТИСТИЧЕСКИЙ МАШИННЫЙ ПЕРЕВОД, СУПЕРВИЗОР, ТРАНСФЕРНАЯ СИСТЕМА МАШИННОГО ПЕРЕВОДА, ТЕКСТОВЫЙ КОРПУС, ЯЗЫКОВАЯ ПАРА.

Работа посвящена разработке распределенной статистической системы перевода естественных языков. Актуальность темы оправдана появлением большого количества научно-технических документов и необходимостью оперативного их перевода на другие языки. В работе проведен краткий обзор существующих типов систем машинного перевода, описана теоретическая база статистических систем машинного перевода, изложен нетрадиционный подход к созданию таких систем. В результате работы было создано распределенное программно-информационное обеспечение статистической модели перевода научно-технических текстов на примере русского и английского языков. Система представляет набор приложений взаимодействующих с общей базой данных. Набор приложений можно разделить на два класса:

а) приложения необходимые для обучения системы по уже имеющимся переводам, которые выполнены человеком;

б) приложения осуществляющие подбор наиболее подходящих переводных эквивалентов.

Алгоритмы обучения системы были разработаны с учетом особенностей научных текстов и слабо применимы для других стилей литературы. За неимением текстов нужного объема и качества, в рамках данной работы обучение системы проводилось на комбинированном наборе переводов, состоящим преимущественно из официально-делового и публицистического стилей литературы. Для подбора наиболее подходящих переводных эквивалентов используется жадный инкрементный поиск. Его основным преимуществом является высокая скорость работы, что может оказаться важным для оперативного перевода. Качество перевода разработанной системы несколько уступает существующим аналогам. Это объясняется особенностями исходных данных и характером используемых алгоритмов. Скорость работы системы в несколько раз превосходит скорости доступных систем подобного класса. Для сравнения систем использовался одинаковый набор данных. В экономической части проведен расчет стоимости разработанной стемы. В разделе посвященном охране труда и окружающей среды описано каких последствий можно избежать при использовании созданной системы.

СОДЕРЖАНИЕ

СПИСОК ТЕРМИНОВ И ИХ СОКРАЩЕНИЙ

СМП — система машинного перевода.

ССМП — статистическая система машинного перевода.

Корпус (лингвистический корпус) — называют совокупность текстов, собранных в соответствии с определенном принципами, размеченных по определенному стандарту, в этой работе корпусом называют совокупность предложений на конкретном языке, разделенных символом перевода строки.

n-грамма — подпоследовательность из n элементов из данной последовательности текста или речи, в данной работе рассматривается как подпоследовательность слов.

ОТП (OTP) — открытая телекоммуникационная платформа (open telecom platform).

Супервизор (в терминах ОТП) — процесс (как совокупность взаимосвязанных и взаимодействующих действий), следящий за дочерними процессами, отвечающий за их запуск и остановку.

Приложение (в терминах ОТП) — компонент, который можно запускать и останавливать как единое целое, и который также может быть использован повторно в других системах.

Дерево контроля (супервизии) — совокупность рабочих процессов вычислительной системы, их супервизоров представленное в виде древовидной структуры.

1. ОСНОВНАЯ ЧАСТЬ

1.1. ВВЕДЕНИЕ

Мы живем в мире информационных технологий, которые прочно вошли в нашу жизнь. Мы пользуемся современными средствами связи. Компьютер превратился в неотъемлемый элемент нашей жизни не только на рабочем месте, но и в повседневной жизни. Быстрое развитие новых информационных технологий свидетельствует о всевозрастающей роли компьютерной техники в мировом информационном пространстве.

С каждым днем увеличивается число пользователей Интернета. Все больше сетевые технологии оказывают влияние на развитие самой науки и техники. За последние годы сильно начал меняться характер образования, переходя на уровень дистанционного. Этот переход осуществляется даже в классических вузах. Развитие науки и образования, да и вообще формирование мирового информационного пространства значительно тормозится из-за так называемого языкового барьера. Эта проблема пока не нашла своего кардинального решения.

Последние годы объем предназначенной для перевода информации увеличился. Создание универсального языка типа Эсперанто, «эльфийских языков» или какого-либо другого языка не привели к изменению ситуации. Использование традиционных средств межкультурной коммуникации может быть достойным выходом. Нынешний век диктует свои условия: информация меняется двадцать четыре часа в сутки, широко применяются электронные средства связи. В такой ситуации классический подход к осуществлению перевода не всегда оправдывает себя. Он требует значительных капиталовложений и временных затрат. В некоторых случаях более целесообразным представляется использование машинного или автоматического перевода.

Целью работы является создание статистической системы машинного перевода. Обозначенная цель подразумевает проектирование распределенной системы, разработку алгоритмов статистического анализа текстов, реализацию и тестирование программного обеспечения.

Цель определила следующие задачи:

- исследование существующих статистических систем машинного перевода;
- изучение математических основ построения статистических систем машинного перевода;
- изучение лингвистических основ машинного перевода;
- изучение возможных вариантов хранения данных в рамках задачи машинного перевода;
- составление требований и ограничений системы;
- разработка численного алгоритма обучения системы;
- разработка алгоритма поиска верного варианта перевода на основе обученной модели;
- составление требований к входным данным численного алгоритма;
- составление требований к выходным данным алгоритма поиска ;
- разработка структуры хранения данных;
- разработка распределенной архитектуры;
- разработка работающей обучающейся модели на тестовых входных данных;
- разработка работающего поискового модуля на тестовых входных данных;
- подбор нужных корпусов текстов;
- разработка распределенной обучающейся системы;
- разработка алгоритмов предварительной обработки входных корпусов текста;
- корректировка системы с учетом входных данных;
- тестирование приложения в совокупности отладка всей системы.

1.2. К ПРОБЛЕМЕ МАШИННОГО ПЕРЕВОДА

В настоящее время имеется достаточно широкий выбор пакетов программ, облегчающих труд переводчика, которые условно можно подразделить на две основные группы:

- электронные словари;
- системы машинного перевода.

Системы машинного перевода текстов с одних естественных языков на другие моделируют работу человека-переводчика. Их полезность зависит от того, в какой степени в них учитываются объективные законы языка и мышления. Законы эти пока еще изучены плохо. Поэтому, решая задачу машинного перевода, необходимо учитывать опыт межнационального общения и опыт переводческой деятельности, накопленный человечеством. В процессе перевода в качестве основных единиц смысла выступают не отдельные слова, а фразеологические словосочетания, выражающие понятия. Именно понятия являются элементарными мыслительными образами. Только используя их можно строить более сложные образы, соответствующие переводимому тексту. В современной лингвистике можно выделить ряд направлений использования компьютера:

- машинный перевод;
- отдельные виды автоматизации лингвистических исследований;
- автоматизация лексикографических работ;
- автоматический поиск библиографической информации.

В этой работе мы будем подробно рассматривать системы машинного перевода. На данный момент выделяют три типа систем машинного перевода:

- полностью автоматический;
- автоматизированный машинный перевод при участии человека (MT[1]-системы);

[1]Machine Translation.

- перевод, осуществляемый человеком, с использованием компьютера (ТМ[2]-системы).

Полностью автоматические системы машинного перевода являются несбыточной мечтой, чем реальной идей. В этой работе мы их рассматривать не будем. Все системы машинного перевода (МТ-системы) работают при участии человека в той или иной мере. ТМ-системы иногда называют еще «памятью переводчика». Они являются скорее просто удобным инструментом, нежели элементом автоматизации.

1.2.1. ПОДХОДЫ К МАШИННОМУ ПЕРЕВОДУ

Системы машинного перевода могут использовать метод перевода основанный на лингвистических правилах. Наиболее подходящие слова из исходного языка просто заменяются словами переводного языка. Часто утверждается, что для успешного решения проблемы машинного перевода необходимо решить проблему понимания текста на естественном языке.

Как правило, метод перевода основанный на правилах использует символическое представление (посредника), на основе которого создается текст на переводном языке. А если учитывать природу посредника то можно говорить об интерлингвистическом машинном переводе или трансфертном машинном переводе. Эти методы требуют очень больших словарей с морфологической, синтаксической и семантической информацией и большого набора правил. Современные системы машинного перевода делят на три большие группы:

- основанные на правилах;
- основанные на примерах;
- статистические.

[2]Translation memory.

1.2.2. СМП ОСНОВАННЫЕ НА ПРАВИЛАХ

Системы машинного перевода основанные на правилах — общий термин, который обозначает системы машинного перевода на основе лингвистической информации об исходном и переводном языках. Они состоят из двуязычных словарей и грамматик, охватывающих основные семантические, морфологические, синтаксические закономерности каждого языка. Такой подход к машинному переводу еще называют классическим. На основе этих данных исходный текст последовательно, по предложениям, преобразуется в текст перевода. Эти системы противопоставляют системам машинного перевода, которые основаны на примерах. Принцип работы таких систем — связь структуры входного и выходного предложения.

Эти системы делятся на три группы:

- системы пословного перевода;
- трансфертные системы;
- интерлингвистические.

ПОСЛОВНЫЙ ПЕРЕВОД

Такие системы используются сейчас крайне редко из-за низкого качества перевода. Слова исходного текста преобразуются как есть в слова переводного текста. Часто такое преобразование происходит без лемматизации и морфологического анализа. Это самый простой метод машинного перевода. Он используется для перевода длинных списков слов, например, каталогов. Так же он может быть использован для составления «словаря-подстрочечника» для ТМ-систем.

ТРАНСФЕРТНЫЕ СИСТЕМЫ

Как трансфертные системы, так и интерлингвистические, имеют одну и ту же общую идею. Для перевода необходимо иметь посредника, который в себе несет смысл переводимого выражения. В интерлингвистических системах посредник не зависит от пары языков, в то время как в трансфертных — зависит.

Трансфертные системы работают по очень простому принципу: к входному тексту применяются правила, которые ставят в соответствие структуры исходного и переводного языков. Начальный этап работы включает в себя морфологический, синтаксический, а иногда и семантический анализ текста для создания внутреннего представления. Перевод генерируется из этого представления с использованием двуязычных словарей и грамматических правил. Иногда на основе первичного представления, которое было получено из исходного текста, строят более «абстрактное» внутренне представление. Это делается для того, чтобы акцентировать места важные для перевода и отбросить несущественные части текста. При построении текста перевода преобразование уровней внутренних представлений происходит в обратном порядке.

При использовании этой стратегии получается достаточно высокое качество переводов, с точностью в районе 90% (сильно зависит от языковой пары). Работа любой системы трансфертного перевода состоит как минимум из пяти частей:

- морфологический анализ;
- лексическая категоризация;
- лексический трансфер;
- структурный трансфер;
- морфологическая генерация.

ИНТЕРЛИНГВИСТИЧЕСКИЙ МАШИННЫЙ ПЕРЕВОД

Интерлингвистический машинный перевод — один из классических подходов к машинному переводу. Исходный текст трансформируется в абстрактное представление, которое не зависит от языка (в отличие от трансфертного перевода). Переводной текст создается на основе этого представления. Можно доказать математически, что в рамках этого подхода, создание каждого нового интерпретатора языка для такой системы будет удешевлять ее, по сравнению, например, с системой трансфертного перевода. Кроме того, в рамках

такого подхода можно реализовать «пересказ текста», перефразирование исходного текста в рамках одного языка.

До сих пор не существует реализаций такого типа систем, которая корректно работала хотя бы для двух языков. Многие эксперты высказывают сомнения в возможности реализации. Самая большая сложность для создания подобных систем заключается в проектировании межъязыкового представления. Оно должно быть одновременно абстрактным и независящим от конкретных языков, но в тоже время оно должно отражать особенности любого существующего языка. С другой стороны, в рамках искусственного интеллекта, *задача выделения смысла* текста на данный момент до сих пор не решена.

1.2.3. СМП ОСНОВАННЫЕ НА ПРИМЕРАХ

Перевод основанный на примерах — один из подходов к машинному переводу, при котором используется двуязычный корпус текста. Этот корпус текста во время перевода используется как база знаний. Предполагается, что люди разлагают исходный текст на фразы, потом переводят эти фразы, а далее составляют переводной текст из фраз. Причем, перевод фраз обычно происходит по аналогии с предыдущими переводами. Для построения системы машинного перевода, основанной на примерах потребуется языковой корпус, составленный из пар предложений. *Языковые пары* — тексты, содержащие предложения на одном языке и соответствующие им предложения на втором, могут быть как вариантами написания двух предложений человеком — носителем двух языков, так и набором предложений и их переводов, выполненных человеком.

Перевод, основанный на примерах, лучше всего подходит для таких явлений как фразовые глаголы. Значения фразовых глаголов сильно зависит от контекста. Фразовые глаголы очень часто встречаются в разговорном английском языке. Они состоят из глагола с предлогом или наречием. Смысл такого выражения невозможно получить из смыслов составляющих частей. Классические методы перевода в данном случае неприменимы. Этот метод перевода можно использовать для определения контекста предложений.

Как показано далее, реализовать примитивную систему машинного перевода основанную на примерах крайне просто.

1.2.4. СТАТИСТИЧЕСКИЙ МАШИННЫЙ ПЕРЕВОД

Статистический машинный перевод — это метод машинного перевода. Он использует сравнение больших объемов языковых пар, так же как и машинный перевод основанный на примерах. Статистический машинный перевод обладает свойством «самообучения». Чем больше в распоряжении имеется языковых пар и чем точнее они соответствуют друг другу, тем лучше результат статистического машинного перевода. Статистический машинный перевод основан на поиске наиболее вероятного перевода предложения с использованием данных из двуязычных корпусов текстов. В результате при выполнении перевода компьютер не оперирует лингвистическими алгоритмами, а вычисляет вероятность применения того или иного слова или выражения. Слово или последовательность слов, имеющие оптимальную вероятность, считаются наиболее соответствующими переводу исходного текста и подставляются компьютером в получаемый в результате текст.

В статистическом машинном переводе ставится задача не перевода текста, а задача его расшифровки. Статья, написанная на английском языке, на самом деле является статьей написанной на русском, но текст зашифрован (или искажен шумом). При таком подходе становится понятно почему, чем «дальше» языки, тем лучше работает статистический метод, по сравнению с классическими подходами.

МОДЕЛЬ ШЕННОНА

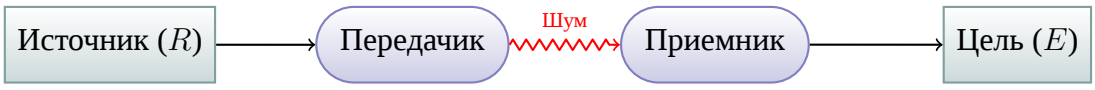

Рис. 1.1. Модель зашумленного канала.

Модель состоит из пяти элементов: источника информации, передатчика, канала передачи, приемника и конечной цели, расположенных линейно. Передатчик кодирует информацию, полученную от источника, и передает ее на канал. По каналу передачи, на который действует шум — помехи любого рода, искажающие информацию, данные поступают в приемник, где они декодируется и передаются к конечной цели.

Из-за шума полученная приемником информация в общем случае не совпадает с информацией, отправленной передатчиком. Однако, согласно модели Шеннона, создавая избыточную информацию, исходные данные можно восстановить со сколь угодно высокой вероятностью. Для обнаружения ошибок используются контрольные суммы, для их исправления — специальные корректирующие коды, при условии, что степень шума не превосходит некоторой границы. Стоит отметить, что любая информация в некотором роде избыточна [20]. Человеческая речь избыточна — чтобы уловить смысл предложения, зачастую необязательно слышать его полностью. Аналогично, письменная речь, тоже избыточна, и при переводе этим можно воспользоваться. Если предложение в целом понятно, но есть несколько незнакомых слов, то обычно не трудно догадаться об их значении.

Таким образом, для перевода текста необходимо найти способ декодирования, использующий естественную избыточность, в связи с чем декодирование должно быть вероятностным. Задача такого декодирования заключается в том, чтобы, при данном сообщении найти исходное сообщение, которому соответствует наибольшая вероятность. Для этого же необходимо для любых двух сообщений уметь находить условную вероятность того, что переведенное сообщение, пройдя через канал с шумом, преобразуется в исходное сообщение. В данном случае нужна модель источника (модель языка) и модель канала (модель перевода). Модель языка дает оценку вероятности фразам переводного языка, а модель перевода оценивает вероятность исходной фразы при условии фразы на переводном языке. Если нам нужно перевести фразу с русского на английский, то мы должны знать, что именно обычно говорят по-английски и как английские фразы искажаются до состояния русского языка. Сам по себе перевод превращается в процесс поиска такой английской фразы, которая максимизировала бы произведения безусловной вероятности английской фразы и вероятности русской фразы-оригинала при условии данной английской фразы.

$$\max_{\varphi_e} P(\varphi_e|\varphi_r) = \max_{\varphi_e} \left(P(\varphi_e) \cdot P(\varphi_r|\varphi_e)\right), \text{ где}$$

- φ_e — фраза перевода (английская);
- φ_e — фраза оригинала (русская).

В системах статистического перевода, в качестве модели языка используются варианты n-граммной модели (например, в переводчике Google, используется 5-граммная модель). Согласно этой модели, правильность выбора того или иного слова зависит только от предшествующих $(n-1)$ слов. Самой простой статистической моделью перевода является модель пословного перевода. В этой модели, известной как Модель IBM №1, предполагается, что для перевода предложения с одного языка на другой достаточно перевести все слова, а расстановку их в правильном порядке обеспечит модель языка. Единственным массивом данных, которым оперирует Модель №1, является таблица вероятностей парных переводных соответствий слов двух языков [55]. Обычно используются более сложные модели перевода.

Работа статистических систем, так же как и систем основанных на примерах происходит в двух режимах: обучения и эксплуатации. В режиме обучения просматриваются параллельные корпуса текста и вычисляются вероятности переводных соответствий. Строится модель языка перевода. Тут же определяются вероятности каждой n-граммы. В режиме эксплуатации, для фразы из исходного текста ищется фраза переводного текста, так, чтобы максимизировать произведение вероятностей.

1.2.5. СРАВНЕНИЕ РАЗЛИЧНЫХ ТИПОВ СМП

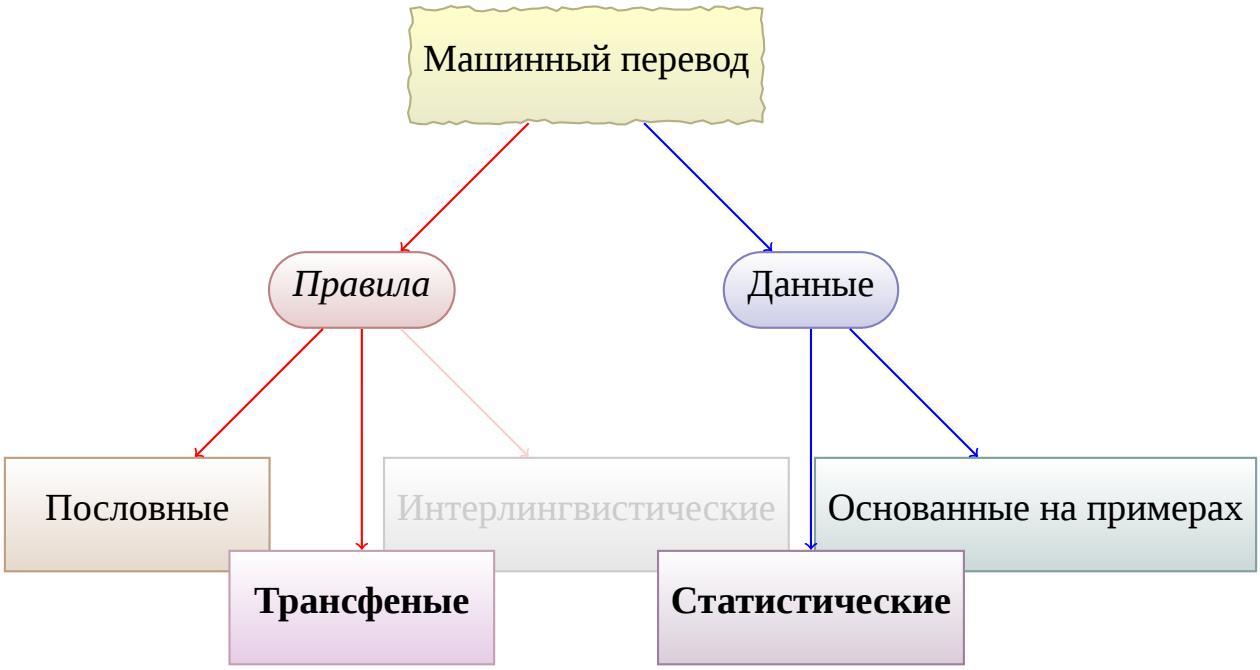

Рис. 1.2. Классификация систем машинного перевода.

Рассмотрим кратко преимущества и недостатки существующих систем.

СИСТЕМЫ ПОСЛОВНОГО ПЕРЕВОДА

Системы пословного перевода на данный момент используются только для составления подстрочечника, как отмечалось ранее.
Преимущества:

- простота;
- высокая скорость работы;
- не требовательные к ресурсам.

Недостатки:

- низкое качество перевода.

