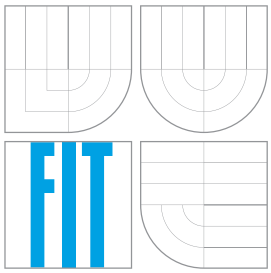
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

EXTRAKCE METADAT Z VĚDECKÝCH ČLÁNKŮ

METADATA EXTRACTION FROM SCIENTIFIC PAPERS

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

TOMÁŠ LOKAJ

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. LUBOMÍR OTRUSINA

BRNO 2010

Abstrakt

Tato práce se zabývá extrakcí metadat z vědeckých článků. Je zde obecně popsán problém extrakce informací se zaměřením na zpracování textových dokumentů. Dále je představen autorem vytvořený program `clanky2meta.py` určený k vyhledávání potřebných informací ve vědeckých publikacích. V závěru práce je provedeno srovnání toho programu s jinými systémy, především se systémem `CiteSeer`^{X}.

Abstract

This work deals with the Metadata Extraction from Scientific Papers. There is generally described issue of information extraction, focusing on the processing of text documents. There is also presented programme `clanky2meta.py` designed to search for relevant information in scientific publication, created by the author. At the end of this work is a comparsion of systems dealing with the same issue, especially with the `CiteSeer`^{X} system.

Klíčová slova

extrakce informací, dolování dat, dolování v textu, metainformace, metadata, vědecký článek, citeseerx, psyco.

Keywords

information extraction, data mining, text mining, metainformation, metadata, scientific paper, citeseerx, psyco.

Citace

Tomáš Lokaj: Extrakce metadat z vědeckých článků, bakalářská práce, Brno, FIT VUT v Brně, 2010

Extrakce metadat z vědeckých článků

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Lubomíra Otrusiny.

.....
Tomáš Lokaj
17. května 2010

Poděkování

Chtěl bych poděkovat svému vedoucím Ing. Lubomíru Otrusinovi za vedení a odborné rady při tvorbě této práce.

Obsah

Seznam obrázků

Seznam tabulek

Kapitola 1

Úvod

V dnešní moderní době je trendem vytvářet většinu publikací v elektronické podobě nebo ji alespoň později do této formy převést. Díky globální síti Internet je možné tato díla nekontrolovaně šířit po celém světě. V záplavě tak nepřeberného množství dat je proto žádoucí tyto informace klasifikovat a třídit, aby uživatelé mohli získat ucelený obraz o daném tématu.

Touto problematikou se zabývá *extrakce informací*, která umožňuje z informačních zdrojů získat potřebná data. Ta je poté třeba analyzovat, ověřit a uložit do některé z *digitálních knihoven*. Tyto knihovny slouží jako rozcestník a zdroj informací pro vědce na celém světě.

Cílem této bakalářské práce bylo vytvořit prostředek, který bude ve vědeckých článcích vyhledávat určitá metadata a srovnat jeho úspěšnost s jinými systémy. K tomuto účelu byl vytvořen program `clanky2meta.py`, jehož implementaci, použitým metodám a dosaženým výsledkům je věnována převážná část této práce. V závěru je shrnuta jeho úspěšnost a možné využití.

V druhé kapitole (2) je obecně popsán problém *extrakce informací* společně s pojmy jako je *vědecký článek* nebo *metadata*. Třetí kapitola (3) se zabývá vytvořeným programem `clanky2meta.py`, popisuje jeho implementaci a metody použité pro hledání dat. Je zde také představen systém *ReReSearch* a využití programu `clanky2meta.py` v tomto systému. Následující dvě kapitoly srovnávají program s jinými systémy (kapitola 4) a shrnují jeho dosažené výsledky a nedostatky (kapitola 5).

Kapitola 2

Extrakce metadat z vědeckých článků

Tato kapitola se zabývá problémem **extrakce metadat z vědeckých článku**, vysvětluje související pojmy a obecně popisuje *extrakci informací*.

2.1 Co jsou to metadata?

Metadata[1] neboli **metainformace** jsou „data o datech“, která mají určitou strukturu. Pomáhají pochopit a interpretovat význam popisovaných dat v konkrétním kontextu. Kromě samotného významu dat obsahují také informace o jejich formátu, původu a možných hodnotách [10].

Příkladem metadat mohou být dodatečné informace u digitální fotografie, jejíž metadata popisují velikost obrázku, datum a čas pořízení, typ fotoaparátu atd. Také hudební skladby ve formátu MP3 mají doplňující údaje (název, interpret, žánr apod.) uloženy v metadatech. Jedná se o tzv. ID3 tagy. Podobně je při extrakci metadat z vědeckých článků myšleno nalezení a vytvoření informací o jejich názvu, autorech, obsahu apod.

