Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.9
DOI: https://doi.org/10.24867/06BE35Petkovic

ПРОШИРЕЊЕ MONGODB БАЗЕ ПОДАТАКА ОТВОРЕНОГ КОДА ЗА ИНДЕКСИРАЊЕ И ПРЕТРАЖИВАЊЕ СРПСКИХ ТЕКСТОВА

AN EXTENSION OF THE MONGODB OPEN SOURCE DATABASE FOR INDEXING AND SEARCHING SERBIAN TEXTS

Стефан Петковић, *Факултет техничких наука, Нови Сад*

Област – ЕЛЕКТРОТЕХНИКА И РАЧУНАРСТВО

Кратак садржај – *Рад описује имплементацију проширења, MongoDB базе података отвореног кода, за подршку индексирања и претраживања српских текстова писаних на оба писма.*

Кључне речи: *Дигитални документи, индексирање, претрага, претпроцесирање, стеминг, Snowball, MongoDB*

Abstract – *This paper describes the implementation of the extension for the open-source MongoDB database that adds support for indexing and searching of Serbian texts written in both alphabets.*

Keywords: *Digital documents, indexing, search, preprocessing, stemming, Snowball, MongoDB*

1. УВОД

Људи су одувек имали потребу да забележе одређене важне догађаје, стечена искуства и знање и тако сакупљене информације пренесу наредним генерацијама. Слике на зидовима пећине Ласко, записи о кретању звезда, медицинске књиге које описују анатомију људског тела – све оно што су људи научили и открили се бележи и преноси [1]. Свака писана или цртана репрезентација мисли или неког догађаја која нам пружа одређене информације, доказе или се користи као службени записник сматра се документом [2].

Данас када кажемо документ можемо мислити на две ствари:

1. традиционални папирни документ и
2. дигитални документ [3].

Дигитални документ је заправо рачунарски обрађена информација којом се рукује као основном јединицом обраде. Примери дигиталних докумената су:

- текстуални дигитални документи (текстуални описи или поруке)
- графички документи (слике, цртежи, дијаграми, графикони)
- структурирани документи – *Hypertext Markup Language*, *Extensible Markup Language* [3].

НАПОМЕНА:
Овај рад проистекао је из мастер рада чији ментор је био др Драган Ивановић, ванр. проф.

MongoDB база података омогућава индексирање и претрагу садржаја складиштених у текстуалним пољима докумената. Тренутна верзија ове базе података подржава индексирање и претрагу текстуалних садржаја на данском, енглеском, италијанском, мађарском, немачком, норвешком, португалском, румунском, руском, турском, финском, француском, холандском, шведском и шпанском језику.

Тема мастер рада јесте да прикаже опис имплементације проширења функционалности текстуалне претраге. Проширење описано у овом раду имплементирано је за потребе претраге текстуалних садржаја на српском језику, али је на исти начин могуће додати подршку и за друге језике.

2. ТЕОРИЈСКЕ ОСНОВЕ

Системи за проналажење информација нам омогућују да пронађемо оно што нам је од значаја у мору информација унутар великих колекција дигиталних докумената. Своју потребу за информацијама корисници систему представљају задавањем одговарајућег упита. Задати упит се прво обради, а затим се на основу обрађеног упита врши претрага колекције. Обрада упита и претрага колекције, за информацијама од значаја, представљају процес претраживања. Како би се обезбедило ефикасно претраживање неопходно је извршити припрему постојећих докумената у колекцији, а такође и сваког новог документа приликом додавања. Ова припрема представља процес индексирања.

Приликом претраге великих колекција дигиталних докумената не захтева се од корисника да буде упознат са колекцијом коју претражује, нити се захтева прецизно дефинисање критеријума претраге (подразумева се непрецизност). Као резултат претраге добија се листа докумената која често поред релевантних резултата садржи и оне који то нису. Системи за проналажење информација теже да пронађу све релевантне документе, а да се при томе број нерелевантних докумената сведе на минимум.

