ՀՀ ԳԻՏՈՒԹՅՈՒՆՆԵՐԻ ԱԶԳԱՅԻՆ ԱԿԱԴԵՄԻԱՅԻ
ԻՆՖՈՐՄԱՏԻԿԱՅԻ և ԱՎՏՈՄԱՏԱՑՄԱՆ ՊՐՈԲԼԵՄՆԵՐԻ ԻՆՍՏԻՏՈՒՏ

Արամ Աշոտի Ավետիսյան

ԲՆԱԿԱՆ ԼԵԶՎԻ ԵՎ UNL ԼԵԶՎԻ ԿԱՌՈՒՑՎԱԾՔՆԵՐԻ ՓՈԽԱԴԱՐՁ ԱՎՏՈՄԱՏԱՑՎԱԾ ՆԵՐԿԱՅԱՑՄԱՆ ՀԱՄԱՐ ԵՐԿԽՈՍԱԿԱՆ և ՈՒՍՈՒՑՎՈՂ ՀԱՄԱԿԱՐԳԻ ՄՇԱԿՈՒՄԸ

Ե.13.04 – «Հաշվողական մեքենաների, համալիրների, համակարգերի և ցանցերի մաթեմատիկական և ծրագրային ապահովում»
տեխնիկական գիտությունների թեկնածուի
գիտական աստիճանի հայցնամ ատենախոսության

ՍԵՂՄԱԳԻՐ

ԵՐԵՎԱՆ 2011

ИНСТИТУТ ПРОБЛЕМ ИНФОРМАТИКИ И АВТОМАТИЗАЦИИ НАЦИОНАЛЬНОЙ АКАДЕМИИ НАУК РА

Аветисян Арам Ашотович

РАЗРАБОТКА ДИАЛОГОВОЙ И ОБУЧАЕМОЙ СИСТЕМЫ ДЛЯ АВТОМАТИЗИРОВАННОГО ВЗАИМНОГО ПРЕДСТАВЛЕНИЯ СТРУКТУР ЕСТЕСТВЕННОГО ЯЗЫКА И ЯЗЫКА UNL

АВТОРЕФЕРАТ

диссертации на соискание ученой степени кандидата
технических наук по специальности
05.13.04 – «Математическое и программное обеспечение вычислительных машин, комплексов, систем и сетей»

ЕРЕВАН 2011

Ատենախոսության թեման հաստատվել է ՀՀ ԳԱԱ ինֆորմատիկայի և ավտոմատացման պրոբլեմների ինստիտուտում:

Գիտական ղեկավար՝ ԳԱԱ թղթ. անդ., ֆ.մ.գ.դ. Ի.Դ. Զասլավսկի

Պաշտոնական ընդդիմախոսներ՝ Ֆիզ. մաթ. գիտ. դոկտոր Է.Մ. Պողոսյան
տեխ. գիտ. թեկնածու Ա.Խ. Մանուկյան

Առաջատար կազմակերպություն՝ Երևանի Պետական Համալսարան

Պաշտպանությունը կկայանա 2011 թ. հունիսի 10-ին, ժամը 16^{00} -ին, ՀՀ ԳԱԱ ԻԱՊԻ-ում գործող ԲՈՀ-ի 037 «Ինֆորմատիկայի և հաշվողական համակարգերի» մասնագիտական խորհրդի նիստում, հետևյալ հասցեով՝ 0014 , Երևան, Պ. Սևակի 1:

Ատենախոսությանը կարելի է ծանոթանալ ՀՀ ԳԱԱ ԻԱՊԻ-ի գրադարանում:
Սեղմագիրն առաքված է 2011թ. մայիսի 10-ին:

Մասնագիտական խորհրդի գիտական քարտուղար՝
ֆիզ. մաթ. գիտ. դոկտոր, պրոֆ. Մ.Ե. Հարությունյան

Тема диссертации утверждена в Институте проблем информатики и автоматизации НАН РА

Научный руководитель: чл.-корр. НАН РА, д.ф.-м.н. И.Д. Заславский

Официальные оппоненты: доктор физ.-мат. наук Э.М. Погосян
кандидат тех. наук А.Х. Манукян

Ведущая организация: Ереванский Государственный Университет

Защита состоится 10-го июня 2011г. в 16^{00} часов на заседании специализированного совета 037 “Информатика и вычислительные системы” ВАК, при ИПИА НАН РА по адресу: 0014, г. Ереван, ул. Паруйра Севака, 1.

С диссертацией можно ознакомиться в библиотеке ИПИА НАН РА.

Автореферат разослан 10-го мая 2011г.

Ученый секретарь специализированного совета
доктор физ. мат. наук, проф. М.Е. Арутюнян

ОБЩАЯ ХАРАКТЕРИСТИКА РАБОТЫ

Актуальность работы.

В последние годы существенно возросли объемы информации в виде разноязычных текстов в электронном виде. Для возможности быстрого анализа и передачи этой информации необходимо создание систем автоматизированного перевода с возможно меньшим потреблением машинных ресурсов и времени.

Универсальный сетевой язык (Universal Networking Language, сокр. UNL)[1] предоставляет возможность перевода с одного естественного языка на другой и может быть использован в качестве средства для хранения текстовых ресурсов в универсальном, семантическом виде, позволяя сравнительно легко анализировать информацию. Для реализации этих возможностей необходима разработка эффективного программного обеспечения, которое позволит преобразовывать тексты естественных языков в UNL.

Сложность и специфичность естественных языков делает практически невозможным создание совершенных словарных и грамматических ресурсов, необходимых для качественного перевода. По этой причине, в течение всей истории машинного перевода, в большинстве случаев прибегали к методу смешанного перевода (с пред- и постредактированием). Решением этой проблемы может стать способность программы обучаться, на основе решений, принятых человеком.

Цель диссертационной работы.

Целью диссертационной работы является разработка и реализация алгоритма, способного совершать NL→UNL и UNL→NL преобразования, как в полностью автоматическом режиме, так и в интерактивном. Разработанная программа должна обладать способностью анализа человеческого вмешательства в процессе интерактивного преобразования (обучаться) и дальнейшего использования накопленных знаний.

Научная новизна.

1. Ранее известные преобразователи EnConverter (NL→UNL) и DeConverter (UNL→NL) предоставляли возможность редактирования лишь двух узлов предложения на каждом шаге процесса преобразования. Разработанные в диссертации алгоритмы способны не только анализировать, но и редактировать неограниченное количество узлов предложения на каждом шаге. Более того, стало возможным распознавать и работать с многоуровневыми структурами предложений (подграфами).
2. До настоящего времени алгоритмы преобразования предложений из естественного языка в UNL структуры и обратно, использовали грамматические правила, которые преобразовывают списковые структуры конкретного языка сразу в семантические сети

[1] "UNL Specifications" UNL Center of UNDL Foundation, 2005.