2.2 Vědecký článek

Vědecký článek se od ostatních typů článků liší především svou celkovou strukturou a nutností dodržovat určité typografické konvence. Tyto nároky většinou pocházejí z požadavků odborných periodik, pro něž je článek určen. Důraz je kladen na spisovný jazyk a pevnou strukturu jednotlivých částí, mezi nimiž by neměly chybět následující [10]:

- krátký a výstižný **název**
- jména **autorů**
- dodatečné informace o autorech (**e-maily**, **adresy**, atd.)
- srozumitelný **abstrakt**
- **klíčová slova**
- úvod

[1] *Informace jsou tzv. data, která mají sémantiku (význam).* [1]

- vlastní obsah článku, materiály, výsledky
- závěr
- poděkování
- zdroje a **použitá literatura**

Typografická pravidla například určují, jaký má mít článek formát (např. A4), okraje, velikost a tvar písma, nadpisů a vzorců, číslování a formy jednotlivých sekcí, rovnic, obrázků a tabulek, a další [11]. Pro vědecký článek je také typické používání různých termínu, které jsou charakteristické pro obor, do kterého daný článek spadá. Tyto termíny dodávají textu odbornost a jednoznačnost [3].

Velmi důležitou součástí vědeckých publikací jsou *bibliografické citace* odkazující na jiná díla. Bibliografické citace informují o článcích, ze kterých bylo čerpáno, nebo se zabývají podobným problémem. Jedná se o velice efektivní zdroj informací k získávání a shromažďování dalších článků, které mohou svým obsahem obohatit jak čtenáře, tak také například databáze digitálních knihoven.

Bohužel ještě neexistuje jednotný standard, který by přesně určoval formu zápisu jednotlivých informací, jako například autorů článku či použité literatury. Autoři mohou mít svá jména různě zkrácena a bibliografické citace mají zase mnoho norem, podle kterých mohou být zapsány. To všechno znesnadňuje problém, kterým se zabývá tato bakalářská práce, tj. *extrakci metainformací.*

2.3 Extrakce informací

Extrakci informací lze popsat jako činnost, při které jsou získávána podrobná data pro předem specifikované požadované informace. Tyto informace jsou poté strukturovaně reprezentovány v podobě **metainformací** [2]. Jedná se o proces podobný získávání znalostí z databází, avšak narozdíl od dat v databázi nemají tyto dokumenty pevnou strukturu v podobě tabulek či záznamů. Jsou psány přirozeným jazykem, formátovány podle potřeby autora či jednoho z obrovského množství vzorů. Jedná se o velmi aktuální téma, protože v dnešní době většina publikací vzniká v elektronické podobě a jejich šíření pomocí Internetu nic nebrání. Je proto žádoucí tyto publikace automaticky třídit a klasifikovat.

2.3.1 Historie extrakce informací

Automatická extrakce informací se rozšířila v 80. letech 20. století. Zpočátku se zpracovávaly hlavně novinové články. Hnacím motorem však byl obrovský nárust elektronických článků a konference **Message Understanding Conference** organizovaná agenturou *DARPA*[2], na které různé výzkumné týmy pracují na stanoveném tématu. Jedním z těchto témat byla i extrakce informací. Systémy byly nejprve vytvářeny ručně, později se začalo využívat metod *strojového učení*[3] [5].

[2] Americká obranná agentura `http://www.darpa.mil/`

[3] Strojové učení se zabývá technikami, které umožní počítači přizpůsobit se okolnímu prostředí.

2.3.2 Dolování dat

Extrakce informací patří do oblasti **dolování dat**[4], což je proces hledání informací a znalostí ve velkém objemu informací. Dolování dat využívá mnoho různých metod, postupů a algoritmů, které umožní odhalit a plně využít vztahy, závislosti a vzory ukryté v datech. Mezi tyto metody například patří [6]:

- rozhodovací stromy
- neuronové sítě
- nejbližší soused
- klasifikace
- vzorkování
- regrese
- analýza asociací