Текстуални дигитални документи се састоје од речи које представљају одређени број знакова у тексту. Токен представља инстанцу речи која се појављује у документу, а једна класа еквиваленције речи представља терм. Ако се речи након примене одређених правила за претпроцесирање текста своде на исти низ знакова, онда припадају истој класи еквиваленције [3]. На пример следећа реченица:

Миш уз пушку, миш низ пушку.

садржи шест токена, док се број термова зна тек након примене правила за претпроцесирање текста. Јасно нам је да трећи и шести токен („пушку”) представљају један терм, али први („Миш”) и четврти токен („миш”) могу припадати истој класи еквиваленције само ако се примени правило пребацивања свих слова у мала слова.

Претпроцесирање текста представља примену одређених правила на текстуалне садржаје који обезбеђују флексибилност претраге [3]. Иста правила се примењују и на документе и на упите, а сам квалитет претраге у великој мери зависи од ових правила.

Лематизација представља редукцију различитих граматичких облика речи (лица, родови, падежи, времена, итд.) на њихов базни речнички облик, односно лему. Стеминг је груби хеуристички процес који одсеца крајеве речи са циљем да се постигне резултат што је могуће сличнији ономе који се постиже правилном лематизацијом базираном на лингвистичком знању [3]. Резултат стеминга (стем) представља одређени низ карактера који није нужно лема, односно корен улазне речи, а не мора чак ни постојати као реч у оквиру језика. Важно је једино да употреба стеминга допринесе повећању перформанси система.

3. ТЕХНОЛОГИЈЕ КОРИШЋЕНЕ ЗА ИМПЛЕМЕНТАЦИЈУ

MongoDB јесте документ оријентисана база података отвореног кода. Сваки рекорд складиштен у *MongoDB* бази података представља један BSON документ (*Binary JSON – JavaScript Object Notation*) кога чине парови поља и њихових вредности.

MongoDB складишти документе у колекцијама (*collections*), а колекције у базама података (*databases*). Колекције се могу упоредити са табелама у релационим базама података.

Базе података и колекције у *MongoDB* не морају се експлицитно креирати већ ће оне бити креиране приликом првог складиштења докумената у њима – имплицитно креирање. Због ове чињенице могуће је изабрати непостојећу базу података и у нову колекцију убацити документ. Убацивањем документа у колекцију креирају се и нова база података и нова колекција, уколико нису већ постојале.

Основне операције за рад са документима обухватају операције за креирање (додавање), измену, брисање и претрагу, односно читање докумената. Све наведене операције се извршавају над једном колекцијом. Атомичност (*atomicity*) операција за манипулацију документима (креирање, измена и брисање) остварена је на нивоу једног документа, чак и онда када се операција врши над документом који садржи више других докумената. Уколико једна операција врши модификацију већег броја независних докумената модификација сваког документа понаособ је атомична, али операција у целини није.

Приликом коришћења операција за претрагу докумената потребно је задати одговарајући критеријум претраге, односно филтер. Уколико се филтер изостави, односно уколико се проследи празан филтер резултујућа листа ће садржати све документе колекције.

Ефикасније извршавање операција за претрагу постиже се коришћењем индекса. Уколико се над колекцијом не креирају индекси претрага захтева проверу и поређење сваког документа у колекцији са задатим упитом (филтером).

Индекси представљају специјалне структуре података које чувају мали део скупа података у облику који се лакше претражује. У *MongoDB* индекси представљају Б-Стабла у којима се чувају сортиране вредности једног или више поља.

Snowball је прост језик за обраду ниски (*string*) креиран по узору на SNOBOL (*String Oriented and Symbolic Language*).

Основни типови података којима *Snowball* рукује јесу ниске, означени цели бројеви (*signed intigers*) и Булов тип података (*booleans*). Називи могу садржати слова, цифре и доњу црту, али морају почињати словом. Везују се за типове података, рутине (*routines*), групације (*groupings*) и споља видљиве интерфејсе (*externals*). Сви називи морају бити декларисани.

