Алгоритам за реконструкцију реченица из PDF документа

УДК 81'322.2:004.912

Весна Пајић,
svesna@agrif.bg.ac.rs
Универзитет у Београду, Пољопривредни факултет

Сташа Вујичић Станковић
stasa@matf.bg.ac.rs
Универзитет у Београду, Математички факултет

Милош Пајић
paja@agrif.bg.ac.rs
Универзитет у Београду, Пољопривредни факултет

САЖЕТАК: Употреба PDF докумената у обради природних језика постала је уобичајена и свакодневна активност истраживача у области рачунарске лингвистике и њој сличних. Издвајање текста из PDF докумената помоћу постојећих софтверских алата доводи до озбиљног нарушавања структуре реченице и параграфа, што представља велики проблем за лингвистички оријентисана истраживања. У овом раду представљамо нов алгоритам за реконструисање реченица и параграфа из PDF докмената, назван алгоритам за реконструисање реченице (енг. *Sentence Recovery Algorithm*) или скраћено SR алгоритам. Овај алгоритам као улаз користи текст издвојен из PDF документа и покушава да реконструише реченице из њега. Алгоритам узима у обзир проблеме настале погрешним тумачењем краја линије текста, прекидања реченице или параграфа насталим због уметнутих табела или слика, затим проблема насталих због хифенације и сличних. Осим описивања и евалуације алгоритма, представићемо и један случај имплементације алгоритма у Јава програмском језику, за обраду научних чланака оригинално записаних у PDF фомату.

КЉУЧНЕ РЕЧИ: обрада природних језика, језички ресурси, Java програмирање, процесирање PDF докумената.

ДАТУМ ПРИЈЕМА РАДА: 27. децембар 2014.
ДАТУМ ПРИХВАТАЊА РАДА: 18. април 2015.

1. Увод

Рачунарска обрада текстова на природним језицима, позната и као обрада природних језика (енг. *NaturalLangugaeProcessing* или NLP) интензивно се развија последњих година. Развој је праћен и њеном интеграцијом са другим областима рачунарства, као што су истраживање текста, претрага информација, екстракција информација, машинско превођење и друге. Све ове подобласти рачунарства користе текст писан природним језиком као улаз, а затим га обрађују и трансформишу на различите начине.

Текстуални ресурси који се том приликом користе разликују се међусобно по форматима датотека. Последично, они морају да буду и различито обрађивани, у складу са форматом. Обичне текстуалне датотеке (.TXT) и текстуалне датотеке преузете са веба у форми хипертекста (.HTML) неће бити обрађиване на исти начин. Па ипак, иако имају различиту структуру, формати као што су .TXT, .HTML или .XML могу бити

сматрани текстуалним датотекама, у смислу да је текст у њима представљен непрекинутом секвенцом карактера. HTML и XML датотеке имају додатне делове текста који представљају ознаке елемената и које не припадају природном језику, међутим јасно дефинисана синтакса HTML/XML елемената дозвољава њихово једноставно уклањање из текста. Због тога ћемо, са становишта обраде природних језика, сматрати да су ови формати еквивалентни. Данас постоји велики број софтверских алата помоћу којих је могућа адекватна обрада ових типова датотека. Неки од њих су UNITEX (Paumier, 2011), NooJ (Silberztein, 2003), GATE (Cunningham et al. 2002), различити омотачи (Muslea et al. 1999; Kushmerick 2000.; Liu et al. 2000; Baumgartner et al. 2001.) и слични.

Са друге стране, а посебно на вебу, последњих година се као формат размене докумената намеће PDF формат. С обзиром да велики број истраживача све више користи веб као корпус, сви они су, у неком тренутку, суочени са обрадом тесктуалних података у PDF документима. На несрећу, обрада докумената у PDF формату помоћу постојећих алата има неколико сметњи. Један од основних проблема за лингвистичку обраду PDF текстова јесте нарушена оригинална структура реченице и текста. Реченице могу бити прекинуте ознакама за крај линије текста или неким другим објектима. У овом раду ћемо представити нов метод за аутоматско трансформисање датотека из PDF формата у TXT формат, који омогућује превазилажење проблема нарушене структуре реченице и обраду текста уобичајеним методима и алатима.

У поглављу 2. описана је структура PDF датотека, приказани су неки од алата за конвертовање PDF докумената у текстуалне датотеке и описани проблеми који настају у том процесу, са лингвистичког становишта. Један од највећих проблема је нарушавање оригиналне структуре реченице. У поглављу 3. представљен је алгоритам (SR алгоритам) за реконструкцију структуре реченице до нивоа који омогућује даље процесирање текста. Имплементација овог алгоритма у Јава програмском језику и пример употребе приказан је у поглављу 4. У поглављу 5. дата је евалуација алгоритма, која показује да је алгоритам веома добар са становишта обраде природних језика. На крају је дат закључак и указано на правце будућих истраживања и активности неопходне за решавање представљеног проблема.

2. Трансформисање PDF датотека у TXT датотеке – тренутно стање

2.1. Portable Document Format (PDF)

PDF (енг. *Portable Document Format*) је формат осмишљен од стране компаније *AdobeSystems*[1], са намером да се документи представе независно од софтверских и хардверских платформи. Дизајниран је да сачува оригинални изглед документа, па ће тако PDF документ изгледати исто на екрану, као и одштампан на папиру, без обзира на тип рачунара или штампача корисника. Додатно, PDF документи су компримовани, омогућавајући тако да комплексне информације буду ефикасно преузете са веба. Као такав, PDF је постао стандард за електронску размену докумената, који одржава ISO организација.[2]

Структура и синтакса PDF датотеке је строго дефинисана. PDF документ је структура података састављена од мањег скупа основних типова података, који се користе за представљање делова PDF докумената: страна, фонтова, анотација и других. На најнижем нивоу, PDF документ је секвенца бајтова. Ови бајтови се групишу на основу специфичних синтаксних правила. Једна или више таквих група формирају синтаксне ентитете (објекте) вишег нивоа, представљајући садржај документа, али истовремено и начин на који тај садржај треба да буде уметнут и приказан на страни документа. За више детаља о спецификацији PDF документа, читаоцу предлажемо званичну документацију.

Овде ћемо дати само један пример дела садржаја PDF документа, који илуструје како PDF чува информације о тексту. Текст “ABC” је постављен 10 инчи од дна стране и 4 инча од леве ивице, користећи фонт *Helvetica* величине 12pt. Одговарајући део PDF документа би тада изгледао као у наставку:

1 http://www.adobe.com/devnet/pdf/pdf_reference.html

2 http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=51502

```
BT
/F13 12 Tf
288 720 Td
(ABC) Tj
ET
```

Горњих пет линија имају следеће значење:

- линија 1 – започиње текстуални објекат;
- линија 2 – поставља фонт и величину слова, као параметре у оквиру текста. Фонт *Helvetica* је представљен идентификационим кодом F13;
- линија 3 – одређује почетну позицију на страни;
- линија 4 – исцртава глифове низа карактера на датој позицији;
- линија 5 – завршава текстуални објекат.

Као што се може видети у примеру, унутар PDF документа је записано много више информација него што је то потебно за лингвистичку анализу и обраду текста. Највећи део информација се односи на визуелну репрезентацију текста, која је ирелевантна за NLP истраживања.