2.3.3 Dolování v textech

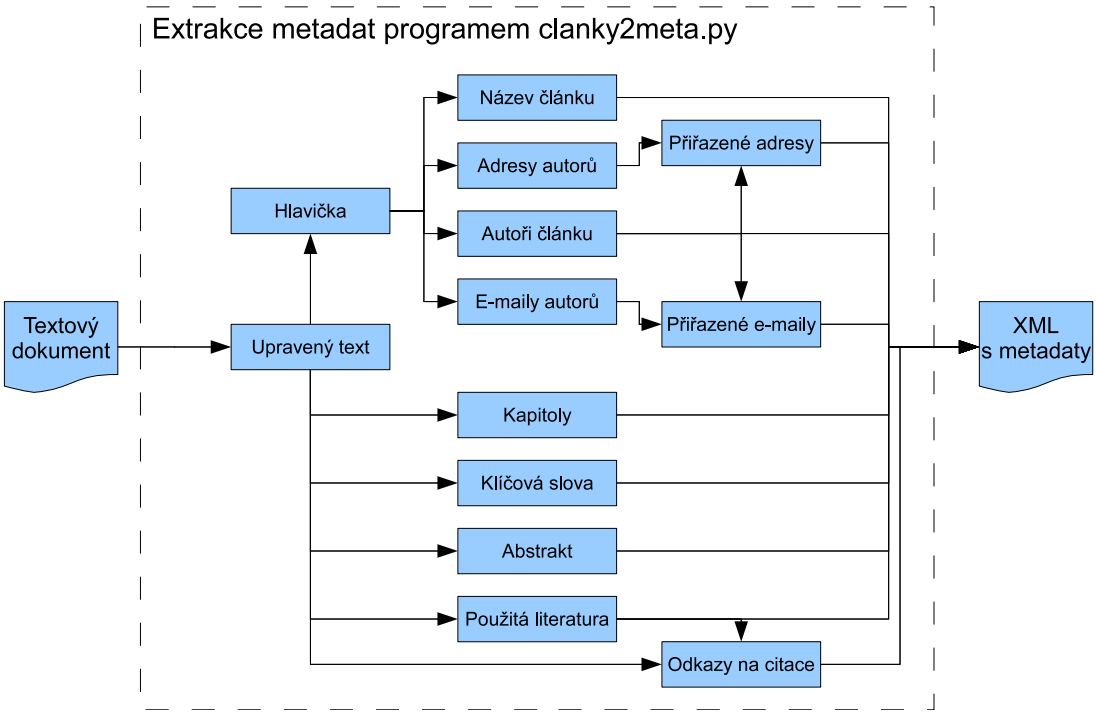
Dolování v textech hledá a analyzuje informace v textových dokumentech, jenž by mohly být zajímavé pro uživatele a nejsou nijak předem známé [4]. Podle nedávných studií má v dnešní době dolování v textech velký potenciál pro komerční využití, protože například mnoho společnosti má zhruba 80% svých informací v různých textových dokumentech [8]. Dolování v textech je také daleko komplexnější než původní dolování dat, protože většinou zpracovává nestrukturovaná a nejasná data.

[4] anglicky Data Mining

Kapitola 3

Program clanky2meta.py

Praktickou částí této bakalářské práce bylo vytvoření programu pro extrakci metainformací z vědeckých článků. Tento program byl nazván `clanky2meta.py` a následující kapitola se zabývá jeho implementací. Podrobně jsou zde popsány metody hledání jednotlivých informací a použité technologie.



Obrázek 3.1: Schéma programu clanky2meta.py

Program `clanky2meta.py` je napsán v programovacím jazyce *Python* ve verzi 2.6.2. Jak ukazuje obrázek 3.1, vstupními daty jsou textové soubory, které vznikly převodem z jiných

formátů. Tyto formáty jsou nejčastěji `pdf`[1] nebo `ps`[2]. Pomocí různých vzorů a *regulárních výrazů* jsou poté vyhledávány metainformace, mezi které patří název článku, autoři článku a jejich adresy a e-maily, abstrakt a kapitoly článku, použitá literatura a její citace a také klíčová slova. Výstupem programu je soubor v *XML* formátu, který obsahuje získaná metadata.

3.1 Použité technologie

Tato podkapitola popisuje technologie, které byly použity v programu `clanky2meta.py` k extraci metadat a zlepšení výkonu.

3.1.1 Python

Jazyk `Python` je objektově orientovaný dynamický interpretovaný programovací jazyk, který svým charakterem bývá řazen i k jazykům scriptovacím. Součástí `Pythonu` je také množství modulů, které umožňují jeho všestranné použití od jednoduchých konzolových scriptů, přes práci se službami internetu, až po rozsáhlé aplikace s grafickým uživatelským rozhraním. Python má srozumitelnou, dobře čitelnou syntaxi. V čem však tento jazyk zaostává oproti kompilovaným jazykům (`C`, `C++`, `Java` atd.) je rychlost. Jeho interpret však lze lehce rozšířit o moduly psané v jazyce `C` nebo `C++`, což pomáhá k celkové optimalizaci výsledného programu. `Python` je multiplatformní jazyk, který už dnes bývá základní součástí většiny běžných Linuxových distribucí.

3.1.2 Psyco

Program pro extrakci metainformací je určen pro zpracování velkého množství dat, proto je nutné, aby zpracování jednotlivých článků netrvalo příliš dlouho. Při vyhledávání informací se používá výhradně *regulárních výrazů*, které jsou samy o sobě dosti pomalé. Také se často prohledávají slovníky, např. při identifikaci křestních jmen autorů. Bylo tedy potřeba najít způsob, jak tyto operace zoptimalizovat. To se částečně podařilo s použitím knihovny `Psyco`.

	Celkový čas běhu	**Zpracování jednoho souboru**
bez Psyco	45 min 1 s	1,81 s
s Psyco	28 min 54 s	1,15 s

Tabulka 3.1: Srování dob trvání extrakcí při použití knihovny Psyco

Knihovna `Psyco` pracuje na principu podobnému *JIT* [3] kompilátoru. Během běhu programu je každý blok zkompilován do více binárních verzí, které jsou optimalizovány pro různé situace. Autoři na své stránce [15] uvádějí, že běžné zrychlení bývá typicky čtyřnásobné. Nevýhodou tohoto řešení však je, že program při svém běhu zabírá

[1] Portable Document Format - univerzální formát souborů vhodný k platformově a softwarově nezávislému přenosu

[2] PostScript - programovací jazyk optimalizovaný pro tisk grafiky či textu na různá média (papír, film, CRT monitor apod.) [12]

[3] Just In Time označuje metodu překladu, která urychluje běh programu interpretovaných jazyků jeho překládáním do strojového kódu během jeho provádění.

více paměti. Zrychlení bylo testováno programem `clanky2meta.py` na **1 495** souborech, výsledky testování zobrazuje tabulka 3.1.