универсального языка. Разработанный алгоритм основан на более гибком формате правил. Новый формат правил позволяет свободно манипулировать структурой графа предложения, например, предоставляет возможность непосредственного построения синтаксического дерева предложения. Синтаксическое представление является промежуточным, но важным (с лингвистической точки зрения) этапом преобразования предложений.

3. Впервые разработан алгоритм, позволяющий проводить интерактивный процесс преобразования предложений из естественного языка в UNL и обратно. В отличие от существующих преобразователей EnConverter и DeConverter, данный алгоритм позволяет пользователю вмешиваться в произвольную стадию процесса преобразования (разбиение предложения, выбор слов из словаря и применение грамматических правил) и делать необходимые изменения для корректирования результата.
4. Разработана автоматизированная система, которая позволяет NL↔UNL преобразователям накапливать знания (обучаться) с помощью анализа результатов вмешательства пользователя. Накопленные знания представляются в виде новых определяющих правил, интерпретирующих логику каждого вмешательства, и применяются в дальнейших преобразованиях.
5. Разработан новый алгоритм для сравнения UNL –узлов и грамматических правил на предмет совпадения требуемого множества или подмножества атрибутов. Алгоритм использует метод сопоставления каждому атрибуту однозначного бинарного числа, упрощая тем самым процедуру сравнения каждой пары "узел-условие" к одной бинарной операции *И (AND)*.
6. Преобразователи EnConverter и DeConverter в алгоритме словарного поиска используют метод индексации словаря для достижения приемлемой скорости преобразования. С целью сохранения высокой скорости словарного поиска, при использовании словарей больших размеров, представляется новый алгоритм словарного поиска, основанный на автомате Ахо-Корасик[2]. Алгоритм производит поиск в заранее построенном префиксном дереве словаря, не загружая его в оперативную память машины. Данный алгоритм также использует технику индексации. Примененные усовершенствования позволяют иметь стабильную скорость словарного поиска, не зависящего от размера словаря.
7. В отличие от предыдущих программ, разработанный преобразователь работает в обе стороны (из естественных языков в UNL или обратно). Результат работы программы зависит только от направленности грамматических правил, принимающих участие в процессе преобразования.
8. Впервые NL↔UNL преобразователи имеют возможность работы в средах с произвольными операционными системами (благодаря использованию языка

[2] V. Aho, M. J. Corasick, "Efficient String Matching: An Aid to Bibliographic Search" Communications of the ACM, 1977

программирования Java2). Реализована структура, позволяющая преобразователям запускаться неограниченным числом потоков, предоставляя тем самым возможность их использования в сетевых сервисах и прокси–серверах.

Используемые технологии.

Исходя из требований к характеристикам разрабатываемого программного обеспечения, проанализированы существующие технологии и выбраны наиболее подходящие для решения поставленных задач. Для обеспечения условий платформенной независимости и многопоточности используется язык программирования Java2. Система приспособлена для работы в сетевой среде, для чего использованы технологии сетевого программирования, такие как: Java Servlet, Java Server Pages, JavaScript, CSS2, HTML4 и технология AJaX.

Для словарного поиска в словарях UNL используется алгоритм, основанный на автомате Ахо-Корасик, с применением технологии индексации с целью ускорения процесса поиска. Изучены альтернативные методы и технологии словарного поиска, такие как: алгоритмы словарного поиска Кнута-Морриса-Пратта (автомат КМП) и Бойера-Мурра, а также открытые поисковые системы Apache Lucene и Cassandra. Проведен сравнительный анализ данных технологий и алгоритмов.

Практическая ценность.

Разработка автоматического и полуавтоматического двустороннего (NL↔UNL) преобразователя, со способностью обучения, позволит получить систему, способную переводить тексты с естественного языка в UNL и обратно, с целью хранения и семантического анализа или же дальнейшего преобразования в другой естественный язык.

Благодаря сделанным усовершенствованиям в алгоритмах применения грамматических правил и в методах словарного хранения и поиска, существенно снижается объем употребляемых машинных ресурсов и ускоряются процессы переноса используемых данных. Это позволяет системе оперировать на мобильных устройствах, что в наше время является важным условием для распространения UNL в различные сферы информационных технологий.

Основные положения, выносимые на защиту.

На защиту выносятся следующие положения:

1. Разработка и реализация алгоритма, предоставляющего возможность проведения процесса интерактивного преобразования NL↔UNL методом пользовательского вмешательства в стадиях анализа предложения и применения грамматических правил.
2. Разработка автоматизированного механизма обучения преобразователя, интегрированного в процесс преобразования, методом анализа пользовательского

вмешательства и генерирования новых определяющих правил для интерпретации каждого конкретного исправления.

3. Разработка алгоритма применения правил преобразования, позволяющего совершать двухстороннее (NL↔UNL) преобразование предложений естественного языка в UNL и обратно.
4. Разработка и реализация алгоритма поиска в UNL-словарях и в словарях естественных языков на основе автомата Ахо-Корасик. Проведение сравнительного анализа существующих алгоритмических и программных решений для задач данной области.
5. Применение бинарных операций в построении алгоритма для сравнения UNL-узлов на предмет совпадения требуемого множества или подмножества атрибутов.
6. Реализация централизованной сетевой системы программ, позволяющих:
    a. совершать преобразования предложений естественного языка в UNL и обратно, в автоматическом или в интерактивном режиме
    b. хранить и редактировать языковые ресурсы (словари, грамматические правила, документы на естественных языках или на УНЛ) и делиться ими с другими пользователями.

Описанные программы, как и сама система, являются платформенно независимыми приложениями и сконструированы с расчетом на использование в многопоточном режиме, что является необходимостью для их применения в качестве сетевых сервисов или прокси-серверов.

Результаты исследований диссертационной работы используются:

- в рамках проекта “UNL+3” швейцарской организации “UNDL Foundation”, созданной при поддержке ООН, призванного разработать централизованную сетевую среду для создания и развития UNL-ресурсов и необходимых инструментов для работы с ними (Женева, Швейцария);
- UNL-центром Армении, в рамках проекта по разработке UNL↔Армянских словарей и грамматических правил.

Апробация диссертационной работы.

Основные материалы диссертационной работы были представлены на:

- международной научной конференции CSIT в 2009 году, Ереван, Армения;
- международной научной конференции по принципам разработки и применению многоязычных сетей WordNet (Principles, Construction and Application of Multilingual Wordnets, 5th Global Wordnet Conference). Mumbai, India, 2010;
- На научном семинаре ИПИА НАН РА, 2011;

Публикации.