Ярких представителей на рынке нет, в данном случае удобнее создавать новую систему под конкретную задачу.

ТРАНСФЕРТНЫЕ СИСТЕМЫ

Трансфертные системы распространены очень широко. Наиболее известными представителями являются ImTranslator, PROMPT. Все подобные системы имеют сходные преимущества и недостатки.
Преимущества:

- высокое качество перевода (при наличие нужных словарей и правил);
- выбор тематики текста, который повышает качество перевода;
- возможно уточнение перевода, благодаря внесению изменений в базу данных переводчика (таким образом, пользователь получает потенциально бесконечное множество терминов, с которыми можно свободно оперировать, и можно достигнуть «бесконечного» качества перевода).

Недостатки:

- высокая стоимость и время разработки;
- для добавления нового языка, приходиться переделывать систему заново;
- нужна команда квалифицированных лингвистов, для описания каждого исходного и каждого переводного языка;
- требовательность к ресурсам на этапе составления базы.

ИНТЕРЛИНГВИСТИЧЕСКИЕ СИСТЕМЫ

Интерлингвистические системы перевода так и не были доведены до уровня промышленных систем. Предполагаемые преимущества:

- высокое качество перевода, независимо от выбора языка;
- выделение смысла из исходного текста происходит один раз и потом записывается на любой язык, в том числе исходный (получаем «пересказ текста»);
- низкая стоимость трудозатрат на добавления нового языка в систему.

Недостатки:

- спорность потенциальной возможности;
- высокая сложность разработки;
- системы не масштабируются.

СМП, ОСНОВАННЫЕ НА ПРИМЕРАХ

СМП, основанные на примерах, так же не имеют ярких представителей. Существующие прототипы используются в академической среде для иллюстрации самого метода. Часто они поставляются не в виде готового продукта, а в виде набора библиотек: Marclator – СМП Дублинского Университета, Cunei гибридная СМП, основанная на переводе по аналогии и на статистическом переводе.

Преимущества:

- высокое качество перевода (при наличие достаточно долгой тренировке системы);
- хорошо справляется со многими контекстными задачами (фразовые глаголы);
- квалифицированные лингвисты не нужны непосредственно для построения системы, нужны только инженеры;
- логическая простота устройства;
- возможно обучение системы во время ее эксплуатации.

Недостатки:

- для обучения системы нужны большие параллельные корпуса текста, размеченные определенным образом;
- качество перевода зависит от исходных корпусов;
- продолжительное время обучения;
- требовательность к ресурсам на этапе обучения.

ССМП

ССМП активно разрабатывались (и разрабатываются) компанией IBM. Благодаря ее разработкам, были созданы модели перевода IBM Model 1-5. Но наибольшую известность этот метод приобрел благодаря компании Google. Кроме переводчика Google существует еще ряд систем и библиотек, использующих статистический подход.
Преимущества:

- высокое качество перевода:
  - для фраз, которые целиком помещаются в n-граммную модель
  - при наличии достаточно долгой тренировке системы.
  - при наличии качественных корпусов текста;
- квалифицированные лингвисты не нужны непосредственно для построения системы, нужны только инженеры;
- труд человека минимизирован для создания таких систем;
- не требуется перестраивать систему при добавлении нового языка;
- возможно обучение системы во время ее эксплуатации.

Недостатки:

- для обучения нужны большие параллельные корпуса текста;
- сложный математический аппарат;
- качественный перевод возможен только для фраз, которые целиком помещаются в n-граммную модель;
- качество перевода зависит от исходных корпусов;
- при добавлении нового языка приходится анализировать большие объемы данных;
- продолжительное время обучения;
- требовательность к ресурсам на этапе обучения.

1.3. МАТЕМАТИЧЕСКАЯ БАЗА ССМП

Пусть φ_r — фраза оригинала, русская. Требуется найти φ_e — фразу перевода, английскую. Нужно максимизировать $P(\varphi_e|\varphi_r)$. Если вспомнить модель зашумленного канала (модель Шеннона), то получаем:

$$P(\varphi_e|\varphi_r) = \frac{(P(\varphi_e) \cdot P(\varphi_r|\varphi_e))}{P(\varphi_r)} \Rightarrow$$

$$\varphi_{eg} = \arg\max_{\cup\varphi_e} P(\varphi_e|\varphi_r) = \arg\max_{\cup\varphi_e} (P(\varphi_e) \cdot P(\varphi_r|\varphi_e))$$

$P(\varphi_r)$ — нам известна, ее не учитываем. Величина $P(\varphi_e)$ называется моделью языка. $P(\varphi_r|\varphi_e)$ — модель перевода. Работа любой статистической системы перевода состоит из двух этапов:

- обучения — вычисляются модели языка и перевода;
- эксплуатации — вычисляется величина $\arg\max_{\cup\varphi_e} P(\varphi_e|\varphi_r)$ при данной φ_r (процесс вычисления называют декодированием).

1.3.1. ОБУЧЕНИЕ ССМП

ВЫЧИСЛЕНИЕ ЯЗЫКОВОЙ МОДЕЛИ

В качестве модели языка в системах статистического перевода используются преимущественно различные модификации n-граммной модели, утверждающей, что «грамматичность» выбора очередного слова при формировании текста определяется только тем, какие $(n–1)$ слов идут перед ним. Вероятность каждого n-грамма определяется по его встречаемости в тренировочном корпусе [55].

$$P(\omega_1 \ldots \omega_l) = \prod_{i=0}^{i=l+n-1} P'(\omega_i|\omega_{i-1} \ldots \omega_{i-n+1})$$

n — n-граммность модели.

$$P'(\omega_m|\omega_1 \dots \omega_{m-1}) = K_n \cdot P(\omega_m|\omega_1 \dots \omega_{m-1}) +$$

$$+K_{m-1} \cdot P(\omega_{m-1}|\omega_1 \dots \omega_{m-2}) +$$

$$+K_2 \cdot P(\omega_2|\omega_1) + K_1 \cdot P(\omega_1) + K_0;$$

$$P(\omega_1) = \frac{\textit{частота}\ (\omega_1)}{|\Theta|};$$

$$P(\omega_m|\omega_1 \dots \omega_{m-1}) = \frac{\textit{частота}\ (\omega_1 \dots \omega_{m-1}\omega_m)}{\textit{частота}\ (\omega_1 \dots \omega_{m-1})};$$

K_i — коэффициенты сглаживания. Они могут быть выбраны различными способами. Чаще всего используется линейная интерполяция.

$$K_i > K_{i+1};$$

$$\sum_{i=0}^{i=n} K_i = 1.0;$$

В этом случае придется подбирать и экспериментально, например для трехграммной модели $K_3 = 0.8, K_2 = 0.15, K_1 = 0.049, K_0 = 0.001$ [34]

P' можно вычислить иначе, используя адаптивный метод сглаживания

$$P'(\omega_m|\omega_1 \dots \omega_{m-1}) = \frac{\delta + \textit{частота}\ (\omega_1 \dots \omega_m)}{\sum\limits_i \left(\delta + \textit{частота}\ (\omega_{1_j} \dots \omega_{m_j})\right)}$$

$$= \frac{\delta + \textit{частота}\ (\omega_1 \dots \omega_m)}{\delta \cdot V + \sum\limits_i \left(\textit{частота}\ (\omega_{1_j} \dots \omega_{m_j})\right)}$$

V – количество всех n-грамм в используемом корпусе. Наиболее простым случаем аддитивного сглаживания является метод, когда $\delta = 1$ – метод сглаживания Лапласа [58].

Существуют и другие техники сглаживания вероятностей (Гуда-Тьюринга, Катца, Кнезера-Нейя [2]),

ВЫЧИСЛЕНИЕ МОДЕЛИ ПЕРЕВОДА

Обозначим:

- Θ_e — «английский» текст (множество предложений);
- Θ_r — «русский» текст;
- Π_e — «английское» предложение (последовательность слов);
- Π_r — «русское» предложение;
- ω_e — «английское» слово;
- ω_r — «русское» слово;
- $l_e \leftarrow |\Pi_e|$;
- $l_r \leftarrow |\Pi_r|$;
- $\pi_{\omega_r} \leftarrow$ позиция ω_r в Π_r;
- $\pi_{\omega_e} \leftarrow$ позиция ω_e в Π_e.

Пусть $P(\Pi_e|\Pi_r)$ — вероятность некоторой строки (предложения) из e, при гипотезе перевода из r. По аналогии с моделью языка можно предположить, что

$$P(\Pi_e|\Pi_r) = \frac{\textit{частота}\ (\Pi_e, \Pi_r)}{\textit{частота}\ (\Pi_r)};$$

Однако это не верно. Для вычисления модели перевода нужно:

- разделить предложение на меньшие части;
- ввести новую переменную a, представляющую выравнивания между отдельными словами в паре предложений.

$$P(\Pi_e|\Pi_r) = \sum_a P(\Pi_e, a|\Pi_r);$$

Вероятность перевода:

$$P(\Pi_e, a|\Pi_r) = \frac{\varepsilon}{(l_r+1)^{l_e}} \prod_{j=1}^{l_e} t(\omega_{ej}|\omega_{ra(j)})$$

t – это вероятность слова оригинала в позиции j при соответствующем ему слове перевода $\omega_{ra(j)}$, определенном выравниванием a. ε — нормализующая «константа». В этой работе ε выбирается равным 1, но если рассуждать более строго, ε — распределение вероятностей длин предложений каждого из языков

$$\varepsilon = \varepsilon(l_e|l_r)$$

Для приведения $P(\Pi_e, a|\Pi_r)$ к $P(a|\Pi_e, \Pi_r)$, т.е. к вероятности данного выравнивания при данной паре предложений, каждая вероятность $P(\Pi_e, a|\Pi_r)$ нормализуется по сумме вероятностей всех выравниваний данной пары предложений:

$$P(a|\Pi_e, \Pi_r) = \frac{P(\Pi_e, a|\Pi_r)}{\sum\limits_a P(\Pi_e, a|\Pi_r)}$$

Имея набор выравниваний с определенными вероятностями, можно подсчитать частоты каждой пары слов, взвешенные по вероятности выравниваний, в которых они встречаются. Например, если какая-то пара слов встречается в двух выравниваниях, имеющих вероятности 0.5 и 1, то взвешенная частота ($counts$) такой пары равна 1.5 [55].

$$t(\omega_e|\omega_r) = \frac{counts(\omega_e|\omega_r)}{\sum\limits_{\omega_e} counts(\omega_e|\omega_r)} = \frac{counts(\omega_e|\omega_r)}{total(\omega_r)};$$

Требуется оценить вероятности лексического перевода $t(\omega_e|\omega_r)$ из параллельного корпуса (Θ_e, Θ_r). Но чтобы сделать это нужно вычислить a, которой у нас нет. Возникает так называемая «проблема курицы и яйца». Для оценки параметров модели нужно знать выравнивания. Для оценки выравнивания нужно знать параметры модели.

При решении этой проблемы используют EM-алгоритм (Витерби), который более детально рассмотрен в приложении. EM-алгоритм:

1) Инициализируем параметры модели (одинаковыми значениями, на первой итерации);
2) Оценим вероятности отсутствующей информации;
3) Оценим параметры модели на основании новой информации;
4) Перейдем к следующей итерации.

1.3.2. ДЕКОДИРОВАНИЕ ССМП

Декодер нужен для осуществления непосредственно перевода. Он вычисляет величину

$$\arg\max_{\cup\varphi_e} \left(P(\varphi_e) \cdot P(\varphi_r|\varphi_e)\right)$$

Задача декодирования является NP-полной [7]. Существует несколько способов и методов декодирования. Обычно выделяют:

- полный перебор;
- поиск по первому наилучшему совпадению (A*):
  - стековый поиск,
  - мультистековый поиск;
- жадный инкрементный поиск;
- сведение к обобщенной (асимметричной или симметричной) задаче коммивояжера:
  - задача ЛП,
  - метод Лина-Кернигана,
  - генетические алгоритмы,
  - «муравьиная оптимизация».

Каждый из методом обладает своими достоинствами и недостатками. Вариант полного перебора мы рассматривать не будем, так как, если мы ограничим наш поиск в строке не более чем в два раза длины m строки исходного языка, то мы получим наивный метод с сложностью $O(m^2v^{2m})$ [7].

В данном случае можно рассмотреть только часть пространства возможных состояний. При этом, скорее всего, мы можем пропустить самое хорошее решение, но сможем найти «достаточно хорошее». Для алгоритмов декодирования важными параметрами являются скорость, поиск ошибок, качество перевода.

ПОИСК ПО ПЕРВОМУ НАИЛУЧШЕМУ СОВПАДЕНИЮ

Поиск учитывает как расстояние от начального состояния и оценки расстояния до цели. В приложении декодирования имеем следующий алгоритм для стекового поиска:

1) Инициализируем стек пустой гипотезой.
2) Достаем лучшую гипотезу h из стека.
3) Если h — все предложение, выводим его и завершаем выполнение.
4) Для всех возможных следующих слов ω расширить гипотезу h и положить обратно в стек.
5) Перейти ко второму шагу.

«Стековым» поиск назван по историческим причинам, на самом деле используется очередь с приоритетами.

Большим минусом этого поиска является, то что более короткие гипотезы имеют приоритет. Мультистековый поиск отличается наличием отдельного «для гипотез» разного размера. Крайне не экономичен по памяти. Для уменьшения пространства поиска возможны различные ухищрения, основанные на знаниях о структуре предложения.

ЖАДНЫЙ ИНКРЕМЕНТНЫЙ ПОИСК

Простой поиск, позволяет достичь решения путем выбора лучшей альтернативы, чтобы добраться до цели в настоящее время. Эвристическая функция определяется как стоимость самого дешевого пути из текущего состояния в целевое состояние. Жадный инкрементный поиск имеет следующие особенности:

- «плохой» вариант перевода получаем сразу;
- последовательно применяя набор операций можем улучшить перевод:
  - изменить перевод слова (группы слов),
  - удалить слово (группу слов),
  - поменять слова местами.

В оригинальной работе [4] приводится иной набор операций, но он относится исключительно к моделям высшего порядка (IBM 3-5). Дело в том, что модели высших порядков учитывают наличие фертильности слова — величину показывающую сколько слов языка перевода способно породить данное слово исходного языка. С понятием фертильности связано понятие нулевого слово — слово исходного языка не имеющее графического начертания в тексте, и какого либо еще своего проявления, кроме того, что оно обладает ненулевой фертильностью. Кроме того, в классических моделях используется биграммная модель языка. В связи с этим, к описанным выше опирается сразу добавляются замены слов в зависимости от их фертильности, вставки слов в какой либо участок между словами, вставка слов и их одновременная замена. В этой работе мы не вводим формальное описание моделей высших порядков, так как это займет достаточно внушительный объем. Потому мы привели упрощенную версию алгоритма жадного поиска. Именно такая версия поиска используется нами в практической части работы. Выносить отдельно описание мы тоже не стали, так как ничем особенно новым этот вариант поиска не отличается. Всего скорее такой алгоритм будет проходить по циклу операции быстрее оригинального, но этот факт требует дополнительных исследований и количественных измерений. В любом случае ясно, что за

меньшее число операций придется платить большим количеством итераций алгоритма.

Жадный поиск может стартовать из начального состояния, которое ни приведет к конечной цели. Не является полным или оптимальным. Однако, это самый быстрый из существующих методов. Жадный инкрементный поиск используется в данной работе.

СВЕДЕНИЕ К ОБОБЩЕННОЙ ЗАДАЧЕ КОММИВОЯЖЕРА

Исходными данными для задачи является множество вершин, разбиение этого множества на подмножества, и также матрица стоимостей перехода из одной вершины в другую. Задача заключается в нахождении кратчайшего замкнутого пути, который бы посетил по одной вершине в каждом подмножестве.

Если для двух смежных вершин $|\overset{\frown}{\mathrm{AB}}| \neq |\overset{\frown}{\mathrm{BA}}|$, то задача является асимметричной. Для декодирования имеем:

- подмножество вершин — слово исходного текста ω_r;
- вершина — вариант перевода ω_e (обычно выбирают 10 наиболее вероятных вариантов);
- расстояние — величина неопределенности $-\log\left(P(\varphi_e) \cdot P(\varphi_r|\varphi_e)\right)$.

Обобщенная задача коммивояжера (ОЗК) сводится к простой задаче коммивояжера (ЗК) при помощи алгоритма Нуна-Бена [19]. Далее ее решают средствами линейной оптимизации (крайне медленный вариант), эвристических алгоритмов, генетических алгоритмов, методов *роевого* интеллекта.

Для данной работы сведения задачи декодирования к обобщенной задаче коммивояжера практического значения не имеет, но представляет особенной интерес в рамках статистического машинного перевода. В ходе исследований, сопутствующих работе нами был рассмотрен метод решения обобщенной задаче коммивояжера с помощью метода «муравьиной оптимизации». В результате, было сделано очень важное наблюдение — для ЗК не важен город с которого был начат обход графа, а для задачи декодирования важен — есть заданный порядок слов.

Метод «муравьиной оптимизации» как раз относился к таким не устойчивым методам. Относительный порядок слов (расположение n соседних слов) может быть установлено с помощью модели целевого языка, однако, например при циклическом сдвиге слов выходного предложения, модель языка будет бессильна.

Сам по себе метод «муравьиной оптимизации» оптимизации представляет из себя приближенный метод решения ЗК. Он основан на наблюдениях за поведением муравьев в природе. Перемещаясь от одного пункта до другого, муравьи оставляют за собой тропы из феромонов Если другие муравьи находят такие тропы, они, пойдут по ним. Тем самым укрепят воздействие феромонной тропы. Со временем феромонная тропа испаряется. Чем больше времени требуется для прохождения, тем сильнее испарится феромонная тропа. Для ЗК муравьи сначала в случайном порядке выходят из каждого города, после заданного числа переходов отбирается путь который менее всего «испарился».

В рамках задачи перевода при решении ЗК методом «муравьиной оптимизации» можно использовать выходе циклический сдвиг полученных результатов, но в это потребует дополнительных накладных расходов, которые могут перекрыть преимущества приближенного метода.

Можно также предложить вариант метода, когда каждый полученный результат можно будет циклически сдвинуть на определенную величину и потом целиком проверить по модели языка, но в этом случае, мы получим очередной и очень экзотический вариант жадного инкрементного поиска.

1.4. ПРЕДЛАГАЕМЫЙ ПОДХОД К РАЗРАБОТКЕ ССМП

Основными задачами разрабатываемой системы являются:

а) обработка входного параллельного корпуса с опорой на особенности научного текста;

б) создание моделей языка и перевода на основании входного параллельного корпуса;

в) декодирование исходного текста на основании моделей языка и перевода (с опорой на особенности научного текста);

г) иметь возможность выполнять описанные выше операции распределено.

В соответствии с перечисленными задачами весь код системы можно разбить на 3 класса приложений, каждый из которых соответствует своей задаче.

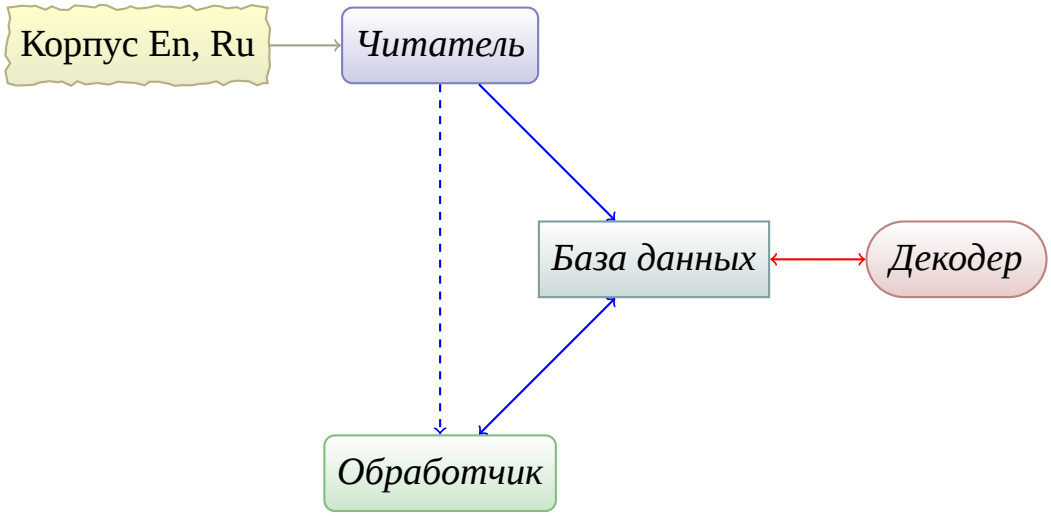

Рис. 1.3. Упрощенная схема системы.

Кроме того, для взаимодействия с системой пользователя необходимо предусмотреть интерфейс взаимодействия с пользователем и с внешними приложениями. С учетом этого и условия распределенности общая схема может иметь вид как представлено ниже.