Autorem této optimalizační knihovny je Armin Rigo a dostupná je na stránce `http://psyco.sourceforge.net/`.

3.1.3 XML

`eXtensible Markup Language` je velice rozšířený značkovací jazyk, který je jazykem odvozeným z rodiny jazyků *SGML*[4]. Pomocí `XML` lze snadno ukládat různé informace tak, aby je bylo možno dále strojově zpracovávat. Existuje mnoho nástrojů pro zpracování `XML` dokumentů a podpory se jim také dostává v řadě programovacích jazyků. Díky jejich textové podobě jsou tyto dokumenty člověkem snadno čitelné a také upravovatelné i obyčejným textovým editorem. `XML` byl vyvinut a standardizován konzorciem **W3C**.

3.1.4 Regulární výrazy

Jedná se o velice užitečnou pomůcku při práci s textem. `Regulární výraz`[5] je sám o sobě řetězec, který umožňuje pomocí konečné množiny znaků definovat nekonečnou množinu řetězců. Používá se například ve skriptovacích jazycích k úpravám a vyhledávání v textu. Je-li třeba nalézt určitou část textu, ale není přesně známá její podoba, lze pomocí `regulárního výrazu` definovat její přibližný popis. Poté jsou nalezeny části textu, které odpovídají popisu v zadaném `regulárním výrazu`. Používají se tak například k vyhledání e-mailových adres nebo parsování zdrojových kódů.

3.2 Extrakce metadat

Lidský mozek dokáže při pohledu na článek v jeho původní podobě rozeznat jednotlivé metainformace. Kromě přirozené inteligence mu k tomu pomáhá také styl formátování dokumentu, kdy jsou použity různé druhy fontů a zvýraznění. Program má k dispozici pouze prostý text bez těchto dodatečných informací. Jedinou pomůckou mu mohou být pouze seskupení bílých znaků, jako jsou mezery či nový řádek.

Jak již bylo zmíněno dříve, největším pomocníkem při získávání potřebných metainformací jsou **regulární výrazy**. Pomocí nich program prochází různé části článku a v každé z nich vyhledává určité informace. Například v *hlavičce* bývají údaje o autorech a názvu článku. Dalším pomocným prvkem jsou *slovníky*, které jsem převzal z diplomové práce Tomáše Petránka [4]. Ty pomáhají zjistit, zda-li je určité slovo křestním jménem či názvem státu apod.

3.2.1 Úprava vstupního textu

Před samotným vyhledáváním je nejprve nutné upravit vstupní text. Ten totiž může vlivem špatného převodu ze svého původního formátu obsahovat rozházené odstavce, shluky špatných znaků, které reprezentují obrázky či vzorečky, a problémy vycházející z použití diakritiky v různých jazycích (např. přehlásky v němčině nebo háčky a čárky v češtině). Příliš mnoho špatných znaků, tzn. znaky nereprezentující písmena, číslice, interpunkční znaménka

[4] Standard Generalized Markup Language je složitý značkovací metajazyk, který umožňuje definovat podmnožiny jednodušších značkovacích jazyků.

[5] anglicky Regular Expression

a další znaky běžné v textu, způsobuje zpomalení procesu extrakce, špatné výsledky, někdy dokonce pozastavení se celého programu. Je-li v článku více než 35% znaků špatných, program jej prohlásí za nepoužitelný a dále jej nezpracovává.

Odstranění diakritiky z textu

Kvůli používání slovníků, které obsahují slova bez diakritiky, je nutné z textu diakritiku odstranit. Jinak by například nefungovalo vyhledávání a identifikace křestních jmen a s ním spojená extrakce autorů. Diakritika může také během převodu z původního formátu na text způsobit znehodnocení slova. To dělají například přehlásky, kdy se místo `Universität Tübingen` v textu vyskytuje `Universita"t Thu"bingen`. Příklad znehodnocení autorova jména je ukázán v tabulce 5.4 v kapitole 5.2.