Како би ефикасно обрадио PDF документ, истраживач мора или да добро познаје структуру PDF формата, или да се ослони на постојеће софтвере за конверзију из PDF у TXT формат.

2.2. Софтвер за конверзију из PDF у TXT формат

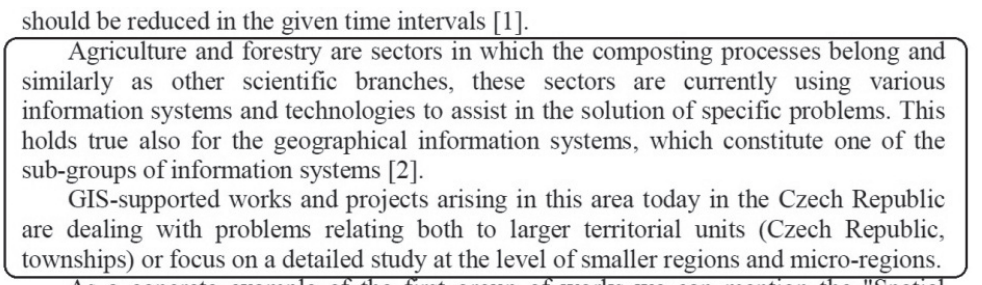
Данас постоји велики број софтверских алата за обраду и управљање PDF датотекама, од којих већина има опцију конвертовања у обичне текстуалне фајлове (.TXT). Без жеље да препоручимо било који од њих, поменућемо неке само као илустрацију стања у овој области.

Један од најважнији је званични софтвер компаније *Adobe* за управљање PDF датотекама. Тренутно актуелна верзија је названа *Adobe Acrobat XI*[3]. Овај софтвер допушта уређивање и креирање PDF докумената, сједињавање и комбиновање датотека, заштиту докумената и конвертовање у друге формате, као што су .TXT, .DOC, .HTML и други. *ABBYY PDF Transformer* 3.0[4] је још један комерцијалан софтвер, са сличним могућностима. Овај алат је вишејезични и омогућава једноставно конвертовање у формате који дозвољавају мењање и претрагу докумената, са сачуваним оригиналним изгледом. Осим ових, доступан је и велики број бесплатних алата на вебу, као што су *Some PDF*[5], *Convert PDF*[6], *Convert PDF to TXT*[7] и други. Постоји и *Google*-ов алат за прегледање PDF датотека у HTML облику. Можда најбољи са становишта очувања структуре текста јесте *GATE*-ов модул за конвертовање из PDF у HTML формат. Иако *GATE* нема могућност конвертовања у TXT директно, коректно препознати параграфи у HTML формату могу лако бити обрађивани као текст.

За програмере постоје API сучеља за скоро сваки програмски језик, који омогућавају управљање PDF документима директно из програмског кода. Овде вреди поменути *PDF Clown* и *Ice PDF*, с обзиром да су изузетно функционални, са занемарљивим бројем грешака, добро документовани и једноставни за коришћење. *PDF Clown*[8] је бесплатан скуп API класа отвореног кода, написан за више програмских језика (Java™ 6 и C#/.NET 4.0). *Ice PDF*[9] је отворен скуп класа у Javi за прегледање, штампање и манипулисање PDF документима. Може бити коришћен као самостална апликација или уграђен у било коју Java апликацију. Осим PDF рендеровања, може бити коришћен и за конверзију слика, PDF претрагу или екстракцију текста и слика из PDF докумената.

Без обзира који софтвер се користи, приликом конверзије из PDF у TXT формат јављају се слични проблеми са становишта лингвистичке обраде. Сваки од алата доводи до нарушавања структуре реченице и параграфа током конверзије у текстуалне датотеке. Разлог за то не лежи у несавршености самих алата, већ у природи самог PDF формата, који првенствено чува информације неопходне за обезбеђивање истог изгледа документа на било којој платформи.

3 http://www.adobe.com/rs/products/acrobat.html
4 http://pdftransformer.abbyy.com/
5 http://download.cnet.com/Some-PDF-to-Txt-Converter/3000-2079_4-10836740.html
6 http://convertonlinefree.com/PDFToTXTEN.aspx
7 http://www.convertpdftotext.net/
8 http://www.stefanochizzolini.it/en/projects/clown/index.html
9 http://www.icesoft.org/java/projects/ICEpdf/overview.jsf


should be reduced in the given time intervals [1].

Agriculture and forestry are sectors in which the composting processes belong and similarly as other scientific branches, these sectors are currently using various information systems and technologies to assist in the solution of specific problems. This holds true also for the geographical information systems, which constitute one of the sub-groups of information systems [2].

GIS-supported works and projects arising in this area today in the Czech Republic are dealing with problems relating both to larger territorial units (Czech Republic, townships) or focus on a detailed study at the level of smaller regions and micro-regions.

As a concrete example of the first group of works we can mention the "Spatial

Слика 1. Извод из PDF документа (текст је преузет из Часописа за пољопривредну технику, Број 4, 2012).

2.3. Проблеми настали након конверзије

Погледајмо мало детаљније процес конверзије документа из PDF у TXT формат. Као што смо напоменули у секцији 2.2, овај процес је сличан без обзира на то који се алат користи. Улаз у систем је увек неки документ у PDF формату (назваћемо га File.PDF), а излаз је текстуална датотетка креирана од стране алата за конверзију (означићемо се ја File.TXT). У примеру који следи ми смо користили *Ice PDF*, уколико није назначено другачије. Постоји неколико резултата процеса конверзија, који могу бити посматрани као проблеми за лингвистичко оријентисану обраду.

2.3.1. Убацивање ознаке за крај линије текста (EOL)

Ознака за крај линије текста (eng. *end-of-line* или EOL) се састоји од једног или више карактера који означавају крај линије или параграфа у текстуалним датотекама. Који скуп карактера ће бити коришћен као EOL зависи од софтверске платформе. На *Windows* оперативним системима, користи се тзв. *carriage return* (CR) карактер праћен *linefeed* (LF) карактером (означавају се и са '\r\n' или 0x0D0A). За визуелну репрезентацију ових карактера, када је то потребно, користи се *pilcrow* карактер (¶).

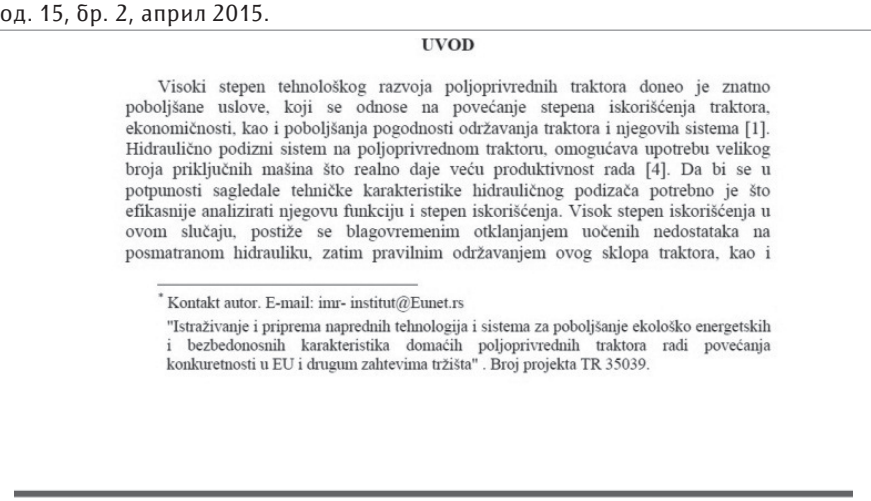
У већини случајева конверзије из PDF у TXT, EOL карактери се убацују на позиције где се линије визуелно преламају, а не само на позиције где параграфи текста заиста завршавају. На тај начин се губи информација о параграфу и реченици; један параграф из датотеке File.PDF (slika 1) се конвертује у неколико параграфа (слика 2) у датотеци File.TXT (по један параграф за сваку визуелну линију текста).