Материалы диссертационной работы опубликованы в 4 работах, в том числе в материалах двух международных конференций, и в двух статьях, список которых приведен в конце автореферата..

ОСНОВНОЕ СОДЕРЖАНИЕ РАБОТЫ

Во введении обоснована актуальность темы исследования, обозначена цель, основные задачи и методы их решения.

В первой главе проведен обзор современных исследований и литературы по алгоритмическим и техническим решениям, применимым в NL↔UNL преобразованиях. Обзор показывает существующие трудности и сферы необходимых усовершенствований.

Приведены некоторые сведения из истории машинного перевода и обзор подходов к переводу с использованием промежуточных искусственных языков. Кратко описывается язык UNL, его синтаксис, и сферы возможных применений.

Описаны имеющиеся программные средства, предназначенные для преобразования текстов естественных языков в UNL (EnConverter) и обратно – преобразования UNL-структур в тексты естественных языков (DeConverter). Описаны принципы работы программ EnConverter и DeConverter, существующие трудности их использования и необходимые усовершенствования, такие как:

- Разработка нового программного решения для преобразования предложений естественного языка в UNL и преобразования UNL-структур обратно в тексты естественных языков, основанного на едином алгоритме.

- Необходимость интерактивного процесса преобразования предложений естественного языка в UNL и обратно.
  - Изменение автоматическим способом выбранное разбиение предложения естественного языка, задав собственное.
  - Выбор наиболее подходящего универсального слова для произвольной части (слово, словосочетание, часть речи и т.п.) разбиения.
  - Вмешательство в ход применения правил методом ручного выбора более предпочтительных правил преобразования в произвольной стадии процесса.

- Разработка способности обучения преобразователей методом анализа интерактивного вмешательства в процессе преобразования.

- Разработка метода быстрой проверки UNL-предложения на соответствие с грамматическими правилами.

- Методы представления UNL-словарей и алгоритм ускоренного словарного поиска.

- Поддержка новой более гибкой структуры грамматических правил преобразования и возможности синтаксического редактирования.

Во второй главе проводится детальный обзор структур NL, UNL и NL↔UNL словарей и существенных проблем, связанных с постепенным возрастанием их объема. Проводится сравнительный анализ некоторых наиболее эффективных алгоритмических и программных методов для решения выявленных проблем. Предлагается решение поставленной задачи методом разработки и применения алгоритма представления (компиляции) NL-UNL словарей в древовидной структуре.

Описывается разработанный алгоритм словарного поиска, основанный на автомате Ахо-Корасик , и его применение. Также, предлагаются некоторые методы оптимизации для алгоритмов компиляции и словарного поиска, существенно ускоряющие процесс преобразования.

В спецификации UNL определены три типа словарей[3]:

1. Словарь UNL или словарь универсальных слов. Это список универсальных слов в алфавитном порядке с описанием их семантических свойств.

2. Словарь естественного языка (NL). Это список словарных записей естественного языка с описанием своих грамматических свойств.

3. Словарь UNL-NL. Это словарь, сопоставляющий записи UNL и NL-словарей, наряду с их описаниями.

Словарь UNL-NL может быть как направленным (UNL→NL или NL→UNL), так и двусторонне направленным (NL↔UNL). Словарь UNL→NL используется для преобразования UNL-структур в тексты естественных языков, а словарь NL→UNL используется для преобразования текстов естественных языков в UNL-структуры.

Словари представляют собой текстовые файлы, где каждая словарная запись (dictionary entry) записана одной строкой.

Для достижения высокой скорости и качества результата, в разработках современных алгоритмов программ словарного поиска, очень часто прибегают к существующим решениям из теории автоматов. В этом параграфе рассмотрены некоторые из этих алгоритмов, а именно: алгоритмы полного перебора (brute force), Бойера-Мура (Boyer–Moore, сокр. БМ), Кнута-Морриса-Пратта (Knuth–Morris–Pratt, сокр. КМП) и Ахо-Корасик (Aho–Corasick, сокр

[3] UNDL Foundation, www.unlweb.net, 2011

АК). Эти алгоритмы широко применимы и уже доказали свою эффективность в решении подобных задач.

Приведены особенности оценки сложностей каждого алгоритма, а именно:

- Особенность алгоритма полного перебора – это его простота. Время работы алгоритма можно считать приемлемым только в случае работы с набольшими объемами информации.
  Если дан текст, длиною в *m* символов, а суммарная длина всех искомых слов составляет *n* символов, тогда сложность алгоритма полного перебора оценивается в $O(m \bullet n)$.

- Алгоритм КМП является одним из самых распространенных алгоритмов для решения задач данной области. Преимущество этого алгоритма, по сравнению с полным перебором, в том, что когда в процессе поиска встречается несовпадение, то данная ситуация уже хранит в себе информацию, позволяющую определить, где именно следующее совпадение может находиться. Сложность алгоритма КМП оценивается в $O(m+n)$.

- В отличии от КМП, алгоритм БМ, сопоставляя начальные символы ключевого слова и текста поиска, начинает процесс сравнивания не с первого символа, а с последнего символа искомого слова, и продвигаясь справа налево, производит сравнивания символов. До начала поиска ключевое слово анализируется, что позволяет затем, при появлении несовпадений, совершать более широкие прыжки по тексту поиска. Данный алгоритм во многом зависит от структуры ключевого слова и текста поиска. Сложность алгоритма БМ, как и в случае КМП, оценивается в $O(m+n)$. Было доказано, что в наихудшем случае алгоритм БМ производит $3 \bullet m$ сравнений, а в наилучшем – m/n

- Еще одним эффективным решением является алгоритм словарного поиска, основанный на автомате предложенным Альфредом Ахо и Маргарет Корасик. Данный алгоритм предусмотрен для осуществления поиска строк в словарях или в других, структурированных подобным образом, текстовых документах.
  Работа этого алгоритма состоит из двух этапов. На первом этапе словарь, в котором должен осуществляться поиск, преобразовывается в древовидную структуру (префиксное дерево). А уже на втором этапе производятся поиски ключевых слов. Общее время работы алгоритма АК оценивается в $O(n+m+k)$. Но если учесть, что префиксное дерево строится лишь раз и затем может быть использовано многократно, то справедливо будет извлечь из оценки сложности этап построения префиксного дерева и оставить оценку самого процесса поиска. В таком случае получаем оценку в $O(n+k)$, где *k*-количество найденных слов.