3.2.2 Vyříznutí hlavičky a její úpravy

Hlavička je část článku, která bývá na jeho začátku. Zpravidla obsahuje název díla a jeho autory společně s adresami a e-maily a většinou končí před *abstraktem článku*. Její vyříznutí urychlí vyhledávání zmíněných informací.

```
Anup K. Sen
Indian Institute of Management Calcutta
Joka, D. H. Road
Calcutta 700 027, INDIA
sen@iimcal.ac.in

Amitava Bagchi
School of Management
University of Texas at Dallas
Richardson, Texas 75083
abagchi@utdallas.edu
```

Obrázek 3.2: Hlavička formatovaná do sloupců

Hlavičky se běžně vyskytují formátované do sloupců (obrázek 3.2). To znemožňuje správné rozdělení autorů a také to způsobuje zpřeházení částí adres. Program se proto snaží takovou hlavičku upravit a naskládat za sebe informace patřící k sobě. To také pomůže při přiřazování adresy a e-mailu k danému autorovi (obrázek 3.3).

```
Anup K. Sen, Indian Institute of Management Calcutta, Joka, D. H. Road,
Calcutta 700 027, INDIA, sen@iimcal.ac.in
Amitava Bagchi, School of Management, University of Texas at Dallas,
Richardson, Texas 75083, abagchi@utdallas.edu
```

Obrázek 3.3: Upravená hlavička

3.2.3 Extrakce názvu článku

Název článku je pravděpodobně nejdůležitější metainformace získávaná z článků. Jeho nalezení však není tak triviální, jak by se mohlo na první pohled zdát. Zpravidla jej lze nalézt na samém začátku textu. Mohou se před ním však vyskytovat i další informace, jako například informace o konferenci, ze které článek pochází, datum jeho vydání, copyright, URL jeho původního umístění apod.

K vyhledání názvu článku bylo zkoušeno několik postupů. Díky lepší úspěšnosti byl nakonec zvolen takový, při kterém jsou postupně procházeny všechny řádky hlavičky

a ty jsou podrobeny další analýze. Ve většině případů by měl název článku být řetězec psaný celý na jednom řádku. Někdy však bývá autorem napsán na řádků více. To může také nastat vlivem špatného převodu článku z původního formátu na text. Pokud je tato situace poznatelná (řádek končí předložkou, spojkou či dvojtečkou), program se pokusí název spojit. Dalším nežádoucím problémem vzniklým převodem může být, když se na řádek s názvem článku připojí nějaká další informace, např. jeho autor. V tomto případě pomůže, když je název psán velkými písmeny, jak zobrazuje obrázek 3.4. Potom lze jednoduše pomocí regulárního výrazu oddělit tyto údaje od sebe. Není-li název celý napsán velkými písmeny a nastane tato situace, je na řádku vyhledána informace o autorech. V případě jejího nalezení je řádek opět rozdělen.

```
EFFICIENT INFORMATION DISCOVERY AND RETRIEVAL IN WIRELESS AD HOC NETWORKS by Zhao Cheng
```

Obrázek 3.4: Název článku spojený s další metainformací

Po případných úpravách všech řádků hlavičky se řádky znovu postupně procházejí. Každý řádek je analyzován a pokud projde kontrolou, je přidán do seznamu možných názvů článku. Kontrola spočívá ve vyhledávání specifických řetězců či potenciálních jmen autorů na řádku. Pokud je takový řetězec nalezen, řádek už nemůže být názvem článku. Specifické řetězce jsou reprezentovány regulárním výrazem a vždy přestavují část nějaké jiné informace, než je hledaný název článku. Jejich příklady zobrazuje tabulka 3.2.

Regulární výraz	Význam
`ISSN|[0-9]{4}-[0-9]{4}`	jedná se o ISSN publikace
`in proc[.]|in proceedings|proceedings of`	článek pochází z konference
`vol(ume)?[.]? *[0-9]+`	číslo sborníku
`(pages?|p[.]) [0-9]+`	informace o pozici článku ve sborníku

Tabulka 3.2: Příklady některých specifických řetězců

Nyní je k dispozici seznam potenciálních názvů. Jako název článku je určen ten, který se v textu vyskytuje nejdříve.

3.2.4 Extrakce jmen autorů

Dalšími velmi důležitými informacemi, které lze z dokumentů získat, jsou jména jejich **autorů**. Většinou jde o jednoho nebo více autorů, kteří se rozhodujícím způsobem podíleli na tvorbě publikace. Bohužel také tato extrakce patří k nejsložitějším a tím pádem i nejméně úspěšným, se kterými se musí program vypořádat.

Jak již bylo zmíněno na počátku této kapitoly, program má k dispozici pouze čistý text, ve kterém nemusí být jména autorů od sebe nijak oddělena, protože jakékoliv zvýraznění není k dispozici. Nastávají tedy případy, kdy je při výskytu autora majícího více než dvě jména, omylem jedno z nich přiřazeno k jinému autorovi. Obrázek 3.5 znázorňuje situaci, kdy jsou v původním textu od sebe jednotliví autoři odděleni tabulátorem. Při převodu do textové podoby dochází k chybě, kdy je tabulátor považován za mezeru, a jména jsou naskládána hned za sebou. Mezi jmény tedy není žádný výrazný oddělovač, takže algoritmus pro nalezení jmen autorů omylem přiřadí část jména jednoho autora k autorovi druhému. Místo autorů David Robins, Jonathan Gibbs, Devorah DeLeon Penka a Nate

DeBry jsou nalezeni autoří David Robins, Jonathan Gibbs, Devorah DeLeon a Penka Nate DeBry.

```
David Robins    Jonathan Gibbs    Devorah DeLeon Penka    Nate DeBry
```