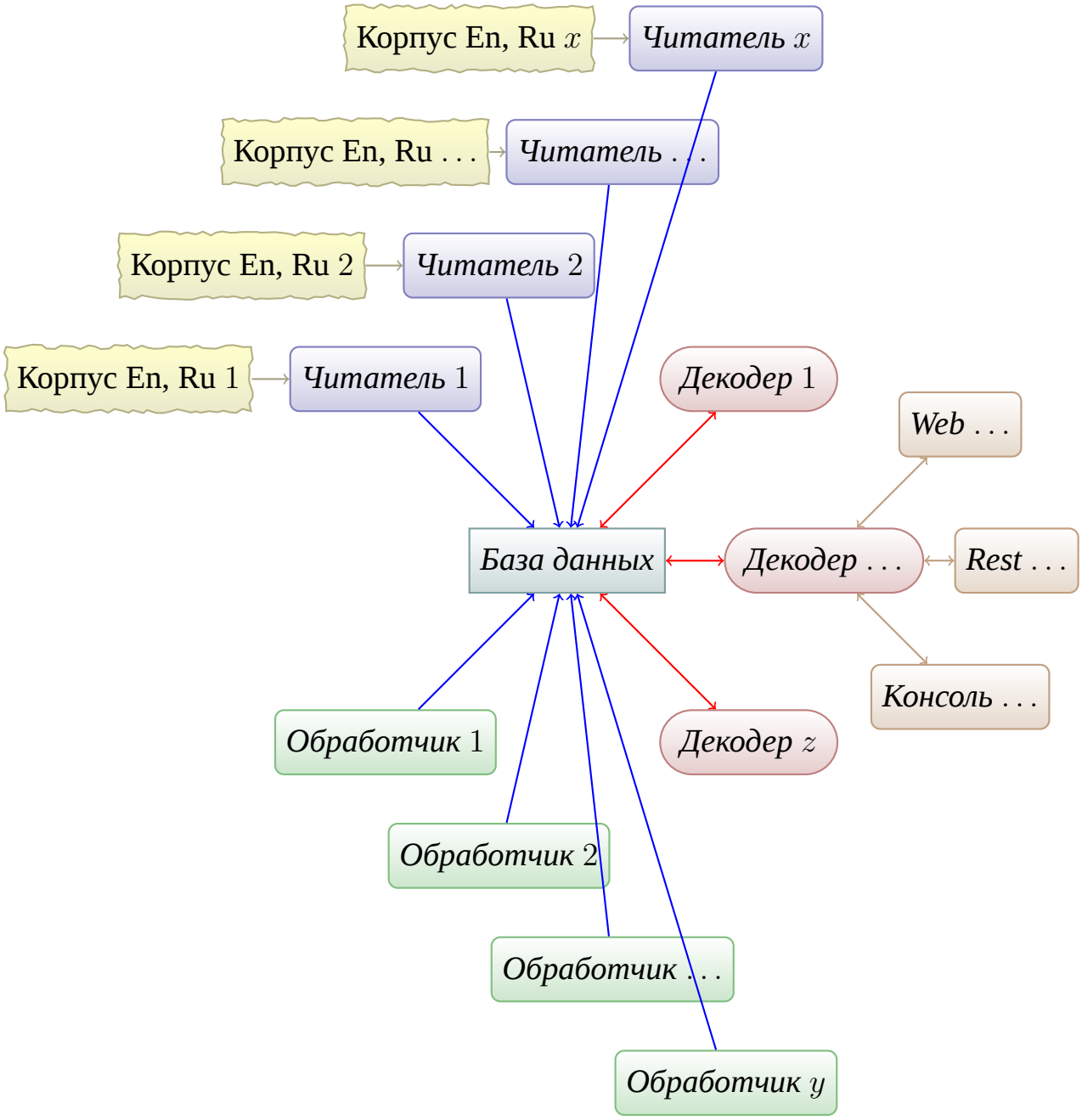

Рис. 1.4. Полная схема системы.

Для эффективной работы системы предполагается, что $x < y$. Все три типа модуля могут быть удалены географически (однако, при этом может упасть скорость взаимодействия). База данных может быть распределена. Использование (возможно, удаленной) базы данных является скорее чисто техническим моментом реализации системы. Намного естественнее и проще было бы проводить все вычисления в локальной памяти машины, а для распределения вычислений осуществлять пересылку сообщений от одного узла

к другому. Однако, память компьютера может быть не в состоянии вместить все необходимые для вычисления данные даже при очень большом объеме. А пересылка сообщений приведет к переполнению очередей сообщений для процессов, которые медленно осуществляют обработку информации. На единицу времени они будут принимать больше, чем смогут обработать. Потому самым оптимальным решением стало использование базы данных класса «ключ-значение» для каждой вычислительной операции, которая потребует длительного хранения. При создании систем машинного перевода текстов с одних естественных языков на другие возникает вопрос о том, какими минимальными единицами смысла следует оперировать в таких системах [60].

Система ориентирована на перевод исключительно научно-технической литературы. Тексты научной литературы на русском и английском языках сходны по своей структуре. Стиль научно-технической прозы отличается:

а) устойчивыми формальными выражениями;

б) прямым порядком слов;

в) стереотипной структурой предложений;

г) важностью локального порядка слов.

Для перевода научного текста, на основании принципов изложенных в теоретической части, мы предлагаем следующие приемы:

а) вместо слов использовать их группы или n-граммы;

б) использовать выравнивание по крупным группам n-грамм;

в) использовать модели низких порядков (IBM 3-5 могут сильно исказить локальный порядок слов).

Как показывают статистические исследования, словосочетания длиной более 10 слов повторяются в текстах очень редко и в совокупности покрывают их менее чем на 1% [60].

Предлагаются следующие ограничения:

а) максимальный размер n-грамм положить равным 5 слов;

б) максимальный размер считываемой строки ограничить 256 символами;

в) максимальный размер предложения ограничить 40 словами;

Все эти ограничения являются настраиваемыми, однако именно с такими настройками проводилась реализация и тестирование системы. Ниже рассмотрим каждое приложение в отдельности.

1.4.1. ЧИТАТЕЛЬ

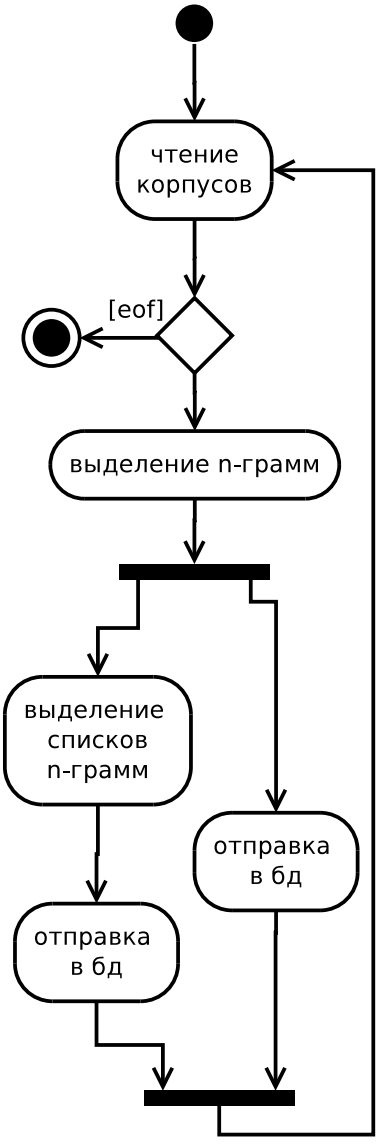

Рис. 1.5. Диаграмма активности приложения «читатель».

Приложение считывает параллельный корпус строка за строкой из двух файлов корпуса одновременно. После прочтения каждая строка разбивается на слова, по синтаксическим признакам конца слова (пробелы, знаки препинания). Из списков слов формируются группы n-грамм начиная с максимального n, заканчивая слова. Одним из принципиальных моментов системы является метод выделения списков n-грамм. Имея список n-грамм верхнего уровня, скажем пятого, составить список более низкого уровня можно без обращения к исходной строке. Таким образом, выделения списка занимает время строго меньшее чем $O(l^2)$ по длине строки. Формирование списков n-грамм и отправка полученных n-грамм в базу может происходить параллельно. Сама по себе отправка представляет собой пару операций: инкремент счетчика заданной n-граммы в базе и добавление слова множество n-грамм. Обе эти операции также можно проводить параллельно, более того отправку в базу данных может быть осуществлена асинхронно, так как для нас не важен возврат результата выполнения команды. n-граммы в списке n-граммы отсортированы по убыванию их длин, n-граммы одинаковой длинны, отсортированы по возрастанию позиции первого слова n-граммы в исходном предложении. Добавление в базу списков n-грамм представляет собой асинхронную запись каждого списка в базу данных подобно добавлению элемента в стек. Общая схема приложения представлена ниже. Стрелками показаны потоки данных. Прерывистыми линиями изображены косвенные связи.

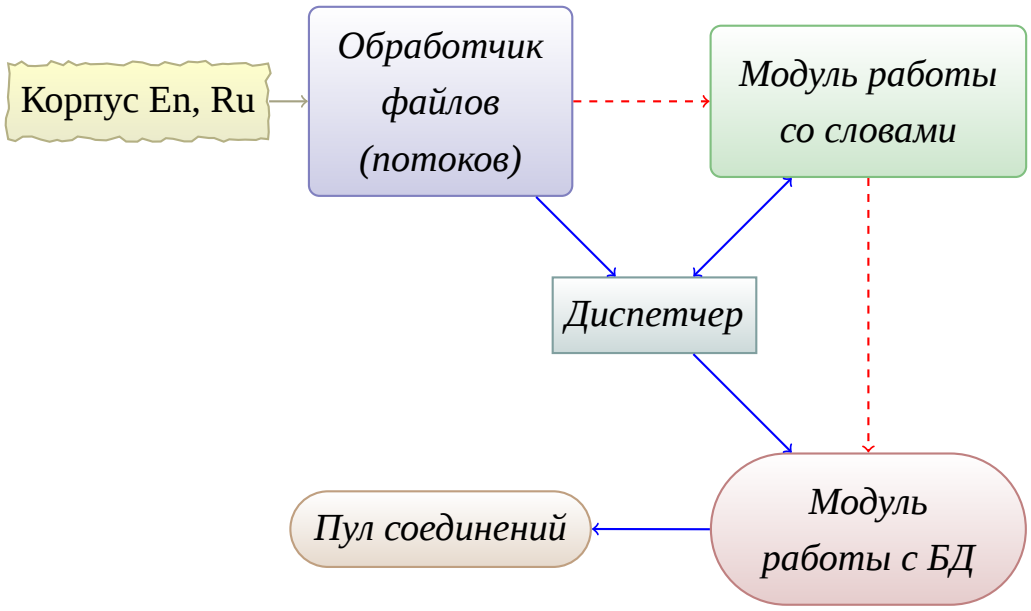

Рис. 1.6. Схема модулей приложения «читатель».

В одной из модификаций этого приложения мы сразу в приложении читателя составляли список сопоставления n-грамм разных языков. Это позволяло несколько разгрузить приложение обработчика кроме того давало возможность гибко настраивать сопоставление n-грамм разного размера.

Полный список сопоставления n-грамм представляет собой декартово произведения списков n-грамм на разных языках. Но использование полного списка для n-грамм большого размера является не экономичным с точки зрения хранения информации. Тем более, учитывая, стилистические особенности научного текста, крупные группы слов отражающие сходные понятия в научных текстах находятся примерно на одинаковом удалении от начала предложения.

Для этого случая нами был разработан следующий подход:

а) строить полный список сопоставлений (декартово произведение) для 1-грамм, то есть слов;

б) строить m-диагональную матрицу для всех остальных n-грамм.

m выбирается экспериментально, по умолчанию полагается равным 3. Такой подход позволяет снизить количество сопоставляемых n-грамм, но повышает вероятность ошибки обучения системы. Как показала практика, не все научные тексты удовлетворяют ограничению представленному выше. Кроме того, это предположение является ошибочным для текстов иных стилей. В текущей реализации мы отказались от такого подхода.

1.4.2. ОБРАБОТЧИК

Обработчик нужен для обучения модели перевода на основании данных полученных у читателя.

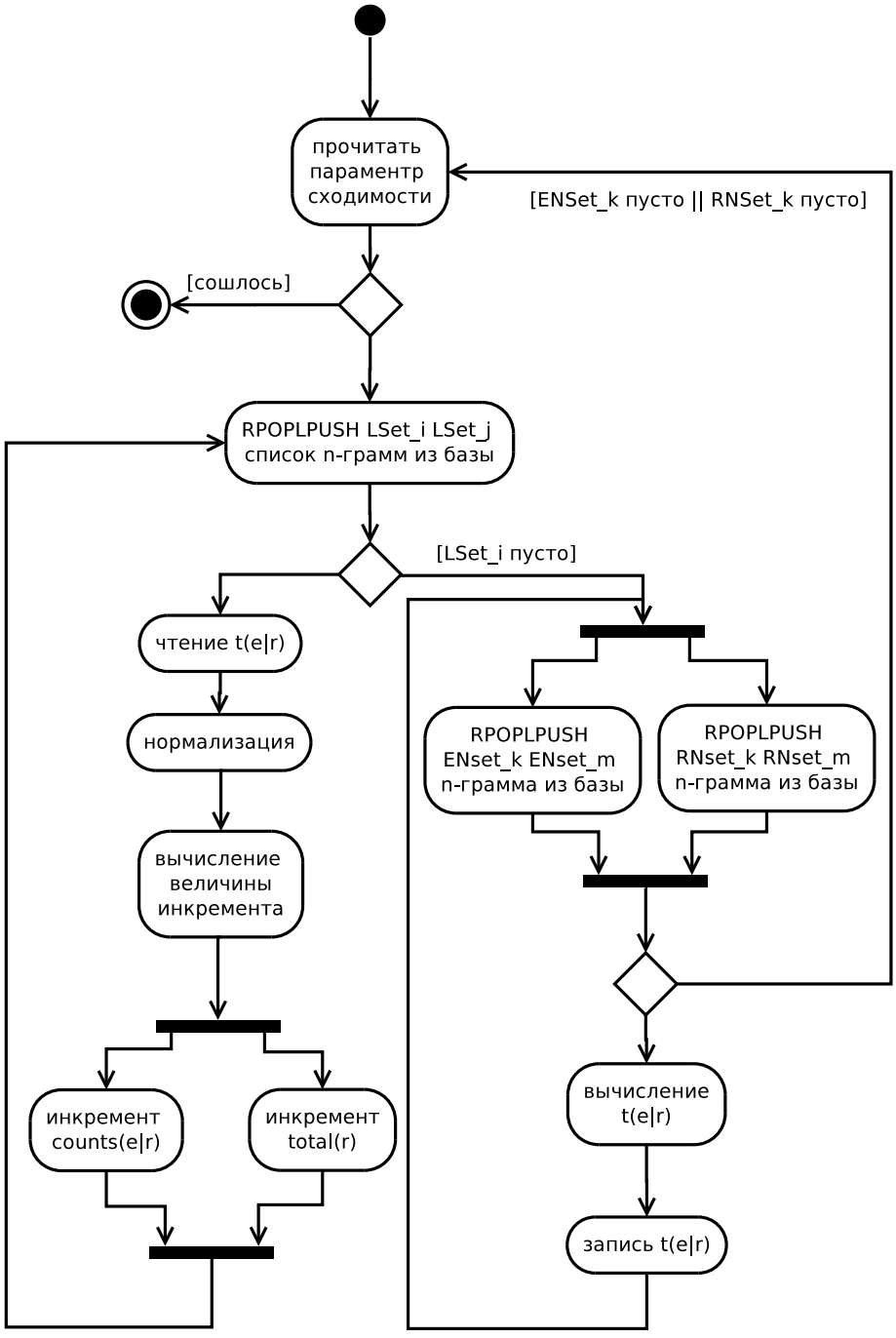

Рис. 1.7. Диаграмма активности приложения «обработчик».

Принцип работы обработчика во многом определяется алгоритмом обучения описанным в выше. Условие сходимости предлагается не вычислять как математическое неравенство. Как показала практика на текстах ограниченного объема это условие может никогда не выполниться, тем более нам не столь важен сам факт сходимости, сколько относительные отношения вероятностей парных соответствий n-грамм.

Важными архитектурными моментами работы приложения являются методы доступа к данным. Предполагается что все операции на запись данных буду асинхронными. Операция инкремента величины подсчетов (см. алгоритмы в приложении), должны быть атомарными и асинхронными. В противном случае, если мы используем распределенные вычисления, то данные рискуют быть или неверно прочитаны или перезаписаны в неподходящий момент (оптимистическая блокировка). Использование пессимистической блокировки может привести к значительному замедлению при большой конкуренции запросов.

Lset_0 — пары списков n-грамм перемещенные туда читателем. На момент начала работы обработчика предполагается что это множество не пусто. На каждом проходе по множеству элемент этого множества считывается в обработчик и перемещается в Lset_1 (это выполняется как атомарная операция rpoplpush). При каждой i-й итерации алгоритма обработчик считывает пару списков n-грамм из Lset_0 и кладет их в множество Lset_{i+1}. Ситуация с вычислением вероятности по всем n-граммам корпуса аналогична, за исключением того, что считывание и перемещение ведется сразу по двум множествам. Тут так же предполагается, что используемая база данных поддерживает атомарные операции такого вида. В противном случае могут возникнуть конфликты записи данных и есть возможность, что алгоритм будет многократно отрабатывать с ограниченным объемом одних и тех же данных.

Использование множеств в данном контексте может показаться странным. Более того, оно может показать не самым эффективным решением. Однако это не так. Выбор терминологии и структур хранения данных был сделан из практических соображений. В ходе проектирования и промежуточных реализаций системы было опробовано много различных вариантов, которые описаны в ниже.

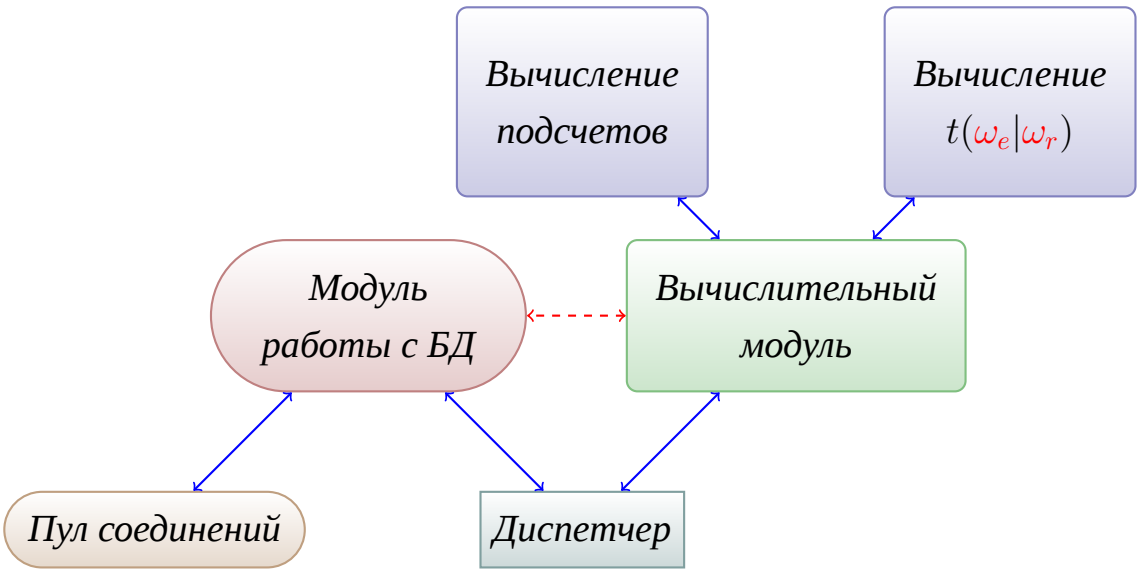

Рис. 1.8. Схема модулей приложения «обработчик».

Выше представлена схема модулей приложения обработчика. Стрелками показаны потоки данных. Прерывистыми линиями изображены косвенные связи.

Для системы крайне критично время доступа к базе данных, так как для всех численных операций база используется как временное хранилище данных между итерациями алгоритма обучения. Именно поэтому мы на архитектурной схеме изображаем такую техническую деталь как пул соединений.

1.4.3. ДЕКОДЕР

Согласно описанному ранее алгоритму жадного инкрементного поиска декодер имеет два режима работы.

В первом режиме работы на вход принимается исходный текст. Далее ищется ее наиболее вероятный эквивалент, последовательным разбиением фразы на n-граммы наибольшего размера (аналогично алгоритму системы перевода основанной на примерах в приложении), и параллельным поиском их в базе данных. После того, как эквивалент всего текста был найден, вычисляется величина неопределенности текста:

$$\text{ВН} = 2^{-\left(\frac{1}{S_{\eta e}}\sum_{i=1}^{S_{\eta e}}\log_2 P(\eta_{ei})+\frac{1}{S_{\omega e}}\sum_{j=1}^{S_{\omega e}}\log_2 P(\omega_{rj}|\omega_{ej})\right)}$$

- η_e — n-граммы высокого порядка найденные в созданном тексте;
- S_{η_e} — количество таких n-грамм;
- $P(\eta_e)$ — вероятность n-грамм согласно языковой модели (вычисляется как указано раннее);
- ω_e — n-граммы (слова) как результат перевода согласно модели перевода;
- S_{ω_e} — количество таких n-грамм (слов);
- $P(\omega_{rj}|\omega_{ej})$ — вероятность перевода фразы ω_{ej} на ω_{rj}.