Рутине представљају секвенце једне или више наредби. Након дефинисања, рутине се могу позвати једноставним навођењем њиховог имена.

Ток извршавања *Snowball* програма контролише се имплицитном употребом сигнала, уместо експлицитне употребе различитих наредби за контролу тока (*if, else, break*, итд). Све наредбе у оквиру *Snowball* језика резултују позитивним (*true*) или негативним (*false*) сигналима. Уколико наредба резултује негативним сигналом наредбе које следе након ње се игноришу, а у супротном даље извршавање наредби се наставља.

Snowball програм се састоји од декларација након којих следе дефиниције групација и рутина. Декларисане променљиве доступне су из било које тачке програма и постоје све до краја његовог извршавања. Позив програма остварује се путем *Snowball* API (*Application Programming Interface*) кроз његове споља видљиве интерфејсе (*externals*). Snowball API дефинише појам тренутне ниске (*current string*) над којом споља видљиви интерфејси врше обраду. Вредност тренутне ниске, након извршене обраде, јесте резултат *Snowball* програма.

4. СТЕМЕР ЗА СРПСКИ ЈЕЗИК

Српски језик припада породици индоевропских језика, тачније словенској групи, а јужнословенској подгрупи језика. Због чињенице да је српски језик флективан језик његов стемер садржи далеко већи број правила од стемера за језике који то нису. Као основа стемера за српски језик коришћена су правила за стеминг хрватског језика, аутора Николе Љубешића и Ивана Панџића [4]. Постојећа правила су модификована и допуњена за потребе стеминга српског језика. За опис алгоритма коришћен је *Snowball* језик.

Главне рутине алгоритма за стеминг српског језика су:

Транслитерација (пресловљавање) ћирилице у латиницу – Српски језик користи два писма, ћирилицу и латиницу, а како је алгоритам за стеминг развијен са циљем претраживања информација, доступност информација на оба писма је од великог значаја.

Припрема (*prelude*) – Српски језик, поред два писма, користи и два дијалекта: екавски и ијекавски. На пример, речи сенка, сјенка и сијенка имају исто значење али су написане различитим дијалектом. Наведене речи потребно је сврстати под исту класу еквиваленције (терм), што се постиже трансформацијом екавице у ијекавицу. Такође, све је чешће у употреби неправилна комбинација латиничних слова „d" и „j" уместо слова „đ", па је и токене као на пример „Djoković" и „Đoković" потребно третирати као исти терм.

Обележавање региона (*mark regions*) – Алгоритам за стеминг српског језика користи један регион *R1* којим се одређује да ли ће стеминг бити примењен или не. Уколико најдужи суфикс из листе упада у регион *R1*, односно уколико не прелази тај регион, стеминг ће бити примењен. Регион *R1* може бити:

1. Регион након првог самогласника уколико се изван њега налазе најмање два слова, у супротном је то регион након првог сугласника који прати самогласник.
2. Регион након првог слова „p" уколико се изван њега налазе најмање два слова, у супротном је то регион након било којег првог слова које прати слово „p".

Поред региона *R1* постоји и тестна рутина *R2* којом се проверава постојање слова са дијакритицима. Одређена правила за стеминг се примењују само уколико тестна рутина резултује позитивним сигналом, односно уколико реч не садржи слова са дијакритицима.

Морфолошке промене – Представљају последњи корак пре примене стеминга и извршавају се са циљем добијања истих стемова за различите облике речи. На пример, за следеће речи потребно је генерисати исте стемове:

1. „правилан" - мушки род, једнина
2. „правилна" - женски род, једнина
3. „правилно" - средњи род, једнина

Како би добили исти стем за претходно наведене речи, прву реч („правилан") је потребно из једнине мушког рода трансформисати у множину мушког рода („правилни"), а затим применити правила за стеминг.

Стеминг – Процес стеминга подељен је у два корака, односно две рутине. Примарна рутина садржи највећи број правила, док секундарна садржи само неколицину. Секундарна рутина се примењује само у случају када примарна рутина резултује негативним сигналом, односно не изврши стеминг речи.

