ЮГОЗАПАДЕН УНИВЕРСИТЕТ „НЕОФИТ РИЛСКИ”

Природо–математически факултет **Катедра „Информатика”**

Надежда Пламенова Борисова

Семантичен уеб - модели и приложения

АВТОРЕФЕРАТ

на дисертационен труд

за присъждане на образователната и научна степен „ДОКТОР“
в област на висше образование 4. Природни науки, математика и информатика;
професионално направление 4.6 Информатика и компютърни науки;

Научни ръководители: доц. д-р Димитрина Полимирова
доц. д-р Елена Каращранова

БЛАГОЕВГРАД

2015

Дисертационният труд е обсъден и насочен за защита на разширено заседание на катедра “Информатика” при Природо-математически факултет на Югозападен университет “Неофит Рилски”, състояло се на 25.08.2015 г.

Дисертационният труд “Семантичен уеб – модели и приложения” е с общ обем от 129 страници. Състои се от увод, шест глави, заключение, списък с научно-приложни приноси и 2 приложения. Използваната литература включва 85 източника, от които 75 на английски и 10 на български език.

Списъкът на авторските публикации се състои от пет заглавия, четири от които в съавторство.

Защитата на дисертационния труд ще се състои на 5.10.2015 година от ..... часа в зала ..... на Югозападен университет „Неофит Рилски“.

Съдържание

Увод 5
1. Семантичен уеб - теоретични основи, инструменти и приложения 7
1.1 Тенденции в развитието на уеб технологиите 7
1.2 Семантичен уеб – технологии и стандарти 8
1.3 Семантичен уеб и онтологии 10
1.4 Характерни приложения на онтологиите в извличането на информация и управлението на знания 10
1.5 Изводи 12
2. Извличане на информация и семантичен уеб 12
2.1 Класификация на задачите за извличане на информация 13
2.1.1 Named Entity recognition (NE) 14
2.1.2 Coreference resolution (CO) 14
2.1.3 Template Element construction (TE) 14
2.1.4 Template Relation construction (TR) 14
2.1.5 Scenario Template production (ST) 15
2.2 Онтологично базирано извличане на информация 16
2.3 Изводи 17
3. Лингвистични технологии, свързани със семантичния уеб 18
3.1 GATE 19
3.2 UIMA 20
3.3 OpenNLP 21
3.4 Изводи 21
4. Подход за автоматично маркиране на неправилно съгласуване на прилагателно име със съществително име в български текст 22
4.1 Сегментация на текста на ниво изречения 23
4.2 Токънизация 23
4.3 Анотиране на отделните токъни с тяхната респективна категория за част на речта (POS tagging) 23
4.4 Маркиране на двойките прилагателно-съществително, които не са съгласувани по род и число 24
4.5 Изводи 26
5. Реализиране на инструмент за лематизация на български текст в GATE 26

5.1 Извличане на списък с леми от BG Office project 29
5.2 Извличане на списък с леми от Bulgarian-language Wiktionary project 29
5.3 Използване на вграден речник 30
5.4 Използване на BGLemmatizer като инструмент в платформата GATE 30
5.5 Изследване на ефективността на BGLemmatizer като средство за обработка на български текст 30
5.6 Изводи 35
6. Извличане на онтологично базирани понятия от неструктуриран текст на български език 36
6.1 Токънизация 37
6.2 Разделяне на текста на изречения 39
6.3 Анотирането на отделните токъни с респективната им част на речта (*POS tagging*) 39
6.4 Лематизация 40
6.5 Маркиране на термини в текстовия документ, съвпадащи с лингвистичните данни за концептите в онтологията 41
6.6 Тестване 42
6.7 Изводи 44
Заключение 45
Използвана литература 47

Увод

Семантичният уеб (*Semantic Web*) цели изграждане на машинно обработваем слой от метаданни над съществуващата мрежа от хипертекстови документи, с което да предостави възможност за търсене и споделяне на ресурси по смисловото им съдържание. Това поражда необходимост от такова представяне на информацията в глобалната мрежа, с което нейният смисъл ще бъде възприет и използван едновременно от потребители и от компютри [Berners-Lee et al., 2001].

Семантичните уеб технологии разширяват съществуващите уеб услуги, като предоставят възможност за реализиране на хранилища на данни в мрежата, изграждане на речници от понятия и задаване на правила за обработка на данни. Технологичните стандарти на World Wide Web Consortium (W3C), предназначени за описване и свързване на ресурсите в уеб пространството, целят да трансформират WWW от хипертекстова в разпределена система, базирана на знания.

Развитието в посока към семантичен уеб поставя за изпълнение редица задачи, които са актуални и отворени към момента. Част от тези задачи са езиково независими (например създаване на онтологии), докато други са специфични за конкретен език. От технологична гледна точка, семантичният уеб цели добавяне на семантична информация към уеб съдържание, създавайки среда, в която софтуерните агенти ще изпълняват задачите си по-ефективно. Реализацията на тази визия изисква автоматизация на следните процеси:

- Създаване на **семантична анотация** (метаданни) за значението или смисъла на данните, съдържащи се в хипертекстови документи;
- Създаване и взаимодействие с **онтологии**;
- Свързване на **уеб съдържание с онтологии**;
- Компютърна **обработка на естествен език** (Natural Language Processing, NLP);
- **Извличане на знания** от текст;

Хипертекстовите документи съдържат предимно информация на естествен език. Следователно, степента на автоматизация на гореспоменатите процеси зависи от ефективността на NLP – техники, използвани за обработка на уеб съдържанието.

Въпреки разнообразните подходи за извличане на знания от текст, резултатите към момента са ограничени, поради семантичната разлика между естествения (човешки) език и формалните езици за описание на знания.

Посочените задачи са решими чрез използване на съвместими със семантичния уеб инструменти за езикови технологии и разработване на набор от свободно достъпни, адаптивни компоненти, които интегрират данните за езика със семантичните уеб данни под формата на онтологии.

В резултат на изследване, проведено от Европейската мрежа за върхови постижения META-NET [1] [Blagoeva et al., 2012], е анализирано развитието в областта на езиковите технологии и техният потенциал като възможности за сътрудничество и споделяне на знание между крайни потребители. Авторите са представили основните сфери на приложение на езиковите технологии, както и оценка на актуалната ситуация в областта на езиковите технологии. От направеният обзор и оценка на състоянието на наличните езикови технологии за всеки от изследваните 30 езика, които се говорят в Европа, нито едно състояние не е оценено като „отлично". Ресурсите за английски език са получили висока оценка, а **българският език е** категоризиран като „*фрагментарно развит*", което означава, че е сред езиците **с висок риск от отмиране в дигиталната епоха.** Развитието на изследванията за български език в тази област се затруднява и от факта, че като славянски език притежава „*богата флективна система*".

Настоящият дисертационен труд частично допълва този инструментариум, като представя подход за онтологично базирано извличане на информация от неструктуриран текст на български език.

В много отношения резултатите в дисертационния труд са постигнати с използване на специфична методика за анализ на текст и теоретични разработки, свързани със спецификата на българския език.

Цел на настоящия дисертационен труд е изследване на възможностите за онтологично базирано извличане на информация от неструктуриран текст на български език.

За реализация на тази цел са поставени следните **конкретни задачи**:

1. Да се изследва текущото развитие на технологиите, свързани със семантичния уеб;
2. Да се анализират характерни конкретни приложения на онтологиите в извличането на информация и управлението на знания;
3. Да се анализират съществуващи платформи с отворен код за анализ на текст, реализиращи дейности по извличане на информация, като се анализират и съществуващите в тях средства за обработка на български текст;
4. Да се реализира инструмент за обработка на български текст в платформа с отворен код, с цел изпълнение на дейности по извличане на информация и да се изследва ефективността на предложения инструмент за обработка на български текст;
5. Да се създаде модел за извличане онтологично базирани понятия от неструктуриран текст на български език.

[1] В META-NET членуват 60 изследователски центъра от 34 страни, сред които е и Институтът за български език „Проф. Любомир Андрейчин"

Основната част от съдържанието е разделено в пет глави, като за решаването на всяка една от първите четири задачи е отделена по една глава, а петата задача присъства във всяка една от последните три глави.

Макар че реализираният подход (Глава 6) в тази дисертация обхваща етапи от предварителната обработка на текста (маркиране на думите с граматична категория за част на речта и лематизация), в Глави 4 и 5 е представена по-пълна картина на тези етапи, с която да се покаже ролята и приложението на тези резултати в развитието на семантичния уеб и онтологиите. С други думи, ролята в решаването на различни проблеми в развитието на семантичния уеб.

1. Семантичен уеб - теоретични основи, инструменти и приложения

1.1 Тенденции в развитието на уеб технологиите

Всеки етап от развитието на Интернет се характеризира с промяна на технологиите и основните принципи на публикуване и споделяне на информация:

- **Web 1.0** (според Бърнърс-Лий "*read-only web*") се характеризира с бавни връзки и статични сайтове. Съдържанието се създава и публикува на уеб сайтове с ограничено интерактивно взаимодействие, а потребителите могат единствено да търсят информация и да я четат [Aghaei et al., 2012].

- **Web 2.0** (или „*read-write web*") се въвежда за първи път от Тим О'Райли през 2003 г. и се популяризира от компанията O'Reilly Media през 2004 [O'Reilly 2005]. За разлика от предходни години, основната цел на Web 2.0 е да предостави възможност за сътрудничество и обмен на данни между отделните потребители, като голяма част от тези услуги са безплатни. При Web 2.0 навлизат технологии като: софтуер за изграждане и поддържане на уикита (wiki software), флаш дизайн (flash design), ajax програмиране. Постепенно се налага и тенденцията да се използва софтуер с отворен код (open source software). Според [O'Reilly & Battelle, 2006] "*Мрежата като платформа*" предоставя много повече от "*софтуер като услуга*", по отношение подобряване на предлаганите услуги чрез обратна връзка с потребителите.

- **Web 3.0** – (според [Getting 2007] "*read-write-execute web*") най-общо се характеризира с представяне на висококачествено съдържание и услуги. Този етап на развитие включва изкуствен интелект, семантични мрежи, представяне на знания, онтологии, софтуерни агенти и др. Точният смисъл на този термин все още се дискутира, но често се отъждествява със *Семантичен уеб*. В контекста на семантичния уеб, Web 3.0 е развиващо се продължение на World Wide Web, в което всяко уеб съдържание може да се представя не само на естествен език, но и в машинно „*разбираем*" вид, който да се тълкува и използва

от софтуерни агенти. По този начин се постига по-бързо и лесно намиране, споделяне и интегриране на информацията.

1.2 Семантичен уеб – технологии и стандарти

Терминът “Semantic Web” е въведен от Тим Бърнърс-Лий като "мрежа от данни, които могат да бъдат обработвани пряко и косвено от машини" [Berners-Lee et al., 2001]. Проектът Семантичен уеб се фокусира върху създаване на специализирана система за обмен на знание. По думите на Бърнърс-Лий, семантичната мрежа ще позволи да се създават уеб приложения, които по функционалност ще превъзхождат многократно всяка една от съвременните услуги.

Глобалната семантична мрежа е проект на World Wide Web Consortium (W3C), с който се цели преминаване от класическия "*Уеб на документи*" към "*Уеб на данни*”. Такава мрежа ще предостави възможност на компютрите да извършват повече полезна работа чрез използването на системи, които осигуряват надеждни взаимодействия по мрежата. Терминът "*Семантичен уеб*" се отнася до визията на W3C за мрежа от свързани данни. Семантичните уеб технологии позволяват да се създават хранилища на данни в мрежата, да се изграждат речници и да се задават правила за обработка на данните. Свързаните данни могат да се използват чрез технологии като: RDF (Resource Description Framework), SPARQL (SPARQL Protocol and RDF Query Language), OWL (Web Ontology Language) и SKOS (Simple Knowledge Organization System) [W3C].

Основните езици, които съставляват Семантичният уеб, формират единен стек (Фигура 1-1) и е прието всяко ново развитие в тази област да е надграждане над нивата на съществуващите езици. Всяко следващо ниво се очаква да бъде съвместимо с предходните и по този начин всяка инвестиция, направена в по-стара технология, да се запази.

Според [GEIST 2010/2011] за реализацията на семантичния уеб са необходими:

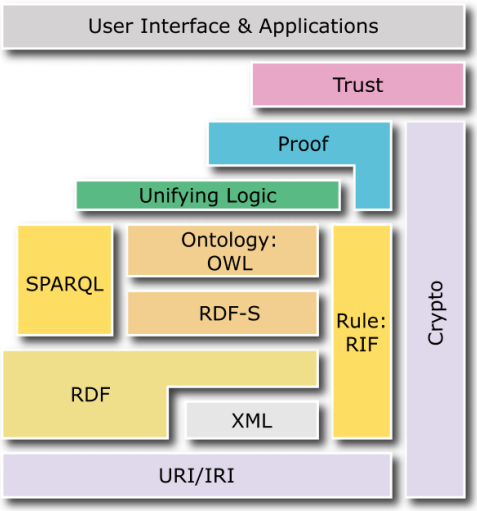
- Машинно обработваеми метаданни (Explicit Metadata), чрез които се задава значението или смисълът на данните в уеб съдържанието.
- Онтологии (Ontologies) за осигуряване на контролиран речник на понятията от конкретна област, в който всяко понятие е с прецизна дефиниция и машинно-обработваема семантика. Понятията (terms), наричани още класове, концепти или фреймове, в същината си са описание на обекти и се характеризират със свойства (properties), чрез които могат да се свързват помежду си. Свойствата, като начин за представяне на съществуващите връзки между понятията в конкретна област, са основата на йерархичната и мрежова структура на онтологията. В рамките на онтологията чрез аксиоми и ограничения се дефинират правила, които са винаги верни и осигуряват проверка за коректността на въвежданите данни.

- Откриване на неочаквани връзки и противоречия в онтологичното знание (Logic and inference). Извършването на автоматичен логически извод (reasoning), в заключение от предоставеното знание, се реализира чрез машини за извод (reasoners).
- Софтуерни агенти (Agents), които в глобалната семантична мрежа:
  - Получават задачи и предпочитания от потребителя;
  - Търсят уеб ресурси, комуникирайки си с други агенти;
  - Сравняват и селектират информацията според потребителските предпочитания и изисквания;
  - Предоставят отговори на потребителя.

Текущо състояние и стандартизация

В стъпаловидния модел на структурата на семантичния уеб (Фигура 1-1), средните слоеве съдържат стандартизирани от W3C технологии [W3C 2015] за изграждане на семантични уеб приложения: RDF, RDF-S, OWL, SPARQL и RIF (Rule Interchange Format).

Най-горните слоеве (*Cryptography*, *Proof* и *Trust)* съдържат идеи, с които да се верифицира надеждността на източниците, да се предостави възможност за логически извод на нови твърдения и сигурност на използваните данни и услуги. За тяхната реализация все още не са стандартизирани технологии.