Obrázek 3.5: Možná chyba při hledání jmen autorů s prostředními jmény

Vyhledávání pomocí slovníku křestních jmen

První používanou metodou pro nalezení jmen autorů je metoda, která využívá slovníku křestních jmen. Tento slovník obsahuje přes 22 000 položek. Program postupně prochází všechna slova v hlavičce a pomocí algoritmu *binárního vyhledávání* (viz tabulka 3.3) zjišťuje, jestli se jedná o křestní jméno. V případně pozitivního nálezu jsou okolo křestního jména vyhledávány následující varianty zápisu autorova jména:

- PŘÍJMENÍ Jméno
- Jméno Příjmení Příjmení+
- Jméno Příjmení
- J. Příjmení
- Přijmení J.

U posledních dvou možností se už nepoužívá slovník křestních jmen, ale obdobně se po slovech vyhledávají iniciály, které ve vyhledávacím algoritmu zastupují křestní jméno. Použitý regulární výraz pro *Příjmení* obsažený ve výčtu nevyhledává jen celá slova začínající velkým písmenem, což je u jmen podmínka, ale také iniciály, prostřední jména (von, van der, de atd.) a případně i jeho pozici v rodové linii (I., II., III., IV. atd.).

```
def binarySearch(seznam, hodnota, vlevo, vpravo):
    while vlevo <= vpravo:
          stred = (vpravo + vlevo) / 2
          if seznam[stred] == hodnota:
             return True
          if hodnota < seznam[stred]:
             vpravo = stred - 1
          else:
             vlevo = stred + 1
    return False
```

Tabulka 3.3: Použitý algoritmus binárního vyhledávání

Vyhledávání pomocí údajů z e-mailové adresy

Před samotnou extrakcí jmen autorů článku jsou již známy e-mailové adresy, které se nachází v hlavičce. Nabízí se tedy možnost vyhledání dalších jmen autorů pomocí první části e-mailové adresy, která většinou obsahuje část autorova jména.

Vyhledání pomocí již nalezených jmen

Při každém nalezení autorova jména je jeho jméno vyjmuto z textu a nahrazeno značkou `%X%`, kde `X` je pořadí, ve kterém bylo autorovo jméno vyextrahováno. Tohoto značení se využívá také později při přiřazování adres. Pokud se tedy nepodařilo nalézt celé jméno některého autora pomocí předchozích metod a jsou extrahována jména autorů okolo něj, dá se toto jméno autora doplnit. Podmínkou je, aby od sebe byla jednotlivá jména autorů řádně oddělena například čárkou. Tuto situaci znázorňuje obrázek 3.6.

```
%6%, Andreas Vogel, %8% and %9%
```

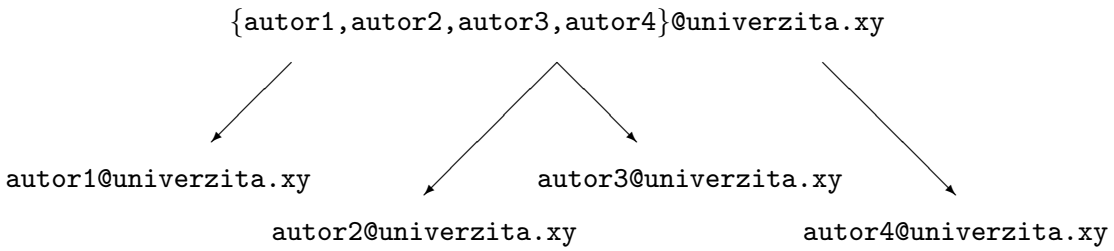
Obrázek 3.6: Dohledání neextrahovaných jmen autorů

Další použité slovníky

Program kromě slovníku křestních jmen využívá také další slovníky. Jedná se například o seznamy slov, která určitě nereprezentují příjmení. Během extrakce jmen autorů je proto každé potenciální příjmení kontrolováno tímto slovníkem. Další slovníky obsahují například jména států, měst a slova objevující se v adresách. Toho se zase využívá při vyhledávání adres (viz dále v kapitole 3.2.6).

3.2.5 Extrakce e-mailových adres

V hlavičce se také vyskytují informace o **e-mailových adresách**. Pro e-mail existuje striktně daný tvar, takže jeho nalezení nemusí být složitou záležitostí a měl by postačit jednoduchý regulární výraz. Vědecké články však mnohdy mívají více autorů, kteří pocházejí z jednoho pracoviště či univerzity. Často proto své e-mailové adresy spojují do jednoho řetězce, který uvádějí ve své práci. Nalezení takového řetězce už vyžaduje složitější vzor a je nutné jej potom rozložit na jednotlivé e-maily. Příklad složeného e-mailu a jeho rozdělení znázorňuje obrázek 3.7.



Obrázek 3.7: Rozdělení složeného e-mailu

Přiřazení e-mailu k autorovi

Nalezené e-mailové adresy je třeba přiřadit ke správnému autorovi. Přiřazení se provádí rozdělením celého jména autora na všechny možné podřetězce. Tyto podřetězce se následně vyhledávají v části e-mailu před zavináčem a k autorovi je přiřazena ta adresa, která obsahuje největší počet jeho podřetězců.