Во втором режиме работы на вход принимается исходный текст (ИТ), переводной текст (ПТ) с предыдущей итерации и величина неопределенности (ВН). Применяются операции описанные выше для алгоритма жадного инкрементного поиска, если полученное новое значение имеет меньшую величину неопределенности чем предыдущее, то оно подается на выход. В противном случае итерации продолжаются.

Интерфейсная часть декодера представляет собой набор отдельных приложений, которые вызывают функции декодера при поступлении запросов на перевод. Общая схема декодера представлена ниже. Стрелками показаны потоки данных. Прерывистыми линиями изображены косвенные связи.

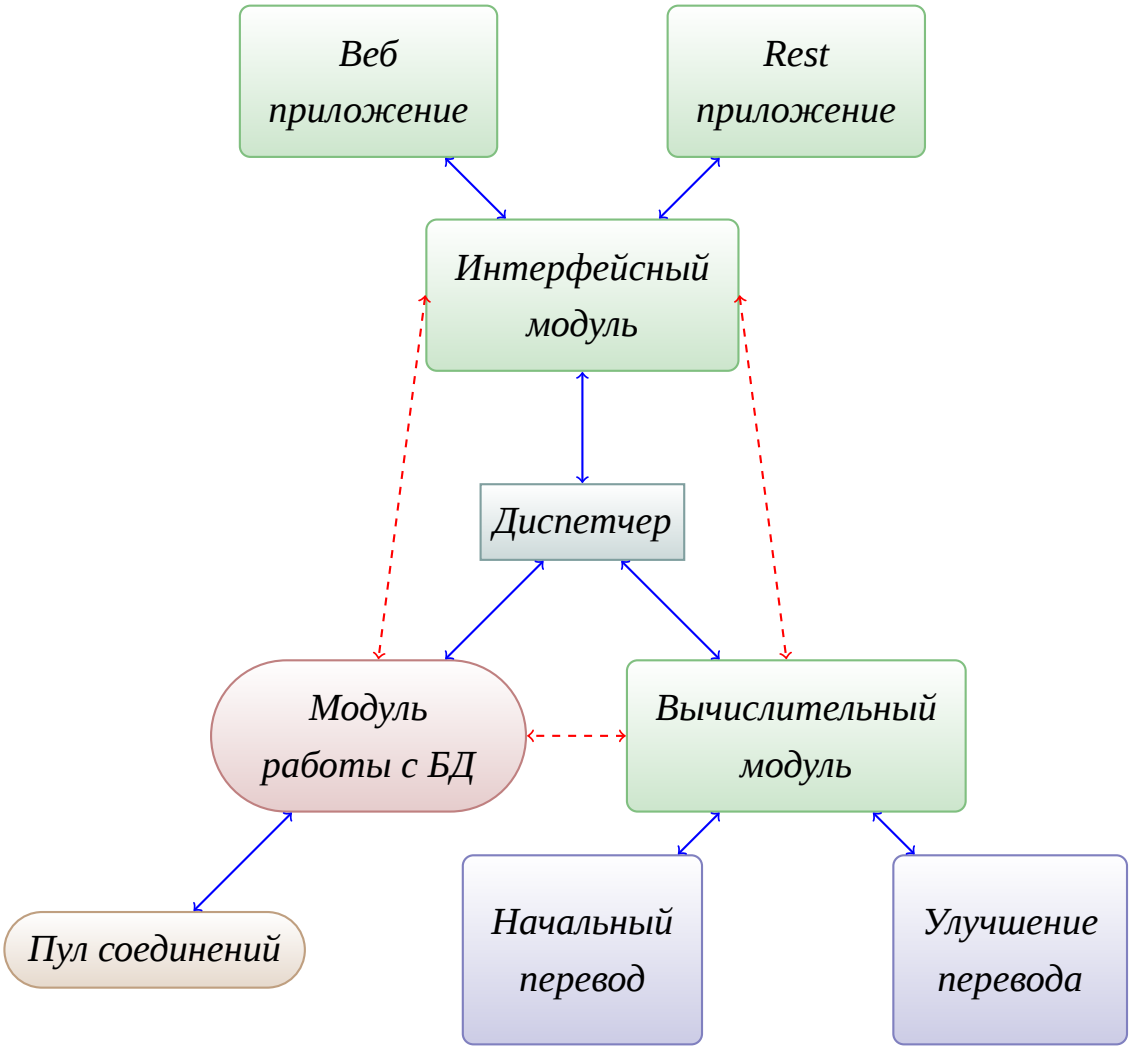

Рис. 1.9. Схема модулей приложения «декодер».

Rest-приложение представляет из себя HTTP RESTful веб-службу. К ней по определенному адресу поступает HTTP запрос POST вида. В теле запроса передается исходный текст для перевода, который отправляется для декодирования. После первой итерации декодирования Rest-приложение, отвечает запросившему клиенту первым «наихудшим» вариантом перевода. Далее не разрывая соединения генерируются другие варианты перевода и сразу передаются клиенту. Это продолжается пока клиент самостоятельно не разорвет соединение.

Веб-приложение представляет из себя обычный веб-ресурс, где пользователю предлагается ввести исходный текст для перевода. После осуществления первой итерации перевода, пользователю будет предложено улучшить полученный вариант перевода. Количество итераций по улучшению перевода определяет в данном случае сам пользователь.

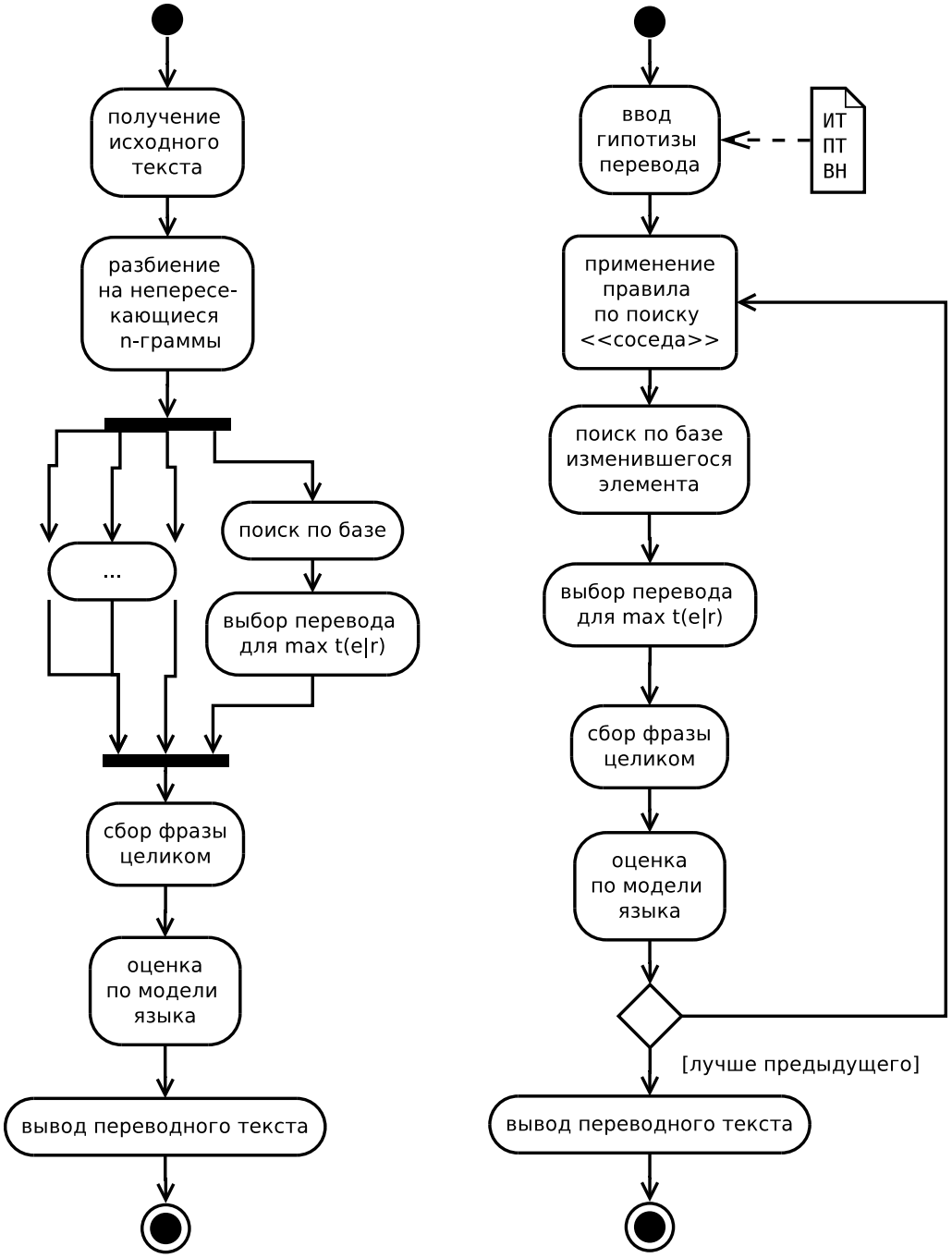

Рис. 1.10. Диаграммы активности приложения «декодер».

1.4.4. СРАВНЕНИЕ С РЕШЕНИЯМИ ДРУГИХ ПРОИЗВОДИТЕЛЕЙ

На данный момент существует несколько открытых платформ для создания систем машинного перевода. Наиболее крупные из них Moses и Chaski. Принципы работы этих систем построенных на основе той и другой платформы во многом сходны, только с тем отличием, что Chaski позволяет строить распределенные системы. Для обучения в обоих случаях используется библиотека GIZA++, однако Chaski включает в себя многопроцессорную версию этой библиотеки.

Для тестирования разрабатываемой системы, были созданы еще две ССМП на основе Moses и Chaski. Ниже приведены сравнения этих систем с разработанной.

Таблица 1.1. Сравнение модулей обучения для различных систем.

Система	Распределенная	Отказоуст.
Текущая	Да	Да
Moses (GIZA++)	Нет	Нет
Chaski (MGIZA++)	Да	Нет

Таблица 1.2. Сравнение модулей декодирования для различных систем.

Система	Распределенная	Отказоуст.	Web	Rest
Текущая	Да	Да	Да	Да
Moses (GIZA++)	Возможно	Нет	Да	Нет
Chaski (MGIZA++)	Возможно	Нет	Возможно	Нет

Самым большим достоинством разработанной системы является то, что она ориентирована на перевод текстов исключительно научно-технического характера, которые имеют явную стилистически подчеркнутую принадлежность к жанру научной прозы. Такая узкая специализация системы является и основным недостатком.

1.5. РЕАЛИЗАЦИЯ ССМП

1.5.1. ИСПОЛЬЗОВАНИЕ КОРПУСА

В ходе работы мы попытались построить свой собственный корпус для обучения системы на основе «Лолиты» Набокова. Но проблема заключается в том, что такой корпус крайне сложно выровнять. Русский и английский варианты местами сильно отличаются. При попытке разбить на предложения наивным методом (по знакам препинания) расхождение возникло на 11-м предложении. При попытке разбить на предложения с помощью подробного синтаксического анализа, ошибка возникла примерно на 200-м предложении. Возможно, имеет смысл сначала разбивать текст на абзацы, а внутри абзацев разбивать на предложения. Ошибка будет иметь меньшее влияние. Однако сложно утверждать, что такой вариант не обнаружит расхождения в абзацах и в любом случае потребуется ручная правка и много времени. Для маленьких корпусов существуют более сложные методы, такие как метод Потемкина-Кедровой [52], но чаще всего они предполагают наличие заранее подготовленного словаря.

Обычно построение большого корпуса ведется с учетом гипотезы Гайла-Черча, о том, что существует прямая зависимость между длинами предложений в исходном тексте и в переводе. В рамках данного подхода каждой паре предложений из текстов на разных языках, соответствует некоторая характеристика возможности сопоставления. Эта характеристика вычисляется на основе разницы длин предложений (в символах, включая пробелы) и дисперсии этой разницы. Далее, с помощью методов динамического программирования, находится такое соответствие предложений, при котором характеристики возможности сопоставления максимальны [41]. Само по себе построение корпуса является самостоятельной задачей корпусной лингвистики. Потому было принято решение пользоваться уже готовыми корпусами.

Для обучения рассматриваемой ССМП использовалась комбинация из нескольких русско-английских корпусов:

а) корпус резолюций ООН;

б) русско-чешско-английский корпус UMC (ÚFAL Multilingual Corpora);

в) некоторая выборка из Национального корпуса русского языка (была выкачена с веб-сайта корпуса и автоматически сформирована по атрибутам тегов);

г) некоторая выборка из работы Д. Кнута «Искусство программирования» в качестве примера корпуса научных текстов.

Общий объем корпуса составил примерно 180‘000 предложений (5‘854‘095 лексических единиц) на каждом из языков.

Важно отметить, что сложно назвать полученную комбинацию корпусом научных текстов, но за неимением хороших исходных данных пришлось довольствоваться этим.

Большие корпуса научного всего скорее у издательств, и если мы захотим развивать систему дальше, то можно будет по определенной лицензии эти корпуса приобрести.

Корпуса не обязательно будут выровнены по предложениям, но опираясь на свойства научного текста выравнивать их самостоятельно достаточно просто. Например это можно будет сделать, благодаря наличию формул и международных терминов.

1.5.2. СРЕДСТВО РАЗРАБОТКИ

В качестве основного инструментального средства используется функциональный язык erlang.

Основные преимущества:

а) нет побочных эффектов присущих императивным языкам;

б) модель легковесных процессов;

в) удобные средства разработки распределенных приложений;

г) высокая производительность (меньше чем для C, но сопоставима с C++, при компиляции erlang в машинный код).

Очень часто erlang используют для написания высоконагруженных распределенных приложений. Рассматриваемая система относится именно к таким приложениям.

В ходе выполнения работы как альтернативный вариант рассматривался язык программирования python, но после анализ данных производительности в рамках данной задачи предпочтение было отдано erlang.

Для увеличения надежности приложений было решено организовать взаимодействия процессов с учетом принципов ОТП (открытая телекоммуникационная платформа — OTP, open telecom platform), там где это возможно. Диспетчер каждого приложения и модуль пула соединения с базой данных имеет структуру дерева контроля («дерева супервизии»), как показано ниже на рисунке. Рабочий процесс может быть обычным вычислительным сервером (в терминах OTP), или супервизором (более низкого уровня). Супервизор в данном случае нужен для контроля за рабочими процессами. Если у какого либо рабочего процесса произойдет сбой и процесс «упадет», то супервизор перезапустит его.

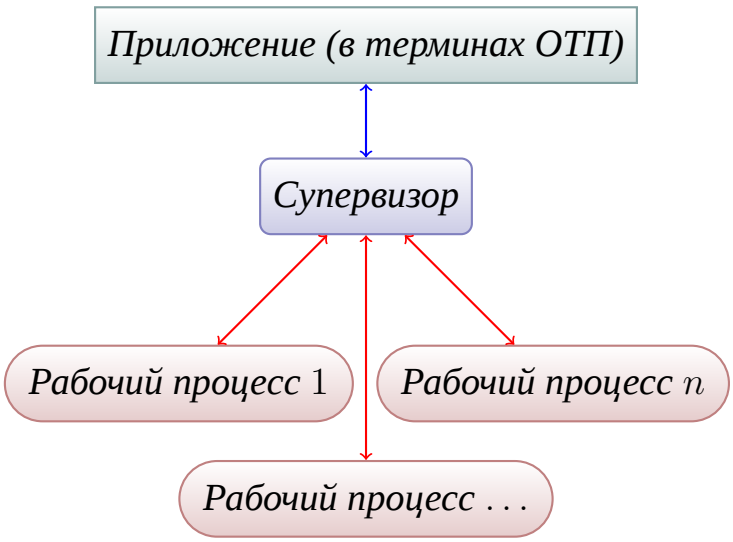

Рис. 1.11. Дерево контроля ОТП.

Если сбои будут происходит в течение заранее определенного промежутка времени то супервизор передаст сигнал о том, что про процесс перестал работать вверх по дереву контроля.

Если приложение (в терминах ОТП) контролируется другим супервизором, то процесс повторится снова. И так пока не дойдет до самого верхнего уровня, что приведет к отказу самого приложения.

1.5.3. БАЗА ДАННЫХ

В качестве базы данных используется Redis Server. Ранее были рассмотрены прочие варианты:

1) Хранить все словари в памяти, а по окночании скидывать на диск — очень быстро, но опасно, ибо когда память кончится программа начинает работать не стабильно.

2) Использовать локальное отображение в файл — но такой вариант:

   - очень медленный;
   - не масштабируемый.

3) Попробовали в качестве базы данных использовать Memcache, но оно оказалось медленно, и нет поддержки атомарных операций.

4) Попробовали в качестве базы данных использовать родные для erlang dts и mnesia, но первая имеет ограничение в размере хранимых данных, вторая в количестве хранимых ключей.

В результате долгих поисков и анализа, мы пришли к текущему варианту Кроме того, Redis версии ниже 2.4 оказался чувствителен к длине ключей и значений.

Важной особенностью Redis является поддержка атомарных операций обновления, работа с множествами и списками, работа с хранилищами типа ключ-значение. Для реализации алгоритмической части были крайне необходимы следующие операции:

а) для работы с n-грамами:

- sadd key value — добавить элемент в множество,
- srandmember key — получить случайный элемент множества,
- smove source destination member — переместить элемент из одного множества в другое;

б) для работы со списками n-грам:

- lpush key value — добавление элемента в начало списка,
- rpush key value — добавление элемента в конец списка,
- lpop key — получение первого элемента списка и его удаление,
- rpop key — получение последнего элемента и его удаление,
- rpoplpush srckey dstkey — получение и удаление последнего элемента списка «srckey» с добавлением его в «dstkey»;

в) для работы с подсчетами и оценкой вероятности:

- hset key field value — установить значение поля,
- hget key field — получить значение поля,
- hincrby key field integer — увеличить значение поля «field» на заданную величину «integer»,
- hexists key field — проверить существование поля.

Все эти операции имеют константное время доступа.

Учитывая характер системы, достаточно узким ее местом является скорость доступа к базе данных, особенно, если база является удаленной. Наличие асинхронных операций записи при вычисления подсчетов в данном случае является огромным преимуществом, Мы используем параллельный доступ к базе данных. Это особенно оказывается критично на синхронных операциях чтения.

Для параллельного доступа к базе данных, и исключения блокировок мы используем пул соединений с базой данных. Ниже на рисунке представлено дерево контроля потоков пула соединения.

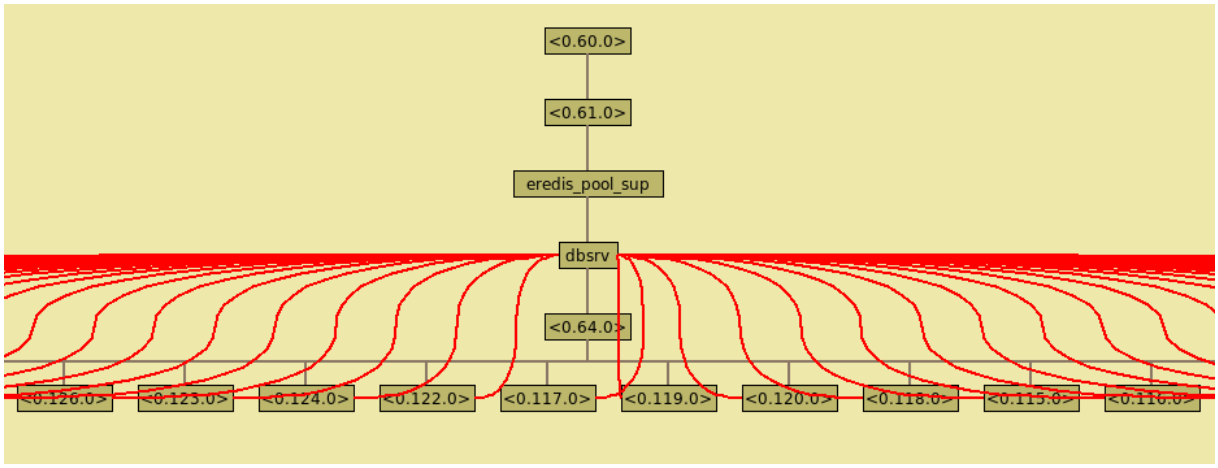

Рис. 1.12. Дерево контроля для пула соединения.

1.5.4. АРИФМЕТИКА С ФИКСИРОВАННОЙ ЗАПЯТОЙ

В результате деления при осуществлении сбора подсчетов используется вещественное деление. Для инкремента подсчетов удобно использовать атомарную операцию hincrby. Однако, она работает только с целыми числами. В этом случае мы предлагаем следующий подход.

В качестве вероятности, мы будем использовать числа с фиксированной запятой по основанию 576460752303423488 (2^{59}). Такое число выбрано не случайно, это максимальное положительное число используемое в erlang, которое может поместиться в машинное слово 64-битной архитектуре.

$$Base := 576460752303423488;$$

$$\text{adiv}(X, Y) \to \left\lfloor \frac{Base}{Y} \cdot X \right\rfloor.$$

$$\text{amul}(X, Y) \to \left\lfloor \frac{X}{Base} \cdot Y \right\rfloor.$$

При таком подходе возможно появления ошибок гораздо большего порядка, чем существующее машинное «эпсилон», но они все равно будут пренебрежимо малы. Для скорости вычислений и экономии памяти, важно соблюсти последовательность вычислений.