Фигура 1-1. Технологичен стек на семантичния уеб [W3C, 2007]

Архитектурата на семантичния уеб продължава да се обсъжда и може да бъде предмет на бъдещи промени и подобрения.

Фокусът в настоящия дисертационен труд е насочен към изследване на възможността за онтологично базирано извличане на понятия от неструктуриран текст на български език. Представеният подход обхваща основните компоненти от технологичния стек, свързани с представяне на знания в семантичния уеб, като използва методика за анализ на текст, съобразена със спецификата на българския език.

1.3 Семантичен уеб и онтологии

Основната цел на семантичния уеб е всеобхватно и машинно „*разбираемо*“ знание, за което се разчита основно на формални онтологии за структуриране на основните данни. Следователно, успешната реализация на семантичния уеб изисква автоматизация на процесите, свързани с проектиране, създаване, разпространение и взаимодействие с онтологии.

С използването на онтологии в глобалната семантична мрежа се цели да се улесни споделянето на информация от конкретна област между потребители и софтуерни агенти. Чрез онтологията се осигурява организация на знания от областта в обща структура, която може да бъде анализирана, споделяна и многократно използвана. В такава структура ясно могат да се дефинират понятията в областта от връзките между тях. По този начин структурата на знанията може да бъде използвана повторно за анализ от различен характер.

Исторически понятието „онтология” произлиза от философията - “*the study of the nature of existence*”, но в контекста на семантичната глобална мрежа онтологията описва понятията, съществуващи в конкретна област и релациите между тях.

Автор на най-известната дефиниция за „онтология” е Грубер през 1993г. [Gruber 1993]: „*an explicit specification of a conceptualization*” (експлицитна спецификация на концептуализацията). Формално онтологията се състои от понятия, организирани в таксономия, техните дефиниции, атрибути и свързаните с тях аксиоми и правила за извод.

Според Грубер „*A conceptualization is an abstract, simplified view of the world that we wish to represent for some purpose.*” С други думи предметната област може да бъде концептуализирана по различен начин съгласно представите на автора и целите на онтологията.

Различните дефиниции в литературата разкриват специфични аспекти на ролята, която онтологията играе. Всички дефиниции обаче споделят идеята, че онтологията осигурява описание на дадена гледна точка за предметната област. [Атанасова 2011]

1.4 Характерни приложения на онтологиите в извличането на информация и управлението на знания

Използването на онтологии за моделиране на знания от специфични области представлява ключов аспект за интеграция на информацията, постъпваща от различни източници, за подпомагане на съвместната работа в рамките на виртуални общности, за подобряване на процеса на извличане на информация и е от значение за извършването на автоматични логически изводи от предоставените знания.

Онтологиите са ключов елемент за реализиране на идеята за Семантичния уеб. Популярността им нараства, тъй като предлагат елегантни

решения на предизвикателства в областта на информационните технологии, като: оперативна съвместимост (interoperability), споделяне на знания (knowledge sharing), повторна използваемост на знания (knowledge reuse/reusability) и др. [Нишева 2013].

Онтологиите имат потенциал за приложения, които предоставят речник на предметна област, предназначен за потребители и програмни системи. Също така, могат да се използват при проектиране на навигацията, структурата и нивата на достъп до съдържанието на уеб сайтове и информационни системи, както и за разширяване на потребителски заявки за търсене в Интернет. С тяхна помощ могат да бъдат проектирани интелигентни интерфейси, които базирайки се на заложените правила, персонализират диалога с потребителя на основата на прости изводи от наличните ограничения върху определени свойства.

Представените в тази част подходи конкретизират характерни приложения на онтологиите в уеб базирани системи.

В [Moreno et al., 2012] се използва предметна онтология, описваща типове туристически дейности, с което се предоставя възможност за извод на съответствия между характеристиките на определен тип дейност и конкретни потребителски предпочитания. Концептите в използваната предметна онтология и техните взаимовръзки са определени от експерти в областта на туризма. **Онтологията съдържа класове**, чрез които са описани типове дейности, **а не съдържа екземпляри**, представящи самите дейности. Дейностите се съхраняват в отделна GIS - база данни, тъй като могат да се променят.

В [Gaeta et al., 2009] е представен подход за управление на жизнения цикъл на предметни онтологии, използван за дефиниране на персонализиран подход за електронно обучение. Целта на описания метод е подпомагане на общност от експерти в моделирането на области от образованието. Поставен е акцент върху набор от инструменти за съвместно изграждане и редактиране на онтологии, които могат да бъдат използвани от експертите без конкретен опит в инженеринга на знания.

В [Castellanos-Nieves et al., 2011] е анализиран прецизно подход за подпомагане на оценяването на отворени въпроси, който съчетава предметни онтологии, семантични анотации и семантични измервания за сходство. Този подход включва алгоритъм за извличане на знания от отговорите на обучаемите. Фазата на подготовка в алгоритъма е реализирана чрез стандартни NLP техники. Във фазата на търсене се прилагат *стеминг алгоритми* за разпознаване на кандидат – лингвистични изрази. Стеминг алгоритмите се използват за откриване на думи в база от знания, които са подобни на неанализираните думи в текста. Освен в този конкретен случай за испански език, стеминг алгоритмите се използват и за други езици като например английски. За процеса на извличане на информация от текст на български език, стеминг алгоритмите не са решение, поради флексивността на езика. По тази причина, в предложения в Глава 6 от дисертационния труд подход за извличане на понятия от

неструктуриран текст на български език, е използван алгоритъм, известен като *лематизация* за определяне на основните форми на думите.

1.5 Изводи

В резултат на извършените анализи в първа глава са предствени:

1. Тенденции в развитието на уеб технологиите;
2. Технологии, свързани със семантичния уеб, както и текущо състояние и стандартизация;
3. Класификации на онтологиите и среди за проектиране;
4. Езици за представяне на знания в семантичния уеб;

За целите на дисертационното изследване са проучени конкретни приложения на онтологиите като компонент на семантичните уеб технологии и по-специално в областта на управлението на знания и извличането на информация.

В резултат на извършеното проучване, може да се направи следният извод:

Използването на онтологии за моделиране на знания от специфични области осигурява интеграция на информацията, постъпваща от различни източници, подпомага съвместната работа в общности от експерти и е от значение за процеса на извличане на информация. Резултатът от процеса на извличане на знания от текст е свързан с ефективността на NLP-техники, използвани за предварителна обработка на съдържанието. Прилагането на тези техники е необходимо да бъде съобразено със спецификата на конкретния език. В процеса на извличане на информация от текст на български език е необходимо предварителната обработка на съдържанието да включва алгоритъм, изпълнението на който да осигури разпознаване на лингвистичните изрази, въпреки флексивността на езика.

2. Извличане на информация и семантичен уеб

Както беше изяснено, концепцията на семантичния уеб е насочена към добавяне на семантика (метаданни) към съществуващото уеб съдържание. Реализацията на тази визия осигурява по-ефективно управление и достъп на приложения до точно структурирана информация, която могат да обработват и интерпретират.

Като се има предвид броя на съществуващите уеб страници, както и тяхната хетерогенност по отношение на съдържанието, единственият възможен подход за описание на ресурсите с метаданни е чрез използване на автоматизирани техники.

Information Extraction (IE), като форма на анализ на естествен език, е процес за извличане на структурирана информация от неструктурирани документи. Тази информация може да се използва непосредствено за

визуализация от потребителите, да се съхранява в база от данни за последващ анализ или да се използва от системи за търсене и извличане на информация (Information Retrieval, IR).

За да се разграничи IE от IR, ще бъдат направени следните уточнения:

- Предмет на системите за IR е извличането на релевантни документи чрез заявка за търсене в бази от данни и интернет.
- Приложенията за IE анализират текст и предоставят недвусмислена информация според зададени критерии.

В сравнение с IR, IE е по-труден процес, защото е свързан с интензивно използване на знания, които са обвързани в различна степен с конкретни области и сценарии. Въпреки това, еднозначността на резултатите от IE осигурява възможност за относително ясното им представяне на повече от един език, в сравнение с възможностите на IR за превод на многоезични текстове.

За да се насърчи развитието на системите за IE и да се създаде обща база за оценка на тяхната ефективност, се установяват няколко проекта, от които най-значими са:

- *Message Understanding Conference* (MUC) – провеждана в периода от 1987 до 1998 г.
- *Automatic Content Extraction* (ACE) – приемник на MUC с първо пилотно проучване през 1999 г.

Бързото развитие на областта на IE е основно повлияно от MUC. Провежданите от MUC конференции координират няколко изследователски групи и държавни агенции с цел да подобрят технологиите за IE и IR. Като резултат от тези конференции са определени няколко основни вида задачи за IE, които възникват в реалните приложения и илюстрират основните функционални възможности на съвременните системи за IE.

2.1 Класификация на задачите за извличане на информация

Задачите на IE, определени в MUC са фокусирани върху извличане на информация от новинарски статии (например по отношение на терористични прояви). Според MUC – 7 (1998г.) основните задачи, изпълнявани от системите за IE се класифицират по следния начин [MUC]:

- Named Entity recognition (NE) – установява субектите в текста;
- Coreference Resolution (CO) – установява идентичност между субекти в текста;
- Template Element construction (TE) – установява атрибути на субектите;
- Template Relation construction (TR) – установява релации между субектите;
- Scenario Template construction (ST) – установява събитията, в които субектите участват.

Качеството на резултатите, получени от изпълнението на всяка задача, зависи силно от качеството на резултатите, получени от предходно изпълняваните задачи.

В следващата част ще бъдат конкретизирани описанието на задачите, което се основава на [Cunningham 2004]:

2.1.1 Named Entity recognition (NE)

С фазата на *Named Entity recognition* системата за IE идентифицира всички собствени имена, съдържащи се в текста, като например имена на лица, организации и места. Разпознаването на именувани единици (NE) идентифицира също дати, часове, валута и др.

Както е отбелязано в [Cunningham 2004], *Named Entity recognition* е най-надеждната технология за IE. Тази задача е „*слабо*“ зависима от тематичната област и ефективността на постигнатите резултати (F-Measure [2]) може да достигне 95%.

2.1.2 Coreference resolution (CO)

С тази задача системата за IE идентифицира еквивалентност между субекти в текста, които са били разпознати в предходната фаза (NE).

Изпълнението на CO е от голямо значение за следващите фази (TE, TR и ST), с които се цели придобиване (натрупване) на субектите с атрибути и релации. Тази технология може да се използва например за създаване на връзки между документи.

CO е неточен процес и постигнатите резултати се различават значително в зависимост от тематичната област. Системите за IE на MUC – 7 постигат сравнително ниска ефективност на резултатите (F-Measure): 62%.

2.1.3 Template Element construction (TE)

TE надгражда NE и CO, асоциирайки описателна информация (атрибути) със субекти. Атрибутите за даден субект са дефинирани в зависимост от тематичната област и с изпълнението на тази фаза се извличат и групират в шаблонови елементи (*template element*).

Ефективността на постигнатите от MUC – 7 резултати при изпълнението на тази задача е 87% (F-Measure).

2.1.4 Template Relation construction (TR)

Целта на тази фаза е установяване на съществуващи отношения между шаблонни елементи, които са извлечени от предходната фаза.

Извличането на отношения между субекти е централен елемент в почти всяка задача за IE. Изпълнението на тази задача слабо зависи от тематичната област и ефективността на постигнати резултати е около 75%.

[2] F-measure е пояснена в част 5.5.

2.1.5 Scenario Template production (ST)

Целта на тази фаза е да установи събития, в които участват шаблонните елементи. В резултат от изпълнението на тази задача се установяват TE и TR, които съответстват на определени сценарии на събития.

ST е сложна задача и ефективността на изпълнението й зависи от тематичната област и от ефективността на резултатите, получени от NE, TR и TE. Например ефективността на резултатите, постигната от MUC – 7, е около 60%.

Automatic Content Extraction (ACE), продължава идеята на MUC за формулиране на обща база на системите за IE, като се отличава със следните съществени разлики [ACE]:

1. Част от задачите, дефинирани от MUC, се смесват в ACE. Задачите за *NE* и *CO* се обединяват в една обща задача - *Entity Detection and Tracking (EDT)*, както и задачите за *TE* и *TR* се обединяват в *Relation Detection and Characterization (RDC)*. Задачата за *ST* се преименува на *Event Detection and Characterisation (EDC).*
2. Задачите в ACE са по-сложни в сравнение с дефинираните в MUC. Например: таксономията на субектите в задачата за *EDT* е по-прецизна и е необходимо системите за IE да интерпретират метонимично срещаните субекти, което изисква семантичен анализ на текста.
3. Резултатите от ефективността на ACE не са публично достъпни, а са ограничени до участниците. По този начин, ползата от програмата (за неучастващите в нея) намалява.

В контекста на семантичния уеб, основните изисквания към системите за IE се отнасят до въпросите за преносимост на задача и тематичната област. Според [Hendler 2001] реализацията на семантичния уеб ще се основава на голям брой малки онтологични компоненти, вместо няколко големи и сложни онтологии, споделяни от голям брой потребители. Такива компоненти ще бъдат постоянно създавани, разширявани и обединявани. Следователно услугите (*anotation services*), свързани с тях, ще трябва да бъдат преработвани в съответствие с промените. Това поставя редица ограничения и изисквания към технологията за анотиране по отношение на използваемост, преносимост и възможност за поддръжка.

Системите за IE могат в известна степен да се адаптират към нова задача и тематичната област чрез техники за машинно обучение (*machine learning*). Въпреки постигнатите резултати (чрез използване на машинно обучение), преносимостта на системите за IE към нови тематични области все още остава отворен въпрос. Интеграцията на онтологии в процеса на IE осигурява възможност за постигане на преносимост и ефективност на изпълняваните задачи.

2.2 Онтологично базирано извличане на информация

Онтологично базираното извличане на информация (*Ontology-based information extraction*, OBIE) се очертава като подобласт на IE. Решаваща роля в този процес имат онтологиите, които осигуряват формални и експлицитни спецификации на концептуализации. Поради използването на онтологии, тази област е свързана с представяне на знания и има потенциал да подпомогне развитието на семантичният уеб.

OBIE поставя следните основни предизвикателства:

- Идентифициране на понятия от дадена онтология в текст;
- Автоматично попълване на дадена онтология с екземпляри, които са разпознати в текст.