2.3.2. Прекидање параграфа објектима

PDF датотеке често имају много графичких елемената (објеката) уметнутих у текст на различите начине. То могу бити табеле, слике, графикони, формуле, објекти стране (заглавље, подножје, број стране) и слично. Уколико је објекат уметнут у параграф у датотеци File.

Agriculture·and·forestry·are·sectors·in·which·the·composting·processes·belong·and·¶
similarly·as·other·scientific·branches,·these·sectors·are·currently·using·various·¶
information·systems·and·technologies·to·assist·in·the·solution·of·specific·problems.·This·¶
holds·true·also·for·the·geographical·information·systems,·which·constitute·one·of·the·¶
sub-groups·of·information·systems·[2].·¶
GIS-supported·works·and·projects·arising·in·this·area·today·in·the·Czech·Republic·¶
are·dealing·with·problems·relating·both·to·larger·territorial·units·(Czech·Republic,·¶
townships)·or·focus·on·a·detailed·study·at·the·level·of·smaller·regions·and·micro-regions.·¶

Слика 2. Текст са слике 1 након конверзије у TXT формат; приказани су и скривени карактери, као што су размаци и крај параграфа.



UVOD

Visoki stepen tehnološkog razvoja poljoprivrednih traktora doneo je znatno poboljšane uslove, koji se odnose na povećanje stepena iskorišćenja traktora, ekonomičnosti, kao i poboljšanja pogodnosti održavanja traktora i njegovih sistema [1]. Hidraulično podizni sistem na poljoprivrednom traktoru, omogućava upotrebu velikog broja priključnih mašina što realno daje veću produktivnost rada [4]. Da bi se u potpunosti sagledale tehničke karakteristike hidrauličnog podizača potrebno je što efikasnije analizirati njegovu funkciju i stepen iskorišćenja. Visok stepen iskorišćenja u ovom slučaju, postiže se blagovremenim otklanjanjem uočenih nedostataka na posmatranom hidrauliku, zatim pravilnim održavanjem ovog sklopa traktora, kao i

* Kontakt autor. E-mail: imr- institut@Eunet.rs

"Istraživanje i priprema naprednih tehnologija i sistema za poboljšanje ekološko energetskih i bezbedonosnih karakteristika domaćih poljoprivrednih traktora radi povećanja konkuretnosti u EU i drugum zahtevima tržišta" . Broj projekta TR 35039.

90 Grozdanić B., et al.: Optimizacija hidrauličnog podizača .../ Polj. tehn. (2012/4), 89 - 94

pravilnim korišćenjem uputstva za rad istog. Nakon sprovedene sveobuhvatne analize predhodnog hidrauličnog podizača predložen je i ugrađen inovirani hidraulični podizač

Слика 3. Крај једне и почетак друге стране; параграф је подељен фуснотом и заглављем стране.

PDF (слика 3), тада је текстуална секвенца у фајлу File.TXT прекинута додатним линијама (слика 4), које су производ конверзије објекта.

UVOD·¶
¶
Visoki·stepen·tehnološkog·razvoja·poljoprivrednih·traktora·doneo·je·znatno¶
poboljšane·uslove,·koji·se·odnose·na·povećanje·stepena·iskorišćenja·traktora,·¶
ekonomičnosti,·kao·i·poboljšanja·pogodnosti·održavanja·traktora·i·njegovih·sistema·[1].·¶
Hidraulično·podizni·sistem·na·poljoprivrednom·traktoru,·omogućava·upotrebu·velikog¶
broja·priključnih·mašina·što·realno·daje·veću·produktivnost·rada·[4].·Da·bi·se·u·¶
potpunosti·sagledale·tehničke·karakteristike·hidrauličnog·podizača·potrebno·je·što·¶
efikasnije·analizirati·njegovu·funkciju·i·stepen·iskorišćenja.·Visok·stepen·iskorišćenja·u·¶
ovom·slučaju,·postiže·se·blagovremenim·otklanjanjem·uočenih·nedostataka·na¶
posmatranom·hidrauliku,·zatim·pravilnim·održavanjem·ovog·sklopa·traktora,·kao·i·¶
----------¶
¶
*¶
·Kontakt·autor.·E-mail:·imr-·institut@Eunet.rs·¶
"Istraživanje·i·priprema·naprednih·tehnologija·i·sistema·za·poboljšanje·ekološko·energetskih·¶
i·bezbedonosnih·karakteristika·domaćih·poljoprivrednih·traktora·radi·povećanja·¶
konkuretnosti·u·EU·i·drugum·zahtevima·tržišta"·.·Broj·projekta·TR·35039.¶
¶
Grozdanić·B.,·et·al.:·Optimizacija·hidrauličnog·podizača·:.../·Polj.·tehn.·(2012/4),·89·-·94·¶
90·¶
pravilnim·korišćenjem·uputstva·za·rad·istog.·Nakon·sprovedene·sveobuhvatne·analize·¶
predhodnog·hidrauličnog·podizača·predložen·je·i·ugrađen·inovirani·hidraulični·podizač·¶
nakon·čega·je·sprovedeno·ispitivanje·odnosnog,·rekonstruisanog,·hidrauličnog·podizača·¶
koji·je·ugrađen·na·nekim·traktorima·IMR-a·[5].·Ispitivanje·je·obavljeno·u·autentičnim·¶
eksploatacionim·uslovima·sa¶
¶
ciljem·da·se·dobiju·relevantni·rezultati·za·ocenu·¶

Слика 4. Конвертован текст са слике 3.

2.3.3. Проблем хифенације

Хифенација је процес убацивања карактера "-" којима се речи на крају визуелне линије текста преламају у следећи ред. Уколико су речи у датотеци File.PDF преломљене, тада се оне неће конвертовати у адекватне форме у датотеци File.TXT. На пример, ако је реч "*metabolic*" преломљена као "*meta-bolic*", биће конвертована у секвенцу "*meta-*", EOL карактер и секвенцу "*bolic*". Најједноставнији приступ би био једноставно уклонити карактере EOL и "-",и произвести једну реч ("*metabolic*"). Међутим, постоје и случајеви као што је, на пример, "*amino-glycosides*" када је неоподно задржати карактер "-".

2.3.4. Погрешна интерпретација карактера

С обзиром да је PDF формат дизајниран тако да задржи исти изглед документа, чест проблем који се јавља приликом конверзије је неадекватна репрезентације карактера. Овај проблем је очигледнији у обради текстова који садрже нелатиничне карактере. Овде је битно поменути да се наведени проблем појавио приликом употребе *ABBYY PDF Transformer*, при чему је *Ice PDF* конвертовао карактере коректно. Без обзира на то, битно је поменути овај проблем, с обзиром да се истраживачи често суочавају са њим.