Často se také e-maily vyskytují ve tvaru `firstName.lastName@pracoviste.xy`. V tomto případě je jejich přiřazení jednoznačné, resp. ke každému autorovi je jeho e-mail vytvořen podle zadaného předpisu.

3.2.6 Extrakce adres

Posledními zbývajícími informacemi, které se vyskytují v hlavičce, jsou **adresy autorů**. Ne vždy je se jedná o klasickou adresu obsahující ulici, město, PSČ apod. Může to být například jen název univerzity či oddělení a případné další informace, které jsou v textu uvedeny na stejném místě.

Jejich vyhledání spočívá v rozdělení hlavičky na řádky a následném procházení jednotlivých slov. Tato slova jsou následně vyhledávána ve slovnících obsahujících výrazy, které charakterizují část adresy (města, státy, instituce atd.). Při úspěšném nálezu je řádek ořezán o případné jiné informace, např. e-mailové adresy a značky autorů, a přidán do seznamu možných adres.

Přiřazení adresy k autorovi

Přiřazení adresy ke správnému autorovi se provádí dvěma způsoby. První z nich využívá značek, které někdy bývají za jmény autorů a před začátkem adresy. Pomocí nich lze snadno určit, kterému autorovi patří daná adresa. Tyto značky bývají nejčastěji čísla, křížek nebo hvězdička. Jeden z takový případů znázorňuje obrázek 3.8. V některých článcích se však používají zvláštní symboly, jako například †. Ty se bohužel ztratí či znehodnotí během převodu z původního formátu do textové podoby.

```
Anup K. Sen*    Amitava Bagchi**    David Robins***

* Indian Institute of Management Calcutta, Joka, D. H. Road, Calcutta 700 027, INDIA
** School of Management, University of Texas at Dallas, Richardson, Texas 75083
*** Computer Science Department, Washington University, St. Louis, Missouri 63130
```

Obrázek 3.8: Adresy s označením u autorů

Nejsou-li značky k dispozici, jsou adresy přirovnány podle pozice v textu vzhledem k autorům. V ideálním případě jsou autoři na jednom řádku společně s adresou (viz obrázek 3.9). Potom lze pomocí čísel ve značení přiřadit adresy ke správným autorům.

```
%4%, and %5% USC Institute for Creative Technologie, 13274 Fiji Way, Suite 600,
Marina del Rey, CA 90292-7008
```

Obrázek 3.9: Adresy bez označení u autorů

3.2.7 Extrakce abstraktu

Nedílnou součástí každého vědeckého článku je jeho **Abstrakt**. Jedná se o odstavec, který zpravidla následuje hned za hlavičkou. Bývá v něm stručně popsán obsah článku, což pomáhá čtenáři ve vyhledávání článků s podobným zaměřením.

Abstrakt musí být vždy uvozen slovem `Abstract`. Program tedy vyhledává takovou část článku, ve které se za tímto uvozujícím slovem vyskytuje odstavec textu. Právě tento odstavec je abstraktem článku. Chybou v převodu článku do textové podoby se může stát, že je abstrakt rozdělen nechtěnou vertikální mezerou. Abstrakt by měl být vždy řádně ukončen tečkou. Program se proto přednostně snaží nalézt odstavec i s případnou mezerou, ale správně zakončený.

3.2.8 Extrakce názvů kapitol

Každý vědecký článek bývá rozdělen do několika samostatných **kapitol**. K lepšímu pochopení jeho smyslu a obsahu se kromě abstraktu hodí znát i názvy jeho kapitol, aby měl čtenář představu, jakou cestu jeho autor k popsání problému zvolil a čím vším se přesně zabývá.

Extrakce z obsahu

Lepším zdrojem k vyhledání názvů kapitol jsou dokumenty, které mají na začátku vypsán svůj **obsah**, což není nic jiného, než seznam kapitol. Samo se potom nabízí získat jejich názvy právě odtud. V anglicky psané literatuře bývá obsah značen jako *Table of Contents* nebo jenom *Contents*. V článku je vyhledána sekce uvozená právě takovými hesly a z ní jsou následně vyextrahovány názvy kapitol. Tento postup je nejúspěšnější a o správnosti získaných dat není žádných pochyb. Pokud ovšem nedošlo během převodu z původního formátu na text ke znehodnocení informací v obsahu kapitol. Ten totiž bývá psán stylem, kdy po názvu kapitoly následuje několik teček a číslo stránky. Pravděpodobně právě kvůli těmto tečkám dochází někdy k rozházení, rozdělení či přeskupení názvů kapitol. Aby se předešlo nesprávným výsledkům, snaží se program o zpracování pouze nepoškozeného obsahu. Další metodou, která se používá v případě nenalezení obsahu nebo při jeho poškození, je využití *„seznamu běžných názvů kapitol“*.