Евалуација стемера за српски језик извршена је употребом програма аутора Петра Дамјановића [5]. Резултат програма представљају прецизност, поврат, тачност и Ф-мера (*F-measure*) које означавају колико добро одређени стемер ради над колекцијом текстуалних докумената. Резултати евалуације стемера за српски језик, добијени употребом овог програма:

```
Precision : 0.9468657878535981
Recall : 0.8793230936500129
F-measure : 0.9118453871663618
```

5. ТЕКСТУАЛНА ПРЕТРАГА У ОКВИРУ *MONGODB*

MongoDB омогућава претрагу садржаја складиштених у текстуалним пољима докумената коришћењем *$text* оператора.

Како би се текстуална претрага могла користити, над колекцијом докумената је потребно дефинисати текстуални индекс. Текстуални индекс чува по један запис за сваку класу еквиваленције сваког индексираног поља у оквиру сваког документа унутар колекције. Креирање текстуалног индекса, над колекцијом докумената, постиже се извршавањем операције:

```
db.collection.createIndex({fieldName:"text"})
```

где *fieldName* представља назив поља чији садржај се индексира, а „*text*" представља текстуални литерал којим се специфицира текстуални тип индекса.

Потпроцес токенизације и претрпоцесирања се, у зависности од одабране верзије текстуалног индекса, обавља у оквиру *moveNext* методе једне од класа: *BasicFTSTokenizer* или *UnicodeFTSTokenizer*. Наведена метода проналази наредни токен унутар текстуалног садржаја, а затим врши његово претпроцесирање. Претпроцесирање токена, за све верзије текстуалних индекса, укључује филтрирање стоп речи и стеминг. Новину у трећој верзији текстуалних индекса представља брисање дијакритика након стеминга, што резултује индексима који нису осетљиви на слова са и без истих. Листа стоп речи, као и алгоритам за стеминг одређују се на основу изабраног језика приликом креирања индекса.

Први корак имплементације проширења представља регистрацију новог језика. Језици су представљени *BasicFTSLanguage* и *UnicodeFTSLanguage* класама, које поседују методе за креирање инстанци *FTSTokenizer* класа. Макро директивом:

```
MONGO_FTS_LANGUAGE_DECL(Serbian,"serbian","rs")
```

региструје се нови (у овом случају српски) језик за другу и трећу верзију текстуалног индекса, док се макро директивама:

```
MONGO_FTS_LANGUAGE_DECLARE(languageSerbianV1,
"serbian",TEXT_INDEX_VERSION_1);
```

и

```
MONGO_FTS_LANGUAGE_DECLARE(languageSerV1, "ser",
TEXT_INDEX_VERSION_1);
```

региструје нови језик за прву верзију текстуалног индекса.

Након регистрације језика потребно је додати нову листу стоп речи за дати језик. Листе стоп речи чувају се у текстуалним датотекама где сваки ред представља једну реч. Назив датотеке задаје се у формату:

```
stop_words_nazivJezika.txt
```

Креирану листу потребно је регистровати у оквиру скрипте за изградњу *FTS* модула.

Наредни корак представља додавање алгоритма за стеминг новорегистрованог језика, уколико одговарајући не постоји у оквиру *MongoDB*. Како је алгоритам за стеминг описан *Snowball* језиком, пре додавања исти је потребно превести у *ANSI C*. Након превођења нови алгоритам треба дефинисати у оквиру *libstemmer_c* библиотеке коју користи *MongoDB*.

Последњи корак представља проширење *codepointRemoveDiacritics* функције. Наведена функција уклања дијакритике, односно трансформише слова са дијакритицима у њима одговарајућа слова без дијакритика. Овај корак није обавезан када се сва потребна правила већ налазе у оквиру *codepointRemoveDiacritics* функције, међутим у случају проширења за српски језик недостајала су правила за трансформацију слова „Đ” и „đ” у њима одговарајућа слова без дијакритика „D” и „d”.