По проведенном тестам прирост скорости составил примерно 14 раз при применении операции hincrby с числами с фиксированной запятой, по сравнению с последовательными считывании и записью в базу данных.

1.5.5. ОБУЧЕНИЕ СИСТЕМЫ

Основой системы являются базовые алгоритмы моделей IBM 1 и IBM 2 (можно найти в приложении). Причем с точки зрения эффективности автор отдает предпочтении IBM 1. При использовании IBM 2 при падении производительности улучшения качества работы системы не замечено было.

В большинстве случаев для полной сходимости было достаточно около 10 итераций. Потому в рабочей системе предлагается условие сходимости заменить проверкой значения счетчика цикла.

ИНИЦИАЛИЗАЦИЯ

В базовом алгоритме указано, что необходимо перед началом его работы инициализировать величины $t(\omega_e|\omega_r)$ для всех ω_e и ω_r в параллельном корпусе. Логично было бы для этого использовать абсолютную частоту слова в каждом подкорпусе. Однако, для этого необходимо прочитать весь корпус целиком. Если внимательно посмотреть на базовый алгоритм, то становится ясно, что от начального значения величин $t(\omega_e|\omega_r)$ ничего не зависит, если они все одинаковы. В таком случае, можно их инициализировать заранее какой-либо константой, например в данной работе используется константа равная 0.5.

ВОЗМОЖНЫЕ ОШИБКИ

Очень важным моментом реализации алгоритма обучения является то, что оценка вероятности модели и вычисления подсчетов должны происходить сразу по всему корпусу текстов.

На начальных этапах реализации системы нами была допущено серьезная ошибка. Приложения читателя, проходила по корпусу и считывало 1000 строк. После этого эта тысяча строк отправлялась на обработки и по ней строилась модель перевода. Затем считывалась следующая порция строк.

В результате этих действий подсчеты вычислялись только для локального набора строк, и затем по ним вычислялась вероятность переводных соответствий и перезаписывала вероятность вычисленную ранее.

После окончания работы алгоритма обучения мы получали вероятности переводных соответствий обученные только на последней тысяче строк, с некоторым влиянием остального корпуса.

Установить ошибку удалось только после запуска алгоритма в предельном случае на корпусе состоящим из двух строк, при проходе полного алгоритма обучения на каждой из них.

1.5.6. РЕАЛИЗАЦИЯ ИНТЕРФЕЙСОВ ДЕКОДЕРА

Рассматриваемая система обладает тремя видами интерфейса:

- консольные;
- простой веб-интерфейс;
- rest-интерфейс.

Консольное приложение является стандартным отладочным интерфейсом ОТП приложений. Для перевода достаточно просто вызывать соответствующую функцию модуля.

Веб-интерфейс декодера выполнен в виде отдельного ОТП приложения и представляет из себя полноценный веб-ресурс, реализованный с помощью библиотеки mochiweb.

Мы не будем в работе приводить как внешне выглядит клиентская часть Веб-приложения, которая отображается в браузере, так как она представляет собой всего лишь текстовый элемент ввода, две кнопки («перевести» и «улучшить перевод»), и текстовый элемент вывода.

Важнейшей частью Веб-приложения является XSLT-преобразователь. В данном случае он был написан на C с использованием libxml2 и libxslt. Из erlang он вызывается как порт-сервер. На вход подается строка XML которая содержит в себе данные (в нашем случае, это перевод исходного текста в случае ответа на запрос) и шаблон XSL, в котором описано как данные передавать. Для увеличения быстродействия XSLT преобразователь (на стороне кода на C) использует хэш-таблицу, в которой записываются использованные ранее XSL шаблоны. Сделано для того, чтобы не загружать при повторном обращении каждый шаблон снова. В рамках проекта было реализовано две версии преобразователя с запоминанием шаблонов и без него. Версия без хэш-таблицы работает медленнее, но удобнее при «горячей замене» шаблона уже на работающем коде. Кроме того, преобразователь является не блокирующим и при одновременных обращениях вызывается параллельно. Последнее дает прирост производительности при наличие нескольких процессоров или ядер.

Rest-интерфейс декодера также выполнен в виде отдельного ОТП приложения и представляет из себя полноценный веб-ресурс реализованный с помощью библиотеки mochiweb.

После поступления POST запроса приложению, оно начинает передавать текстовый (Content-Type: text/plain) HTTP (chunked) поток с наиболее вероятными вариантами перевода. Этот поток будет передаваться до тех пор, пока клиент не закроет соединение, или не будут перебраны все варианты улучшения перевода.

Поток представляет из себя набор чанков в рамках спецификации протокола HTTP 1.1. Каждый чанк содержит количество передаваемых байт и последовательность передаваемых данных.

Каждая строка полного перевода исходного текста передается в отдельном чанке и обязательно заканчивается символом перевода строки.

В отличие от web-интерфейса в этом случае клиенту не нужно будет самому запрашивать улучшение перевода. Самый первый «жадный» вариант перевода передается в первом чанке потока.

1.6. ТЕСТИРОВАНИЕ РАЗРАБОТАННОЙ ССМП

1.6.1. ОЦЕНКА КАЧЕСТВА ПЕРЕВОДА

Для оценки качества перевода достаточно взглянуть на таблицу ниже. В ней приведены переводы одного и того же фрагмента для различных статистических систем машинного перевода.

Таблица 1.3. Различные переводы исходного текста

Исходный текст	adopted at the 81st plenary meeting
Проф. переводчик	принята на 81-м пленарном заседании
Moses	приняли на 81 полный встречи
Google-переводчик	принятой на 81-е пленарное заседание
Данная система	принята без голосования на 81 пленарном заседании в Брюсселе

Все правильно, заседании было в Брюсселе, но в исходном тексте этого не было. Однако не совсем корректно сравнивать системы, обученные на разных корпусах текстов. Потому мы будем использовать систему машинного перевода основанную на пакете Moses (декодировщик Moses и обучающая система GIZA++), обученную на том же самом входном корпусе, что и наша система.

Для оценки качества перевода существует набор формальных метрик.

- BLEU (Bilingual Evaluation Understudy);
- METEOR (Metric for Evaluation of Translation with Explicit ORdering);
- NIST (названа по имени университета National Institute of Standards and Technology);
- WER (Word error rate).

Чаще всего системы машинного перевода оценивают по метрике BLEU:

- высокая корреляция с оценкой человека;
- для оценки требуется готовый перевод тестовых предложений, выполненный профессиональным переводчиком.

$$\text{BLEU} = Bp \cdot e^{\left(\sum_{n=1}^{N} W_n \log(p_n)\right)}$$

$$Bp = \begin{cases} 1, & l_c > l_h; \\ e^{\left(1-\frac{l_h}{l_c}\right)}, & l_c \leqslant l_h. \end{cases} \quad \text{и} \quad p_n = \frac{\sum\limits_{C \in S_c} \sum\limits_{\eta_c \in C} \textit{число}_{\text{среза}}(\eta_c)}{\sum\limits_{C \in S_c} \sum\limits_{\eta_c \in C} \textit{число}(\eta_c)}$$

- S_c — множество кандидатов на перевод;
- C — кандидат на перевод;
- η_c — n-грамма кандидата на перевод;
- l_c — длинна кандидата перевода;
- l_h — длинна экспертного перевода (выполненного человеком);
- $W_n = \frac{1}{N}$ — вес, для метрик основанных на BLEU, но не эквивалентных ей, может отличаться для каждого n, например NIST назначает больший вес более редким n-граммам;
- $N = 4$, n-грамность оценки.

Таким образом, BLEU есть взвешенное среднее числа совпадений n-грамм (слов) кандидата и n-грамм в переводе эксперта. Метрика является инвариантом порядка n-грамм, важно наличие совпадений [34]. Чем выше значение метрики BLEU, тем перевод хуже.

Метрики NIST и METEOR основаны на BLEU, WER — на вычислении расстояния Левенштейна.

Для оценки перевода система мы использовали метрику BLEU. Для подсчета метрики было выделено 1024 предложений в качестве примеров экспертного перевода, и 1024 предложений было переведено рассматриваемой системой МП. Для этого набора предложений метрика BLEU = 0.243 для

единственной итерации работы декодера, и 0.209 для лучшего варианта из 100 полученного в результате нескольких итераций. Система МП, построенная на основе пакета Moses, дает метрику BLEU = 0.209 для модели IBM 3 и BLEU = 0.173 для модели IBM 5.

Таблица 1.4. Оценки перевода текста

Система	**BLEU**
Текущая (1)	0.243
Текущая (100)	0.209
Moses (IBM 3)	0.201
Moses (IBM 5)	0.173

Высокие значения метрики BLEU показывают, что система проявляет себя не в лучшем свете на тестовом корпусе, однако сам корпус весьма отличается отличается от тех для которых создана данная система.

Существует целый ряд неформальных способов оценки систем машинного перевода. Чаще всего используют технику «обратного перевода».

- исходный отрывок переводят системой с языка L_1 на язык L_2;
- полученный отрывок переводят в обраПритную сторону с языка L_2 на язык L_1.
- сравнивают два варианта отрывка на языке L_1 и по их близости судят о качестве перевода.

На наш взгляд такой метод оценки не является вполне корректным в случае статистических систем. Так как согласно уравнению Байеса, в итоге мы оцениваем только модель языка используемой системы. Более того при многократном применении «обратного перевода» к одному и тому же отрывку, с некоторой итерации перевод на L_1 и на L_2 перестанут меняться.

1.6.2. ОЦЕНКА СКОРОСТИ

ОЦЕНКА СКОРОСТИ ОБУЧЕНИЯ

Ниже мы сравним скорости обучения и работы трех систем машинного перевода. Для того чтобы показать разницу распределенных (многопоточных) и не распределенных систем эксперименты проводились на разных машинах. Для тестирования рассматриваемой системы было на каждой машине запускались соответствующие приложения (читатель и обработчик), пропорционально числу ядер процессоров. Тестирование как и ранее проводилось на смешанном корпусе описанном выше.

Высокая скорость текущей (представленной в работе) системы во многом объясняется ее простотой. Однако, для более крупных корпусов текста, скорость обучения может оказаться критичной.

Оценка скорости обучения на одном и том же корпусе.

Таблица 1.5. Оценка скорости обучения, процессор: Intel Core2 Duo, 1 ядро 64 бит, ОП 4Гб, ФС: ext4

Система	Время, ч
Текущая (1)	≈ 5
Moses (GIZA++)	≈ 25
Chaski (MGIZA++)	≈ 26

Таблица 1.6. Оценка скорости обучения, процессор: Intel Xeon E5506, 8 ядер 64 бит, ОП 10Гб, ФС:xfs

Система	Время, ч
Текущая (1)	≈ 1
Moses (GIZA++)	≈ 22
Chaski (MGIZA++)	≈ 3

ОЦЕНКА СКОРОСТИ ПЕРЕВОДА

Скорость перевода будем оценивать в микросекундах. Причем, скорость запуска декодера (Moses стартует в течение минуты на Intel Core2 Duo) и время на прочие накладные вычисления учитывать не будем.

Таблица 1.7. Оценка скорости перевода, процессор: Intel Core2 Duo, 1 ядро 64 бит, ОП 4Гб, ФС: ext4

Система	Время, μс
Текущая (1)	1132
Текущая (100)	7108124
Moses (IBM 3)	$\approx$ 10000000
Moses (IBM 5)	$\approx$ 30000000

Таблица 1.8. Оценка скорости перевода, процессор: Intel Xeon E5506, 8 ядер 64 бит, ОП 10Гб, ФС: xfs

Система	Время, μс
Текущая (1)	1012
Текущая (100)	1119024
Moses (IBM 3)	$\approx$ 5000000
Moses (IBM 5)	$\approx$ 6000000

Как можно заключить из приведенных тестов использование параллельных вычислений позволяет достичь значительного прироста производительности на этапе обучения системы машинного перевода.

На этапе декодирования особенный рост наблюдать трудно. Это будет весьма заметно на большом количестве одновременных запросов.

2. ЭКОНОМИЧЕСКАЯ ЧАСТЬ

2.1. ВВЕДЕНИЕ

По мере того как расширяется информатизация современного общества, возрастает значение прикладной (вычислительной, компьютерной, инженерной) лингвистики, науки, находящейся на стыке глубоко человечной, гуманитарной науки лингвистики (языкознания), изучающей законы развития и пользования могучим средством мышления и коммуникации — языком, — и компьютерного знания, с помощью которого машине передастся все большая часть интеллектуального труда человека [44].

В данной главе будет рассмотрена экономическая сторона вопроса. Вначале будет построена сетевая модель работ над проектом и её графическое представление. Будут рассчитаны ранние и поздние сроки начала и завершения работ над проектом, найден критический путь и его продолжительность, вероятность завершения комплекса работ по проекту в срок. После этого будет рассчитана сумма расходов на разработку проекта. Будет подсчитана цена разработанной системы, капитальные вложения, связанные с её внедрением, а также расходы, связанные с её эксплуатацией.

2.2. ПОСТРОЕНИЕ СЕТЕВОЙ МОДЕЛИ

Сложность и комплексность разработки и реализации описанного проекта, необходимость параллельного выполнения работ, зависимость начала многих работ от результатов других, значительно осложняют планирование разработки.

При наличии множества взаимосвязанных работ, наиболее удобными являются системы сетевого планирования и управления (СПУ). Они основанны на применении сетевых моделей планируемых процессов, которые допускают использование современной вычислительной техники.

Сетевые модели планируемых процессов позволяют быстро определить последствия различных вариантов управляющих воздействий и находить наилучшие из них.

СПУ дают возможность своевременно получать достоверную информацию о состоянии дел, о возникших задержках и возможностях ускорения хода работ, концентрируют внимание на «критических» работах, определяю-

щих продолжительность проведения разработки в целом, заставляют совершенствовать технологию и организацию работ, непосредственно влияющих на сроки проведения разработки, помогают составлять рациональные планы работ.

2.2.1. ПЕРЕЧЕНЬ РАБОТ И СОБЫТИЙ

Составим полный перечень событий и работ сетевой модели. Каждая работа имеет определенную продолжительность. Однако не всегда заранее известно точное время выполнения работ, поэтому дадим продолжительности каждой работы две вероятностные оценки:

- минимальную (оптимистическую) — t_{min};
- максимальную (пессимистическую) — t_{max}.

Эти величины являются исходными для расчета ожидаемого времени выполнения работ: .

$$t_{\text{ожидаемое}} = \frac{3 \cdot t_{min} + 2 \cdot t_{max}}{5}$$

Рассчитаем дисперсии работ по формуле:

$$\delta^2 = \left(\frac{t_{max} - t_{min}}{5} \right)^2$$

Таблица 2.9. Перечень работ и событий, t_{min}, t_{max}, $t_{ож}$ измеряются в днях.

№	Наименование события	Код работы	Наименование работы	t_{min}	t_{max}	$t_{ож}$	δ^2
0	Начало работ	0 — 1	Анализ проблемы и составление плана-графика	2	5	3.8	0.16
1	Завершение анализа и составления плана	1 — 2	Исследование существующих статистических систем машинного перевода	4	10	6.4	1.44
		1 — 3	Изучение математических основ построения статистических систем машинного перевода	10	20	14.0	4.0
		1 — 4	Изучение лингвистических основ машинного перевода	7	10	8.2	0.36
		1 — 5	Изучение возможных варинтов хранения данных в рамках задачи машинного перевода	3	4	3.4	0.04
2	Завершение исследования существующих систем	2 — 6	Составление требований и ограничений системы	1	2	1.6	0.04
3	Завершение изучения математических основ построения статистических систем машинного перевода	3 — 7	Разработка численного алгоритма обучения системы	5	7	5.8	0.16
		3 — 8	Разработка алгоритма поиска верного варианта перевода на основе обученной модели	6	8	6.8	0.16
4	Завершение изучения лингвистических основ машинного перевода	4 — 9	Составление требований к входным данным численного алгоритма	1	2	1.6	0.04
		4 — 8	Составление требований к выходным данным алгоритма поиска	1	2	1.6	0.04
5	Завершение изучения возможных вариантов хранения данных	5 — 7	Разработка структуры хранения данных	2	3	2.4	0.04
6	Завершение составления требований к системе на основе аналогов	6 — 10	Разработка распределенной архитектуры	3	4	3.4	0.04
7	Завершение разработки численного алгоритма и структру хранения данных	7 — 10	Разработка работающей обучающейся модели на тестовых входных данных	5	7	5.8	0.16
8	Завершение разработки алгоритма поиска и выработки требований к выходным данным	8 — 13	Разработка работающего поискового модуля на тестовых входных данных	6	9	7.2	0.36

Таблица 2.9 *(продолжение)*. Перечень работ и событий, t_{min}, t_{max}, $t_{ож}$ измеряются в днях.

№	Наименование события	Код работы	Наименование работы	t_{min}	t_{max}	$t_{ож}$	δ^2
9	Завершение составления требований к входным данным численного алгоритма	9 — 11	Подбор нужных корпусов текстов (корпуса текста все еще остаются тестовыми, но составляются на основе реальных данных)	5	7	5.8	0.16
10	Окончание разработки распределенной архитектуры и и модели обучающейся на тестовых входных данных	10 — 12	Разработка распределенной обучающейса системы	2	4	2.8	0.16
11	Окончание подбора нужных корпусов текстов	11 — 12	Разработка алгоритмов предварительной обработки входных корпусов текста	1	2	1.4	0.06
12	Окончание разработки распределенной обучающейса системы и алгоритмов предварительной обработки входных	12 — 13	Корректировка системы с учетом входных данных	1	2	1.4	0.06
13	Окончание корректировки обучающейся системы и разработки работающего поискового модуля	13 — 14	Тестирование приложения в совокупности отладка всей системы	10	15	12.0	1.0
14	Завершение отладки	14 — 15	Прогон системы на реальных корпусах текста	1	2	1.5	0.06
15	Завершение работ						

2.2.2. ГРАФИЧЕСКОЕ ПРЕДСТАВЛЕНИЕ СЕТЕВОЙ МОДЕЛИ

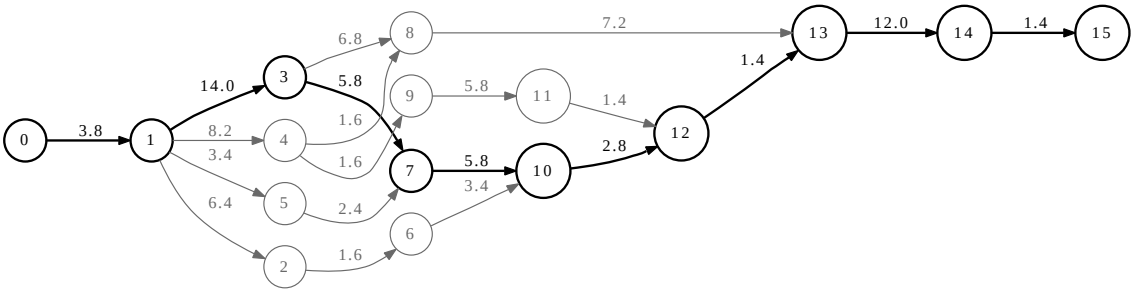

Рис. 2.13. Сетевой граф работ и критический путь.

Построенный граф состоит из 15 событий (вершины графа). Дуги графа — работы. Каждая дуга графа подписана сверху соответствующей ей продолжительностью работ.

Построенный граф удовлетворяет условию независимости событий i от события j для все $i > j$. Выполнение этого условия очевидно, так как ни одна дуга не заканчивается в вершине с номером, меньшим, чем номер вершины из которой это дуга начиналась. Это позволяет корректно выполнять дальнейшие расчеты.

2.2.3. РАСЧЕТ ПАРАМЕТРОВ СЕТЕВОЙ МОДЕЛИ

Рассчитаем некоторые характеристики сетевой модели. Характеристики сетевой модели позволяют определить степень напряженности всего комплекса работ в целом и каждой работы в отдельности, а также принять решение о перераспределении ресурсов.

Для всех работ рассчитаем следующие показатели:

- ранний срок начала работы: $t_{\text{рн},i-j} = \max\limits_{k<i} (t_{\text{ожидаемое},k-i} + t_{\text{рн},k-i})$;
- ранний срок окончания работы: $t_{\text{ро},i-j} = t_{\text{рн},i-j} + t_{\text{ожидаемое},i-j}$;
- поздний срок начала работы: $t_{\text{пн},i-j} = t_{\text{по},i-j} - t_{\text{ожидаемое},i-j}$;
- поздний срок окончания работы: $t_{\text{по},i-j} = \min\limits_{k>j} (t_{\text{по},j-k} - t_{\text{ожидаемое},j-k})$;

- полный резерв времени: $r_{\text{п},i-j} = t_{\text{пн},i-j} - t_{\text{рн},i-j}$;
- свободный резерв времени: $r_{\text{с},i-j} = t_{\text{рн},j} - t_{\text{рн},i} - t_{\text{ожидаемое},i-j}$.