На пример, неке од погрешних конверзија које смо приметили су замена ћириличног слова **з**

глифом **3**, ћириличног слова **и** латиничним словом **u** (курзив слова **и** је ***u***), ћириличног слова улатиничним словом **y**. Даље, појављује се проблем конвертовања два карактера, као што је **fl** једним глифом **fl** (као у "*single flagellum*") и многи други. Обично је веома тешко уочити да је дошло до грешке, с обзиром да замењени карактери изгледају исто или веома слично као и прави.

3. Алгоритам за реконструкцију реченције (SR алгоритам)

У већини NLP техника и метода, реченица је основна јединица обраде текста. Због тога је формат добијен након конверзије из PDF у TXT (File.TXT) неадекватан за лингвистичку обраду и анализу. Како, са друге стране, PDF формат доминира као начин представљања докумената, посебно на вебу, додатна обрада и припрема ових фајлова је неопходна како би се на њх применили NLP алати. Овај посао може бити временски захтеван уколико се ради ручно, посебно јер су колекције докумената обично веома обимне.

Имајући сличан проблем у нашим истраживањима и покушавајући да их превазиђемо, развили смо алгоритам који аутоматизује процес припреме текста за даље процесирање. На тај начин се смањује људско ангажовање у процесу обраде. Алгоритам је развијен тако да може да буде примењен и у другим истраживањима. Довољно је прецизан да може бити коришћен у NLP истраживањима са високом поузданошћу. Додатно, алгоритам је језички независтан, с обзрром да се ослања само на статистичке особине текста.

Алгоритам је примарно развијен за процесирање докумената који се углавном састоје од текста и имају структуру сличну штампаним документима као што су научни чланци, књиге или новине. Овакви документи, иако могу да садрже и различите графичке елементе (фотографије, графиконе, табеле и др.), углавном представљају информације у текстуалној форми. Додатно, текст је организован у параграфе, са спорадичним насловима и поднасловима. Остали PDF документи, као што су PPT слајдови или различити каталози са пуно слика, не могу бити процесирани SR алгоритмом ефикасно и нису предмет овог истраживања.

3.1. Главни ток SR алгоритма

Улаз у SR алгоритам је текст добијен након иницијалне конверзије документа из PDF у TXT формат постојећим алатима. Особине таквог текста су описане у претходним секцијама, али је главна особина оваквог текста та да је структура реченице и параграфа на неки начин прекинута и нарушена, па процесирање уобичајеним NLP алатима није могуће.

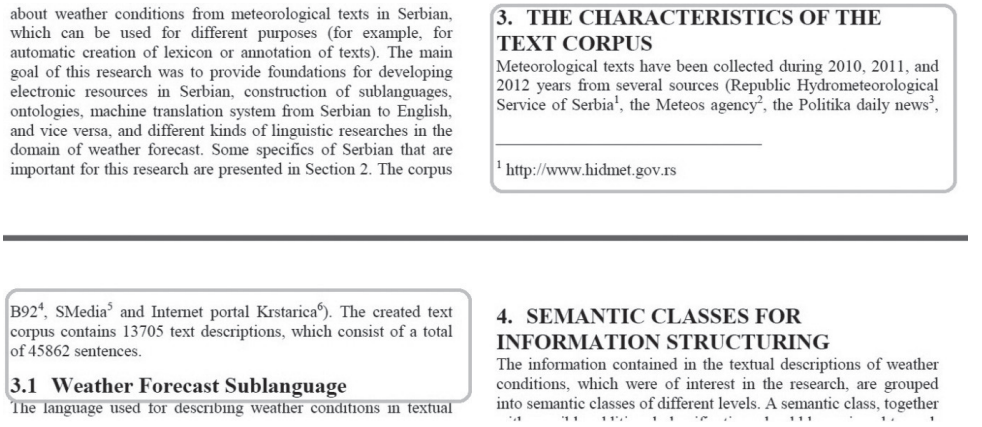
С обзиром да је реторичка структура текста нарушена, улазни текст се посматра као секвенца линија текста. SR алгоритам покушава да сваку од прочитаних линија текста класификује у једну од следећих класа:

- насловна линија (или део наслова);
- почетак, централни део или крај параграфа;
- наслов објекта (табеле, слике и др.);
- део конвертованог објекта (на пример, делови табеле или конвертоване формуле);
- елемент стране (заглавље, подножје, број стране и сл.).

Након идентификације (класификације) линије, SR алгоритам на основу врсте линије предузима одређене акције. Ове акције могу да варирају у зависности од коначне намене процесирања текста. На пример, уколико корисника занима само проучавање самог језика, тада су табеларни подаци вероватно ирелевантни, па могу бити изостављени из излазног текста. Са друге стране, уколико се врши издвајање информација из текста, управо табеле могу садржати кључне податке, па ће бити задржане у тексту како би се омогућило њихово даље процесирање. У том смислу, могуће су модификације основног алгоритма.

Основни облик SR алгоритма, представљеног у овом раду, обухвата следеће акције:

- узастопне линије препознате као насловне конвертују се у један параграф;
- EOL карактери се уклањају из линија препознатих као линије које су на почетку или у средишту параграфа, са посебним процесирањем преломљених речи;
- наслови објеката се задржавају као посебни параграфи;



about weather conditions from meteorological texts in Serbian, which can be used for different purposes (for example, for automatic creation of lexicon or annotation of texts). The main goal of this research was to provide foundations for developing electronic resources in Serbian, construction of sublanguages, ontologies, machine translation system from Serbian to English, and vice versa, and different kinds of linguistic researches in the domain of weather forecast. Some specifics of Serbian that are important for this research are presented in Section 2. The corpus

3. THE CHARACTERISTICS OF THE TEXT CORPUS

Meteorological texts have been collected during 2010, 2011, and 2012 years from several sources (Republic Hydrometeorological Service of Serbia[1], the Meteos agency[2], the Politika daily news[3],

[1] http://www.hidmet.gov.rs

B92[4], SMedia[5] and Internet portal Krstarica[6]). The created text corpus contains 13705 text descriptions, which consist of a total of 45862 sentences.

3.1 Weather Forecast Sublanguage

The language used for describing weather conditions in textual

4. SEMANTIC CLASSES FOR INFORMATION STRUCTURING

The information contained in the textual descriptions of weather conditions, which were of interest in the research, are grouped into semantic classes of different levels. A semantic class, together

Слика 5. Део PDF документа; у наставку ће бити објашњено процесирање дела текста унутар сивог оквира.

- делови конвертованих објеката се уклањају из излазне датотеке;
- елементи стране се уклањају из излазне датотеке.