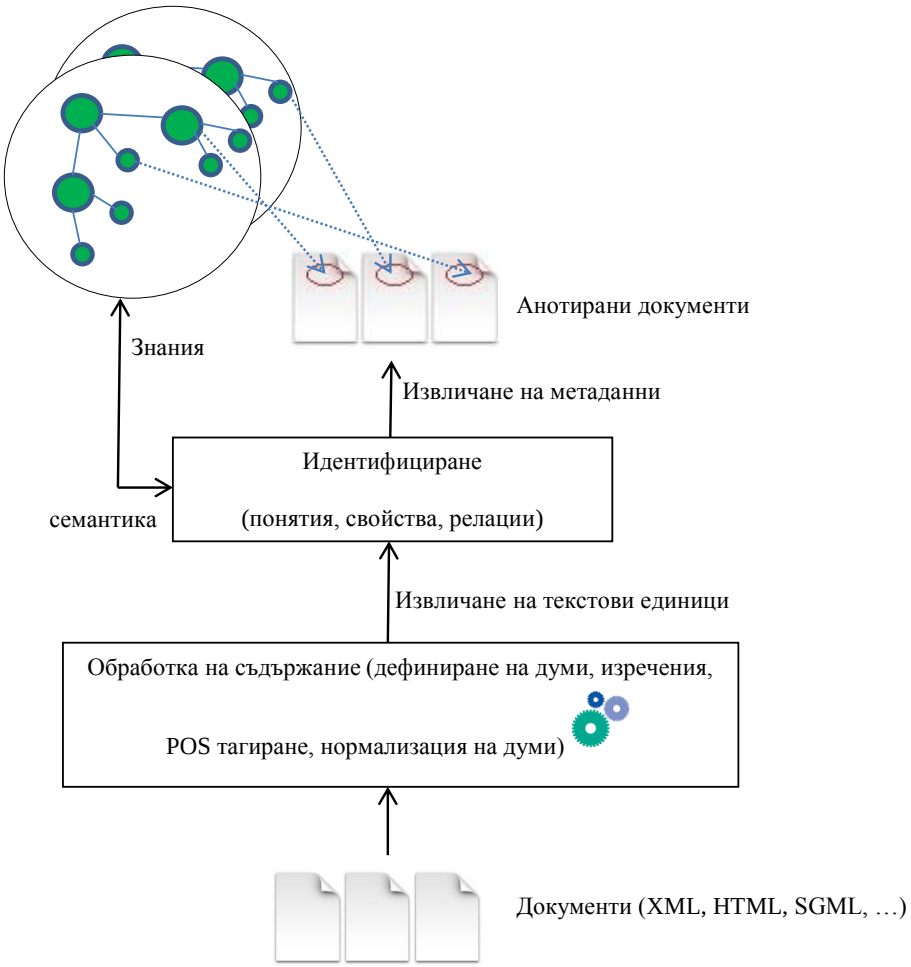
Успешната реализация на първото предизвикателство, а именно идентифициране на понятия от дадена онтология в неструктуриран текст, естествено води до предоставяне на възможност за автоматично попълване на използваната онтология с екземпляри, които са разпознати в текста.

За автоматичното попълване на дадена онтология, целта на OBIE е да идентифицира екземпляри в текста, принадлежащи на понятия в онтологията, и да добави тези екземпляри на правилното място. Ако използваната онтология вече е попълнена с екземпляри, задачата на системата за OBIE може да се преформулира като: идентифициране на екземпляри от онтологията в текста.

Автоматичното попълване на онтология се извършва като разновидност на OBIE, докато основното извличане на информация се реализира чрез предварителна лингвистична преработка, последвана от компоненти за разпознаване, като: списък (gazetteer); граматика, базирана на правила; техники за машинно обучение.

На Фигура 2-1 е визуализирана обща схема за онтологично базирано извличане на анотации от неструктуриран текст. Процесът на идентифициране при OBIE, както и при традиционните системи за IE, е неизменно свързан с NLP техники за обработка на текстовото съдържание. По тази причина, предмет на системите за IE са лингвистичните и семантичните ресурси.

Лингвистичните ресурси (*tokenizers, part of speech taggers*) в системите за IE са свързани с обработка на текстово съдържание и обикновено се изпълняват в т.нар. подготвителна фаза. След тази фаза може да се изпълни фаза на търсене или идентификация, която се реализира чрез семантични ресурси (*gazetteers, ontologies*). Подготвителната фаза обикновено включва следните етапи: разделяне на текста на изречения; определяне на потенциалните думи; задаване на граматическа категория за част на речта на всяка потенциална дума. Първите два етапа могат да се реализират от езиково независими лингвистични ресурси. Последният етап е специфичен за всеки език и инструментите, с които се реализира (*part of speech taggers*), използват морфологичен модел на съответния език. Морфологичният модел се базира на анотационна схема, изготвена от специалисти в областта.



Фигура 2-1. Извличане на онтологично базирани анотации

За реализиране на целта на настоящия дисертационен труд е предложен модел, който предоставя възможност за онтологично извличане на информация от неструктуриран текст на български език. Важно е да се уточни, че предложеният **модел** е фокусиран върху алгоритъм за обработка на съдържанието, **съобразен със спецификата на българския език**, с който да се осигури възможност за изпълнение на следващия етап, а именно идентифициране на онтологични понятия.

2.3 Изводи

В резултат на извършените анализи във втора глава са предствени:

1. Основните задачи, изпълнявани от системите за IE, степента им на зависимост от тематичната област и ефективността на постигнатите резултати за всяка от тях;

2. Обща схема за онтологично базирано извличане на информация от неструктуриран текст;
3. Основни предизвикателства за онтологично базираното извличане на информация;

Като заключение може да се направи следният извод:

В контекста на семантичния уеб интеграцията на онтологии в процеса на IE е ключов аспект за постигане на преносимост и ефективност на изпълняваните задачи. Настоящото дисертационно изследване е фокусирано върху фазите за предварителна обработка на съдържанието и идентификация от общата схема за OBIE, представена на Фигура 2-1. Специфичен и езиково зависим етап от подготвителната фаза е определяне на граматическа категория за част на речта на всяка потенциална дума. Инструментите, реализиращи този етап използват морфологичен модел на съответния език, който се базира на анотационна схема, изготвена от специалисти в областта. Следователно основен момент при проучването на платформи с отворен код за анализ на текст, реализиращи дейности по извличане на информация, е възможността за POS-тагиране на български език и използване на семантични ресурси.

3. Лингвистични технологии, свързани със семантичния уеб

Както вече е уточнено, специфичен и езиково зависим етап от анализа на текст е определяне на граматична категория на думите за част на речта.

Платформите за анализ на текст предоставят инструменти (т.нар. POS тагери), които маркират думите в текста с респективните им категории за част на речта, като за целта прилагат морфологичен модел. Морфологичният модел за всеки език е съобразен с анотационна схема, създадена от езикови специалисти.

За да се осигури възможност за анализ на текстове на различни езици, платформите с отворен код предлагат т.нар. *езиково независими „тагери“*. Езиково независимите тагери позволяват да се зададе морфологичен модел, който да се прилага в обработката на текста. Пример за такива тагери, които включват и модел за български език, са *TreeTagger* и *LingPipe* [TreeTagger] [LingPipe]. И в двата случая моделът за работа с български език е базиран на анотационната схема, разработена от BulTreeBank [3].

Съобразно този аргумент, в тази глава се разглеждат платформи за анализ на текст, които биха могли да се използват при определяне на граматична категория на думите в текст на български език.

[3] Анотационната схема на BulTreeBank е пояснена в 4.3.

3.1 **GATE**

General Architecture for Text Engineering (GATE) е платформа с отворен код, разработвана от 1995 г. от екипа за Natural Language Processing (NLP) към Университета Шефилд, Великобритания. Към настоящия момент GATE е най-популярната платформа за анализ на текст и глобален лидер в NLP инфраструктурата, използвана от хиляди потребители за различни проекти, включително извличане на информация (IE) на различни езици [GATE].

Компонентите в архитектурата са разделени на типове, известни като *ресурси*, като всеки компонент е със специфично предназначение и представлява софтуерна единица за многократна употреба. Основните типове ресурси са [Tablan et al., 2002]:

- Езикови ресурси (Language Resources - LRs) – съхраняват и предоставят достъп до лингвистични данни (документи, корпуси, онтологии).
- Средства за обработка (Processing Resources - PRs) - използват се за обработка на данните, предоставени от един или повече LRs.
- Визуални ресурси (VRs) - графични компоненти, които позволяват визуализация и редактиране на езикови ресурси и средства за обработка.

Основното предимство на GATE е, че предоставя възможност за добавяне на нови компоненти към стандартните в платформата. По този начин могат да разширяват типовете анализ и обработка на текст.

Приложенията (*applications*) в GATE представляват набор от средства за обработка, които се изпълняват в определена последователност. Най-често използваните типове приложения са: *Pipeline*, което като цяло изпълнява зададена последователност от средства за обработка върху документ, и *Corpus Pipeline*, което изпълнява зададената последователност върху корпус от документи.

Основната цел на GATE е анотиране на документи. Анотацията може да се извършва изцяло автоматично (като се използват приложения), ръчно (от потребителя), или полуавтоматично (чрез стартиране на приложение върху корпус и след това ръчно коригиране/добавяне на нови анотации).

GATE поддържа документи в различни формати, включително XML, RTF, EML, HTML, SGML и обикновен текст. Във всички случаи, форматът се анализира и се превръща в един унифициран модел на анотация, които се характеризира с *тип* и набор от *характеристики*, представени като двойки атрибут-стойност. Анотациите се съхраняват в структури, наречени *annotation sets* [Cunningham et al., 2002]. Предимството на единното представяне (т.е. анотациите) е, че NLP приложенията не е необходимо да бъдат адаптирани за различните формати на документите.

Наборът от ресурси, интегриран в GATE, е известен като CREOLE (*Collection of REusable Objects for Language Engineering*). CREOLE съдържа езикови ресурси и средства за обработка, като част от тях са езиково независими

или могат да бъдат конфигурирани за работа и с други езици (освен за английски).

В CREOLE са предоставени и специализирани средства за обработка на следните езици: френски, немски, италиански, румънски, арабски, китайски, хинди, руски и български. Част от ресурсите са „*базови*“ и могат да послужат като основа при разработване на приложение.

Към настоящия момент, за анализ на текст на български език могат да се използват част от стандартните средства на платформата, с които:

- Да се определят потенциалните думи;
- Да се определят границите на изреченията;
- Да се определят граматични категории на думите за част на речта;
- Да се изпълни алгоритъм за стеминг на думите;

Важно е се поясни, че GATE предоставя:

- *LingPipe POS Tagger PR,* за който са осигурени два модела за български език в платформата;
- *TreeTager PR,* за който не е осигурен модел за български език, но може да се инсталира допълнително.

3.2 **UIMA**

Unstructured Information Management Architecture (UIMA) е платформа за анализ на неструктурирано съдържание, първоначално разработена от IBM (International Business Machines Corporation), но към момента се поддържа от Apache Software Foundation. UIMA е сходна с GATE по отношение на представяне на документите (като текст плюс анотации) и предоставяне на възможност за потребителите да дефинират конвейери (*pipelines*) от анализатори, които манипулират документа [UIMA].

UIMA позволява създаваните приложения да използват отделни компоненти, като всеки компонент има определено специфично предназначение, а платформата управлява потока на данни между отделните компоненти.

Платформата UIMA предоставя среда на изпълнение, в която потребителите могат да разработват и използват собствени компоненти, които заедно с други компоненти да включват в приложения.

Алгоритмите за анализ на документи са реализирани като отделни компоненти, които се наричат *анотатори* (*Annotators*).

Анотаторът за маркиране на думите по части на речта в UIMA (Hidden Markov Model Tagger, HMM) включва два морфологични модела – за английски и немски език. Други модели за предварително анотирани корпуси могат да бъдат обучавани извън средата на UIMA.

3.3 **OpenNLP**

Apache OpenNLP поддържа най-често срещаните NLP задачи, като токънизация, сегментация на изречения, POS тагиране и др., които са основни етапи при обработката на текст за изграждане на по-сложни услуги.

Целта на проекта OpenNLP е да се създаде актуален инструментариум за посочените по-горе задачи. Допълнителна цел е предоставяне на голям брой предварително изградени модели за различни езици, както и анотирани текстови ресурси, които са в основата на тези модели [OpenNLP].

Apache OpenNLP съдържа няколко компонента, които позволяват да се изгради конвейер за обработка на естествен език. Тези компоненти позволяват да се изпълни съответната задача за обработка на естествен език, да се обучава модел и често да се направи оценка на този модел. Всеки един от тези компоненти е достъпен чрез приложно-програмен интерфейс (API), като в допълнение се предоставя и интерфейс с команден ред.

За OpenNLP са достъпни модели за английски, немски, испански, холандски и др. езици, но не е наличен модел за български език.

Съществуват и други платформи с отворен код, които предоставят в различна степен възможности за анализ на текст и извличане на съдържание. Предствените в тази глава платформи са едни от най-често използваните, като основен момент в направеното проучване е възможността за анализ на текст на български език.

3.4 **Изводи**

В резултат на извършеното проучване за платформи с отворен код за анализ на текст, могат да се направят следните изводи:

1. Успехът на платформи, поддържащи анализ на текст и извличане на съдържание, като GATE и UIMA, демонстрират значителния интерес за управление на неструктурирана информация.
2. В настоящия дисертационен труд за анализ на текст на български език и извличане на съдържание е подходяща платформата GATE, поради следните основни причини:
   - GATE е безплатна софтуерна платформа с отворен код;
   - GATE е безспорно най-развитата платформа;
   - GATE е активно разработвана и се използва от голяма част от потребителите;
   - GATE предлага богат набор от средства за обработка на текст, които подпомагат извличането на съдържание;
   - В GATE са осигурени два морфологични модела за определяне на думите като част на речта за текст на български език.

4. Подход за автоматично маркиране на неправилно съгласуване на прилагателно име със съществително име в български текст

В тази част авторът представя подход за автоматично маркиране на неправилно съгласуване на прилагателно име със съществително име в български текст.

За реализация на предложения подход е използвана възможността за прилагане на регулярни изрази върху анотации в GATE с цел откриване на неправилно съгласуване на прости фрази, образувани от две думи – атрибутивно прилагателно и съществително, като атрибутивното прилагателно предхожда съществителното име.

Прилагателните имена в българския език, в ролята на съгласувано определение, приспособяват граматическите си показатели за род и число към граматическите показатели за род и число на определяемото. Например във фразата "*щастливо деца*" прилагателното име и съществителното име не са съгласувани по число (прилагателното "*щастливо*" е в единствено число, а съществителното "*деца*" – в множествено). Аналогичен пример е фразата "*щастлив дете*", в която прилагателното и съществителното не са съгласувани по род (прилагателното "*щастлив*" е в мъжки род, докато съществителното "*дете*" е среден род). Такъв тип грешки се среща в автоматичния превод и често се допуска от изучаващите български език като чужд език. В този случай средствата за автоматично откриване на определени типове грешки могат да бъдат полезни.

Необходимо условие за изпълнението на задачи, свързани с обработката на естествен език, е разпознаването и маркирането на „*единици*“ в текста. Разделянето на текста на единиците, които го съставят, се извършва чрез идентифициране на границите на изреченията, определяне на потенциалните думи и тяхната граматична категория [Коева 2008].