В результате сравнительного анализа, выделены две основные причины выбора алгоритма АК в качестве наиболее подходящего, для решения нашей задачи. Причины следующие:

- При преобразованиях документов с большим количеством предложений и использовании для этого громоздких словарей, время, потраченное на конструирование словарного дерева, не играет большой роли, так как делается это лишь один раз. С другой стороны, мы получаем алгоритм более быстрого поиска слов.

- Вторая причина выбора алгоритма АК состоит в том, что алгоритмы КМП и БМ ориентированы на поиск в обычных текстовых документах, и при их использовании в словарном поиске мы теряем то преимущество, которое дает нам структура словаря.

Для возможности хранения префиксного дерева словаря на жестком диске машины и освобождения оперативной памяти (словари могут доходить до нескольких миллионов словарных записей в каждом, что не позволяет использовать их в оперативной памяти машины), был разработан новый формат, названный DXL, представляющий дерево словаря в текстовом формате.

Еще одна причина выбора алгоритма АК – это наличие возможностей его оптимизации и усовершенствования для максимального удовлетворения требованиям процессов NL↔UNL преобразований.

С целью ускорения процесса словарного поиска были совершены некоторые оптимизации.

1. Алгоритм АК был оптимизирован для нахождения всех словарных записей, имеющих полное совпадение с какой-либо частью предложения, методом выявления префиксных совпадений при поиске слова. С целью оценки эффективности предлагаемой оптимизации проводились исследования с использованием предложений различных длин, так как ее эффективность зависит от длины предложения и его содержания. При рассмотрении предложений длиною более 150 символов, можно утверждать, что получаем скорость, в десятки раз превышающую скорость обычного применения алгоритма АК.
2. Во многих современных поисковых системах используют технику индексации данных. В нашем случае, также было решено прибегнуть к данной технологии. Для этого был создан индексный массив, который ставил первые шесть символов слов всех словарных записей, вошедших в список компилированного словаря (префиксного дерева), в соответствие с их байтовыми адресами в DXL файле. Если слово короче шести символов, то в соответствие ставится все слово. Другими словами, мы получаем адреса всех узлов дерева, находящихся не дальше шестого уровня. Таким образом, для ключевого слова $K = \{ k_1, k_2, \ldots \ldots, k_n \}$ $(n>6)$, из индексного массива программа сможет найти местоположения словарных записей

для {"k_1", "k_1k_2", "$k_1k_2k_3$", "$k_1k_2k_3k_4$", "$k_1k_2k_3k_4k_5$", "$k_1k_2k_3k_4k_5k_6$"} в DXL-документе. Индексный массив создается в процессе компиляции словаря и записывается в отдельный файл вместе с документом DXL.

Выбор числа 6 для количества индексных символов основывается на тестированиях, и является оптимальным количеством, в соотношении размера массива и времени поиска. На рис. 1 изображен график, показывающий среднюю зависимость скорости общего процесса (T – среднее время анализа одного предложения) от количества индексируемых символов слов (N).

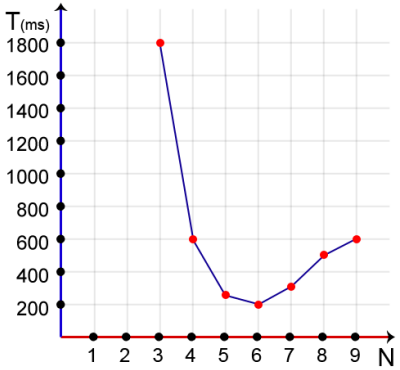

Рис. 1: Зависимость времени анализа от количества индексируемых символов.

Для получения представленных результатов, были проведены опыты с 20 случайно отобранными предложениями, длиною от 100 до 200 символов. Согласно графику, для наибольшей скорости анализа, оптимальным является число 6.

3. С целью ускорения переходов от одного узла к другому, во время компиляции словаря все узлы первого подуровня текущего (родительского) узла записываются в алфавитном порядке, а словарные записи записываются в самом начале. Таким образом, сделав переход в какой-либо уровень, при наличии словарной записи в этом уровне, она встречается в первую очередь. Затем, по очереди, рассматриваются все узлы данного уровня, в поисках нужного. Учитывая то, что все узлы этого уровня упорядочены в алфавитном порядке, во время проверки каждого, мы сравниваем его с искомым символом и, если оказывается, что рассматриваемый узел уже "больше" нужного символа, то продолжать процесс не стоит, так как все остальные узлы также будут "больше". В таком случае, сразу же вызывается функция ошибки и алгоритм завершает поиск, возвращая текущий выходной список, даже если он пуст. Благодаря данной оптимизации, экономится то время, которое затрачивалось на бессмысленную проверку ненужных узлов, и появляется

уверенность в том, что алгоритм не пропустит какое-либо префиксное совпадение, совершив преждевременный переход.

Разработанная программа на основе алгоритма АК, была также сравнена с некоторыми современными технологиями (Apache Lucene и Cassandra), которые также получили широкое применение в различных сферах разработок поисковых систем во всем мире. Усовершенствованный алгоритм АК показал результаты, превосходящие результаты данных технологий, чем и доказал свою эффективность в решении проблемы словарного поиска в процессах NL↔UNL преобразований.

В третьей главе дается детальное описание грамматических правил и их роль в процессах NL↔UNL преобразований. Приводится анализ функциональности грамматических правил для EnConverter и DeConverter и методов их усовершенствований. Описывается структура нового формата для правил и разработка их программной реализации.

Приводится описание экспериментальной работы по созданию альтернативы программы "DeConverter", названной "JDeCo". Описывается результат применения в JDeCo нового алгоритма для сопоставления правил преобразования и узлов предложения.

"Энконвертацией" называется процесс преобразования предложений естественного языка в UNL с помощью программы EnConverter, и "деконвертацией" -процесс обратного направления, производимый DeConverter–ом.

Грамматические правила энконвертация описывают конкретные условия и действия (такие, как изменение атрибутов узлов или построение синтактического дерева), выполняемые при удовлетворении этих условий.

Грамматические правила имеют возможность морфологического синтаксического и семантического анализа и преобразования узлов, таким образом, создавая UNL- сеть для входного предложения.