3.·THE·CHARACTERISTICS·OF·THE·¶
TEXT·CORPUS·¶
Meteorological·texts·have·been·collected·during·2010,·2011,·and·¶
2012·years·from·several·sources·(Republic·Hydrometeorological·¶
Service·of·Serbia1,·the·Meteos·agency2,·the·Politika·daily·news3,·¶
.....¶
1·http://www.hidmet.gov.rs¶
·B924,·SMedia5·and·Internet·portal·Krstarica6).·The·created·text·¶
corpus·contains·13705·text·descriptions,·which·consist·of·a·total·¶
of·45862·sentences.·¶
3.1·Weather·Forecast·Sublanguage·¶

Слика 6. Текст након иницијалне конверзије из PDF у TXT формат; скривени карактери (EOL и размаци) су приказани да би демонстрирали праву структуру текста.

У наставку је дат пример улазног текста у PDF формату, као и након иницијалне конверзије. Слика 5 приказује део PDF документа. Пример процесирања је приказан за део текста унутар сивог оквира. Тај део текста се састоји од једног наслова, параграфа који се протеже на две стране, уметнуте фусноте и једног поднаслова.

Након конверзије PDF документа уобичајеним софтвером (у овом случају коришћен је *Ice PDF*, због његових особина објашњених у 2.3.4), добијено је једанаест линија текста приказаних на слици 6.

Идеално, SR алгоритам би требало да обради ове линије текста на следећи начин:

- Линије 1 и 2 би требале да буду препознате као насловне и да буду спојене у један параграф (то ћемо постићи анализом последњег карактера у линији, врсти слова и дужини линије);
- Линије 3, 4 и 5 би требале да буду препознате као један параграф и спојене заједно; EOL карактер ће бити уклоњен са краја линије и биће додати размаци;
- Линија 6, која садржи само размаке и EOL, треба да буде потпуно уклоњена (биће утврђено да се претходни параграф није завршио још, па линија 6 може бити само део неког конвертованог објекта – фусноте у овом случају);
- Линија 7 треба да буде препозната као фуснота и уклоњена потпуно (то ће бити постигнуто анализом њене дужине);
- Линије 8, 9 и 10 би требале да буду препознате као остатак параграфа и спојене са претходним параграфом, уклањајући при том EOL карактере, сем из линије 10, која у ствари и завршава параграф;
- Линија 11 треба да буде препозната као наслов и да остане задржана како јесте, тј. као један параграф.

Након процесирања, текст би требало да изгледа као на слици 7.

3.·THE·CHARACTERISTICS·OF·THE·TEXT·CORPUS·¶

Meteorological·texts·have·been·collected·during·2010,·2011,·and·2012·years·from·several·sources·(Republic·Hydrometeorological·Service·of·Serbia1,·the·Meteos·agency2,·the·Politika·daily·news3,·B924,·SMedia5·and·Internet·portal·Krstarica6).·The·created·text·corpus·contains·13705·text·descriptions,·which·consist·of·a·total·of·45862·sentences.·¶

3.1·Weather·Forecast·Sublanguage·¶

Слика 7. Текст након примене SR алгоритма; скривени карактери (EOL и размаци) су приказани да би демонстрирали праву структуру текста.

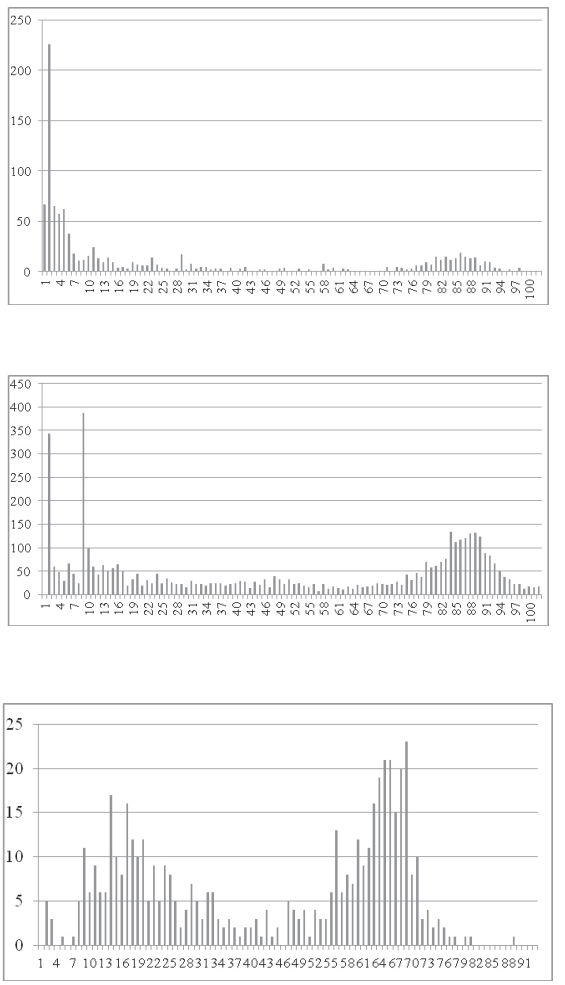
3.2. Предуслови за ефикасну примену SR алгоритма

Током истраживања процесирали смо велики број PDF докумената, углавном научних чланака и енциклопедијског текста. С обзиром да су сви ти документи били креирани углавном за штампање, имали су сличну форму и неке заједничке особине, које су искоришћене приликом креирања самог алгоритма. Због тога су ове особине и задржане као предуслови које документ треба да задовољи, како би на њега био ефикасно примењен SR алгоритам. Обрада докумената који не задовољавају ове предуслове SR алгоритмом је могућа, али са смањеном укупном ефикасношћу и прецизношћу.

Предуслови су:

1. Документ се састоји углавном од текста, организованог у параграфе и наслове, са спорадично уметнутим објектима.
2. Параграфи текста који нису наслови имају исту величину слова којом се приказују, тако да линије текста које припадају параграфу имају апроксимативно једнак број карактера (једнаку дужину) у већем делу документа. Ова дужина одговара ширини једне колоне текста.
3. Наслови се могу разликовати од остатка текста не само на основу форматирања у главном документу, већ и на основу других особина као што су сва велика слова, прво слово сваке речи је велико, имају празну линију испред и иза и сл.
4. Наслови су углавном краћи него линије параграфа; имају мање карактера него линија у параграфу текста.

На основу предуслова 2, анализирали смо дужину линија текста у документу. Иако документи имају различите формате, текстови попут научних чланака, новина или књига имају слично дистрибуцију дужина линија текста. Као пример, приказана је дистрибуција за три различита документа на слици 8a, 8b i 8c.



Слика 8. Дистрибуција броја линија које имају одређену дужину; хоризонтална оса приказује број карактера по линији текста, а вертикална оса приказује број линија текста које имају одређен број карактера. 8.a i 8.b приказују дистрибуцију за два различита документа са једном колоном текста; 8.c приказује дистрибуцију за документ са две колоне текста.

Документи чија је дистрибуција приказана на сликама 8.a и 8.b, садржали су текст у једној колони (ступцу), а документ са слике 8.c је имао текст сложен у две колоне. Приликом обраде већег броја докумената (примери дати на слици 8 то и илуструју), приметили смо да ће већина линија бити рапоређена око две вредности (два мода на графицима на слици 8). Прва од њих се налази са леве стране хистограма, у близини вредности 0, и она представља просечну вредност кратких линија текста. Уочљивија је на сликама 8.a и 8.b, где одражава присуство великог броја линија које имају дужину мању од 10 карактера. То су углавном празне линије, делови конвертованих објеката, линије које садрже само број стране и сличне. Друга вредност око које се групишу линије текста позиционирана је у десном делу хистограма и представља просечну дужину дугачких линија текста.