Като се има предвид спецификата на GATE, задачата за автоматично маркиране на неправилно съгласуване на прилагателно име със съществително име в български текст може да бъде разделена на следните свързани подзадачи:

1. Сегментация на текста на ниво изречения;
2. Определяне на потенциалните думи в текста (токъни);
3. Анотиране на отделните токъни с тяхната респективна категория за част на речта (POS);
4. Маркиране на двойките прилагателно-съществително, които не са съгласувани по род и число.

Първите три подзадачи са стандартни NLP задачи, за които са предоставени средства за обработка в GATE. За изпълнението на всяка от тях са използвани последователно ресурси, налични в плъгин *LingPipe*, а именно: *LingPipe Tokenizer PR, LingPipe Sentence Splitter PR* и *LingPipe POS Tagger PR*. Нашият избор за използване на *LingPipe* се основава на факта, че в момента това

е единственият плъгин в дистрибуцията на GATE, с POS тагер, който включва модели за български език.

За последната подзадача е използван *JAPE (Java Annotation Patterns Engine) Transducer* като средство за прилагане на регулярни изрази върху анотации.

В следващата част последователно ще бъде показано изпълнението на всяка от гореспоменатите подзадачи.

4.1 Сегментация на текста на ниво изречения

Идентифицирането на изреченията, съдържащи се в текста, се извършва автоматично от ресурс, наречен *Sentence Splitter*. Резултатът от този модул е необходим за изпълнението на POS тагера. *Sentence Splitter* използва списък от съкращения, за да разграничи препинателните знаци, използвани в края на изреченията от друг тип тяхна употреба. След изпълнението на този ресурс, всяко изречение, съдържащо се в текста, се анотира с тип „*Sentence*“.

4.2 Токънизация

„*Потенциалните*“ думи, съдържащи се в текста, при компютърната обработка на естествен език са известни като „*токъни*“, а процеса за определяне им „*токънизация*“ [Симов 2012].

Токън най-общо се определя като поредица от символи между два отделящи символа - пунктуационен знак, интервал.

В настоящото дисертационно изследване под *токънизация* се има предвид процесът на разделяне на текста на основни езикови единици – думи, числа, пунктуационни знаци и др.

От токънизацията на документ се генерират анотации от тип *Token* и *SpaceToken*, които не се припокриват и обхващат цялото съдържание на документа.

4.3 Анотиране на отделните токъни с тяхната респективна категория за част на речта (POS tagging)

Отправна точка при извличането на информация е определяне на граматичната категория за всяка съдържаща се дума. Алгоритмите за автоматично определяне на думата като част на речта са известни като POS тагери (*part-of-speech taggers*). За изпълнението на POS тагер се изисква предварителна обработка на текста, която се състои в определяне на границите на изреченията и определяне на потенциалните думи в текста. В резултат от изпълнението на POS тагер, всяка дума се анотира с респективна й категория за част на речта, в съответствие с анотационна схема от описателни морфосинтактични маркери (тагсет).

Анотирането с морфосинтактични маркери се извършва в съответствие със зададен морфологичен модел (набор от данни, получени от обучение на тагер с анотиран корпус). В GATE са включени два модела за български език (*bulgarian-full.model* и *bulgarian-simplified.model*), базирани на разработения от BulTreeBank тагсет BTB-TS (BulTreeBank Morphosyntactic Tagset) [Simov & Osenova 2003], [Simov et al. 02], [Simov et al. 04a].

В основата на BTB-TS, по която са анализирани морфологично и синтактично 15 000 изречения в корпуса BulTreeBank, е заложен граматичен модел, представен в [Осенова & Симов, 2007]. Анотационната схема е приложена върху изолирани и свързани изречения от различни жанрове, като преобладават от съвременната художествена литература и периодични издания.

Морфологичният модел *bulgarian-full.model* използва позиционната анотационна схема за десетте части на речта и съответните им граматически характеристики, представена в [Simov et al., 2004b]. Създадените мнемонични имена за граматическите характеристики съдържат буква или цифра за всяка характеристика. С първата главна буква се определя част на речта. След нея следват букви/цифри, които представляват съответстващите й морфологични характеристики. Характеристиките за всяка част на речта са представени в определен позиционен ред. Ако характеристиката е неустановена, съответната позиция съдържа знак тире, за да се запази композицията на представената част на речта. Например: с таг ***A-pi*** се описва прилагателно (***A***djective) име в множествено число (***p***lural), нечленувана форма (***i***ndefinite) и знакът тире замества неустановен род.

Опростеният морфологичен модел *bulgarian-simplified.model* използва съкратени преди всички тирета маркери за подобряване на производителността. Например: *Pca–p, Pca–s-f, Pca–s-m, Pca–s-n*, и *Pce-as-m* са обединени като *Pca*.

LingPipe POS Tagger PR анотира всеки токън с характеристика *category*, която в зависимост от използвания режим на изпълнение съдържа най-подходящия описателен маркер от използваната анотационна схема или списък от най-подходящи описателни маркери със съответните им доверителни оценки (confidence scores).

4.4 Маркиране на двойките прилагателно-съществително, които не са съгласувани по род и число

Последният етап е разпознаване и маркиране на двойките прилагателно – съществително име, които не са съгласувани по род и число.

След изпълнението на POS тагера, съдържащите се в текста токъни са анотирани с характеристика *category*, която съдържа граматичните им характеристики. За маркиране на несъгласуваните по род и число двойки прилагателно – съществително име, е използвано средство за обработка *JAPE-Plus Transducer*, с което се предоставя възможност за прилагане на регулярни изрази върху създадените анотации.

С помощта на регулярни изрази са формулирани JAPE граматически правила. Всяко граматическо правило разпознава и маркира двойки прилагателно – съществително име в текста, които според зададените в правилото критерии не са съгласувани по род и число. Например, в резултат на дефинираното правило *PluralSingularPair,* всяка двойка прилагателно – съществително име в текста, в която прилагателното име е в множествено число и съществителното е в единствено число, се анотира с тип *PSAgrError*.

Чрез *JAPE-Plus Transducer PR* формулираните граматически правила се прилагат върху анотации от тип „*Token*“:

```
Input: Token
```

Всеки токън е анотиран с респективната му част на речта чрез *LingPipe POS* тагер, който използва анотационна схема BTB-TS на BulTreeBank. В резултат от изпълнението на този ресурс за обработка и според спецификацията на BTB-TS тагсет, ако първият знак на POS маркерът е "***A***", съответният токън е прилагателно име, ако тя е "***N***", съответният токън е съществително име. За съществителните имена, четвъртият знак на POS маркера определя съществителното име по число - "***s***" за единствено число и "***p***" за множествено число. Единствено или множествено число за прилагателното име се определя от POS маркера чрез третия му знак.

Следователно за маркиране на анотации от тип "*Token*" с характеристика "*category*", чиято стойност е низ, който започва с главна буква "***A***" (за прилагателно), последвана от произволен символ и малка буква "*p*" (за мн.ч.), използваме следния регулярен израз:

```
{ Token.category =~ "^A.p" }
```

Аналогично, за маркиране на анотации от тип "*Token*" с характеристика "*category*", чиято стойност е низ, който започва с главна буква "***N***" (за съществително име), последвана от произволен символ и малка буква "***s***" (за ед.ч.), използваме:

```
{ Token.category =~ "^N..s" }
```

Обединявайки двата регулярни израза, създаваме правило, което маркира с анотация „*PSAgrError*“ двойките прилагателно-съществително име в текста, в които прилагателното име е в множествено число и съществителното име е в единствено число:

```
Rule: PluralSingularPair
Priority: 20
(
{ Token.category =~ "^A.p" }
{ Token.category =~ "^N..s" }
): pair
-->
:pair.PSAgrError = { rule = "PluralSingularPair" }
```

Аналогично са формулирани всички правила за разпознаване и маркиране на тип грешки при съгласуване на прилагателно със съществително име в текст на български език, приложени в представения подход. Други типове

грешки, допускани в българския език, могат да бъдат автоматично разпознати и маркирани чрез създаване на допълнителни граматически правила.

4.5 Изводи

В тази глава е предложен подход за автоматично маркиране на неправилно съгласуване на прилагателно име със съществително име в текст на български език, като са използвани следните функционални възможности на платформата GATE:

- Сегментация на текста на ниво изречение;
- Определяне на потенциални думи, съдържащи се в текста;
- Анотиране на потенциалните думи (токъни) с респективната им част на речта чрез LingPipe POS тагер, който използва схемата на BulTreeBank тагсет (BTB-TS);
- Възможност за прилагане на регулярни изрази върху анотации.

От реализацията на предлагания подход могат да се направят следните изводи:

1. Многоезичната поддръжка и езиково независими компоненти в платформата осигуряват възможност за разработване на приложения за автоматично откриване и коригиране на широк спектър от типове граматически грешки в текстове на български език.
2. Възможността за прилагане на регулярни изрази върху анотации в GATE предоставя широк набор от техники за решаване на реални проблеми, свързани с компютърната обработка на естествен език.
3. Формулираните граматични правила могат да се използват при внедряване на NLP функции в софтуерни приложения, изпълняващи задачи, като откриване, анотиране и извличане на словосъчетания, които отговарят на определен набор от критерии.

Представеният подход е публикуван в:

Borisova N., G. Iliev, E. Karashtranova. On Detecting Noun-Adjective Agreement Errors in Bulgarian Language Using GATE, Proceedings of the Fifth International Conference of FMNS, FMNS 2013, Blagoevgrad, pp 180-187, 2013

5. Реализиране на инструмент за лематизация на български текст в GATE

Процесът на нормализация на думите е от особено значение за редица задачи, свързани с обработка на естествен език. Нормализирането на обработвания текст също така е задължителна стъпка при формирането на конвейери за извличане на информация от неструктуриран текст. В

компютърната лингвистика при обработка на естествен език с цел автоматично определяне на основните форми на думите се използват алгоритми, изпълнението на които е известно като „*лематизация*“ и „*стеминг*“.

Стемингът използва конкретни характеристики на езика и обикновено се състои от правила за премахване на наставки. При този подход не се получава основната форма на думата, а така наречения „*стем*“.

Лематизацията е автоматизиран процес за определяне на основната форма на думите (лема) за всяка от словоформите в даден текст. Инструментите, реализиращи този процес на нормализация на думите са известни като *лематизатори*. Често, за по-голяма точност при определянето на основната форма, е необходимо да се предостави и граматичната категория, към която принадлежи разглежданата словоформа. Лематизацията е по-точен и по-бавен подход от стеминга, като има следните недостатъци:

- създаденият речник е статичен, за разлика от езика, който се променя постоянно;

- качеството на лематизацията зависи от точността на алгоритъма за определяне частите на речта.

В тази част е представен разработеният от автора речниково базиран лематизатор за български език, който се разпространява със свободен достъп под GPL v3 license. Представеният софтуер е публично достъпен в GitHub [4] като инструмент с отворен код за платформата GATE. Към момента не е известно съществуването на други инструменти с отворен код и свободен достъп, специализирани за лематизация на български текст. Лематизаторът е тестван върху ръчно анотирания корпус BulTreeBank-Morph, съдържащ 273933 токъни [5]. Полученият резултат при откриването на основните форми на думите е с приблизителна точност 95%.

Лематизацията е от особено значение и може да подобри точността на редица други задачи, свързани с обработката на естествен език, като например извличане на информация.

Лематизацията е трудна задача, когато се реализира за силно флективни езици, какъвто е българският език. Например словоформите на прилагателното име „рядък“ в българския език, представени в таблица 5-1, са девет.

За реализацията на BGLemmatizer, като част от BGLangTools, са използвани лексикални ресурси със свободен достъп BG Office [6] и Bulgarian-language Wiktionary [7]. По този начин в BGLangTools се предоставя възможност за автоматично извеждане на списъците с основни форми, предоставяни от тези проекти и автоматично генериране на техните словоформи.

[4] https://github.com/grigoriliev/BGLangTools/releases
[5] http://www.bultreebank.org/btbmorf/
[6] http://bgoffice.sourceforge.net
[7] http://bg.wiktionary.org

таблица 5-1. Словоформи на прилагателното име "рядък"

основна форма	рядък
м.р. ед. ч.	рядък
м.р., ед.ч., непълен член	р**едкия**
м.р., ед.ч., пълен член	р**едкият**
ж.р., ед.ч.	рядк**а**
ж.р., ед.ч., членувано	рядк**ата**
ср.р., ед.ч.	рядк**о**
ср.р., ед.ч., членувано	рядк**ото**
мн.ч.	р**едки**
мн.ч., членувано	р**едките**
разширена форма	р**едки**

Словоформите на лексемите се извеждат с помощта на словообразуващи парадигми (образец, схема на словоизменението). Лексемата представлява множество от думи (словоформи) имащи едно и също основно значение, различаващи се само по род, число, падеж, граматическо време и др. Използвани са описаните 187 парадигми в [Кръстев 1990], които според автора обхващат всички (или почти всички) типове форми на словообразуване на българските изменяеми думи. На практика това означава, че всяка българска изменяема дума може да бъде отнесена към един от описаните 187 типа и по типовия образец може да се образува всяка конкретна словоформа. Всеки тип има свой номер от естествения ред на числата от 1 до 187 и следвайки гореспоменатият пример с прилагателното име „рядък“ в българския език, което се отнася съм тип "83", получаваме:

```
WordEntry[] forms;
forms = BgWordFormGenerator.generateWordForms("рядък",
"83");
```

BG Office и Bulgarian-language Wiktionary са проекти в процес на разработка, което предполага, че предоставяните лексикални ресурси ще претърпят редица допълнения и промени. Поради тази причина BGLangTools предоставя възможност за автоматично импортиране в базата си от данни на новодобавените в тези проекти основни форми. Към настоящия момент речникът съдържа 65 376 основни форми и 1 017 595 словоформи. При последващо доразработване на BGLangTools броят на словоформите в базата от данни ще нарасне, тъй като глаголното словообразуване е разработено частично – към текущата версия на BGLangTools автоматично се генерират само най-често използваните словоформи на глаголите. Това решение се наложи от особеностите на българския език поради факта, че един глагол може да има повече от 2000 словоформи [Кръстев 1990].

Граматичната информация за въведените думи в BGLangTools се съхранява в 32-битово цяло число. Голяма част от наличните към момента POS тагери за български език използват анотационната схема BTB-TS на BulTreeBank за анотиране на думите с POS маркери. За осигуряване на оперативна съвместимост с BTB-TS, в BGLangTools е имплементирана

поддръжка за конвертиране на BTB-TS POS маркера към 32-битово цяло число и обратно.