Таблица 2.10. Характеристики сетевой модели, t и r измеряются в днях.

Код работы	$t_{\text{ож}}$	$t_{\text{рн}}$	$t_{\text{ро}}$	$t_{\text{пн}}$	$t_{\text{по}}$	$r_{\text{п}}$	$r_{\text{с}}$	$t_{\text{ож. кр. пути}}$	$\delta^2_{\text{кр. пути}}$
1	**2**	**3**	**4** $= 2 + 3$	**5** $= 6 - 2$	**6**	**7** $= 5 - 3$	**8**	**9**	**10**
0 — 1	3.8	0.0	3.8	0.0	3.8	0.0	0.0	3.8	0.16
1 — 2	6.4	3.8	10.2	18.0	24.4	14.2	0.0	0.0	0.0
1 — 3	14.0	3.8	17.8	3.8	17.8	0.0	0.0	14.0	4.0
1 — 4	8.2	3.8	12.0	15.2	23.4	11.4	0.0	0.0	0.0
1 — 5	3.4	3.8	7.2	17.8	21.2	14.0	0.0	0.0	0.0
2 — 6	1.6	10.2	19.4	24.4	26.0	14.2	7.6	0.0	0.0
3 — 7	5.8	17.8	23.6	17.8	23.6	0.0	0.0	5.8	0.16
3 — 8	6.8	17.8	24.6	19.6	26.4	1.8	0.0	0.0	0.0
4 — 8	1.6	12.0	13.6	24.8	26.4	12.8	11.0	0.0	0.0
4 — 9	1.6	12.0	13.6	23.4	25.0	11.4	0.0	0.0	0.0
5 — 7	2.4	7.2	9.6	21.2	23.6	14.0	14.0	0.0	0.0
6 — 10	3.4	19.4	22.8	26.0	29.4	6.6	6.6	0.0	0.0
7 — 10	5.8	23.6	29.4	23.6	29.4	0.0	0.0	5.8	0.16
8 — 13	7.2	24.6	31.8	26.4	33.6	1.8	0.4	0.0	0.0
9 — 11	5.8	13.6	19.4	25.0	30.8	11.4	0.0	0.0	0.0
10 — 12	2.8	29.4	32.2	29.4	32.2	0.0	0.0	2.8	0.16
11 — 12	1.4	19.4	20.8	30.8	32.2	11.4	11.4	0.0	0.0
12 — 13	1.4	32.2	33.6	32.2	33.6	0.0	0.0	1.4	0.06
13 — 14	12.0	33.6	45.6	33.6	45.6	0.0	0.0	12.0	1.0
14 — 15	1.4	45.6	47.0	45.6	47.0	0.0	0.0	1.4	0.06
Суммарные время и дисперсия критического пути:								47.0	4.76

2.2.4. АНАЛИЗ СЕТЕВОЙ МОДЕЛИ

На основе посчитанных выше параметров, проведем анализ сетевого графика. Критически путь включает в себя лишь события с нулевым запасом времени. Таким путём является путь из вершин:

$$L_{\text{кр}} = 0 \to 1 \to 3 \to 7 \to 10 \to 12 \to 13 \to 14 \to 15$$

Суммарное время критического пути составляет $T_{\text{кр}} = 47$ дней, а его суммарная дисперсия — 4.76 дня. На графе критический путь выделен жирными стрелками.

Необходимо, чтобы продолжительность критического пути $T_{\text{кр}}$ не превышала продолжительности заданного директивного срока $T_{\text{дир}}$. Если $T_{\text{кр}} > T_{\text{дир}}$, то необходимо принять меры по уплотнению графика работ. В нашем случае директивный срок создания программного комплекса $T_{\text{дир}} = 63$ дня, а продолжительность критического пути $T_{\text{кр}} = 47$ дней, т.е. $T_{\text{кр}} < T_{\text{дир}}$.

Рассчитаем среднеквадратичное отклонение для продолжительности критического пути.

$$\sum_{i-j} \delta^2_{\text{кр. пути}} = 4.76 \Rightarrow$$

$$\Rightarrow \delta_{\text{кр. пути}} = \sqrt{\sum_{i-j} \delta^2_{\text{кр. пути}}} = \sqrt{4.76} \approx 2.181742422927143$$

Построим доверительный интервал:

$$\Delta T = T_{\text{кр}} \pm 3 \cdot \delta_{\text{кр. пути}} = [40.46; 53.54] .$$

Вычислим вероятность выполнения проекта в директивный срок. Для этого необходимо определить значение функции Лапласа (по таблице) в точке, соответствующей директивному сроку:

$$P = \Phi\left(\frac{T_{\text{дир}} - T_{\text{кр}}}{\delta_{\text{кр. пути}}}\right) = \Phi\left(\frac{63 - 47}{2.18}\right) = \Phi(7.334) = 0.999$$

Таким образом, вероятность завершения работы в директивный срок практически равна 1, то есть проект завершится точно в срок.

2.3. РАСЧЕТ ЗАТРАТ НА РАЗРАБОТКУ

Всю работу над проектом можно разбить на следующие этапы

1) Анализ проблемы, выделение ключевых задач и действий (работа 0 — 1).

2) Исследование принципов работы существующих систем статистического перевода, изучение теоретических основ (работы 1 — 2, 1 — 3, 1 — 4, 1 — 5, 2 — 6).

3) Разработка численных алгоритмов для работы системы (работы 3 — 7, 3 — 8, 4 — 9, 4 — 8).

4) Разработка структуры хранения данных и распеределенной архитектуры (работы 5 — 7, 6 — 10).

5) Реализация отдельных модулей приложения (работы 7 —10, 8 — 13, 10 — 12).

6) Прогон системы на данных приближенных к реальности, корректировка системы (работы 9 —11, 11 — 12, 12 — 13).

7) Тестирование и отладка (работы 13 — 14).

8) Прогон системы на реальных данных (работы 14 — 15).

Одной из основных статей расходов является заработная плата персонала, занятого в исследованиях и разработке при проведении данной дипломной работы. Расчет среднемесячной и среднедневной зарплаты работников, задействованных в проекте, приведен ниже.

Таблица 2.11. Расчет среднемесячной и среднедневной зарплаты работников.

№	ИТР	Количество сотрудников	Среднемесячная зарплата (руб.)	Количество рабочих дней в месяце	Среднедневная зарплата (руб.)
1	Системный архитектор	1	60000	21	2857.14
2	Лингвист	1	60000	21	2857.14
3	Математик	2	60000	21	2857.14
4	Разработчик	2	60000	21	2857.14
5	Тестировщик	1	60000	21	2857.14

Трудоемкость вычисляется по формуле:

$$Q(i-j) = t(i-j) \cdot A(i-j) \cdot f$$

- i, j — начальное и конечное события работы $E(i-j)$;
- $Q(i-j)$ — трудоемкость работы, чел/дн.;
- $A(i-j)$ — количество исполнителей, занятых выполнением работы $E(i-j)$;
- f — коэффициент перевода (при необходимости) рабочих дней в календарные, $f = 0.85$ для пятидневной рабочей недели или $f = 1.0$, если перевод в календарные дни не требуется.

В данном случае коэффициент перевода берется $0,85$. Вычислим расходы на зарплату персонала этапам работ обозначенным выше. Важно заметить, что для простоты в таблице зарплаты различных работников не различаются. В таблице выше мы уже привели зарплаты сотрудников — они одинаковы. В противном случае, придется вычислить среднедневную заработную плату, приходящуюся на одного человека в команде (не зависимо от его роли), и далее оперировать этой цифрой.

Таблица 2.12. Расходы на зарплату персонала.

№ этапа	Количество исполнителей (чел.)	Трудоёмкость (чел./дн.)	Среднедневная зарплата (руб.)	Зарплата (руб.)
1	1	3.23	2857.14	9228.56
2	4	28.55	2857.14	81599.91
3	4	13.42	2857.14	38371.39
4	2	4.93	2857.14	14085.70
5	3	13.43	2857.14	38371.39
6	3	7.31	2857.14	20885.69
7	6	65.28	2857.14	186514.09
8	1	1.275	2857.14	3642.853
Итого		137.425		392642.46

Из расчетов выше, следует, что суммарная трудоемкость (≈ 137 человеко-дней) значительно превышает длину критического пути (47 дней).
Это свидетельствует о том, что при данном планировании персонал используется достаточно эффективно. Основным результатом расчета этой таблицы является выявление суммы затрат на заработную плату.

$S_{\text{зп}} = 392642.46$ рублей.

Затрат на закупку программного обеспечения нет, т.к. для разработки планируется использовать открытые продукты.

Таблица 2.13. Затраты на оборудование.

№	Наименование	Количество	Цена (руб.)	Стоимость (руб.)
1	Ноутбук Sony Vaio	7	40000.00	280000.00
Итого				280000.00

Суммарная стоимость оборудования:

$S_{\text{об}} = 280000.00$ рублей.

Рассмотрим оплату Интернета как дополнительную статью расходов:

$S_{\text{Интернета}} = S_{\text{подключения}} + t \cdot S_{\text{мес}}$

- $S_{\text{подключения}} = 1500$ стоимость подключения, руб;
- $S_{\text{мес}} = 600$ — оплата безлимитного тарифа в месяц, руб;
- t — количество месяцев разработки, из расчета, что в месяце только 21 рабочий день;

$S_{\text{Интернета}} = 1500 + 3 \cdot 600 = 3300.00$ рублей.

Кроме того, для повышения качаства системы, возможно, ее придется тестировать на платных корпусах текста. Например:

- «European Corpus Initiative Multilingual Corpus I»;
- «Национальный корпус русского языка».

«European Corpus Initiative Multilingual Corpus I» доступен по цене 2000 рублей, в бессрочное пользование любого характера. Про коммерческую доступность «Национального корпуса русского языка» ничего пока не известно, потому мы не будем его учитывать. Тогда

$S_{\text{корп}} = 2000$ рублей.

Суммарные расходы на разработку могут быть вычислены по формуле:

$$S = S_{\text{зп}} \cdot (1 + \omega_d) \cdot (1 + \omega_c) + S_{\text{об}} + S_{\text{Интернета}} + S_{\text{корп}}$$

- $\omega_d = 0.2$ — коэффициент, учитывающий дополнительную заработную плату (премии);
- $\omega_c = 0.34$ — коэффициент, учитывающий страховые взносы во внебюджетные фонды.

$S = 916669.08$рублей.

Далее, вычислим цену полученной системы:

$$C = \frac{(1 + P_{\text{н}}) \cdot S}{n};$$

- $P_{\text{н}} = 0.2$ — норматив рентабельности, учитывающий часть чистого дохода, включенного в цену (может быть принят равным $0,2$);

- n — количество организаций, которые могут купить разрабатываемое программное обеспечение.

Подобная система может оказаться полезной прежде всего крупным бюро переводов, и крупным многоязычным интернет-порталам. Оценить число последних не представляется возможным. Общее число бюро переводов зарегистрированных в г. Москве насчитывает примерно ≈ 600. Будем считать, что, потенциально, каждое из них может купить данную систему. Тогда,

$C = 1833.33$ рублей.

Вообще, подобная система может оказаться полезной:

- издательствам, занимающимся переводом иностранной технической литературы;
- ведомственным организациям;
- военным организациям;
- конструкторским бюро и научно исследовательским центрам.

Однако, эти типы предприятий в данной оценке, мы использовать не будем. Кроме того, авторы работы убеждены, что системы подобного класса должны поставлять государственным учреждениям бесплатно. Капитальные вложения, связанные с внедрением в организации-пользователе новой программы, равны продажной стоимости системы. На данный момент эта стоимость составляет 1833.33 рублей. Расходы, связанные с эксплуатацией системы (на одну единицу техники, в год) могут быть определены следующим образом:

$$U = T_{\text{м.в.}} \cdot C_{\text{м.в.}} + \frac{C}{T}$$

- $T = 0.5$ — срок морального устаревания системы (условно примем за полгода) ;
- $T_{\text{м.в.}}$ — годовое машинное время вычислительной машины, необходимое для применения внедряемой системы (в данном случае наиболее

эффективно использовать вычислительный кластер, но с учетом высокой цены такого оборудования, вполне может подойти выделенный сервер, или даже простой персональный компьютер);

- $C_{\text{м.в.}} = 12$ рублей, стоимость часа машинного времени.

В данном случае, мы полагаем, что каждое из рассматриваемых бюро переводов обладает своей базой текстов. Таким образом, не будет необходимости в дополнительных расходах, на покупку сторонних платных корпусов текстов. В итоге получаем:

- при работе систему 760 часов в году: $U_{\text{маш},760} = 12786.00$ рублей;
- при работе систему 1993 часов в году (250 рабочих дней, при 40-часовой рабочей неделе): $U_{\text{маш},1993} = 27582.67$ рублей;
- при работе систему 8760 часов в году (в режиме 24 на 7 на 365): $U_{\text{маш},8760} = 108786.67$ рублей.

Кроме того, для использования системы, предприятию придется расширить парк машин. Однако, эти капиталовложения могут зависеть от нужд и возможностей самой компании, и колеблются в диапазоне от 12000 рублей до 13 млн. рублей (покупка вычислительного кластера класса Блэйд с 40 узлами).

2.4. ЦЕЛЕСООБРАЗНОСТЬ ПРИМЕНЕНИЯ СИСТЕМЫ

Системы машинного перевода целесообразно применять только для перевода научно-технической литературы. Это обусловлено стилистическими особенностями научного текста. В других случаях результат машинного перевода не представляет какой-либо ценности.

Системы машинного перевода (пока) не могут полностью заменить человека, однако, они значительно облегчают труд переводчика, делая его работу более эффективной.

Кроме того, системы машинного перевода могут быть необходимы:

- научным сотрудникам исследовательских центров — чтобы в кратчайшие сроки получить общее представление, о той или иной работе, на иностранном языке;
- военным и ведомственным организациям — когда уровень секретности, не всегда позволяет переводчику нужной специализации получить доступ к исходному тексту.

Основное преимущество данной системы, заключается, в том, что ее можно распростанять в коробочном варианте (распределенность желательна, но совсем не обязательна). Это является гарантией, что информация введенная для перевода не будет доступна сторонним лицам. Кроме того, благодаря, тому что система основана на статистике, ее можно настроить на тексты, заданной тематики, что будет весьма полезно первой и второй категории пользователей.

Системы машинного перевода могут быть выгодны:

- крупным бюро перевода;
- издательствам, занимающимся переводом иностранной технической литературы;
- инернет-порталам.

Заработная плата переводчика на конец 2011 года колеблется от 20000 руб до 45000 руб. Если взять среднюю величину и выяснить во сколько организации обходится один переводчик в год, то получится

$$U_{\text{чел},avg} = \frac{45000 + 20000}{2} \cdot 12 \cdot (1 + \omega_d) \cdot (1 + \omega_c) = 627120.00 \text{ рублей.}$$

$$U_{\text{чел},min} = 20000 \cdot 12 \cdot (1 + \omega_d) \cdot (1 + \omega_c) = 385920.00 \text{ рублей.}$$

Напомним:

- $\omega_d = 0.2$ — коэффициент, учитывающий дополнительную заработную плату (премии);
- $\omega_c = 0.34$ — коэффициент, учитывающий страховые взносы во внебюджетные фонды.

Основываясь на расчетах проведенных в предыдущем разделе, совокупная стоимость владения рассматриваемой системы в самом дорогом случае (без учета капиталовложений в оборудование) ниже, чем компании обходится самый низкооплачиваемый переводчик.

$$U_{\text{маш},8760} = 108786.67 < 385920.00 = U_{\text{чел},min}$$

Причем, если рассматривать, только рабочие дни и 40-часовую рабочую неделю, то разница становится более ощутимой.

$$U_{\text{маш},1993} = 27582.67 < 385920.00 = U_{\text{чел},min}$$

Кроме того, скорость профессионального переводчика обычно ограничена двадцатью страницами в день, в то время как, на тот же объем текста, машина потратит в худшем случае несколько минут.

Важно понимать, что статистические системы машинного перевода не могут полностью функционировать без человека. Перед тем как начать переводить, статистические СМП должны обучиться на переводах, которые сделал человек. Причем от качества текстов, на которых машина обучается, будет зависеть и качество результата ее работы. Перевод выдаваемый машиной не всегда удовлетворяет литературным стандартам языка перевода. Потому в этих случаях могут потребоваться услуги корректора. Однако, стоит заметить, что при работе переводчика-человека также бывает необходим корректор. Чаще в его роли выступает переводчик более высокой квалификации. В итоге, можно придти к следующей формуле

$$U_{\text{чел},max} + U_{\text{маш},8760} < U_{\text{чел},min} + U_{\text{чел},max}$$

где, $U_{\text{чел},max}$ — расходы на заработную плату корректора или переводчика

высокой квалификации. (В данном случае переводчиков с низкой квалификацией, можно отправить повышать свою квалификацию.)

На предприятиях, основной деятельностью, которых является перевод текстов с одного языка на другой возможно следующей применение схемы

$$\text{человек}_1 \rightarrow \text{машина} \rightarrow \text{человек}_2$$

- человек_1 — высококвалифицированный переводчик, который продолжает переводить особо важные тексты, самостоятельно, без применения СМП.
- машина — статистическая СМП, которая предварительно обучается. на переводах человека_1;
- человек_2 — высококвалифицированный переводчик, который корректирует результаты СМП.

Выгодность данной схемы требует дополнительных обоснования и практических измерений скорости работы людей и СМП и выходит за рамки данной работы.

3. ОХРАНА ТРУДА И ОКРУЖАЮЩЕЙ СРЕДЫ

3.1. ВВЕДЕНИЕ

Любая технологическая деятельность является потенциально опасной для человека и окружающей средыОбъемный расход подаваемого в помещение свежего воздуха, м^3 на одного человека в час. Разработка дипломного проекта на кафедре «Вычислительная математика и программирование» неизбежно связано с постоянной и долговременной работой с персональным вычислительным устройством и иной электронной техникой.

Распределенное программно-информационное обеспечение статистической модели перевода естественных языков созданное в рамках данной дипломной работы, — это сложный программный программный комплекс. Разработка такого рода систем не подразумевает постоянное взаимодействие с компьютерной техникой. Работа с компьютером характеризуется:

- значительным умственным напряжением;
- нервно-эмоциональной нагрузкой;
- высокой напряженностью зрительной работы;
- не естественным положением корпуса тела;
- нагрузкой на мышцы кистей рук (при работе с клавиатурой).

Большое значение имеет рациональная конструкция и расположение элементов рабочего места, что важно для поддержания оптимальной рабочей позы человека. В процессе работы с компьютером необходимо соблюдать правильный режим труда и отдыха. В противном случае у человека отмечаются значительное напряжение зрительного аппарата с появлением жалоб на неудовлетворенность работой, головные боли, раздражительность, нарушение сна, усталость и болезненные ощущения в глазах, в пояснице, в области шеи и руках. Кроме того, важно соблюдать достаточную площадь на одно рабочее место. Она должна составлять для взрослых пользователей не менее 9 м^2, а объем не менее 20 м^3.

В целом, на рабочем месте должны быть предусмотрены меры защиты от возможного воздействия опасных и вредных факторов производства. Уровни этих факторов не должны превышать предельных значений, оговоренных правовыми, техническими и санитарно-техническими нормами. Эти

нормативные документы обязывают к созданию на рабочем месте условий труда, при которых влияние опасных и вредных факторов на работающих либо устранено совсем, либо находится в допустимых пределах. Эти, а также многие другие факторы необходимо учитывать при длительной и интенсивной работе с компьютером, такой как разработка дипломного проекта. Соблюдение правил безопасности при работе с компьютером позволяет не только сохранить здоровье, но и повысить производительность, уменьшив утомление от длительного взаимодействия с техникой.

3.2. ОСНОВНАЯ ЧАСТЬ

В основной части работы будут кратко описана большая часть требований к условиям труда программиста. Аналогичные требования должны быть предъявлены к условиям труда переводчиков. В конце части более подробно описан уровень шума, возникающий от нескольких некогерентных источников.

3.2.1. МИКРОКЛИМАТ

Микроклимат — суть, состояние внутренней среды помещения, оказывающее воздействие на человека, характеризуемое показателями температуры воздуха и ограждающих конструкций, влажностью и подвижностью воздуха.

Главным процессом, который регулируется параметрами микроклимата является теплообмен человека с окружающей средой. Человеческому организму очень важно поддерживать постоянную температуру тела. Это является необходимым условием жизнедеятельности человека, осуществляемым благодаря процессу терморегуляции.

Терморегуляция — способность человека поддерживать температуру тела в определенных рамках, несмотря на температуру окружающей среды.

Отклонение нормальной для организма температуры в сторону увеличения называется гипертермией, в сторону уменьшения — гипотермией. Главное, что может предоставить комфортный для человека микроклимат — это оптимальные условия для теплообмена тела человека с окружающей средой.

При работе за компьютером в замкнутом помещении человек, как пра-

вило, окружен вычислительной техникой, основная особенность которой — большое количество выделяемого в окружающую среду тепла.

Это является следствием того, что в помещении происходит повышение температуры и снижение относительной влажности воздуха.