EnConverter анализирует узлы и модифицирует их с помощью специальных "окон", а условия и действия для каждого окна описывается в грамматическом правиле. Каждое поле грамматического правила выражает условие либо действие, следуя очередности :

1. Левое окно условия (Left Condition Window, сокр. LCW или PRE)
2. Левое окно анализа (Left Analysis Window, сокр. LAW)
3. Среднее окно условия (Middle Condition Window, сокр. MCW или MID)
4. Правое окно анализа (Right Analysis Window, сокр. RAW)
5. Правое окно условия (Right Condition Window, сокр. RCW или SUF)

Правила энконвертации имеют следующую структуру:

<TYPE>
["(" <PRE> ")" ["*"]]...
"{"|""" [<COND1>] ":" [<ACTION1>] ":" [<RELATION1>] ":" [<ROLE1>] "}"|"""

["(" <MID> ")" ["*"]]...
"{"|""" [<COND2>] ":" [<ACTION2>] ":" [<RELATION2>] ":" [<ROLE2>] "}"|"""
["(" <SUF> ")" ["*"]]...
"P(" <PRIORITY> ");"

Где символы, взятые в двойные кавычки, являются предопределенными знаками ограничений правила.

{ }	обозначает область действия данного фрагмента правила.
{ }... or [ ]...	обозначает повторения одного и того же описания не менее 1 раза.
|	обозначает выбор среди нескольких возможностей.
[ ]	обозначает необязательное присутствие данного объекта.
* после ( )	обозначает повторение узлов, соответствующих описанию, не менее 1 раза.

Выражения, записанные в {} и "", используются для описания правил, применяемых к узлам, указанным окнами анализа. Символы {} используются для описания присутствующего узла, тогда как "" используются для описания внедряемого узла.

{ } указывает на присутствующий узел из узлового списка (Node-list).

" " указывает на узел, который нужно добавить в узловой список (Node-list).

Каждое правило энконвертации имеет возможность добавления только одного узла в узловой список.

Приобщая, функцию грамматического правила можно описать следующим образом:

Когда узел, на который указывает левое окно анализа, удовлетворяет условию содержания атрибутов, описанных в <COND1>, и узел, на который указывает правое окно анализа, удовлетворяет условию в <COND2>, а их соседние узлы слева, справа и посередине удовлетворяют условиям, описанным соответственно в полях <PRE>, <SUF> и <MID>, тогда грамматические атрибуты узлов, указанных окнами анализа, переписываются соответственно командам, приведенным в полях <ACTION1> и <ACTION2>. Если в поле <RELATION1> или <RELATION2> описано какое-либо отношение, тогда между двумя узлами, указанными окнами анализа, создается данное отношение. Все операции совершаются на узловом списке, в соответствии с типом правила (<TYPE>)

Аналогичным образом работает также и алгоритм DeConverter-а.

Таким образом, поочередно рассматриваются все узлы предложения, и каждый раз, при нахождении соответствующего грамматического правила, совершаются необходимые преобразования, выдавая, в конечном итоге, преобразованное предложение.

В процессе разработок усовершенствованных алгоритмов для преобразований, первой попыткой решения выдвинутых задач была разработка программы *jDeCo* для преобразования UNL предложений в естественные языки. Эта программа использует алгоритм схожий с

алгоритмом DeConverter-a, но при этом предлагает заметные улучшения, способствующие решению ранее встречаемых проблем. Следует отметить, что программа написана на языке программирования *Java2 (отсюда и название jDeCo - **J**ava **DeCo**nverter)*. Язык *Java2* гарантирует платформенную независимость и позволяет программам работать в произвольной операционной системе и запускаться неограниченным числом потоков.

В ходе разработки *jDeCo* был предложен алгоритм для ускоренного обнаружения совпадений узлов, включающих в себя словарное описание (атрибуты), с условными полями грамматических правил.

Для этого, каждому атрибуту, использованному в ресурсах UNL, выделяется уникальное число, являющееся некоторой степенью числа 2, таким образом, каждому атрибуту соответствует некоторое число 2^k , где k = 0,1,2 … ∞, а в бинарном виде: $2^0 = 1$, $2^1 = 10$, $2^2 = 100$, $2^3 = 1000$ $2^4 = 10000$ и т.д. После этого любая компонента в ресурсах UNL, которая содержит в себе атрибуты, может быть заменена на аналогичную компоненту, содержащую лишь один атрибут: суммарную величину чисел атрибутов в бинарном виде, однозначно определяющую, какие именно атрибуты были использованы.

В таком случае, чтобы проверить содержит ли узел предложения атрибуты, необходимые для применения того или иного правила, достаточно лишь сравнить их суммарные числа с помощью бинарной операции *“И” (пр. 1001 И 1011 = 1001)*. Таким образом, сравнение узлов предложения и условных полей правил преобразований упрощается до одной бинарной операции, что значительно ускоряет процесс преобразования.

Приведем пример данной операции из процесса преобразования в jDeCo. Рассмотрим следующее UNL предложение:

```
[S:1]
  {org}
    Description de l'Egypte was the outcome of the collaboration of more than 150 prominent
    scholars and scientists who accompanied Bonaparte to Egypt in 1798.
  {/org}
  {unl}
    aoj(outcome(icl>result):0W.@past.@def.@entry, description(icl>action):00.@topic)
    mod(outcome(icl>result):0W.@past.@def.@entry, collaboration(icl>action):1B.@def)
    …
  {/unl}
[/S]
```

В процессе применения морфологических правил появится ситуация, где текущим состоянием предложения будет "Egypt description was the outcome collaboration of more 150 prominent scientists and scholars accompany Bonaparte Egypt 1798", а Окно Анализа будет указывать на узел "collaboration of", с атрибутами “N”, “def”, “gen” и с суммарной величиной

равной 1099797102592. Преобразователь, в поиске подходящих правил, найдет следующее правило:

m(N,@def: -@def , the &)P210;

Где:

m	- морфологическое (morphological) правило.
N, @def:	- если узел содержит атрибуты N и @def .
-@def , the &	- убрать атрибут @def, и добавить "the " в начало узла.
P210	- приоритет правила равен 210 (0-255).

Суммарная величина данного правила равна 1099796840448. Для проверки на соответствие, производится операция бинарного "***И***":

10000000000010001000001000000000000000000 *(1099797102592)*

И

10000000000010001000000000000000000000000 *(1099796840448)*

10000000000010001000000000000000000000000 *(1099796840448)*

Полученный результат совпадает с суммарной величиной условия правила, следовательно, правило будет применено, и атрибут @def будет удален, а в начало узла добавится строка "the ".

Кроме алгоритма сопоставления грамматических правил, был разработан также новый формат представления грамматических правил, позволяющий улучшить их эффективность.

Грамматические правила UNL делятся на две основные категории:

- Правила преобразования (Transformation rules)
- Определяющие правила (Disambiguation rules)

Правила преобразования используются для преобразования UNL предложений в естественные языки и обратно, тогда как определяющие правила используются в качестве мета-правил для улучшения работы применения правил преобразования, определяя, какие именно правила должны применяться, а какие - нет, в зависимости от ситуации. Оба типа правил применяются либо на узлах в структуре предложения естественного языка, либо на узлах в структуре графа UNL.