5.1 Извличане на списък с леми от BG Office project

Целта на проекта BG Office е да предостави възможност за проверка на правопис (spell check) за български език в различни проекти с отворен код, като например OpenOffice (openoffice.org) и Mozilla (mozilla.org). По тази причина, BG Office предоставя софтуерни пакети за разработчици [8], които съдържат лексикални ресурси за български език, в които лемите са сортирани по флективен тип и съхранени в отделни файлове. В последната версия (4.1) на пакета за разработчици, лексикалните ресурси са разположени в под-директории на директорията „*data*“. Лемите се съхраняват в отделни файлове (с разширение *.dat*) според флективния им тип. За обозначаване на флективния тип на съдържащите се леми, имената на файловете започват със символите „*bg*“, последвани от номера на типа и опционален еднобуквен суфикс.

BGLangTools предоставя извличане на лемите от пакета за разработчици на BG Office, като за всяка лема се генерират съответните й словоформи и се добавят в посочения речник:

```
String bgofficePath = "/packages/bgoffice-4.1/";
BgDictionary dict = new BgDictionary();
BgOfficeScanner.getInstance().scan(bgofficePath, dict);
```

Чрез променливата `bgofficePath` се задава абсолютният път до главната директория на BG Office пакета.

5.2 Извличане на списък с леми от Bulgarian-language Wiktionary project

Целта на проекта Wiktionary е да осигури цялостен общодостъпен международен речник. Резервните копия на съдържанието са със свободен достъп [9].

Поддръжката за автоматична обработка в BGLangTools на дъмп файла [10] на Bulgarian-language Wiktionary е реализирана с помощта на софтуерната библиотека Bliki engine [11]. С помощта на следния код могат да се извлекат наличните в посочения дъмп файл основни форми, да се генерират съответстващите им словоформи и да се добавят в посочения речник:

```
String dump;
dump = „/dumps/bgwiktionary-latest-pages-
articles.xml.bz2“;
BgDictionary dict = new BgDictionary();
```

[8] http://sourceforge.net/projects/bgoffice/files/For%20Developers/
[9] http://dumps.wikimedia.org/
[10] http://dumps.wikimedia.org/bgwiktionary/latest/bgwiktionary-latest-pages-articles.xml.bz2
[11] https://code.google.com/p/gwtwiki/

```
BgWiktionaryScanner.getInstance().scan(dump, dict);
```

Чрез променливата `dump` се задава абсолютният път до дъмп файла на Wiktionary.

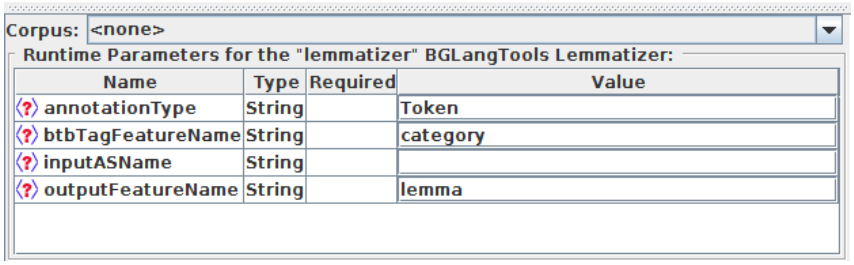
5.3 Използване на вграден речник

Към настоящия момент BGLangTools съхранява във вътрешен формат 1 082 971 токъни, извлечени от проектите BG Office и Bulgarian-language Wiktionary. Зареждането на съхранените токъни с тяхната граматическа информация се изпълнява чрез:

```
BgDictionary dict = BGLangTools.loadBuiltinDictionary();
```

5.4 Използване на BGLemmatizer като инструмент в платформата GATE

BGLemmatizer е реализиран като инструмент за лематизация на български текст в платформата GATE. За изпълнението на BGLemmatizer като средство за обработка се изисква POS маркер за всеки токън, който се получава автоматично от POS тагер. Към момента тагирането на български текст с POS маркери в платформата GATE се изпълнява от LingPipe POS tagger. На Фигура 5-1 е представена примерна конфигурация за BGLemmatizer, където POS маркерът за всеки токън се извлича от характеристиката „*category*“ на антотация от тип „*Token*“ и получената лема се съхранява в характеристика „*lemma*“ за същия тип анотация.



Фигура 5-1. Примерни параметри за изпълнение на BGLemmatizer като инструмент в платформата GATE

5.5 Изследване на ефективността на BGLemmatizer като средство за обработка на български текст

Резултатите от анализа на ефективността на разработения софтуер за автоматична лематизация на български текст (BGLemmatizer) са представени в [Karashtranova et al., 2015]. За определяне качествата на софтуера са използвани

следните статистически методи: извадково оценяване и интервални оценки. Резултатите от анализа на лематизацията показват 95% точност.

Двата основни фактора, които влияят върху резултатите при представително статистическо изследване, са: методът [12], по който се формира извадката, и обемът на извадката [13].

Поради спецификата на изследвания корпус е избран двустъпков модел, който е комбинация между районирана и гнездова извадка.

Районираната извадка (*Stratified*) изисква преди да се извърши изборът на единиците, те да се групират в генералната съвкупност (райони). За целта се използват категориите на един или повече признаци. Извадката се формира, като броят на извлечените единици от всеки район е пропорционален на относителния дял на единиците в района спрямо обема на генералната съвкупност.

При *гнездовата извадка* (*Cluster*) вместо директно формиране на извадката от единиците на съвкупността, най-напред се прави случайна извадка от т.нар. *гнезда* или единици от по висок ред. След това се изследват или всички единици от избраните по случаен начин гнезда, или част от единиците, избрани отново случайно.

Тъй като в конкретния случай се изследва ефективността на софтуер за автоматична лематизация на български текст (BGLemmatizer), с най-висок приоритет е да се оцени неговата ефективност по отношение на най-често срещаните части на речта, а именно: съществителни имена, прилагателни имена и глаголи. Тези части на речта разглеждаме като естествено групиране на данните от генерална съвкупност с обем 149 061 единици.

В конкретното изследване са подходящи традиционните показатели за оценка на ефективността на програми за автоматично извличане на информация ([van Rijsbergen 1979], [Maynard et al., 2006]), тъй като са оценки от тип относителен дял и позволяват прилагането на методи за интервално оценяване.

Успешността на даден метод най-често се оценява с показателите *Точност* (*Precision*) и *Пълнота* (*Recall*).

Precision е отношението на броя вярно идентифицирани единици към общия брой идентифицирани единици. С други думи, *Precision* измерва колко от идентифицираните единици са действително верни, базирайки се на постигнатите резултати, независимо от броя на неидентифицираните единици:

$$Precision = \frac{Correct + 1/2Partial}{Correct + Spurious + Partial}$$

Recall е отношението на броя вярно идентифицирани единици към общия брой единици, които са достъпни в текста. С други думи, *Recall* измерва колко от единиците, които е трябвало да бъдат идентифицирани, са всъщност верни:

[12] Правила и изисквания при формиране на извадката при представителни статистически изследвания.

[13] Определяне на необходимия обем на извадката с оглед получаване на оценки с желана стохастична грешка.

$$Recall = \frac{Correct + 1/2Partial}{Correct + Missing + Partial}$$

Комбинираният показател *F-measure* се определя със следната формула:

$$F - measure = (1 + \beta^2)\frac{P.R}{\beta^2.P + R}$$

където P и R са съответно точност и пълнота, а β>0 е параметър, който показва желаното от нас отношение (тегло) между двата показателя.

Използвахме модифициран вариант, представен в [Стоянова 2012], в който не се включват частично разпознатите лексикални елементи. В този случай точността P и пълнотата R се изчисляват със следните формули:

$$P = \frac{N_{\text{верни}}}{N_{\text{всички\_разпознати}}}$$

$$R = \frac{N_{\text{верни}}}{N_{\text{всички}}},$$

където: $N_{всички}$ е броят на лексикалните единици в корпуса, $N_{всички\_разпознати}$ е броят на разпознатите единици , а $N_{верни}$ е броят на верните.

Както вече посочихме, за анализ на ефективността на BGLemmatizer прилагаме двустъпкова извадка, в която пилотното изследване има следните предимства:

- резултатите от пилотното изследване могат да се използват за получаване на оценки за дисперсиите в отделните райони;
- резултатите от пилотното изследване могат да послужат за определяне на минималния обем на извадката, необходим за получаване на оценки на популационните параметри с определена степен на точност.

Пилотно изследване е проведено с 40 наблюдения от всеки район, като броят на единиците по райони (части на речта) е както следва:

N_1 = 80509 /за съществителни имена/

N_2 = 23159 /за прилагателни имена/

N_3 = 45393 /за глаголи/

Определянето на обемите на извадките за всеки от трите района се изчислява със следната формула:

$$n_j = \frac{N_j\,\sigma_j}{\sum_{i=1}^{3} N_i\,\sigma_i} \cdot n \qquad (j=1,2,3), \tag{1}$$

където σ_i са стандартните отклонения за всеки район.

В резултат на пилотното изследване са получени следните оценки за стандартни отклонения по райони: σ_1 = 0,243, σ_2 = 0,352, σ_3 = 0,195.

След заместване във формула (1), получаваме следните стойности за обемите на извадките по райони: n_1 = 0,534.n; n_2 = 0,224.n; n_3 = 0,242.n

Общият обем на извадката определяме със следната формула:

$$n = \frac{\frac{1}{N}\left(\sum_{i=1}^{3} N_j\, \sigma_j\right)^2}{n\sigma_{\hat{p}}^2 + \frac{1}{N}\sum_{j=1}^{K} N_j \sigma_j^2}, \quad (2)$$

където: N е общият обем на популацията, а $\sigma_{\hat{p}}^2$ е оценка за популационната дисперсия.

За да се постигне 95% точност на доверителния интервал за популационната пропорция, приемаме размера на максимално допустимата стохастична грешка да е 0,02 (т.е. $\sigma_{\hat{p}} = 0{,}01$).

След заместване във формула (2) получаваме, че за постигане на посочената точност минималният обем на извадката е 597.

Резултатите от изследването са получени на базата на извадка с обем от 1373, което ни гарантира постигане на посочената по-горе точност. Разпределението на обемите на извадките по райони е както следва:

n_1 = 0,534.1373 = 732;

n_2 = 0,224.1373 = 308;

n_3 = 0,242.1373 = 333

Като се има предвид, че за пилотното изследване са направени по 40 наблюдения за всеки от районите, броят на наблюденията по райони, включени във втората фаза, са респективно: 692, 268, 293.

Оценката на извадковата пропорция е получена чрез формулата:

$$\hat{p} = \frac{\sum_1^3 n_i \,.\, p_i}{\sum_1^3 n_i}, \quad (3)$$

където p_i са пропорциите на изследваните параметри (показатели) за всеки район и n_i са съответните извадкови обеми.

За построяване на доверителни интервали с 95% точност, прилагаме следната формула:

$$\hat{p} - 1{,}96.\sigma_{\hat{p}} < p < \hat{p} + 1{,}96.\sigma_{\hat{p}}$$

95 % доверителни интервали за показателите точност ($\hat{p} = 0{,}97$) и пълнота ($\hat{p} = 0{,}93$) са:

$$0{,}97 - 0{,}02 < P < 0{,}97 + 0{,}02$$
$$0{,}93 - 0{,}02 < R < 0{,}93 + 0{,}02$$

Използваме най-често прилаганата форма на *F-measure*, с която се задава еднакво тегло на двата показателя точност и пълнота ($\beta = 0{,}5$):

$$F - measure = \frac{P.R}{0{,}5(P+R)} = 0{,}95$$

За постигане на по-голяма обективност, при проверката на извадковите наблюдения частите на речта са извлечени с контекст.

Получените резултати показват висока ефективност и точност на разработения софтуер за автоматична лематизация на български текст.

Предложеният софтуер предоставя възможности за получаване на допълнителна информация за структурата на частите на речта, включени в корпуса. Приложение 1 съдържа честотното разпределение на изследвания

корпус по части на речта. На таблиците по-долу са представени честотни разпределения за всяка от изследваните части на речта на базата на специфични характеристики.

Таблица 5-2. Честотно разпределение на съществителни имена

BTB-TS таг	Честота
N-msi	18667
N-msh	4918
N-msf	2560
N-mpi	5004
N-mpd	3136
N-mt	1966
N-fsi	15816
N-fsd	6127
N-fpi	4992
N-fpd	1836
N-nsi	7398
N-nsd	4288
N-npi	1992
N-npd	986

Таблица 5-3. Честотно разпределение на прилагателни имена.

BTB-TS таг	Честота
Amsi	3256
Amsh	2062
Amsf	1099
Afsi	3287
Afsd	2785
Ansi	2074
Ansd	1492
A-pi	4172
A-pd	2811

Таблица 5-4. Честотно разпределение на глаголи.

BTB-TS таг	Честота	BTB-TS таг	Честота	BTB-TS таг	Честота
V---f-r1s	1769	V---u-o2p	20	V---cv—sfi	669
V---f-r2s	656	V---u-o3p	22	V---cv—sfd	130
V---f-r3s	15582	V---z–2s	217	V---cv—sni	522
V---f-r1p	1391	V---z---p	158	V---cv—snd	92
V---f-r2p	634	V---cao-smi	1204	V---cv–p-i	1305
V---f-r3p	6711	V---cao-smh	55	V---cv–p-d	327
V---f-t1s	42	V---cao-smf	3	V---car-smi	74
V---f-t2s	4	V---cao-sfi	478	V---car-smh	46
V---f-t3s	831	V---cao-sfd	120	V---car-smf	17
V---f-t1p	11	V---cao-sni	314	V---car-sfi	63
V---f-t2p	7	V---cao-snd	25	V---car-sfd	107
V---f-t3p	259	V---cao-p-i	941	V---car-sni	50
V---u-o1s	53	V---cao-p-d	133	V---car-snd	48
V---u-o2s	3	V---cv—smi	1124	V---car-p-i	165
V---u-o3s	112	V---cv—smh	96	V---car-p-d	162
V---u-o1p	5	V---cv—smf	50		

5.6 **Изводи**

В тази глава е представена реализацията на речниково базиран лематизатор за български език (BGLemmatizer), за която са използвани лексикални ресурси със свободен достъп – BG Office и Bulgarian-language Wiktionary.

За реализацията на речниково базирания лематизатор за български език (BGLemmatizer) са изпълнени следните основни задачи:

1. Предоставено е автоматично извличане на лексикални ресурси (списъци с основни форми на думи, категоризирани по съответни типове) от BG Office и Bulgarian-language Wiktionary;
2. Разработена е и софтуерна библиотека за автоматично генериране на словоформи по зададени основна форма и тип. Представеният алгоритъм е базиран на 187 парадигми за словоизменения в българския език.
3. Осигурена е оперативна съвместимост с анотационната схема BTB-TS на BulTreeBank;
4. Лематизаторът е реализиран и като инструмент за платформата GATE. За по-точна работа на лематизатора е необходимо за всяка словоформа да се предостави и информация за граматическите ѝ свойства.