Именно температура и относительная влажность воздуха являются двумя основными параметрами, регулируемыми в санитарных нормах СН-245-71. Подвижностью воздуха в пощение — еще один важный параметр микроклимата. Значения этих параметров должны зависеть от времени года.

Таблица 3.14. Параметры микроклимата для помещений, где установлены компьютеры для холодного времени года.

Параметр микроклимата	Величина
Температура воздуха в помещении	22.24°C
Относительная влажность	40.60 %
Скорость движения воздуха	до 0.1м/с

Таблица 3.15. Параметры микроклимата для помещений, где установлены компьютеры для теплого времени года.

Параметр микроклимата	Величина
Температура воздуха в помещении	23.25°C
Относительная влажность	40.60 %
Скорость движения воздуха	0.1 - 0.2м/с

В помещение, где находятся компьютеры, необходимо осуществлять подачу свежего воздуха. Этот параметр зависит от объема помещения, приходящегося на одного человека, который не должен быть меньше, чем 19,5м^3 на человека.

Таблица 3.16. Расход подаваемого в помещение свежего воздуха

Объем помещения, $м^3$ на чел.	Объемный расход подаваемого в помещение свежего воздуха, $м^3$ на чел. в час
До 20	Не менее 30
От 20 до 40	Не менее 20
От 40	Естественная вентиляция

Обычно, для достижения значений параметров, приведенных в таблицах выше применяют технические средства — кондиционирование воздуха, отопительная система и организационные методы — рациональная организация проведения работ в зависимости от времени года и суток, чередование труда и отдыха.

3.2.2. РЕЖИМЫ ТРУДА И ОТДЫХА

Соблюдение правильного режима может значительно улучшить самочувствие и повысить производительность труда.

В случае несоблюдения режимов труда и отдыха у человека при долгой работе за компьютером могут наблюдаться:

- усталость;
- нарушения сна;
- болезненные ощущения (в глазах, пояснице и области шеи),

В соответствии со СанПиН 2.2.2 546-96 все виды трудовой деятельности, связанные с использованием компьютера, разде-ляются на три группы:

- группа А: работа по считыванию информации с экрана компьютера с предварительным запросом;
- группа Б: работа по вводу информации;
- группа В: творческая работа в режиме диалога с ЭВМ.

Таблица 3.17. Группа А

Уровень нагрузки за рабочую смену, количест-во знаков	Суммарное время регламентированных перерывов, минут	
	Смена 8 часов	Смена 12 часов
до 20000	30	70
до 40000	50	90
до 60000	70	120

Таблица 3.18. Группа Б

Уровень нагрузки за рабочую смену, количест-во знаков	Суммарное время регламентированных перерывов, минут	
	Смена 8 часов	Смена 12 часов
до 15000	30	70
до 30000	50	90
до 40000	70	120

Таблица 3.19. Группа В

Уровень нагрузки за рабочую смену, часов	Суммарное время регламентированных перерывов, минут	
	Смена 8 часов	Смена 12 часов
до 2	30	70
до 4	50	90
до 6	70	120

Время перерывов дано при соблюдении указанных Санитарных правил и норм. При несоответствии фактических условий труда требованиям Санитарных правил и норм время регламентированных перерывов следует увеличить на 30 %.

Эффективность перерывов повышается при сочетании с производственной гимнастикой или организации специального помещения для отдыха персонала с удобной мягкой мебелью, аквариумом, зеленой зоной и т.п.

3.2.3. ЭЛЕКТРОМАГНИТНОЕ И ИОНИЗИРУЮЩЕЕ ИЗЛУЧЕНИЯ

По-мнению ученых, излучение большинства современных мониторов не оказывает пагубного воздействия для взрослого человека.

Тем не менее, исчерпывающих данных по этому вопрос пока нет.

Максимальный уровень рентгеновского излучения от монитора составляет в среднем 10 $\frac{\text{мкБэр}}{\text{ч}}$, а интенсивность ультрафиолетового и инфракрасного излучений лежит в интервале 10-100 $\frac{\text{мВт}}{\text{м}^2}$.

Ниже описаны допустимые значения параметров неионизирующих электромагнитных излучений (в соответствии с СанПиН 2.2.2.542-96).

- Напряженность электрической составляющей электромагнитного поля на расстоянии 50 см от поверхности видеомонитора — $10\frac{\text{В}}{\text{м}}$.
- Напряженность магнитной составляющей электромагнитного поля на расстоянии 50 см от поверхности видеомонитора — $0.3\frac{\text{А}}{\text{м}}$.
- Для взрослых пользователей напряженность электростатического поля не должна превышать — $20\frac{\text{кВ}}{\text{м}}$.

Для снижения воздействия этих видов излучения рекомендуется применять мониторы с пониженным уровнем излучения (MPR-II, TCO-92, TCO-99), устанавливать защитные экраны, а также соблюдать регламентированные режимы труда и отдыха.

3.2.4. ОСВЕЩЕНИЕ

Обычно освещению как рабочего места, так и помещения уделяют мало внимания. При работе за компьютером несоблюдение правил освещения является одной из основных причин ухудшения зрения.

Недостаточность освещения:

- ослабляет внимание;
- приводит к наступлению преждевременной утомленности.

Чрезмерно яркое освещение:

- вызывает ослепление;
- раздражение и резь в глазах;
- приводит к наступлению преждевременной утомленности.

Неправильное направление света на рабочем месте может создавать резкие тени, блики. Для программиста, эти факторы не являются особенно важными, однако, благоприятными их тоже назвать нельзы.

Существует три вида освещения — естественное, искусственное и совмещенное (ес-те-ственное и искусственное вместе).

Естественное освещение — освещение помещений дневным светом, проникающим через световые проемы в наружных ограждающих конструкциях помещений. Естественное освещение характеризуется тем, что меняется в широких пределах в зависимо-сти от времени дня, времени года, характера области и ряда других факторов.

Искусственное освещение применяется при работе в темное время суток и днем, когда не удается обеспечить нормированные значения коэффициента естественного освещения (пас-мурная погода, короткий световой день).

Искусственное освещение бывает трех видов:

- рабочее;
- аварийное;
- эвакуационное.

Освещение, при котором недостаточное по нормам естественное освещение дополняется искусственным, называется совмещенным освещением.

Рабочее освещение, может быть общим или комбинированным.

Общее – освещение, при котором светильники размещаются в верхней зоне помещения равномерно или применительно к расположению оборудования.

Комбинированное – освещение, при котором к общему добавляется местное освещение.

Согласно СНиП II-4-79 в помещений вычислительных центров необходимо применить систему комбинированного освещения.

В качестве источников искусственного освещения обычно используются люминесцентные лампы типа ЛБ или ДРЛ. Они попарно объединяются в светильники, которые должны располагаться над рабочими поверхностями равномерно.

В физике освещенность определяется как отношение светового потока на единицу поверхности:

$$E = \frac{d\Phi}{dS}$$

Единицей измерения освещённости в системе СИ служит люкс (лк).

Световой поток можно рассчитать по формуле:

$$\Phi = \frac{E \cdot K \cdot S \cdot Z}{n}$$

- Φ - рассчитываемый световой поток, лм;
- E - нормированная минимальная освещенность, лк (определяется по таблице). Работу программиста, в соответствии с этой таблицей, можно отнести к разряду точных работ, следовательно, минимальная освещенность будет $= 300$ лк (значения для минимальной освещенности описаны ниже);
- S - площадь освещаемого помещения;
- Z - отношение средней освещенности к минимальной (обычно принимается равным 1.0, 1.1, 2.0);

- K - коэффициент запаса, учитывающий уменьшение светового потока лампы в результате загрязнения светильников в процессе эксплуатации (его значение зависит от типа помещения и характера проводимых в нем работ);

- n - коэффициент использования, (выражается отношением светового потока, падающего на расчетную поверхность, к суммарному потоку всех ламп и исчисляется в долях единицы; зависит от характеристик светильника, размеров помещения, окраски стен и потолка, характеризуемых коэффициентами отражения от стен и потолка).

Требования к освещенности в помещениях, где установлены компьютеры, следующие: при выполнении зрительных работ высокой точности общая освещенность должна составлять 300 лк, а комбинированная — 750 лк; аналогичные требования при выполнении работ средней точности — 200 и 300 лк соответственно. Кроме того все поле зрения должно быть освещено достаточно равномерно — это основное гигиеническое требование.

3.2.5. ШУМ И ВИБРАЦИЯ

При работе с любой техникой необходимо соблюдать спокойствее, не давать волю эмоциям. В противном случае, как для человека, так и для технического средства могут наступить необратимые последствия. Одним из наиболее сильных раздражающих факторов является шум

Под воздействием шума ухудшается внимание, повышается раздражительность. Последнее является весьма опасным последствием работы за компьютером. Человек испытывает головные боли, головокружение.

Шум замедляет реакцию человека на поступающие от технических устройств сигналы. Шум угнетает центральную нервную систему, вызывает изменения скорости дыхания и пульса, способствует нарушению обмена веществ,
возникновению сердечнососудистых заболеваний.

В случае, когда шум воздействует на слуховые органы человека постоянно и его эффективность велика, то это может привести
к частичной или полной потере слуха человека.

Одна из измеримых характеристик звука — это количество заключенной в нем энергии; интенсивность звука в любой точке можно измерить как поток энергии, приходящейся на единичную площадку, и выразить в ваттах на квадратный метр $\left(\frac{\text{Вт}}{\text{м}^2}\right)$.

Такая характеристика не удобна. Возможен очень широком диапазон значений.

При попытке записать в этих единицах интенсивность обычных шумов сразу же возникают трудности, так как интенсивность наиболее тихого звука, доступного восприятию человека с самым острым слухом,
равна приблизительно $0.1 \cdot 10^{-11} \quad \frac{\text{Вт}}{\text{м}^2}$.

Легко видеть, что оперировать числами, выражающими интенсивности звука, лежащие в столь широком диапазоне, очень трудно. Выходом из сложившейся ситуации является использование некоторой *относительной величины*.

Такой величиной является децибел [дБ]. Уровень звука, выраженный в децибелах, численно равен десятичному логарифму безразмерного отношения измеряемой интенсивности звука к эталонной интенсивности звука, умноженному на десять.

$$A_{\text{дБ}} = \log_{10}\left(\frac{A_1}{A_0}\right)$$

- A_1 – измеряемая интенсивность;
- A_0 – эталонная интенсивность, принимаемую за $10 - 12 \quad \frac{\text{Вт}}{\text{м}^2}$.

Напомним, что децибел — это относительная величина. Операции над ней отличаются от операций на абсолютными величинами.

Общий уровень шума от двух источников будет представлять, выраженный в децибелах, не будет равен сумме каждого из них.

Суммировать необходимо интенсивность двух источников, а после этого перейти к децибелам путем увеличения логарифма.

Уровень шума, возникающий от нескольких некогерентных источников, работающих одновременно, подсчитывается на основании принципа энергетического суммирования излучений отдельных источников

$$L_{\Sigma} = 10 \cdot log_{10} \left(\sum_{i=1}^{i=n} \left(10^{0.1 \cdot L_i} \right) \right)$$

- L_i – уровень звукового давления i-го источника шума;
- n – количество источников шума.

Полученные результаты расчета сравнивается с допустимым значением уровня шу-ма для данного рабочего места. Если результаты расчета выше допустимого значения уровня шума, то необходимы специальные меры по снижению шума. Для того, чтобы оценить уровень шума в помещении, обычно используются специализированные устройства — шумомеры. В простейшем случае шумомер состоит из усилителя, к входу которого подключается измерительный микрофон, а к выходу – вольтметр, проградуированный в децибелах. Однако, существуют иные способы измерения. Для этого можно воспользоваться обычным микрофоном и любым профессиональным звуковым редактором *(Sound Forge, Nero Wave Editor)*. Микрофон и программное обеспечение предварительно следует откалибровать. Для оценки шума такого сложного устройства как компьютер может потребоваться, провести эксперименты по измерению уровня шума для каждой составляющей в отдельности. В противном случае придется поверить шумовым характеристикам, которые указанны производителями комплектующих.

Таблица 3.20. Характеристики шума типичного настольного компьютера.

Источник шума	Уровень шума, дБ
Жесткий диск	35
Система охлаждения	45
Монитор	17
Клавиатура	5
Принтер	40

Таким образом можно оценить шум типичного настольного компьютера.

$$L_\Sigma = 10 \cdot log_{10}\left(10^{3.5} + 10^{4.5} + 10^{1.7} + 10^{0.5} + 10^{4.0}\right) =$$

$$= 10 \cdot log_{10}(44838.3) = 46.5165 \text{ дБ}$$

Таблица 3.21. Уровни звука в децибелах на различных рабочих местах.

Категория напряженности труда	Категория тяжести труда			
	Легкая	Средняя	Тяжелая	Очень тяжелая
Мало напряженный	80	80	75	75
Напряженный	70	70	65	65
Очень напряженный	60	60	—	—
Опасно напряженный	50	50	—	—

Уровень шума на рабочем месте программистов не должен превышать 50 дБ, а в залах обработки информации на вычислительных машинах — 65 дБ. Вычисленное значение не превышает допустимый уровень шума рабочего места. Однако, важно понимать, что приведенные расчеты не претендуют на высокую точность. Реальный уровень может быть несколько больше, так как вычислительная машина является далеко не единственным источником шума. Для обеспечения показателей установленных нормой, необходимо использовать звукопоглощающие материалы и виброизоляторы, в которые помещается оборудование. Во многих случаях рекомендуется размещать рабочее место программиста и само устройство удаленно друг от друга.

ЗАКЛЮЧЕНИЕ

В ходе выполнения дипломной работы были получены следующие результаты:

а) реализован основной функционал распределенной системы машинного перевода;

б) разработаны методы создания модели машинного перевода текстов научно-технической литературы;

в) реализован эффективный метод декодирования (перевода) с одного языка на другой;

г) была проверена модель параллельных вычислений для задачи машинного перевода и сделаны выводы о приросте производительности на ряде операций.

д) выяснилось, что используемая модель вычислений не экономична по отношению к базе данных;

е) не получилось достичь высокого качества перевода для текстов общей тематики, используемые алгоритмы могут потребовать корректировки.

В качестве дальнейших приоритетных направлений исследований и развития созданной системы можно выделить следующие:

а) реализация полноценного фразового перевода при использовании классических подходов к статическому машинному переводу совместно с подходом рассмотренном в работе;

б) реализация синтаксического перевода;

в) апробация смешанной трансферно-статистической системы машинного перевода при использовании морфологического анализа для конкретной пары языков (русский, английский);

г) апробация более точные, но менее эффективные методы поиска.

Кроме того, в плане улучшения реализации системы может потребоваться

а) использовать пословное сжатие при хранении в БД;

б) переписать обработчика на C с библиотекой libevent;

в) использовать libevent для rest-интерфейса декодера, чтобы иметь возможность поддерживать 1 млн. одновременных соединений;

г) попробовать перейти на более эффективную реализацию базы данных, например с redis на leveldb.

В экономической части работы проведена оценка стоимости разрабатываемой системы. Суммарные расходы на разработку могут составлять. 916669.08 рублей. Однако, чтобы система окупилась достаточно продавать ее по цене 1833 рублей. Это связано, с большим числом предприятий нуждающихся в оперативном машинном переводе. Совокупная стоймость владения системой при работе в режиме 24 на 7 на 365 составляет 108786 рублей в год. Однако, это все равно ниже чем расходы на заработную плату низкоквалифицированного переводчика за тот же самый период. Даже при наличие корректора, система оказывается очень выгодной.

В разделе посвященном охране труда и окружающей среды кратко описаны основные требования к рабочему месту программиста. Эти же самые рекомендации можно отнести и к рабочему месту профессионального переводчика

Наиболее подробно в работе рассмотрен фактор зашумленности рабочего места. В современном мире переводчик чаще всего использует в своей работе вычислительную технику. Как составление программ, так и перевод текстов — сложная высокоинтеллектуальная деятельность, часто сопряженная с высокими нервными нагрузками. При работе с любой техникой необходимо соблюдать спокойствие, взвешенно и обдумано принимать решения, и не поддаваться внешним раздражителям. Одним из таких раздражителей является шум. Существуют различные методы защиты от шума. Чаще всего это звукоизоляция помещений, специальные установки для устройств, поглощающие вибрацию. В некоторых случаях применяют малошумящие устройства.

Соблюдение рекомендаций, предложенных выше, позволит свести к минимуму вредные воздействия на здоровье человека при длительной работе и сохранить его работоспособность, а с помощью последнего повысить качество результатов его труда.

Кроме того, системы машинного перевода призваны ограничить взаимодействие переводчиков с вычислительной техникой, и тем самым так же свести к минимуму вредные воздействия на их здоровье. Это позволит улучшить качество переводов, на которых будет обучаться система в будущем.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. *Charniak, E.* Syntax-based Language Models for Machine Translation / E. Charniak, K. Knight, Yamada K. // MT Summit IX. — 2003.
2. *Chen, S.F.* An empirical study of smoothing techniques for language modeling / S.F Chen // *Computer Speech and Language*. — 1999. — Vol. 13, no. 4.
3. *DeNero, J.* Fast Consensus Decoding over Translation Forests / J. DeNero, D. Chiang, K. Knight // Association for Computational Linguistics 2009. — Association for Computational Linguistics, 2009.
4. Fast Decoding and Optimal Decoding for Machine Translation / U. Germann, M. Jahr, K. Knight et al. // Artificial Intelligence. — 2003.
5. *Germann, U.* Greedy decoding for statistical machine translation in almost linear time / U. Germann // Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology / USC Information Sciences Institute, Marina del Rey. — Vol. Volume 1. — CA: Association for Computational Linguistics, 2003.
6. *Hoang, H.* Design of the Moses Decoder for Statistical Machine Translation / H. Hoang, P. Koehn // ACL Workshop on Software engineering, testing, and quality assurance for NLP 2008. — Association for Computational Linguistics, 2008.
7. *Knight, K.* Decoding Complexity in Word-Replacement Translation Models / K. Knight // Computational Linguistics, Squibs & Discussion. — Vol. 25. — 1999.
8. *Knight, K.* Teaching Statistical Machine Translation / K. Knight // MT Summit IX Workshop on Teaching Machine Translation. — 2003.
9. *Knight, K.* Statistical Machine Translation / K. Knight, P. Koehn // Tutorial at MT Summit 2003. — 2003.
10. *Knight, K.* What's New in Statistical Machine Translation / K. Knight, P. Koehn // Tutorial at HLT/NAACL 2003. — 2003.

11. *Knight, K.* Introduction to Statistical Machine Translation / K. Knight, P. Koehn // Tutorial at AMTA 2004. — Association for Computational Linguistics, 2004.

12. *Koehn, P.* Challenges in Statistical Machine Translation / P. Koehn // Talk given at PARC, Google, ISI, MITRE, BBN, / Univ. of Montreal. — 2004.

13. *Koehn, P.* Statistical Machine Translation / P. Koehn. — Cambridge: Cambridge University Press, 2010.

14. *Koehn, P.* Factored Translation Models / P. Koehn, Hoang H. // Handbook of Natural Language Processing and Machine Translation. — Springer, 2011.

15. *Koehn, P.* Edinburgh University System Description for the 2008 NIST Machine Translation Evaluation / P. Koehn, J. Schroeder, M. Osborne // NIST MT Evaluation Meeting. — Association for Computational Linguistics, 2008.

16. *Koehn, P.* Moses: Open Source Toolkit for Statistical Machine Translation / P. Koehn et al. — Association for Computational Linguistics, 2007.

17. *Liu, Yang*. Log-linear Models for Word Alignment / Yang Liu, Qun Liu, Shouxun Lin // 43rd Annual Meeting of the Association for Computational Linguistics / Institute of Computing Technology Chinese Academy of Sciences. — Beijing: Association for Computational Linguistics, 2005.

18. *Manning, C. D*. Foundations of Statistical Natural Language Processing / C. D Manning, H. Schuetze. — Second printing with corrections edition. — Cambridge: The MIT Press, 2000.

19. *Noon, C.* An efficient transformation of the generalized traveling salesman problem. — 1993.

20. *Shannon, C.E.* A mathematical theory of communication / C.E. Shannon // *Bell System Technical Journal*. — 1948. — Vol. 27, no. 4. — P. 379–423 and 623–656.

21. *Smith, P. D.* An Introduction to Text Processing / P. D. Smith. — Cambridge, MA:: The MIT Press, 1990.

22. *Sparck, J. K.* Evaluating Natural Language Processing Systems / J. K. Sparck, J. R. Galliers. — Berlin: Springer, 1995.

23. *Stone, M. L.* Web embraces language translation / M. L. Stone. — ZDNN, 1998.

24. *Sumita, Y.* Experiments and prospects of example-based machine translation / Y. Sumita, H. Iida // In Proceedings of the 29th Annual Conference of the ACL. — Berkley: CA, 1991.

25. *Wilks, Y.* Machine Translation, Its Scope and Limits / Y. Wilks. — New York: Springer Science+Business Media LLC, 2009.

26. *Ахманова, Г. И.* Теория и практика английской научной речи / Г. И. Ахманова, О. И. Богомолова; Под ред. М. М. Глушко. — М.: Изд. МГУ, 1987.