Ова друга вредност је веома важна за SR алгоритам, с обзиром да описује дужину линије текста која припада параграфима текста, тј. ширину колоне текста. У наставку ћемо ову вредност обележавати са CW (eng. *column width*). CW вредност се разликује за различите докумнете и зависи од величине стране, величине фонта, маргина и сличних параметара стране. Тако на пример, само посматрајући хистограме на слици 8, може се уочити да је документ представљен на слици 8.a имао CW вредност око 85, а документ са слике 8.b 88. CW вредност за документ на слици 8.c је око 65, што указује на то да је овај документ имао ужу колону текста него прва два документа (ово је заиста и био случај, с обзиром да се ради о тексту сложеном у два ступца).

3.3. Израчунавање CW вредности

SR алгоритам користи CW вредност како би одлучио да ли линија текста припада параграфу или наслову. Пошто сваки документ има своју специфичну CW вредност, неопходно је прво је израчунати.

Текстулани документ који ћемо обрађивати SR алгоритмом, као што смо то већ показали, може да буде посматран као секвенца или низ линија текста. У том смислу, документ D можемо да представимо $D=\{t_i, i=1..|D|\}$, где је $|D|$ број линија текста у документу D, а t_i је једна линија текста. Свака линија текста t_i има своју дужину, т.ј. број карактера у линији, без завршног EOL карактера. Ову дужину ћемо означити са $l(t_i)$. Нека је m једнако $max_D l(t_i)$, тј. максималној дужини линија у документу.

Дистрибуција дужина линија (*DL*) је низ целих бројева $DL = \{x_j \in N | j=0\ldots m\}$, таквих да постоји x_j линија у документу које имају дужину j (слика 8). Као што је показано у секцији 3.2, DL има бимодалну дистрибуцију. За потребе SR алгоритма, потребно је да одредимо десни мод, који ће у ствари представљати CW вредност.

Прво је потребно израчунати просечну дужину l_{avg} свих линија у документу као

$$l_{avg} = \frac{\sum_{i \in 1..|D|} l(t_i)}{|D|} \quad \mathbf{l_{avg}} = \frac{\sum_{\mathbf{i \in 1..|D|}} \mathbf{l(t_i)}}{\mathbf{|D|}}$$

$$\mathbf{l_{avg}} = \frac{\sum_{\mathbf{i \in 1..|D|}} \mathbf{l(t_i)}}{\mathbf{|D|}} \mathbf{l_{avg}} = \frac{\sum_{\mathbf{i \in 1..|D|}} \mathbf{l(t_i)}}{\mathbf{|D|}} .$$

Ова вредност је важна, јер уколико се хистограм DL вредности подели на два дела у односу са l_{avg}, CW вредност остаје у десном делу хистограма. Тада, CW можемо да израчунамо као дужину коју има највећи број линија у десном делу хистограма (тј. које имају већу дужину од просечне):

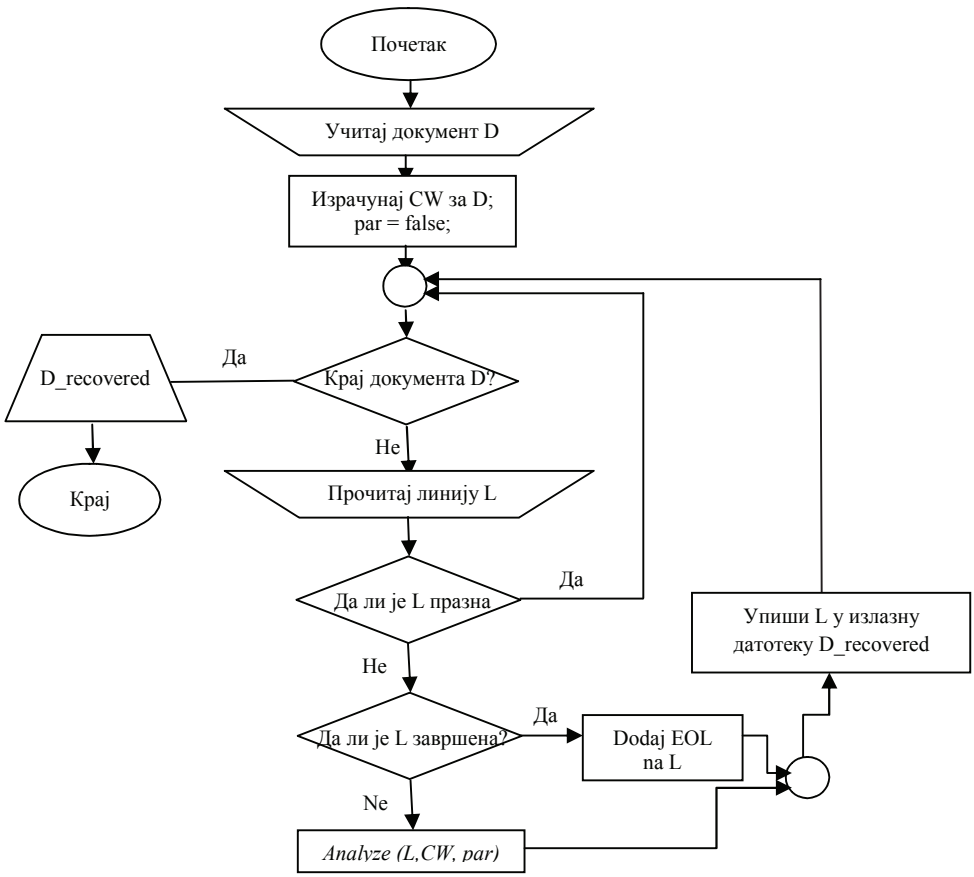
$CW = \{i \mid x_i = \max\{ x_j \mid j \geq l_{avg}\}\}$.

Уколико израчунамо CW вредност помоћу ове формуле, документи представљени на слици 8 имаће CW вредност редом 85, 83 и 68.

3.4. Дизајн и шема SR алгоритма

Главни задатак SR алгоритма је да реконструише реченице чија је структура нарушена током конверзије документа. Због тога је важно обезбедити да се делови текста који припадају једној реченици споје поново у једну реченицу, тј. да реченица не буде подељена на делове помоћу EOL карактера. Било би идеално да се део текста који припада јеном параграфу у оригиналном документу трансформише у један параграф у излазној TXT датотеци, али је могуће да помоћу SR алгоритма он буде трансформисан у два или више параграфа. Ово нећемо сматрати грешком, док год једна реченица почиње и завршава у истом параграфу, тј. не постоји реченица која се протеже кроз два парагарафа.

У том смислу, алгоритам разликује три основна типа улазних текстуалних линија: празне линије (EL), које садрже само нула или више размака



Слика 9. Шема главног тока SR алгоритма.