От реализацията на предложения инструмент за лематизация на български текст могат да се направят следните изводи:

1. С разработения софтуер (BGLemmatizer) се постигнат по-добри резултати и точност при POS анотиране на български текстове с високи нива на граматически грешки.
2. Представените таблици на честотно разпределение могат да бъдат използвани при анализ на грешки и разпознаване на типове грешки, с оглед тяхното премахване. Това ще спомогне за повишаване на точността и ефективността на разработения софтуер при използването му в различни NLP – задачи като извличане на информация.
3. Интеграцията на BGLemmatizer с други лингвистични инструменти е приложима за дейности, свързани с извличане на информация. Разширяването на BGLangTools с приложно-програмни интерфейси (APIs) ще предостави възможности за автоматично генериране на текст.

Публикувани резултати:

- Iliev G., N. Borisova, E. Karashtranova, E., D. Kostadinova, A Publicly Available Cross-Platform Lemmatizer for Bulgarian, Proceedings of the Sixth International Scientific Conference – SWU, FMNS 2015, Blagoevgrad, pp 147-151, 2015

- Karashtranova E., G. Iliev, N. Borisova, Y. Chankova, I. Atanasova, Evaluation of the Accuracy of the BGLemmatizer, Proceedings of the Sixth International Scientific Conference – SWU, FMNS 2015, Blagoevgrad, pp 152-156, 2015

6. Извличане на онтологично базирани понятия от неструктуриран текст на български език

Процесът на извличане на информация (IE) цели да извлече определени типове информация от текст на естествен език чрез автоматична обработка. Например една система за IE може да извлече информация за демографски показатели на страните от набор уеб страници, пренебрегвайки другите видове информация. Онтологично базираното извличане на информация (Ontology based Information Extraction OBIE) се очертава като подобласт на IE, в която решаваща роля имат онтологиите. Поради използването на онтологии, тази област е свързана с представянето на знания и има потенциал да подпомогне развитието на семантичния уеб.

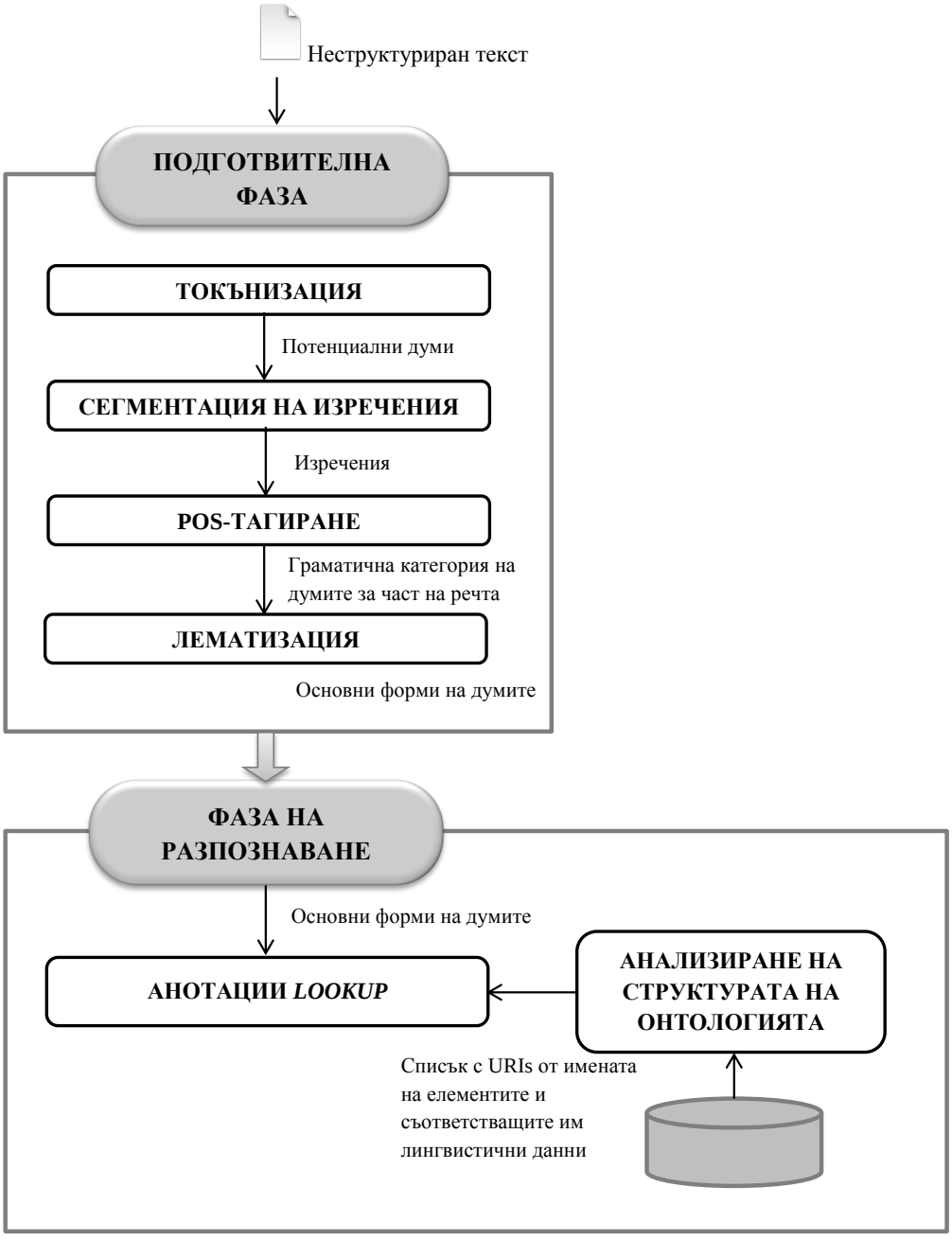
Въпреки разнообразните подходи за извличане на знания от текст, резултатите към момента са ограничени поради семантичната разлика между човешкия естествен език и формалните езици за описание на знания. Под формата на онтология знанието се формализира декларативно и експлицитно. От друга страна, изразено с човешки език е имплицитно и неясно. Освен това, естественият език не е концептуално прецизен, което позволява различни концептуализации да бъдат изразени с едни и същи думи.

В тази част е предложен подход за онтологично базирано извличане на понятия от неструктуриран текст на български език, който може да се използва като основа за автоматично извличане на семантична анотация (метаданни). Автоматичното приписване (assignment) на онтологични класове на текстови елементи е важен аспект на семантичните уеб приложения и услуги.

За реализацията на този подход е използвано приложение *Corpus Pipeline* в платформата GATE, което изпълнява набор от средства за обработка на текст в определена последователност.

Предложеният алгоритъм за откриване на онтологични елементи в неструктуриран текст на български език (Фигура 6-1) съдържа:

1. Подготвителна фаза, която включва следната обработка на текста: токънизация, сегментация на ниво изречение, определяне на граматична категория за всеки токън и лематизация на думите;
2. Фаза на разпознаване, която включва средства за обработка, с които се предоставя възможност за извличане на лингвистични данни от структурата на онтологията и идентифициране на тези данни с резултата от лематизацията на думите в текста.



Фигура 6-1. Алгоритъм за OBIE от неструктуриран текст на български език

В следващата част последователно ще бъде показано изпълнението на всяка от гореспоменатите подзадачи.

6.1 Токънизация

За токънизация на текста е използвано средство за обработка *GATE Unicode tokenizer PR* като решение, независещо от езика. *GATE Unicode tokenizer*

PR разделя текста в документа на прости токъни и предоставя основна информация за типа им.

Токъните се класифицират като цифри, пунктуационни знаци, символи или думи и, в случай на думи, се предвижда информация за тяхната ортография (например с начална главна буква, всички главни букви, всички малки букви и т.н.). С класификацията на токъните се цели постигане на максимална ефективност и по-голяма гъвкавост при използване на граматични правила.

Всяко граматично правило се състои от регулярен израз от лявата страна (LHS) и шаблон за генериране на анотация (RHS):

```
{LHS} > {RHS}
```

Когато моделът, зададен в лявата страна на правилото, съвпада с входни данни, се генерира нов тип анотация според дефинирания в дясната страна на израза шаблон:

```
{LHS} > {Annotation
type};{attribute1}={value1};...;{attribute n}={value n}
```

GATE Unicode tokenizer PR не използва действителните текстови символи, а техните категории, определени от стандарта Unicode [14]. Например граматичното правило за думи, които започват с малка буква и съдържат само букви и пунктуационни знаци (дълги тирета), е следното:

```
"LOWERCASE_LETTER" (LOWERCASE_LETTER|DASH_PUNCTUATION)* >
Token;orth=lowercase;kind=word;
```

След токънизация на изречението „*Масивите, които използват низове като индекси, се наричат асоциативни масиви*.“ за анотациите от тип *Token* се получава следният набор от характеристики (Фигура 6-2):

Type	Set	Start	End	Id	Features
Token		1281	1289	46474	{kind=word, length=8, orth=upperInitial, string=Масивите}
Token		1289	1290	46475	{kind=punctuation, length=1, string=,}
Token		1291	1296	46477	{kind=word, length=5, orth=lowercase, string=които}
Token		1297	1306	46479	{kind=word, length=9, orth=lowercase, string=използват}
Token		1307	1313	46481	{kind=word, length=6, orth=lowercase, string=низове}
Token		1314	1318	46483	{kind=word, length=4, orth=lowercase, string=като}
Token		1319	1326	46485	{kind=word, length=7, orth=lowercase, string=индекси}
Token		1326	1327	46486	{kind=punctuation, length=1, string=,}
Token		1328	1330	46488	{kind=word, length=2, orth=lowercase, string=се}
Token		1331	1338	46490	{kind=word, length=7, orth=lowercase, string=наричат}
Token		1339	1350	46492	{kind=word, length=11, orth=lowercase, string=асоциативни}
Token		1351	1357	46494	{kind=word, length=6, orth=lowercase, string=масиви}
Token		1357	1358	46495	{kind=punctuation, length=1, string=.}

Фигура 6-2. Набор от характеристики за анотации от тип *Token* след токънизация

[14] http://www.fileformat.info/info/unicode/category/index.htm

6.2 Разделяне на текста на изречения

Използвано е средство за обработка *LingPipe sentence splitter PR*, с което се идентифицират границите на всяко изречение и се генерира група анотации от тип "*Sentence*".

6.3 Анотирането на отделните токъни с респективната им част на речта (*POS tagging*)

Анотирането на токъните по части на речта включва въвеждане на граматични характеристики и отстраняване на граматична многозначност. Използвано е средство за обработка *LingPipe POS Tagger PR*, което изисква предварителна обработка на текста (токънизация и сегментация на ниво изречение). *LingPipe* е единственият плъгин с POS тагер в дистрибуцията на GATE, който включва модели за български език.

След POS маркиране на токъните от гореспоменатия пример, към набора от характеристики за анотациите от тип *Token* се добавя характеристиката *category*, чиято стойност съответства на респективната им граматична категория (Фигура 6-3).

Type	Set	Start	End	Id	Features
Token		1281	1289	61471	{category=A-pd, kind=word, length=8, orth=upperInitial, string=Масивите}
Token		1289	1290	61472	{category=PUNCT, kind=punctuation, length=1, string=,}
Token		1291	1296	61474	{category=Pre-op, kind=word, length=5, orth=lowercase, string=които}
Token		1297	1306	61476	{category=Vpitf-r3p, kind=word, length=9, orth=lowercase, string=използват}
Token		1307	1313	61478	{category=Ncmpi, kind=word, length=6, orth=lowercase, string=низове}
Token		1314	1318	61480	{category=R, kind=word, length=4, orth=lowercase, string=като}
Token		1319	1326	61482	{category=Ncfpi, kind=word, length=7, orth=lowercase, string=индекси}
Token		1326	1327	61483	{category=PUNCT, kind=punctuation, length=1, string=,}
Token		1328	1330	61485	{category=Ppxta, kind=word, length=2, orth=lowercase, string=се}
Token		1331	1338	61487	{category=Vpitf-r3p, kind=word, length=7, orth=lowercase, string=наричат}
Token		1339	1350	61489	{category=A-pi, kind=word, length=11, orth=lowercase, string=асоциативни}
Token		1351	1357	61491	{category=Ncmpi, kind=word, length=6, orth=lowercase, string=масиви}
Token		1357	1358	61492	{category=PUNCT, kind=punctuation, length=1, string=.}

Фигура 6-3. Набор от характеристики за анотации от тип *Token* след POS тагиране

Целта на представения подход за OBIE е извличане на термини от текстов документ чрез дефинираните в онтологията концепти. Имената на класовете в онтологията са идентификатори, базирани на URI адреси, и лингвистични данни за концептите могат да се получат от зададените им дефиниции и алтернативни етикети. За да се реализира процесът на маркиране на термините в текстовия документ, съвпадащи с лингвистичните данни за концептите в онтологията, е необходима нормализация на думите.

Към настоящия момент в дистрибуцията на GATE е налично средство за обработка (*BulStem*), извършващо стеминг на български текст [Nakov 2003].

Резултатът от стеминга е така нареченият „стем“, а не основната форма на думата. Например за думата „*асоциативните“* правилото премахва „*ните*” и се получава стем „*асоциатив“*, вместо основната форма „*асоциативен“*.

От текущото състояние на наличните средства за обработка на български текст в GATE възникна необходимостта от алгоритъм за автоматично определяне на основната форма на думите.

За тази цел е интегриран разработеният от автора речниково базиран лематизатор за български език (*BGLangTools Lemmatizer)* като средство за обработка в GATE.

6.4 Лематизация

За автоматично извеждане на основните форми на думите е използван *BGLangTools Lemmatizer*, като основната форма на всяка дума в този подход се извежда в характеристика *root* (Фигура 6-4).

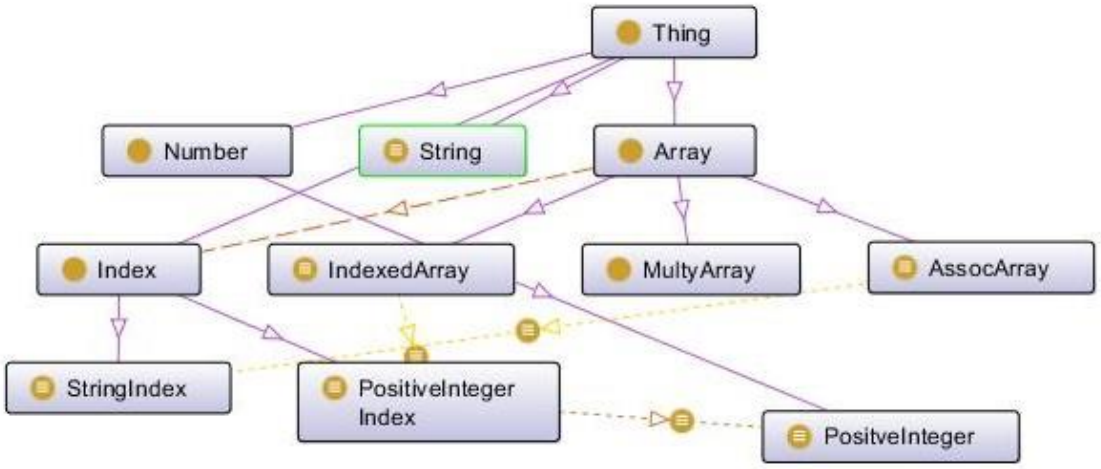
За процеса на лематизацията се изисква допълнителна информация за граматичните категории на токъните, която се предоставя от *LingPipe POS tagger*.