27. *Белоногов, Г. Г.* Компьютерная лингвистика и перспективные информационные технологии / Г. Г. Белоногов. — М.: Русский мир, 2004.

28. *Березин, В. М.* Защита от вредных производственных факторов при работе с ПЭВМ / В. М. Березин, М. И. Дайнов. — М.: Изд-во МАИ, 2003.

29. *Бобков, Н. И.* Охрана труда на ВЦ / Н. И. Бобков, Т. В. Голованова. — М.: Изд-во МАИ, 1995.

30. *Вдовин, В. А.* Экономическая эффективность разработки информационных систем и технологий / В. А. Вдовин, А. В. Дегтярев, В. А. Оганов; Под ред. А.В. Дегтярева; МАИ. — М.: Доброе Слово, 2006.

31. *Дайнов, М.И.* Методические указания к дипломному проектированию «Экологические платежи за загрязнение окружающей природной среды» / М.И. Дайнов, Л.Б. Метечко, В.В. Толоконникова. — М.: Изд-во МАИ, 2001.

32. *Дайнов, М. И.* Борьба с шумами и вибрацией в авиационной промышленности / М. И. Дайнов, Л. И. Малько, В. М. Яров. — М.: Изд-во МАИ, 1998.

33. *Ершов, А. П.* Машинный фонд русского языка: внешняя постановка / А. П. Ершов. — М.: Наука, 1986.

34. *Кан, Д. А.* Применение теории компьютерной семантики русского языка и статистических методов к построению системы машинного перевода: Диссертация канд. ф.м. наук: 05.13.11 / Санкт-Петербургский Государственный Университет. — Санкт-Петербург, 2011.

35. *Караулов, Ю. Н.* Анализ метаязыка словаря с помощью ЭВМ / Ю. Н. Караулов. — М.: Наука, 1982.

36. *Караулов, Ю. Н.* Методология лингвистического исследования и машинный фонд русского языка / Ю. Н. Караулов. — М.: Наука, 1986.

37. *Кнут, Д.* Искусство программирования / Д. Кнут. — 3-е издание, исправленное и дополненное изд. — М.: Вильямс, 2002. — Т. Том 1. Основные алгоритмы.

38. *Ковалев, А. М.* Основы управления проектами в области информационных технологий / А. М. Ковалев, В. А. Ковалев; Под ред. А.В. Дегтярева; МАИ. — М.: Доброе Слово, 2007.

39. *Комиссаров, В. Н.* Практикум по переводу с английского языка на русский / В. Н. Комиссаров, А. Л. Коралова. — М.: Высшая школа, 1990.

40. *Кормалев, Д.А*. Основы теории автоматической обработки текста / Д.А Кормалев, Е.А. Сулейманова; Университет города Переславль-Залесский. — Переславль-Залесский, 2005.

41. *Липатов, А.В.* Автоматизация процесса построения и пополнения двуязычных специализированных словарей / А.В. Липатов, А. А. Мальцев, В. В. Шило // Труды конференции «Диалог». — М.: 2005.

42. *Максименко, О. И.* Формальные методы оценки эффективности систем автоматической обработки текста: Диссертация доктора филологических наук: 10.02.21 / Москва. — М., 2003.

43. *Максименко, О. И.* Машинный семантический анализ русского языка и его применения: Диссертация доктора физико-математических наук:

05.13.11 / Санкт-Петербургский государственный университет. — СПб., 2006.

44. *Марчук, Ю. Н.* Основы компьютерной лингвистики / Ю. Н. Марчук; МПУ. — Издание 2-е дополненное изд. — М.: Изд-во «Народный учитель», 2000.

45. *Марчук, Ю. Н.* Модели перевода / Ю. Н. Марчук. — М.: Изадтельский центр «Академия», 2011.

46. *Мельчук, И. А.* Опыт теории лингвистических моделей «смысл-текст» / И. А. Мельчук. — М.: Наука, 1974.

47. *Мельчук, И. А.* Русский язык в модели «смысл-текст» / И. А. Мельчук; Школа: «Языки русской культуры». — Москва-Вена, 1995.

48. *Моисеева, Н. К.* Управление маркетингом: теория, практика, информационные технологии / Н. К. Моисеева, М. В. Конышева. — М.: Финансы и статистика, 2002.

49. *Ньюэл, М. В.* Управление проектами для профессионалов. Руководство к сдаче сертификационных экзаменов / М. В. Ньюэл. — М.: Кудиц-Пресс, 2008.

50. *Пиотровский, Р. Г.* Текст, машина, человек / Р. Г. Пиотровский. — Л.: Наука, 1975.

51. *Потемкин, С. Б.* Автоматическая оценка качества машинного перевода на основе семантической метрики / С. Б. Потемкин, Г. Е. Кедрова // Труды II Международной научно-практической конференции, посвященной Европейскому Дню языков. — Луганск: 2005.

52. *Потемкин, С. Б.* Использование корпуса параллельных текстов для пополнения специализированного двуязычного словаря / С. Б. Потемкин, Г. Е. Кедрова // III Международный конгресс исследователей русского языка «Русский язык: исторические судьбы и современность». — М.: 2007.

53. *Пумпянский, А. Л.* Информационная роль порядка слов в научной и технической литературе / А. Л. Пумпянский. — Мн.: ТетраСистемс, 2001.

54. *Рассел, С.* Искусственный интеллект: современный подход / С. Рассел, П. Норвиг. — 2-е изд. изд. — М.: Издательский дом «Вильямс», 2006.

55. *Рахимбердиев, Б. Н.* Эволюция семантики экономической терминологии русского языка в XX веке: Диссертация канд. филологических наук: 10.02.21 / Москва. — М., 2003.

56. *Реформатский, А. А.* Введение в языковедение / А. А. Реформатский; Под ред. В. А. Виноградов. — М.: Аспект Пресс, 1996.

57. *Рецкер, Я. И.* О закономерных соответствиях при переводе на родной язык / Я. И. Рецкер. — М.: Наука, 1950.

58. *Романов, А.С.* Подходы к идентификации авторства текста на основе n-грамм и нейронных сетей / А.С. Романов // Молодежь и современные информационные технологии: Сб. тр. VI Всерос. науч.-практ. конф. студентов, аспирантов и молодых ученых. — Томск: Изд-во ТПУ, 2008.

59. *Романов, А.С.* Структура программного комплекса для исследования подходов к идентификации авторства текстов / А.С. Романов // Докл. Том. гос. ун-та систем управления и радиоэлектроники. — 2(18). — Томск: Изд-во ТПУ, 2008.

60. *Хорошилов, А. А.* Теоретические основы и методы построения систем фразеологического машинного перевода: Диссертация доктора технических наук: 05.13.17 / Москва. — М., 2006.

61. *Хроменков, П. Н.* Анализ и оценка эффективности современных систем машинного перевода: Диссертация канд. филологических наук :10.02.21 / Москва. — М., 2000.

62. *Швейцер, А. Д.* Теория перевода / А. Д. Швейцер. — М.: Наука, 1988.

63. *Шевелев, О.Г.* Методы автоматической классификации текстов на естественном языке / О.Г. Шевелев. — Томск: ТМЛ-Пресс, 2007.

ПРИЛОЖЕНИЕ 1. ПРОСТЕЙШАЯ СМП ОСНОВАННАЯ НА ПРИМЕРАХ

```
-module(simple_ebmt_decoder).
-export([decode/1]).

%% Простой фразовый декодировщик для системы машинного перевода основанной на примерах
decode(Input_string) ->
  Word_list   = words:list(Input_string),           %% Разбиваем входную строку на слова.
  Decoded_word_list = decode_word_list(Word_list, 6), %% Переводим список слов.
  make_sentence(Decoded_word_list).                  %% Формируем из него предложение.

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%% Декодирование
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% Переводит список слов Word_list с учетом размера фразы Phrase_Size.
decode_word_list(Word_list, Phrase_Size) ->
  %%  decode_word_list(Word_list, Size, MaxSize)
  decode_word_list(Word_list, Phrase_Size, Phrase_Size).

decode_word_list([], _, _) -> [];
  %% Если входной список слов пуст, значит переводить больше нечего.

decode_word_list([ Unknown_word | Rest_word_list], 0, MaxSize) ->
  %% Если текущий размер рассматриваемой фразы, значитмы, не можем перевести эту фразу с начала.
  %% Попробуем начать со второго слова. А первое слово текущей фразы признаем неизвестным.
  [[Unknown_word] | decode_word_list(Rest_word_list, MaxSize, MaxSize)];

decode_word_list(Word_list, Size, MaxSize)->
  %% Разбиваем список слов на 2 части.
  %% Первая - фраза, которую хотим перевести. Вторая - остаток предляжения.
  case Size < erlang:length(Word_list) of
    true ->
      {First_Ngram, Rest_word_list } = lists:split(Size, Word_list);
    false ->
      First_Ngram = Word_list,
      Rest_word_list = []
  end,
  %% Пытаемяс перевести фразу.
  case try_to_translate(First_Ngram) of
    {no} -> %% Если не удалось, возьмем фразу поменьше
      decode_word_list(Word_list, Size-1, MaxSize);
    Val ->  %% Если удалось, переводим дальше.
      [ Val | decode_word_list(Rest_word_list, Size, MaxSize)]
  end.
```

```
44
45 try_to_translate(Ngram) ->
46   case Ngram of  % Таблица соотвествий слов.
47         ["i", "have", "a", "big", "fat", "cat"]   ->
48           ["u", "menja", "est'", "bolshoj", "zhirnij", "kot"];
49         ["i", "have", "a", "big", "fat", "rat"]   ->
50           ["u", "menja", "est'", "bolshoj", "zhirnij", "krys"];
51         ["i", "have"]   -> ["ja", "imeju"];
52         ["have", "a"]   -> ["imet'"];
53         ["a", "big"]    -> ["bolshoj"];
54         ["big", "fat"]  -> ["ochen'", "zhirnij"];
55         ["fat", "cat"]  -> ["zhirnij", "kot"];
56         ["i"]    -> ["ja"];
57         ["have"]  -> ["imet'"];
58         ["big"]   -> ["bolshoj"];
59         ["fat"]   -> ["zhirnij"];
60         ["cat"]   -> ["kot"];
61         ["rat"]   -> ["krysa"];
62       % ----------------------------------
63           Val -> {no}
64   end.
65
66 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
67 %%% Формирование предложения
68 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
69
70 make_sentence(List) ->
71   string:join(join_phrases(List), [32]).
72
73 join_phrases([]) -> [];
74 join_phrases([Phrase|Tail] = List) ->
75   [join_phrase(Phrase) | join_phrases(Tail)].
76
77 join_phrase(Phrase) ->
78   string:join(Phrase, [32]).
```

ПРИЛОЖЕНИЕ 2. EM АЛГОРИТМ

EM-алгоритм (expectation-maximization) - алгоритм, используемый в математической статистике для нахождения оценок максимального правдоподобия параметров вероятностных моделей, в случае, когда модель зависит от некоторых скрытых переменных. Каждая итерация алгоритма состоит из двух шагов. На E-шаге (expectation) вычисляется ожидаемое значение функции правдоподобия, при этом скрытые переменные рассматриваются как наблюдаемые. На M-шаге (maximization) вычисляется оценка максимального правдоподобия, таким образом увеличивается ожидаемое правдоподобие, вычисляемое на E-шаге. Затем это значение используется для E-шага на следующей итерации. Алгоритм выполняется до сходимости.

EXPECTATION

$$P(a|\Pi_e, \Pi_r) = \frac{P(\Pi_e, a|\Pi_r)}{P(\Pi_e, \Pi_r)}$$

Числитель:

$$P(\Pi_e, a|\Pi_r) = \frac{\varepsilon}{(l_r+1)^{l_e}} \prod_{j=1}^{l_e} t(\omega_{ej}|\omega_{ra(j)})$$

Знаменатель:

$$P(\Pi_e, \Pi_r) = \sum_a P(\Pi_e, a|\Pi_r) =$$

$$= \sum_{a(1)=0}^{l_r} \cdots \sum_{a(l_e)=0}^{l_r} \frac{\varepsilon}{(l_r+1)^{l_e}} \prod_{j=1}^{l_e} t(\omega_{ej}|\omega_{ra(j)}) =$$

$$= \frac{\varepsilon}{(l_r+1)^{l_e}} \sum_{a(1)=0}^{l_r} \cdots \sum_{a(l_e)=0}^{l_r} \prod_{j=1}^{l_e} t(\omega_{ej}|\omega_{ra(j)}) =$$

$$= \frac{\varepsilon}{(l_r+1)^{l_e}} \prod_{j=1}^{l_e} \sum_{i=0}^{l_r} t(\omega_{ej}|\omega_{ri});$$

Таким образом:

$$P(A|\Pi_e,\Pi_r) = \frac{P(\Pi_e,A|\Pi_r)}{P(\Pi_e,\Pi_r)} = \frac{\frac{\varepsilon}{(l_r+1)^{l_e}} \prod_{j=1}^{l_e} t(\omega_{ej}|\omega_{ra(j)})}{\frac{\varepsilon}{(l_r+1)^{l_e}} \prod_{j=1}^{l_e} \sum_{i=0}^{l_r} t(\omega_{ej}|\omega_{ri})};$$

$$P(A|\Pi_e,\Pi_r) = \frac{P(\Pi_e,A|\Pi_r)}{P(\Pi_e,\Pi_r)} = \prod_{j=1}^{l_e} \frac{t(\omega_{ej}|\omega_{ra(j)})}{\sum_{i=0}^{l_r} t(\omega_{ej}|\omega_{ri})};$$

MAXIMIZATION

$$counts(\omega_e|\omega_r;\Pi_e,\Pi_r) = \sum_a P(a|\Pi_e,\Pi_r) \cdot \sum_{j=1}^{l_e} \delta(\omega_e,\omega_{ej}) \cdot \delta(\omega_r,\omega_{ra(j)});$$

$$counts(\omega_e|\omega_r;\Pi_e,\Pi_r) = \frac{t(\omega_e|\omega_r)}{\sum_{j=1}^{l_e} t(\omega_e|\omega_{ra(j)})} \cdot \sum_{j=1}^{l_e} \delta(\omega_e,\omega_{ej}) \cdot \sum_{i=0}^{l_e} \delta(\omega_r,\omega_{ri});$$

$$t(\omega_e|\omega_r;\Pi_e,\Pi_r) = \frac{\sum_{\Pi_e,\Pi_r} counts(\omega_e|\omega_r;\Pi_e,\Pi_r)}{\sum_{\omega_r}\sum_{\Pi_e,\Pi_r} counts(\omega_e|\omega_r;\Pi_e,\Pi_r)};$$

ПРИЛОЖЕНИЕ 3. МОДЕЛЬ IBM 1

Обучить-Модель-IBM-1 ($t(\omega_e|\omega_r)$, Θ_e, Θ_r)

$\forall\, \omega_e \in \Pi_e \in \Theta_e$ **:**
  $\forall \omega_r \in \Pi_r \in \Theta_r$ **:**
    $t(\omega_e|\omega_r) \leftarrow u, u \in \mathbb{R}$;
▷ *Инициализируем таблицу* $t(\omega_e|\omega_r)$ *одинаковыми значениями.*
пока не сойдется **:**
  $\forall\, \omega_e \in \Pi_e \in \Theta_e$ **:** ▷ *Инициализируем остальные таблицы.*
    $\forall \omega_r \in \Pi_r \in \Theta_r$ **:**
      $counts(\omega_e|\omega_r) \leftarrow 0; \quad total(\omega_r) \leftarrow 0;$
  $\forall\, \Pi_e, \Pi_r \in \Theta_e, \Theta_r$ **:** ▷ *Вычисляем нормализациию.*
    $\forall\, \omega_e \in \Pi_e$ **:**
      $stotal(\omega_e) \leftarrow 0;$
      $\forall\, \omega_r \in \Pi_r$ **:**
        $stotal(\omega_e) \leftarrow stotal(\omega_e) + t(\omega_e|\omega_r);$
    $\forall\, \omega_e \in \Pi_e$ **:** ▷ *Собираем подсчеты.*
      $\forall\, \omega_r \in \Pi_r$ **:**
        $counts(\omega_e|\omega_r) \leftarrow counts(\omega_e|\omega_r) + \dfrac{t(\omega_e|\omega_r)}{stotal(\omega_e)};$
        $total(\omega_r) \leftarrow total(\omega_r) + \dfrac{t(\omega_e|\omega_r)}{stotal(\omega_e)};$
  $\forall\, \omega_e \in \Pi_e \in \Theta_e$ **:** ▷ *Оцениваем вероятность.*
    $\forall \omega_r \in \Pi_r \in \Theta_r$ **:**
      $t(\omega_e|\omega_r) \leftarrow \dfrac{counts(\omega_e|\omega_r)}{total(\omega_r)};$

ПРИЛОЖЕНИЕ 4. МОДЕЛЬ IBM 2

Обучить-Модель-IBM-2 ($t(\omega_e|\omega_r)$, Θ_e, Θ_r)

$\forall\, \omega_e \in \Pi_e \in \Theta_e$ **:**
  $\forall \omega_r \in \Pi_r \in \Theta_r$ **:**
    $t(\omega_e|\omega_r) \leftarrow u_1, u_1 \in \mathbb{R}$;
    $\alpha(\pi_{\omega_e}|\pi_{\omega_r}, l_r, l_e) = u_2, u_2 \in \mathbb{R}$;
пока не сойдется **:**
  $\forall\, \omega_e \in \Pi_e \in \Theta_e$ **:** ▷ *Инициализируем остальные таблицы.*
    $\forall \omega_r \in \Pi_r \in \Theta_r$ **:**
      $counts(\omega_e|\omega_r) \leftarrow 0$; $\quad total(\omega_r) \leftarrow 0$;
      $counts_d(\pi_{\omega_e}|\pi_{\omega_r}, l_e, l_r) \leftarrow 0$; $\quad total_d(\pi_{\omega_r}, l_e, l_r) \leftarrow 0$;
  $\forall\, \Pi_e, \Pi_r \in \Theta_e, \Theta_r$ **:** ▷ *Вычисляем нормализациию.*
    $\forall\, \omega_e \in \Pi_e$ **:**
      $stotal(\omega_e) \leftarrow 0$;
      $\forall\, \omega_r \in \Pi_r$ **:**
        $stotal(\omega_e) \leftarrow stotal(\omega_e) + t(\omega_e|\omega_r) \cdot \alpha(\pi_{\omega_e}|\pi_{\omega_r}, l_r, l_e)$;
    $\forall\, \omega_e \in \Pi_e$ **:** ▷ *Собираем подсчеты.*
      $\forall\, \omega_r \in \Pi_r$ **:**
        $c \leftarrow \dfrac{t(\omega_e|\omega_r) \cdot \alpha(\pi_{\omega_e}|\pi_{\omega_r}, l_r, l_e)}{stotal(\omega_e)}$
        $counts(\omega_e|\omega_r) \leftarrow counts(\omega_e|\omega_r) + c$;
        $total(\omega_r) \leftarrow total(\omega_r) + c$;
        $counts_d(\pi_{\omega_e}|\pi_{\omega_r}, l_e, l_r) \leftarrow counts_d(\pi_{\omega_e}|\pi_{\omega_r}, l_e, l_r) + c$;
        $total_d(\pi_{\omega_r}, l_e, l_r) \leftarrow total_d(\omega_r, l_e, l_r) + c$;
  сгладить-искажения ($counts_d$, $total_d$);
  $\forall\, \omega_e \in \Pi_e \in \Theta_e$ **:** ▷ *Оцениваем вероятность.*
    $\forall \omega_r \in \Pi_r \in \Theta_r$ **:**
      $t(\omega_e|\omega_r) \leftarrow \dfrac{counts(\omega_e|\omega_r)}{total(\omega_r)}$;
  $\forall\, (\pi_{\omega_e}, \pi_{\omega_r}, l_e, l_r) \in counts_d$ **:**
    $\alpha(\pi_{\omega_e}|\pi_{\omega_r}, l_r, l_e) \leftarrow \dfrac{counts_d(\pi_{\omega_e}|\pi_{\omega_r}, l_e, l_r)}{total_d(\pi_{\omega_r}, l_e, l_r)}$;

сгладить-искажения ($counts_d$, $total_d$)

$\lambda \leftarrow 1.0$

$\forall\ (\pi_{\omega_e}, \pi_{\omega_r}, l_e, l_r) \in counts_d$ **:**

  $v \leftarrow counts_d(\pi_{\omega_e} | \pi_{\omega_r}, l_e, l_r)$;

  если ($0 < v < \lambda$) **:**

    $\lambda \leftarrow v$;

  $\lambda \leftarrow \frac{\lambda}{2}$;

$\forall\ (\pi_{\omega_e}, \pi_{\omega_r}, l_e, l_r) \in counts_d$ **:**

  $counts_d(\pi_{\omega_e} | \pi_{\omega_r}, l_e, l_r) \leftarrow counts_d(\pi_{\omega_e} | \pi_{\omega_r}, l_e, l_r) + \lambda$;

$\forall\ (\pi_{\omega_r}, l_e, l_r) \in total_d$ **:**

  $total_d(\pi_{\omega_r}, l_e, l_r) \leftarrow total_d(\omega_r, l_e, l_r) \cdot l_r$;