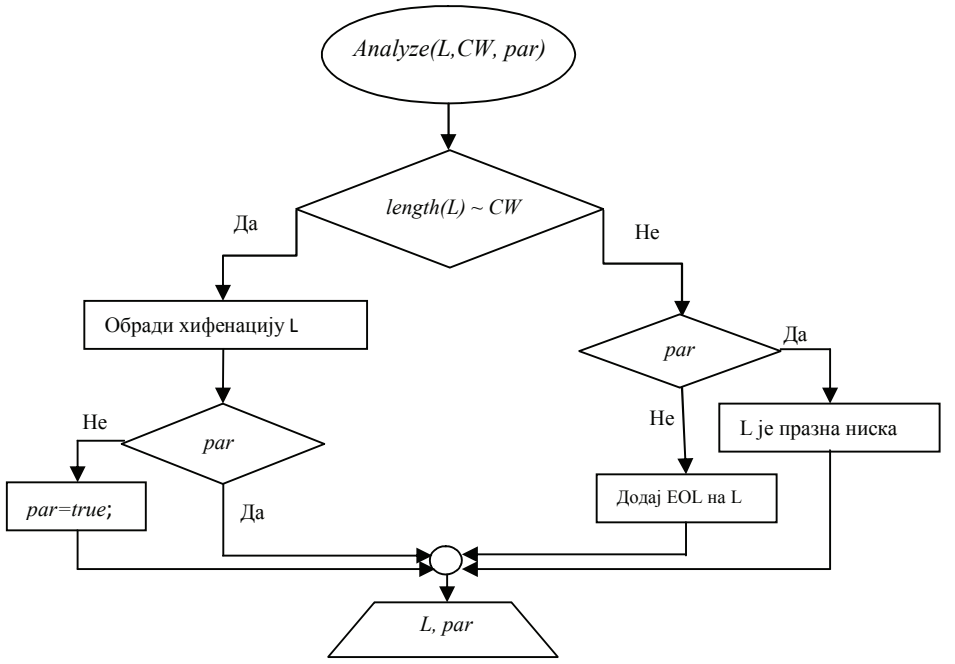
и EOL карактер; завршене линије (FL), које се завршавају неким од знакова из скупа F={. , ?, !} пређеним EOL карактером; и незавршене линије (UL), које не завршавају неким од карактера из скупа F={. , ?, !} праћеним EOL карактером.

Обрада EL і FL линија је тривијална; EL се бришу из излазне датотеке, а на крај FL се додаје EOL карактер. UL линије, са друге стране, морају да буду посебно анализиране и процесиране, у зависности од неколико могућих сценарија (да ли су оне део наслова, параграфа или конвертованог објекта).

Шема главног тока SR алгоритма је приказана на слици 9. Приликом читања линија из улазне датотеке, алгоритам прати да ли се налази унутар параграфа или изван њега, подешавајући вредност промењиве *par* на TRUE или FALSE. Ова вредност је веома важна за анализу UL линија.

Анализа UL линија (означена позивом модула *Analyze (L, CW, par)*) приказана је на слици 10. Заснована је на поређењу дужине UL линије са CW вредности. Зависно од структуре документа (да ли је текст у једној колони или више њих), дужине линија текста унутар параграфа одступају од CW вредности мање или више. Иницијално, алгоритам је дизајниран тако да толерише одступања од CW до 10%; на пример, уколико је CW 85 карактера, све линије које имају између 77 и 93 карактера ће бити сматране делом текстуалних параграфа. Уколико је потребно, могуће је променити ниво одступања, како би се боље обрадио документ одређене структуре. Као резултат овог поређења, доноси се одлука да ли је нека UL линија део текстуалног параграфа или није.

Овде је важно напоменути да анализа UL линија такође зависи и од вредности промењиве *par*, тј. од чињенице да ли је текстуални параграф већ започео или није. На тај начин, алгоритам



Слика 10. Анализа UL линија.

разликује наслове од конвертованих објеката, с обзиром да наслови не могу бити уметнути у параграфе, а објекти могу. Уколико наслови имају нека додатна својства (на пример, свака реч је написана почетним великим словом или су сва слова велика), могуће је искористити ова својства за додатну анализу наслова. Процесирање преломљених речи је могуће урадити најједноставније брисањем карактера "-" и спајањем текуће линије са следећом. Међутим, уколико су на располагању неки додатни језички ресурси, попут лексикона, електронских речника и сличних, могуће је интегрисати их са SR алгоритмом како би се обавила провера да ли је реч преломљена или се ради о сложеници. У примеру који је овде описан користили смо само први начин обраде.

4. Пример имплементације у Javi

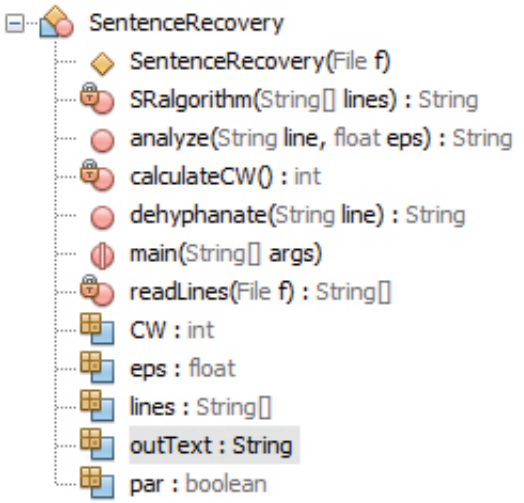
Током истраживања, користили смо SR алгоритам више пута, обрађујући различите типове докумената, од научних чланака до новела. За сопствене потребе развили смо Јава класу, названу *Sentence Recovery*, која имплементура SR алгоритам. С обзиром да смо већ имали одређен број текстова добијених различитим алатима из PDF фајлова, класа *Sentence Recovery* је креиана као самостална класа, али је могуће да се дизајнира и тако да може да се интегрише са постојећим алатима (на пример, са *Ice PDF* класама) за процесирање PDF докумената.

Чланови класе *Sentence Recovery* су приказани на слици 11. Објектни атрибут *lines* је низ текстуалних ниски, које представљају линије из документа који се обрађује (EL, FL и UL линије). Индикатор *par* се користи за праћење текстуалних параграфа, да ли је параграф започео или није. Објектни атрибут CW чува CW вредност документа. Променљива *eps* представља дозвољено одступање од CW вредности. Њена подразумевана вредност је 10%. Објектни атрибут *outTekst* чува излазни текст након процесирања.

Иницијално, након креирања објекат класе *Sentence Recovery*, први корак јесте попуњавање низа lines. Линије се најчешће читају позивом метода *readLines()*. Након тога израчунава се вредност CW помоћу метода *calculateCW()*. Оба метода се позивају аутоматски у конструктору класе.

Методи *SRalgorithm*() и *analyze*() су директне имплементације алгоритама представљених у секцији 3.4. (слике 9. і 10.).

У наставку је дат пример Јава кода који користи класу *Sentence Recovery*:



Слика 11. Чланови класе *Sentence Recovery*.

```
File dir = new File(someDir);
File[] files = dir.listFiles();
for (File f:files){
    SentenceRecoverysr = new
SentenceRecovery(f);
    try{
        FileOutputStream out =
            newFileOutputStream(f.
            getAbsolutePath() + "._SRalg" );
        out.write(sr.outText.getBytes());
        out.close();
    } catch (Exception e) {
        e.printStackTrace();
        System.out.println(sr.outTekst);
    }
}
```

Овај код узима листу фајлова из једног директоријума и генериш реконструисан текст из сваког од њих. Текст је снимљен у новом фајлу унутар истог директоријума. На тај начин је могуће, са само неколико линија кода, обрадити велики број PDF докумената.