6. РЕЗУЛТАТИ И ДИСКУСИЈА

Након имплементације проширења, како би се исто могло испробати, потребно је покренути процедуру за изградњу *MongoDB* базе података (сервера и клијента).

По покретању сервера и клијента, а пре претраге текстуалних садржаја, потребно је креирати колекцију докумената и над њом текстуални индекс за српски језик.

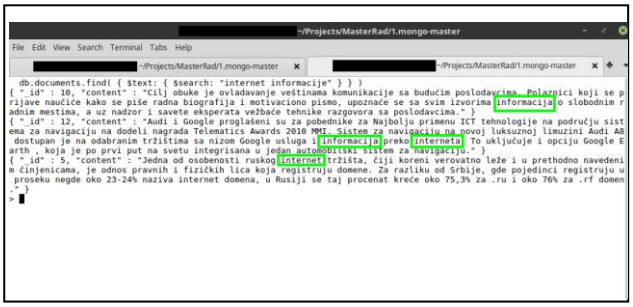
Претрага текстуалног садржаја, на српском језику, у оквиру претходно креиране колекције обавља се операцијом:

```
db.documents.find({$text:{$search:query}})
```

где *query* представља задати упит. На пример, уколико је потребно пронаћи сва документа на српском језику која садрже неку од речи “интернет” или “информације” (у било ком облику), упит се задаје на следећи начин:

```
db.documents.find({$text:{$search:"internet
informacije"}})
```

Резултати текстуалне претраге, на основу претходно задатог упита, приказани су на слици 1.



Слика 1. *Документи који садрже неку од речи: „интернет” или „информације”.*

Проширење текстуалне претраге описано у овом раду представља само један начин на који се одређене ствари могу имплементирати. Процеси транслитерације, трансформације ијекавице у екавицу и трансформације комбинације слова „d” и „j” у слово „đ” не морају бити део стеминг алгоритма већ се могу имплементирати у оквиру *moveNext* методе.

7. ЗАКЉУЧАК

Циљ рада је био да прикаже опис имплементације проширења функционалности текстуалне претраге доступне у оквиру *MongoDB* базе података отвореног кода. Проширење омогућава индексирање и претрагу текстуалних садржаја на српском језику, без обзира на писмо (ћирилица, латиница) или дијалект (екавица, ијекавица) којим је садржај писан.

Простора за даљи развој решења свакако има. Побољшање рутине за означавање региона, смањење броја правила у оквиру алгоритма за стеминг и ревизија листе стоп речи представљају неке од могућих даљих унапређења описаног проширења.

8. ЛИТЕРАТУРА

[1] https://academichelp.net/samples/academics/essays/expository/passing-on-knowledge.html (датум приступа: 2019-08-12)

[2] http://people.ischool.berkeley.edu/~buckland/digdoc.html (датум приступа: 2019-08-12)

[3] Ивановић Д., Милосављевић Б. (2015), Управљање дигиталним документима, ISBN: 978-86-7892-690-7, Факултет техничких наука, Нови Сад, Србија

[4] http://nlp.ffzg.hr/resources/tools/stemmer-for-croatian/ (датум приступа: 2019-08-12)

[5] https://github.com/pedam91/Stemmer-evaluation (датум приступа: 2019-08-12)

Кратка биографија:

Стефан Петковић је рођен у Сремској Митровици, држава Република Србија. Основну школу „Јован Јова-новић Змај” завршио је 2007. године у Сремској Митровици. Средњу економску школу „9. Мај” у Сремској Митровици завршио је 2011. године. Исте године уписао се на Факултет техничких наука у Новом Саду, одсек Рачунарство и аутоматика. Школске 2013/2014. године уписао се на смер Рачунарске науке и информатика. Положио је све испите предвиђене планом и програмом. Дипломирао је 2016. године, након чега уписује мастер студије на истом смеру, са усмерењем на Електронско пословање. Мастер рад на Факултету техничких наука из области Електротехнике и рачунарства – Примењене рачунарске науке о-дбранио је 2019. године.