Type	Set	Start	End	Id	Features
Token		1281	1289	76631	{category=A-pd, kind=word, length=8, orth=upperInitial, root=масив, string=Масивите}
Token		1289	1290	76632	{category=PUNCT, kind=punctuation, length=1, root=,, string=,}
Token		1291	1296	76634	{category=Pre-op, kind=word, length=5, orth=lowercase, root=който, string=които}
Token		1297	1306	76636	{category=Vpitf-r3p, kind=word, length=9, orth=lowercase, root=използвам, string=използват}
Token		1307	1313	76638	{category=Ncmpi, kind=word, length=6, orth=lowercase, root=низ, string=низове}
Token		1314	1318	76640	{category=R, kind=word, length=4, orth=lowercase, root=като, string=като}
Token		1319	1326	76642	{category=Ncfpi, kind=word, length=7, orth=lowercase, root=индекс, string=индекси}
Token		1326	1327	76643	{category=PUNCT, kind=punctuation, length=1, root=,, string=,}
Token		1328	1330	76645	{category=Ppxta, kind=word, length=2, orth=lowercase, root=се, string=се}
Token		1331	1338	76647	{category=Vpitf-r3p, kind=word, length=7, orth=lowercase, root=наричам, string=наричат}
Token		1339	1350	76649	{category=A-pi, kind=word, length=11, orth=lowercase, root=асоциативен, string=асоциативни}
Token		1351	1357	76651	{category=Ncmpi, kind=word, length=6, orth=lowercase, root=масив, string=масиви}
Token		1357	1358	76652	{category=PUNCT, kind=punctuation, length=1, root=., string=.}

Фигура 6-4. Набор от характеристики за анотации от тип *Token* след лематизация

BGLangTools Lemmatizer PR е последното средство от подготвителната фаза за обработка на текстовия документ. Резултатът от всички изброени действия е, че данните от документа са анотирани с характеристики, които позволяват последващото им маркиране с лингвистични данни, извлечени от онтология.

Достъп до създадена онтология се предоставя чрез ресурс *OWLIM LR*, като е важно да поясним, че *OWLIM LR* [Kiryakov et al., 2005] поддържа OWL-Lite версия на езика OWL. Йерархията на понятията (класовете) в създадената примерна онтология е визуализирана на Фигура 6-5.



Фигура 6-5. Йерархия на понятията (класовете) в създадената онтология

Фазата на търсене в предложения алгоритъм (Фигура 6-1) включва средства за обработка, известни като газетери или списъци (*gazetteers*).

Газетерът се състои от набор от списъци, съдържащи имена на елементи. Например за разпознаване на именувани единици в текст (*Named Entity recognition*), газетера съдържа списъци с имена на градове, организации и др., които се използват за откриване на съвпадения в текста. „*Газетер*“ често се използва както за набор от списъци на елементи, така и за средство за обработка, което използва списъци за откриване на съвпадения в текст.

В конкретния случай, под „*газетер*“ имаме предвид средство за обработка на документ, което маркира съвпадения на елементи от списъка с текстови елементи, като създават анотации от тип *Lookup*. Газетерите обикновено използват текстовото съдържание на документа (за откриване на съвпадения) и не изискват *Token* или други анотации.

В дистрибуцията на GATE са налични няколко типа газетери, в зависимост от тяхното предназначение. В нашия пример използваме два от тях, а именно *OntoRoot gazetteer PR* и *Flexible Gazetteer PR*.

6.5 Маркиране на термини в текстовия документ, съвпадащи с лингвистичните данни за концептите в онтологията

OntoRoot gazetteer PR е онтологично базиран газетер и е един от най-използваните компоненти за семантично анотиране.

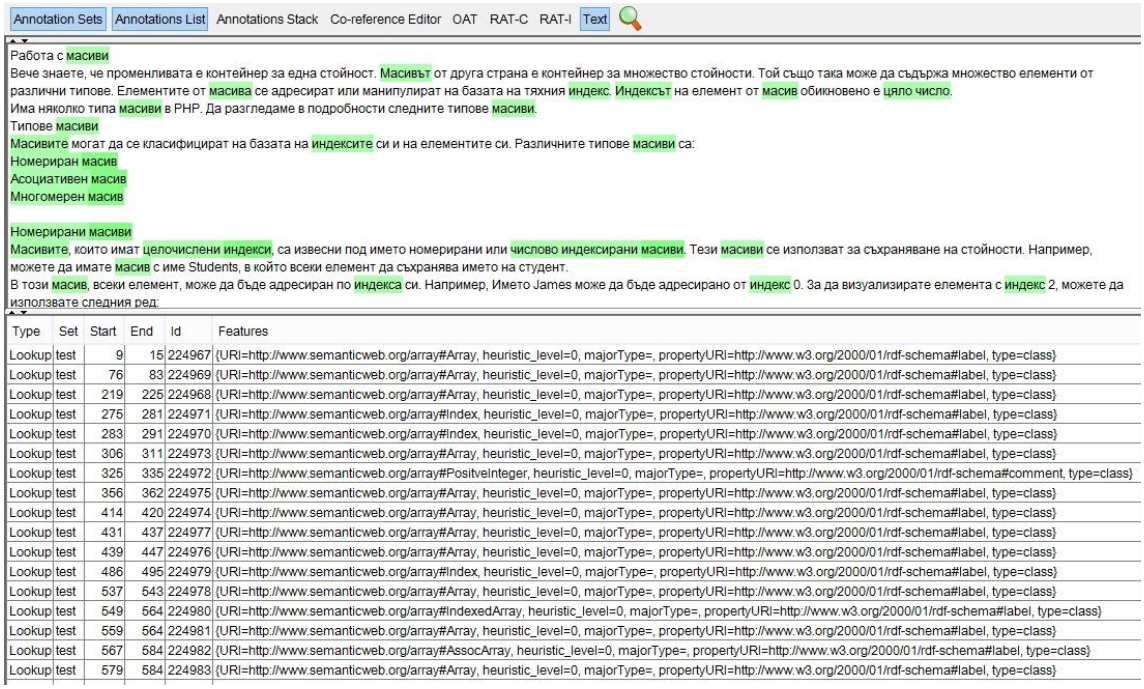
Имената на класовете в онтологията са идентификатори, базирани на URI адреси. Лингвистични данни за концептите могат да се получат от зададените им дефиниции и алтернативни етикети. *OntoRoot gazetteer PR* анализира структурата на онтологията (класове, свойства и екземпляри) и генерира списък с URIs от имената на елементите и съответстващите им лингвистични данни. С други думи, *OntoRoot gazetteer PR* създава анотации тип *Lookup* за открити

съвпадения на езикови елементи от текста с лингвистични данни на елементи от онтологията.

От една страна, лингвистичните данни на елементите в онтологията са въведени в основната им форма (лема). И от друга, използваните в текста термини се срещат в различните им словоформи, поради флексивността на българския език. В този случай *Lookup* анотациите обхващат само тези термини в текста, които са в основна форма и съвпадат с лингвистичните данни за елементи от онтологията. Този недостатък може да бъде преодолян със средство за обработка *Flexible Gazetteer PR.*

Flexible Gazetteer PR предоставя възможност за задаване на входни данни за всеки от наличните типове газетери. По този начин може да се съгласува предназначението на конкретен тип газетер да се изпълни върху зададени характеристики на генерираните анотации.

В представения пример, чрез гъвкавостта на *Flexible Gazetteer PR* задаваме за входни данни на *OntoRoot gazetteer PR* основните форми на думите. Създадените *Lookup* анотации се базират на резултата от предварителната обработка на текста и по-конкретно – на характеристика *root* от анотации тип *Token*, която е получена в резултат от лематизацията на думите (Фигура 6-6).



Type	Set	Start	End	Id	Features
Lookup	test	9	15	224967	{URI=http://www.semanticweb.org/array#Array, heuristic_level=0, majorType=, propertyURI=http://www.w3.org/2000/01/rdf-schema#label, type=class}
Lookup	test	76	83	224969	{URI=http://www.semanticweb.org/array#Array, heuristic_level=0, majorType=, propertyURI=http://www.w3.org/2000/01/rdf-schema#label, type=class}
Lookup	test	219	225	224968	{URI=http://www.semanticweb.org/array#Array, heuristic_level=0, majorType=, propertyURI=http://www.w3.org/2000/01/rdf-schema#label, type=class}
Lookup	test	275	281	224971	{URI=http://www.semanticweb.org/array#Index, heuristic_level=0, majorType=, propertyURI=http://www.w3.org/2000/01/rdf-schema#label, type=class}
Lookup	test	283	291	224970	{URI=http://www.semanticweb.org/array#Index, heuristic_level=0, majorType=, propertyURI=http://www.w3.org/2000/01/rdf-schema#label, type=class}
Lookup	test	306	311	224973	{URI=http://www.semanticweb.org/array#Array, heuristic_level=0, majorType=, propertyURI=http://www.w3.org/2000/01/rdf-schema#label, type=class}
Lookup	test	326	335	224972	{URI=http://www.semanticweb.org/array#PositveInteger, heuristic_level=0, majorType=, propertyURI=http://www.w3.org/2000/01/rdf-schema#comment, type=class}
Lookup	test	356	362	224975	{URI=http://www.semanticweb.org/array#Array, heuristic_level=0, majorType=, propertyURI=http://www.w3.org/2000/01/rdf-schema#label, type=class}
Lookup	test	414	420	224974	{URI=http://www.semanticweb.org/array#Array, heuristic_level=0, majorType=, propertyURI=http://www.w3.org/2000/01/rdf-schema#label, type=class}
Lookup	test	431	437	224977	{URI=http://www.semanticweb.org/array#Array, heuristic_level=0, majorType=, propertyURI=http://www.w3.org/2000/01/rdf-schema#label, type=class}
Lookup	test	439	447	224976	{URI=http://www.semanticweb.org/array#Array, heuristic_level=0, majorType=, propertyURI=http://www.w3.org/2000/01/rdf-schema#label, type=class}
Lookup	test	486	495	224979	{URI=http://www.semanticweb.org/array#Index, heuristic_level=0, majorType=, propertyURI=http://www.w3.org/2000/01/rdf-schema#label, type=class}
Lookup	test	537	543	224978	{URI=http://www.semanticweb.org/array#Array, heuristic_level=0, majorType=, propertyURI=http://www.w3.org/2000/01/rdf-schema#label, type=class}
Lookup	test	549	564	224980	{URI=http://www.semanticweb.org/array#IndexedArray, heuristic_level=0, majorType=, propertyURI=http://www.w3.org/2000/01/rdf-schema#label, type=class}
Lookup	test	559	564	224981	{URI=http://www.semanticweb.org/array#Array, heuristic_level=0, majorType=, propertyURI=http://www.w3.org/2000/01/rdf-schema#label, type=class}
Lookup	test	567	584	224982	{URI=http://www.semanticweb.org/array#AssocArray, heuristic_level=0, majorType=, propertyURI=http://www.w3.org/2000/01/rdf-schema#label, type=class}
Lookup	test	579	584	224983	{URI=http://www.semanticweb.org/array#Array, heuristic_level=0, majorType=, propertyURI=http://www.w3.org/2000/01/rdf-schema#label, type=class}

Фигура 6-6. Набор от характеристики за онтологично базирани анотации от тип *Lookup*

6.6 Тестване

За оценка на ефективността на предложения подход (за извличане на онтологично базирани анотации от неструктуриран текст на български език) се

проведе експеримент с примерен текст [15] от интернет източник. Използваният текст съдържа термини, които са предствени в онтология, визуализирана на Фигура 6-5.

За целта бяха създадени два конвейера за анализ на неструктурираното съдържание. В подготвителната фаза на първия конвейер се прилага алгоритъм за лематизация на думите. За втория конвейер този алгоритъм е заместен със стеминг.

С изпълнението на първия конвейер се разпознават 287 срещания на понятия от онтологията при потенциални 290. Генерираните *Lookup* анотации обхващат всички понятия от използваната онтология.

Резултатът от прилагането на втория конвейер на практика е: разпознати 2 понятия („индекс“ и „масив“), които имат висока честота в конкретния текст. Разпознаването на тези понятия се реализира, защото основната им форма съвпада с резултата от прилагането на стеминга. Във всички останали случаи понятията не могат да бъдат разпознати.

На Таблица 6-1 е представено честотното разпределение на разпознатите понятия в зависимост от използвания алгоритъм (лематизация или стеминг).

Таблица 6-1. Честотно разпределение на разпознатите понятия

Разпознато понятие	Лематизация	Стеминг
Масив	195	195
Индекс	45	45
Цяло число	1	1
Номериран масив	9	0
Асоциативен масив	14	0
Многомерен масив	6	0
Целочислен индекс	1	0
Низов индекс	1	0
Низ	3	0

Идеята за използване на този подход в различни приложения изисква да бъдат обхванати максимално вариантите за разпознаване в неструктуриран текст на въведените лингвистични данни за онтологични понятия. Ефективността на този процес зависи от използването на алгоритми за нормализация на думите. В този случай стемингът не позволява еднозначно тълкуване на срещаните словоформи. Например за изречението „*Информацията е масивна*“, резултатът от този алгоритъм е *stem=масив*, което се разпознава като понятие от конкретната онтология, което не е вярно. Ако се прилага лематизация, резултатът е „*масивен*“, което не съвпада с въведеното понятие.

[15] Източник на съдържанието на използвания примерен документ е: Обучение в PHP - част 5 (Работа с масиви)

6.7 **Изводи**

В тази глава е предложен модел за онтологично базирано извличане на понятия от неструктуриран текст на български език. За реализацията този модел са изпълнени следните задачи:

1. Предложен е подход за OBIE от неструктуриран текст, съобразен със спецификата на българския език;
2. За изпълнението на предложения подход е използван разработения от автора инструмент за лематизация на текст на български език в платформата GATE;
3. Последователно е представено изпълнението на етапите от предложения подход;
4. Предствени са получените резултати от изпълнението всеки етап от алгоритъма;
5. За оценка на ефективността на предложения модел е проведен експеримент с примерен текст от интернет извочник и е представено честотно разпределение на разпознатите понятия;

В резултат на реализацията на предложения модел за онтологично базирано извличане на понятия от неструктуриран текст на български език може да се направи следния извод:

1. Интегрираното средство за лематизация на текст на български език в платформата GATE, в комбинация с други ресурси предоставя възможност за:
   - Широкоспектърни лингвистични изследвания;
   - Създаване на онтологично базирани анотации върху съдържанието на документа, въпреки флексивността на езика.
2. Предложеният подход има приложение в процеси, свързани с генериране на съдържание и управление на информация и знание.