5. Експериментална евалуација и анализа резултата

Како бисмо тестирали и евалуирали SR алгоритам, изведено је неколико експеримената. Експерименти су дизајнирани тако да узму у обзир неке од унапред уочених мањкавости процеса. Најважније је било проценити ефикасност коректно уметнутих EOL карактера.

Анализа линија и одлука да ли уметнути EOL на крај линије или не, зависи од структуре самог документа. Због тога смо одабрали документе три различита типа, обрадили их на исти начин и упоредили резултате. Документи коришћени за тестирање били су научни часопис са једном колоном, научни часопис са две колоне и роман.

Први експеримент је поредио наше резултате са резултатима добијених користећи друге алате који покушавају да утврде почетак и крај параграфа текста. Други експеримент је покушао да установи оптималну вредност епс променљиве, коришћену приликом поређења дужине линије са CW вредности.

5.1. Поређење са постојећим софтвером

Постоји велики број програма за конвертовање из PDF у TXT формате (или сличне, попут HTML или XML). Ипак, не тако много њих има могућност да реконструише реторички структуру документа. Уместо тога, фокусирајући се само на визуелне аспекте, већина софтвера само убацује EOL карактер на места на којим се линије текста визуелно преламају, нарушавајући структуру реченице на тај начин.

Установили смо да, међу свим алатима који могу да обраде PDF документе, једино GATE има неки вид реконструкције реченице и параграфа. Иако и GATE умеће EOL карактере на местима где линије текста визуелно завршавају, он такође покушава да препозна и почетак и крај параграфа текста и умеће и на одговарајуће позиције. Због тога смо одабрали управо GATE као програм са којим ћемо поредити наше резултате.

Како бисмо умањили утицај структуре документа на процес евалуације, обрадили смо документа са три различите структуре. Први документ је био научни часопис са једном колоном текста, други научни часопис са две колоне текста, а трећи роман. Прва два документа су имала више наслова у поређењу са трећим. Додатно, текстуални параграфи у прва два документа су често били прекидани конвертованим објектима, попут табела, математичких формула или слика, док у трећем није било објеката. Са друге стране, трећи

документ је имао пуно параграфа са кратким линијама, који су били делови дијалога.

Као што је напоменуто раније у секцији 3.4, основни задатак SR алгоритма јесте да сачува (реконструише) структуру реченице, тј. да у излазној датотеци нема реченица које се протежу преко два или више параграфа. Главни циљ нашег истраживања био је да припремимо текст за даља лингвистичка истраживања програмима који користе реченицу као основну јединицу обраде. Због тога смо и резултате SR алгоритма анализирали на основу броја реченица које су остале нарушене након обарде. Резултати поређења су приказани у табели 1. Дате су и апсолутне и релативне вредности.

Структура документа	Софтверски алат	Реченице које су остале нарушене након обраде	
		Укупно	%
Научни часопис – један стубац	GATE	10	10%
	SR алгоритам	3	3%
Научни часопис – два ступца	GATE	12	15%
	SR алгоритам	6	8%
Роман	GATE	323	23%
	SR алгоритам	0	0%

Табела 1. Резултат поређења SR алгоритма и програма GATE

5.2. Експерименти са *eps* вредношћу

Параметар *eps* се користи у SR алгортиму за дозвољено одступање од CW вредности и представља релативан број карактера у линији за колико линија текста у текстуалном параграфу може да се разликује од просечне дужине линије. Подразумевана вредност овог параметра је 0.1, што је установљено емпиријски. Покушали смо да променимо ову вредност и видимо да ли и како утиче на ефикаснот алгоритма.

На основу дизајна SR алгоритма јасно је да повећање *eps* вредности доводи до веће толеранције према дужини линије, па ће бити мање случајева у којиме ће UL линија бити погрешно протумачена као или уметнути објекат (и обрисана из текста) или крај параграфа (и тиме подељена на два параграфа). Са друге стране, неки веома дугачки налсови ће бити препознати као делови текстуалних параграфа, па је могуће да се у излазном тексту појаве секвенце текста са некоректном синтаксом. Смањење епс вредности има супротан ефекат.

За тестирање смо користили вредности 0.3, 0.1 и 0.05. Тестирање је изведено на неколико различитих докумената, са различитим реторичким структурама. За обраду текстова попут романа и новела, где је главни текст записан у једној колони и без уметнутих објеката, *eps* вредност је боље повећати на 0.3, посебно ако наслови нису чести у тексту, и релативно су краћи од остатка текста. За обраду текстова попут научних чланака и енциклопедија, *eps* вредност је боље смањити, у зависности од структуре текста. Ако је текст записан у две или више колона, разилке у броју карактера између линија су мање, па је боље узети вредност 0.05.

6. Закључак и правци даљег развоја

SR алгоритам и његова имплементација имају практичну примену у лингвистички оријентисаним истраживањима и обради текста. Истраживачи могу имати јасан бенефит, с обзиром да примена SR алгоритма значајно смањује време потребно за припрему корпуса текста. Иако има одређене недостатке, посебно када се примени на текстове са великим бројем уметнутих објеката, и даље га је могуће користити за коректну конверзију делова текста, остављајући тако мањи део посла истраживачима.

Следећи правци развоја обухватају

имплементацију алгоритма у окружењу који је више прилагођен кориснику, као и његову интеграцију са постојећим језичким ресурсима, попут електронских речника, лексикона и граматика. На тај начин ће истраживачима у области обраде природних језика бити омогућено једноставније процесирање PDF докумената, кроз графичка сучеља. Један од таквих пројеката је *PDF корпус*, који се тренутно развија као део овог истраживања.

Захвалност

Овај рад је резултат истраживања у оквиру пројекта 178006 финансираног од стране Министарства за просвету, науку и технолошки развој Републике Србије.

7. Литература

Baumgartner, R., Flesca, S., Gottlob, G. (2001) Visual Web Information Extraction with Lixto. In *Proceedings of the Conference on Very Large Databases* (VLDB).

Cunningham, H., Maynard, D. Bontcheva, K., Tablan, V. (2002) GATE: A framework and graphical development environment for robust NLP tools and applications. In *Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (ACL'02)*, Philadelphia, July 2002.

Kushmerick, N. (2000) Wrapper induction: Efficiency and expressiveness. *Artificial Intelligence*, 118(1-2): 15–68, 2000.

Liu, L., Pu, C., Han, W. (2000) XWRAP: An XML-enabledWrapper Construction System for Web Information Sources. In *Intern. Conference on Data Engineering (ICDE)*, pages 611–621.

Muslea, I., Minton, S., Knoblock, C. (1999) A hierarchical approach to wrapper induction. In O. Etzioni, J. P. Mueller, and J. M. Bradshaw, editors, *Proceedings of the Third International Conference on Autonomous Agents (Agents'99)*, pages 190–197, Seattle, WA, USA, 1999. ACM Press.

Paumier, S. (2011) *Unitex 2.1 User Manual*, http://www-igm.univmlv.fr/~unitex/UnitexManual2.1.pdf, Université de Marne-la-Vallée.

Silberztein, Max. (2003) *NooJ manual*. http://www.nooj4nlp.net.