Правила преобразования имеют вид:

"α := β;" где левая часть – это условие, а правая часть - описание действий.

Определяющие правила имеют вид:

"α = P;" где левая часть – это условие, а правая часть - некоторое целое число $0 \leqq P \leqq 255$, определяющее вероятность появления ситуации, описанной в α (где P = 0 – ситуация невозможна, и P = 255 – ситуация имеет наибольший приоритет).

Четвертая глава посвящена разработке и реализации нового усовершенствованного преобразователя NL↔UNL. Подробным образом описываются структуры основных компонентов программы и алгоритма. Описывается метод, позволяющий использование единого основного алгоритма для преобразований в обоих направлениях. Детально описывается метод реализации интерактивного преобразования. Предлагается новое усовершенствование, которое заключается в разработке процедуры обучения методом автоматизированного анализа деятельности пользователя в процессе интерактивного преобразования и генерации соответствующих определяющих правил для дальнейшего использования.

В дальнейшем процесс преобразования предложений естественного языка в UNL структуры назовем Анализом (а программу наз. Анализатором), а обратный процесс - Генерацией (а программу наз. Генератором)

Одной из наших целей была разработка единого алгоритма, способного совершать преобразования в обе стороны (NL→UNL и UNL→NL). Очевидно, что для слияния двух произвольных алгоритмов, первостепенным является условие того, что данные, подаваемые на вход первому алгоритму, должны иметь идентичную структуру с входными данными второго алгоритма, а возвращаемый результат первого алгоритма должен иметь структуру, совпадающую соответственно со структурой результата второго.

Структура словарей, как и структура правил преобразования, почти одинакова для обоих направлений преобразования (Анализ и Генерация), что нельзя сказать про структуры входных предложений. Анализатор на вход получает предложение естественного языка (текстовую строку), тогда как Генератор получает предложение UNL, которое хотя и может содержать в себе свой изначальный вид естественного языка, однако эта информация никак не участвует в самом процессе генерации.

Стоит отметить, что в процессе Анализа (в некоторой его стадии), естественное предложение принимает форму UNL-предложения (не преобразовывается в UNL, а принимает UNL-форму описания), после чего только начинает работать основной универсальный алгоритм преобразования.

По своей сути, Анализатор (IAN) является дополнением к Генератору (EUGENE). Генератор получает на вход UNL-документ, грамматические правила и словарь, и сразу подает их на вход универсальному преобразователю. Анализатор, в свою очередь, совершает анализ предложения естественного языка, строит UNL-подобную структуру данного предложения и только затем передает на вход универсальному преобразователю.

Интерактивное преобразование

Первый шаг на пути к интерактивности алгоритма преобразования – это допуск пользователя к управлению анализом предложения естественного языка. Пользователю предоставляется возможность самостоятельно определять как разбиение предложения на части, так и сопоставление этим частям универсальных слов из словаря. В таком случае, запускается процесс ручного WSD (Word Sense Disambiguation). В зависимости от того, какой словарь был выбран (из базы данных или скомпилированный), могут запуститься 2 вида алгоритмов по выявлению и сопоставлению универсальных слов. Алгоритм, работающий при выборе скомпилированного словаря, подробно описан во второй главе. Здесь же описан метод выявления совпадений с использованием словарей из базы данных.

Итак, для того, чтобы найти все возможные совпадения из словаря, необходимо разбить предложение на все возможные части. Для предложения длиною в *n* символов получим число, которое нас не устроит, ведь это число равно $\sum_{k=1}^{n} k$, то есть, сумме от 1 до n. После получения всех вариантов разбиения предложения, базе данных отправляется SQL запрос для выявления словарных записей с полями HW, совпадающими с каким-либо из указанных частиц. Алгоритм работает по принципу "чем больше, тем лучше", предавая предпочтения соответствиям с наибольшим количеством букв или символов. Например, если для слова "республика" найдены словарные записи со значением "республика", то для этой части предложения сопоставляются именно эти записи, а записи со значениями "публика", "лик" или "а" больше не участвуют в сопоставлении с данным отрезком предложения.

Бывают случаи, когда принцип "чем больше, тем лучше" мешает правильному сопоставлению универсальных слов. Возьмем, например, предложение, содержащее отрезок "…этот витамин активно…". В этом случае, процесс WSD может выбрать следующее дробление {"этот", " ", "витамин а", "ктивно"}, несмотря на то, что для слова "ктивно" не будет найдено никаких сопоставлений. В таком случае, пользователь имеет возможность изменить автоматическим образом выбранное дробление на {"этот", " ", "витамин"," ", "активно"}.

Следующим условием достижения полной интерактивности процесса является возможность выбора подходящего универсального слова для каждого отрезка дробленого предложения. Нажимая на каждую из частей предложения, открывается окно с перечисленными вариантами соответствий этой части с кратким описанием каждого варианта. Варианты упорядочены по мере убывания их значения поля "Frequency" (соответствия представляют собой словарные записи, а у словарных записей типа NL→UNL, для определения приоритета записи, используются поля "Frequency"). По умолчанию выбран первый вариант (с наибольшим значением поля "Frequency"). Пользователю дана возможность изменить выбор на другое соответствие. Также пользователь может обозначить эту часть предложения "временной" ("[TEMP] a temporary element"). В основном, так обозначают те части, которые сами по себе не играют никакой роли, но могут охарактеризовать другую часть предложения или даже все предложение целиком. Также

имеется опция "Disregard", при помощи которой пользователь вычеркивает данную часть из предложения.

Как уже было сказано, процесс преобразования практически идентичен для обоих направлений (Генерация и Анализ), и разница состоит лишь в том, в каком направлении действуют правила преобразования. Если правила преобразования имеют направленность {Отношение → Список}, тогда будет иметь место преобразование из UNL-предложений в предложения естественного языка (Генерация), а если правила имеют обратную направленность, тогда предложения естественного языка преобразуются в предложения UNL (Анализ).

Нам известно, что предложение UNL невозможно представить в виде упорядоченного списка, не делая никаких преобразований, а значит, для выполнения вышеупомянутого условия придется представить предложение естественного языка в некоторой форме, соответствующей синтаксису UNL. В этом нам помогает тот факт, что после процесса WSD предложение можно представить в виде упорядоченного списка универсальных слов, а их, в свою очередь, как предложение UNL, где вместо отношений, будет список универсальных слов.