Представеният подход е представен в:

Borisova N., *An approach for Ontology Based Information Extraction (OBIE)*, списание Information Technologies and Control (ITC), под печат.

Заключение

В дисертационния труд е направен преглед на текущото развитие на технологиите, свързани със семантичния уеб. Разгледани са конкретни приложения на онтологиите в извличането на информация и управлението на знания. Изводите от направения обзор са използвани за създаване на концепция и модел за онтологично базирано извличане на понятия от неструктуриран текст на български език. Предложеният модел удовлетворява поставените изисквания и е реализиран в платформа с отворен код за анализ на текст.

Моделът е приложен успешно за извличане на понятия от текст на български език.

Основните приноси на дисертационния труд са:

1. Разработен е речниково базиран лематизатор за текст на български език, който се разпространява със свободен достъп под GPL v3 license.
2. Представеният речниково базиран лематизатор е реализиран и като инструмент с отворен код за платформата GATE, който е публично достъпен в GitHub [16].
3. Изследвана е ефективността на предложения инструмент за лематизация на текст на български език.
4. Предложен е модел за онтологично базирано извличане на понятия от неструктуриран текст на български език, който може да се използва като основа за автоматично извличане на семантична анотация.

Предложеният модел е апробиран, като са описани различните процеси, използвани за: предварителна обработка на текста за определяне на граматични категории за части на речта на срещаните думи [1]; нормализация на думите с цел разпознаване на съвпадения с лингвистични данни на онтологични понятия [2][3]; онтологично базирано извличане на понятия от неструктуриран текст на български език [5].

Идеи за развитие:

1. Изследване на възможностите за идентифициране на:
   - еквивалентност между субекти в текста, които са били разпознати (референция);
   - свойства и връзки на онтологичните понятия.
2. Формулиране на правила за разрешаване на многозначност;
3. Автоматично попълване на онтологията с екземпляри от неструктуриран текст на български език

[16] https://github.com/grigoriliev/BGLangTools/releases

Списък с публикации:

1. **Borisova N.**, G. Iliev, E. Karashtranova. *On Detecting Noun-Adjective Agreement Errors in Bulgarian Language Using GATE*, Proceedings of the Fifth International Conference of FMNS, FMNS 2013, Blagoevgrad, pp 180-187, 2013;
2. Iliev G., **N. Borisova**, E. Karashtranova, E., D. Kostadinova, *A Publicly Available Cross-Platform Lemmatizer for Bulgarian*, Proceedings of the Sixth International Scientific Conference – SWU, FMNS 2015, Blagoevgrad, pp 147-151, 2015;
3. Karashtranova E., G. Iliev, **N. Borisova**, Y. Chankova, I. Atanasova, *Evaluation of the Accuracy of the BGLemmatizer*, Proceedings of the Sixth International Scientific Conference – SWU, FMNS 2015, Blagoevgrad, pp 152-156, 2015;
4. Irena Atanasova, **Nadezhda Borisova**, *Using CLIPS To Realize Quality Of Life Expert System (QLIFEX)*, Proceedings of the Sixth International Scientific Conference – SWU, FMNS 2015, Blagoevgrad, pp 98-104, 2015;
5. **Borisova N.**, *An approach for Ontology Based Information Extraction (OBIE)*, списание Information Technologies and Control (ITC), приета за печат.

Благодарности

Приятно ми е да изкажа своята огромна благодарност към хората, които помогнаха настоящият дисертационен труд да стане възможен и да придобие своя окончателен вид:

- Благодаря на моите научни ръководители доц. д-р Димитрина Полимирова и доц. д-р Елена Каращранова за полезното научно сътрудничество и ценната подкрепа.

- Специално искам да изкажа своята благодарност към гл. ас. Григор Илиев и гл. ас. д-р Иво Дамянов, за ползотворните дискусии по проблемите на дисертационния труд и техните съвети при подготовка на статиите, върху които той бе оформен.

В заключение изказвам благодарност на ръководството на университета и на колегите от катедра „Информатика“.

Използвана литература

[ACE]

Automatic Content Extraction (ACE) Program; <https://www.ldc.upenn.edu/collaborations/past-projects/ace>

[Aghaei et al., 2012]

Aghaei S., Nematbakhsh M. and Farsani H., *Evolution of the World Wide Web: From Web 1.0 to Web 4.0*, International Journal of Web & Semantic Technology (IJWesT) Vol.3, No.1 (January 2012), DOI: 10.5121/ijwest.2012.3101; <http://www.airccse.org/journal/ijwest/papers/3112ijwest01.pdf>

[Berners-Lee et al., 2001]

Berners-Lee, Tim; James Hendler and Ora Lassila (May 17, 2001). *The Semantic Web*, Scientific American Magazine; <http://www.sciam.com/article.cfm?id=the-semantic-web&print=true>

[Blagoeva et al., 2012]

Diana Blagoeva, Svetla Koeva, Vladko Murdarov, *The Bulgarian Language in the Digital Age*, White Paper Series Volume 4, 2012, Springer Berlin Heidelberg, Print ISBN 978-3-642-30167-4, Online ISBN 978-3-642-30168-1, DOI 10.1007/978-3-642-30168-1; <http://www.springer.com/computer/ai/book/978-3-642-30167-4>

[Borisova et al., 2013]

Borisova N., G. Iliev, E. Karashtranova. *On Detecting Noun-Adjective Agreement Errors in Bulgarian Language Using GATE*, Proceedings of the Fifth International Conference of FMNS, FMNS 2013, Blagoevgrad, pp 180-187, 2013

[Castellanos-Nieves et al., 2011]

Castellanos-Nieves D., Fernandez-Breis J.T., Valencia-Garcia R., Martinez-Bejar R., Iniesta-Moreno M., *Semantic Web Technologies for supporting learning assessment*, Information Sciences Volume 181, Issue 9, 1 May 2011, p 1517–1537; <http://dx.doi.org/10.1016/j.ins.2011.01.010>

[Cunningham et al., 2002]

H. Cunningham, K. Bontcheva, D. Maynard, V. Tablan. *GATE — A New Release*, ELSNews, 11(1), 2002; <http://www.elsnet.org/publications/elsnews/11.1.pdf>

[Cunningham 2004]

Cunningham H., *Information Extraction, Automatic*, Preprint submitted to Elsevier Science, 2004; <https://gate.ac.uk/sale/ell2/ie/main.pdf>

[GEIST 2010/2011]

GEIST Research Group, *Semantic Web*, AGH University of Science and Technology Poland; <http://home.agh.edu.pl/~wta/semweb/geist-semweb-intro.pdf>

[Iliev et al., 2015]

Iliev G., N. Borisova, E. Karashtranova, E., D. Kostadinova, A Publicly Available Cross-Platform Lemmatizer for Bulgarian, Proceedings of the Sixth International Scientific Conference – SWU, FMNS 2015, Blagoevgrad, pp 147-151, 2015

[Gaeta et al., 2009]

Gaeta M., Orciuoli F., Ritrovato P., *Advanced ontology management system for personalised e-Learning*, Knowledge-Based Systems Volume 22, Issue 4, May 2009, p 292–301; <http://dx.doi.org/10.1016/j.knosys.2009.01.006>

[GATE]

<https://gate.ac.uk/>

[Getting 2007]

Getting B., (2007) *"Basic Definitions: Web 1.0, Web. 2.0, Web 3.0"*; <http://www.practicalecommerce.com/articles/464-Basic-Definitions-Web-1-0-Web-2-0-Web-3-0>

[Gruber 1993]

Gruber T. R., *A translation approach to portable ontology specifications*, Knowledge Acquisition (1993), vol.5, number 2

[Hendler 2001]

Hendler James, *Agents and the Semantic Web*, IEEE Intelligent Systems Journal, Volume 2, 2001, pp 30-37.

[Karashtranova et al., 2015]

Karashtranova E., G. Iliev, N. Borisova, Y. Chankova, I. Atanasova, *Evaluation of the Accuracy of the BGLemmatizer*, Proceedings of the Sixth International Scientific Conference – SWU, FMNS 2015, Blagoevgrad, pp 152-156, 2015

[Kiryakov et al., 2005]

Kiryakov A., D Ognyanov, D Manov, *OWLIM–a pragmatic semantic repository for OWL*, Web Information Systems Engineering – WISE 2005 Workshops, Lecture Notes in Computer Science Volume 3807, 2005, pp 182-192

[LingPipe]

<http://alias-i.com/lingpipe/>

[Maynard et al., 2006]

Maynard, D., Peters, W., Li, Y. (2006). *Metrics for Evaluation of Ontology-based Information Extraction*, In Proc of WWW 2006 Workshop on ";Evaluation of Ontologies for the Web"; (EON 2006), Edinburgh, Scotland, 2006; <https://gate.ac.uk/sale/eon06/eon.pdf>

[Moreno et al., 2012]

Moreno A., *SigTur/E-Destination: Ontology-based personalized recommendation of Tourism and Leisure Activities*, Engineering Applications of Artificial Intelligence, Available online 17 March 2012; <http://dx.doi.org/10.1016/j.engappai.2012.02.014>

[MUC]

Message Understanding Conference (MUC); <http://www-nlpir.nist.gov/related_projects/muc/>

[Nakov 2003]

Nakov P., *BulStem: Design and Evaluation of Inflectional Stemmer for Bulgarian.*, In Proceedings of Workshop on Balkan Language Resources and Tools

(1st Balkan Conference in Informatics), Thessaloniki, Greece, November, 2003; <http://lml.bas.bg/~nakov/selected_papers_list/nakov_BLRT_BulStem.pdf>

[OpenNLP]

Apache OpenNLP; <https://opennlp.apache.org/>

[O'Reilly 2005]

O'Reilly T., *What Is Web 2.0 Design Patterns and Business Models for the Next Generation of Software* (2005); <http://oreilly.com/web2/archive/what-is-web-20.html>

[O'Reilly& Battelle, 2006]

O'Reilly T., Battele J., *Web Squared: Web 2.0 Five Years On* (2005); <www.web2summit.com/web2009/public/schedule/detail/10194>

[Simov et al., 2002]

K. Simov, G. Popova, and P. Osenova. HPSG-based syntactic treebank of Bulgarian (BulTreeBank). In A. Wilson, P. Rayson, and T. McEnery, editors, A Rainbow of Corpora: Corpus Linguistics and the Languages of the World, p 135–142. Lincom-Europa, Munich, 2002; <www.bultreebank.org/papers/btba1.pdf>

[Simov et al., 2004a]

K. Simov, P. Osenova, A. Simov, and M. Kouylekov. Design and implementation of the Bulgarian HPSG-based treebank. Journal of Research on Language and Computation, 2(4):495–522, 2004.

[Simov et al., 2004b]

Simov K., P. Osenova, and M. Slavcheva, *BulTreeBank morphosyntactic tagset. Technical Report BTB-TR03*, BulTreeBank Project, 2004.

[Simov & Osenova, 2003]

K. Simov and P. Osenova. Practical annotation scheme for an HPSG treebank of Bulgarian. In Proceedings of the 4th International Workshop on Linguistically Interpreteted Corpora (LINC-2003), Budapest, Hungary, 2003.

[Tablan et al., 2002]

Valentin Tablan , Cristian Ursu , Kalina Bontcheva , Hamish Cunningham , Diana Maynard , Oana Hamza , Tony Mcenery , Paul Baker , Mark Leisher, *A Unicode-based Environment for Creation and Use of Language Resources*, In Proceedings of 3rd Language Resources and Evaluation Conference, p 66-71; <http://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=F9063E0E70FAA70A6878E6502D7F0968?doi=10.1.1.18.5528&rep=rep1&type=pdf>

[TreeTagger]

TreeTagger - a language independent part-of-speech tagger, <http://www.cis.uni-muenchen.de/~schmid/tools/TreeTagger/>

[UIMA]

Apache UIMA project, <http://uima.apache.org/>

[1] [van Rijsbergen 1979]

van Rijsbergen, C. J., *Information Retrieval*, London: Butterworths. Second Edition, 1979; <http://www.dcs.gla.ac.uk/Keith/Preface.html>

[W3C]

http://www.w3.org/standards/semanticweb/

[W3C 2007]

W3C, *Semantic Web and Other Technologies to Watch*; <http://www.w3.org/2007/Talks/0130-sb-W3CTechSemWeb/#(24)>

[W3C 2015]

W3C, *Standards*; <http://www.w3.org/TR/tr-status-stds>

[Атанасова 2011]

Атанасова Ир., Дисертационен труд „*Онтология в предметната област Бизнес-анализ*", ЮЗУ „Неофит Рилски", 2011 г.

[Коева 2008]

Коева Св., *Езикови ресурси и компютърни програми с приложение в лингвистичните изследвания*, в: Приложение на информационните технологии в работата на филолога и при изграждането на езикови ресурси, Архимед, София, 54-75, 2009. ISBN 987-954-779-106-0;
<http://dcl.bas.bg/PDF/LanguageResources.pdf>

[Симов 2012]

Симов К., *Електронни корпуси: видове, обработки и използване*, в: Секция за лингвистично моделиране, Институт за информационни и комуникационни технологии, Българска академия на науките;
<http://labling.shu-bg.net/LanguageResources-Shumen2012.ppt>

[Кръстев 1990]

Кръстев Б., *Морфология на българския език в 187 типови таблици*, Наука и изкуство, София, 1990.

[Нишева 2013]

Нишева М., хабилитационен труд „*Създаване на хетерогенни цифрови библиотеки, основано на онтологии*" във връзка с конкурс за професор по информатика и компютърни науки (информатика – системи, основани на знания), ФМИ на СУ „Св.Климент Охридски", 2013 г.; <https://www.fmi.uni-sofia.bg/habil_disert_trudove/habilitacionni_trudove_papka/Habilit_trud_MN.pdf>

[Осенова & Симов, 2007]

Осенова Петя, Кирил Симов, *Формална граматика на българския език*, Институт за паралелна обработка на информацията, БАН, София, 2007, Първо издание ISBN 978-954-92148-2-6

[Стоянова 2012]

Стоянова Ивелина, *Автоматично разпознаване и тагиране на съставни лексикални единици в българския език*, Автореферат на дисертация за присъждане на образователната и научната степен „доктор", Научен ръководител: Проф. Светла Коева, Секция по компютърна лингвистика, Институт за български език, БАН, София, 2012; <http://ibl.bas.bg/wp-content/uploads/2014/10/IStoyanova-avtoreferat.pdf>