На вход преобразователю дается уже не предложение естественного языка, а предложение UNL, которому предстоит пройти процесс преобразования, только после которого оно будет корректно отображать данное предложение естественного языка на UNL.

После передачи предложения универсальному преобразователю, начинается процесс применения правил преобразования.

Преобразователь цикличным образом проверяет имеющиеся правила преобразования, находит применимые правила (кандидаты), и консультируясь с определяющими правилами, применяет наиболее подходящее правило.

Если пользователь заранее выбрал интерактивный процесс применения правил, тогда в выданном результате он увидит всех кандидатов для каждого шага применения правила и тем самым получит возможность изменять выбор правил и перезапускать процесс по новому руслу.

Таким образом, разработанные преобразователи предоставляют возможность полного контроля над процессом преобразования.

Обучение

Несмотря на то, что интерактивность процессов преобразований позволяет значительно улучшить качество результата, следует отметить, что делать одни и те же исправления во множестве подобных друг другу случаев крайне неудобно, а иногда даже невозможно (например, в случае работы с текстами объемом в несколько тысяч предложений). Появляется необходимость нового усовершенствования, которое позволит избежать повторения одинаковых ошибок для преобразования. Иными словами, нужно разработать алгоритм, позволяющий системе обучаться и не совершать ошибки, подобные которым уже совершались. Для достижения данной цели, необходимо решить следующие две задачи:

- Разработка способа анализа пользовательского вмешательства (обучение).
- Применение знаний, полученных при обучении.

Наиболее подходящим способом для анализа пользовательского вмешательства и применения этого опыта, является генерация определяющих правил, выражающих предпочтения пользователя и добавление их к списку определяющих правил при дальнейших преобразованиях.

Когда режим обучения включен, пользователю, после каждого изменения соответствующего универсального слова какой-либо части предложения, предлагается объяснить свой выбор несколькими этапами.

Первый этап – это выбор элементов предложения, повлиявших на решение пользователя. Программа просит пользователя указать все необходимые элементы (слова, словосочетания и т.д.), для генерации нового правила. Указанные элементы могут быть важны как со стороны семантического содержания, так и синтаксического.

На втором этапе, пользователь уже должен указать какие конкретные свойства этих элементов повлияли на его выбор. Указанные на первом этапе элементы предложения представляются в "раскрытом" виде, то есть перечисляются их атрибуты, выбранные универсальные слова и т.д. Пользователь отмечает пункты, необходимые для оправдания сделанного им выбора.

После получения от пользователя всей необходимой информации, генерируется определяющее правило, интерпретирующее логику данного предпочтения на "языке" UNL-грамматики. Сгенерированное правило представляется пользователю для возможности ручного редактирования и затем добавляется в базу знаний системы, для использования в дальнейших преобразованиях.

Аналогичным методом ведется также и процесс обучения выбору наиболее подходящего правила преобразования. Когда пользователь на некотором шаге интерактивного преобразования выбирает другое применимое правило, система также предлагает обосновать свой выбор. Предложение, полученное в результате применения данного правила, показывается в "раскрытом" виде, включающим в себя все, созданные к этому времени, отношения и список узлов. Пользователь отбирает те элементы предложения, которые образуют желаемую структуру, полученную с применением выбранного им правила. Выбрав нужные элементы и/или их компоненты, пользователь получает сгенерированное правило и так же, как и в случае обучения словарного выбора, может отредактировать его и сохранить, добавив в базу знаний.

В заключении подведены основные итоги проведенных исследований.

Таким образом, разработаны новые алгоритмы, работа которых проверена экспериментально.

1. Разработан и реализован алгоритм интерактивного преобразования UNL-структур в структуры естественного языка и обратного преобразования; процесс работы алгоритма может проходить как в полностью автоматическом режиме, так и с допущением вмешательства пользователя. Результаты вмешательства могут быть анализированы с целью усовершенствования алгоритма. Для этого система с помощью пользователя может создавать определяющие правила, интерпретирующие его предпочтения для каждой конкретной ситуации или же определенного класса ситуаций.[4]
2. Построены новые алгоритмы словарного поиска, основанные на учете сравнительного анализа и результатов экспериментов, позволяющие значительно ускорить процесс словарного поиска (результаты приведены в таблице 4).[3][4]
3. Разработан универсальный единый алгоритм, лежащий в основе алгоритмов преобразования предложений естественного языка в UNL и обратно. Алгоритм основан на использовании нового формата грамматических правил, что дает возможность наиболее детального анализа предложения в процессе преобразования, включая возможность не только семантического и морфологического (как в случаи предыдущих реализаций), но и синтаксического анализа.[1][4]
4. Разработан алгоритм минимизации операций поиска и сопоставления грамматических правил , позволяющий свести процедуру сравнения пар "узел-условие" к одной бинарной операции "И" между двумя бинарными числами.[1]

Основные результаты диссертации отражены в следующих публикациях:

1. A. Avetisyan, "Some approaches to the generation of sentences in natural language from UNL", proceedings of the conference CSIT. Yerevan 2009, pp. 268-270.

2. A. Avetisyan, V. Avetisyan, "LOOK4: Enhanecment of web search results with Universal Words and WordNet" Principles, Construction and Application of Multilingual Wordnets, proceedings of the 5th Global Wordnet Conference. Mumbai, India 2010, pp. 111-115.

3. I. Zaslawskiy, A. Avetisyan, V. Gevorgyan, "Implementation of Dictionary Lookup Automata for UNL Analysis and Generation", International Journal "Information Theories and Applications", ISSN 1310-0513, Volume 17, Number 2, Sofia, Bulgaria 2010, pp. 132-139.

4. A. Avetisyan, "Development and Application of Interactive Algorithms for Natural Language to UNL and UNL to Natural Language Transformations", Mathematical Problems of Computer Science Volume 35, IIAP NAS RA, Yerevan, 2011, pp. 5-13.

ԱՄՓՈՓՈՒՄ

Աշխատանքում հետազոտվում են գոյություն ունեցող UNL ռեսուրսները և նրանց հետ աշխատանքի համար մինչ այժմ մշակված ծրագրային միջոցները: Նաև նկարագրվում է նոր և բարելավված ծրագրային միջոցների մշակումը, որոնք օգտագործում են համեմատաբար նոր և արդյունավետ տեխնոլոգիաներ և ալգորիթմներ: Կիրառված բարելավումները ուղղված են բարձրացնելու մշակվող ծրագրային փաթեթի աշխատանքի որակը և արագությունը, ինչպես նաև դարձնել այն հասանելի ինտերնետում: Որակական բարելավման նպատակով մշակված է UNL-ի և բնական լեզուների փոխադարձ ներկայացման համար երկխոսական (ինտերակտիվ) համակարգ, որը թույլ է տալիս օգտագործողին միջամտել նախադասության սերման գործընթացին և հնարավորություն է տալիս համակարգին անալիզի ենթարկել այդ միջամտության արդյունքները (սովորել)՝ ստացած գիտելիքները հետագայում կիրառելու նպատակով:

UNL-ը ներկայացնում է ինֆորմացիան սեմանտիկ ցանցերի տեսքով: Յուրաքանչյուր նախադասություն ներկայացվում է կապակցված և ուղղորդված գրաֆի տեսքով: Ի տարբերություն բնական լեզվի, UNL-ով նկարագրված արտահայտության իմաստը միանշանակ է: UNL սեմանտիկ գրաֆերում հանգույցներին համապատասխանեցված են հասկացություններ (concept), որոնք կոչվում են «ունիվերսալ բառեր» (Universal Word, կրճատ՝ UW), իսկ կողերին՝ իմաստային (սեմանտիկ) կապեր, որոնք կոչվում են «հարաբերություններ» (relation):

Ատենախոսությունը բաղկացած է ներածությունից, չորս գլուխներից, արդյունքների ամփոփումից և օգտագործվող գրականության ցանկից:

Ներածությունում հիմնավորված է հետազոտվող թեմայի արդիականությունը, նշված են նպատակները, հիմնական խնդիրները և նրանց լուծման մեթոդները :

Գլուխ 1.-ում դիտարկված են NL↔UNL փոխադարձ ներկայացման համար կիրառելի ալգորիթմական և տեխնիկական լուծումներ և հետազոտությունների արդյունքներ: Նշված են գոյություն ունեցուղ խնդիրները և բարեփոխումների հնարավոր միջոցներ:

Գլուխ 2.-ում մանրամասնորեն քննարկվում են NL, UNL և NL-UNL բառարանների կառուցվածքը և նրանց չափսերի աստիճանաբար աճի պատճառով առաջ եկող խնդիրները: Ներկայացվում է նշված խնիրների լուծման համար նպատակահարմար

որոշակի ալգորիթմական և ծրագրային միջոցների համեմատական անալիզ: Որպես լուծում, առաջարկվում է Ահո-Կորասիկի ավտոմատի հիման վրա կառուցվող բառարանային փնտրման ալգորիթմի մշակումը և կիրառումը, ինչպես նաև, այդ ալգորիթմի բարելավման համար որոշակի լուծումներ:

Գլուխ 3.-ում ներկայացված են գոյություն ունեցող քերականական կանոների կառուցվածքը և ֆունկցիոնալությունը, նաև նրանց օպտիմիզացիայի և ֆունկցիոնալության բարձրացման նպատակներով կատարված բարեփոխումները:

Բերված են «JDeCo» կոչվող, DeConverter-ին փոխարինող էկսպերիմենտալ աշխատանքի արդյունքները, որոնց թվում է քերականական կանոնների պայմանի և նախադասության հանգույցների համեմատման նոր ալգորիտմը:

Գլուխ 4.-ում ներկայացված է NL↔UNL փոխադարձ ներկայացման նոր ալգորիթմի մշակումը և իրականացումը: Նկարագրված են ալգորիթմի և նրա հիման վրա կառուցված ծրագրի հիմնական բաղադրիչ մասերը: Մշակված է ունիվերսալ ալգորիթմ, որը թույլ է տալիս կատարել նախադասությունների երկկողմանի (UNL-ից բնական լեզու և բնական լեզվից UNL-ի) սերում: Մանրամասնորեն նկարագրված է երկխոսական համակարգի մշակումն ու իրականացումը, ինչպես նաև համակարգի ուսուցման ալգորիթմը:

Ամփոփման մեջ, բերված են ատենախոսության շրջանակներում ստացված հիմնական արդյունքները:

ABSTRACT

Aram Avetisyan

THE DEVELOPMENT OF INTERACTIVE AND LEARNING SYSTEM FOR UNL AND NATURAL LANGUAGE TWO-DIRECTIONAL AUTOMATIC REPRESENTATION

This thesis studies the existing UNL (Universal Networking Language) resources and the available tools developed to work with them. We also describe the approaches in the development of some enhanced solutions, with the usage of newer technologies and more efficient algorithms. The solutions used, are aimed to enhance the quality of the developed program tools, increase the processing speed and to insure their availability on the web. A UNL and natural language (NL) two-directional interactive representation system has been developed to enhance the processing result quality, by providing the user with an opportunity to intervene into NL↔UNL analysis and generation processes and make necessary corrections. Learning ability is developed by allowing the system to analyze the user corrections during the interactive processes and to use the gained knowledge in the future.

UNL represents the natural language texts in a form of sematic networks. Each UNL expression can be presented as a connected directed graph. Unlike natural languages, UNL expressions are unambiguous.

In UNL semantic graphs a concept is presented as a node. Those concepts are also called "Universal Words" or UWs. While the nodes represent concepts, the arcs represent the semantic relations between the concepts (UWs).

The thesis consists of the following: an introduction, four chapters, results summary and a list of referred literature.

The aim and main problems studied in the thesis are briefly presented in the introduction, along with the current status of UNL project developments.

Chapter 1 studies some technologies and algorithmic solutions of machine translation theory, which may become a basis for the enhancement of NL↔UNL representation. This chapter highlights the main obstacles in the usage of currently available software for further UNL development.

Chapter 2 introduces the NL, UNL and NL-UNL dictionaries structure and the problems that arise along with the gradual increase of developed dictionaries sizes. Several most suitable algorithms and technologies are being discussed in a comparative analysis in order find the possible solutions for the mentioned size handling problems.

The Aho-Corasick string matching machine is being suggested as the most suitable basis for the development of NL-UNL dictionary matching algorithm to be used during the natural language analysis and UNL generation processes.

In chapter 3 we present the current UNL grammar structures and the functionality, along with several steps taken for its optimization and enhancement.

A project called "jDeCo" is being introduced, which is an experimental tool, is aimed to enhance the functionality of the DeConverter, by suggesting several algorithmic and technological improvements, including a new efficient rule matching algorithm.

The development and the implementation of a new two-directional NL↔UNL representation algorithm is presented in chapter 4. The main compound parts of both the algorithm and the program are introduced. An algorithm is developed, that allows combining the main mechanisms of NL→UNL and UNL→NL processes into one. Also, a detailed description introducing the interactive processes of the natural language analysis and UNL generation, as well as the developed algorithm of the system learning are provided.

The last section summarizes the main results of the studies and developments described in the thesis.