Extrakce pomocí seznamu běžných názvů kapitol

Druhou metodou k vyhledání názvů článků je použití vytvořeného **seznamu běžných kapitol**. Tento seznam byl vytvořen a postupně doplňován během psaní a testování programu a při procházení různých článků. Zpočátku seznam obsahoval pouze následující názvy kapitol: *introduction, preliminaries, motivation, aknowledgments, references, bibliography, related work, future work, conclusion, conclusions, discussion, summary, result, results, conclusion and future work, conclusions and future work, experimental results, implementation, evaluation, example, background, statistics, methods*. Poté byla provedena testovací extrakce na cca 670 000 článků a byly zaznamenány ty názvy kapitol, které se vyskytly alespoň v jednom procentu případů. Tyto názvy jsou znázorněny v tabulce 3.4. **Seznam běžných kapitol** byl tedy rozšířen o tyto nalezené názvy kapitol.

Princip metody spočívá v tom, že jsou v textu vyhledávány názvy kapitol z výše zmíněného seznamu. Je-li některá z nich nalezena, zaznamená se. Během tohoto procesu je také zjištěn formát, kterým jsou běžné kapitoly zapsány. Je brán zřetel hlavně na uvození

názvu, které bývá buďto **arabskou** nebo **římskou číslicí**, dále na odřádkování před a za názvem kapitoly, na tečky za uvozující číslicí a na velká písmena. Následuje postupné počítání čísel od jedničky. Ke každému číslu je vyhledán řetězec (řádek), který vyhovuje danému formátu. Počítání končí, pokud ke třem po sobě jdoucím číslům nebyl nalezen žádný název a nezbývají už žádné zaznamenané běžné kapitoly.

Název kapitoly	Výskyt	Název kapitoly	Výskyt
Introduction	47%	*Overview*	2%
References	17%	*Evaluation*	2%
Conclusion	13%	*Preliminaries*	2%
Conclusions	12%	*Future Work*	2%
Related Work	10%	*Conclusion and Future Work*	2%
Discussion	7%	*Concluding Remarks*	2%
Results	7%	*Methods*	2%
Summary	4%	*Motivation*	2%
Experimental Results	4%	*Method*	1%
Acknowledgments	4%	*Methodology*	1%
Background	4%	*Example*	1%
Acknowledgements	3%	*Examples*	1%
Conclusions and Future Work	3%	*Results and Discussion*	1%
Bibliography	3%	*Applications*	1%
Experiments	3%	*Previous Work*	1%
Implementation	3%	*Analysis*	1%

Tabulka 3.4: Běžné názvy kapitol

Hledání ničím neuvozených kapitol

Ničím neuvozená kapitola nemá před svým názvem žádnou číslici. To poněkud komplikuje vyhledání jejího názvu, protože se daleko rozšířilo spektrum vyhledaných řetězců. Opět je zde používáno seznamu běžných kapitol a je vyhledáván formát jejich zapsání. Proces je tedy obdobný jako v předchozím případě, ale nepoužívá se žádného počítadla a výsledky bývají hodně nepřesné.

3.2.9 Extrakce klíčových slov

Pomocí **klíčových slov** je možné označit témata článku, kterých se týká. To pomáhá při jeho zařazení a vyhledávání ve vyhledáváčích či katalogu. Pomocí nich lze také nalézt publikaci podobného zaměření.

Jejich extrakce je vcelku jednoduchá, protože bývají uvozeny souslovím `Key Words` nebo `Index Terms`, případně `Keywords`. Samotný výčet klíčových slov pokračuje na stejném či novém řádku. Jednotlivé termy jsou od sebe odděleny jednotným oddělovačem. Díky tomu lze rozpoznat, kdy klíčová slova končí a začíná jiná část textu.

3.2.10 Extrakce použité literatury a odkazů

Poslední částí vědeckých článků bývá **použitá literatura** obsahující **bibliografické citace** neboli také **reference**. Zde jsou uvedeny údaje o citovaných dílech, ze kterých autor článku

čerpal. To pomáhá při vyhledávání dalších děl s podobnou tématikou a umožňuje ověřit uvedené údaje.

Ve světě existuje mnoho norem, podle kterých se struktura bibliografických citací řídí. Taktéž pro každý typ publikace platí trochu rozdílná pravidla. Vždy by však mělo platit, že každá reference začíná na novém řádku. Na to se ovšem při extrakci nemůžeme spolehnout, protože během převodu na text dochází ke ztrátám či konverzi bílých znaků na jiné (například nový řádek na mezeru). Jejich extrakce tedy může být komplikovaná, je nutné je od sebe oddělit. Část článků s použitou literaturou bývá uvozena slovy `References` nebo `Bibliography`.

Oddělení referencí při existenci oddělovače

V případě, že jsou jednotlivé reference od sebe odděleny některým oddělovačem, bývá jejich rozdělení naštěstí jednoznačné i tehdy, když dojde k deformaci struktury textu. Příklady těchto oddělovačů zobrazuje obrázek 3.10.

Hranaté závorky

```
[A04] Kuipers, B. 1978. Modeling spatial knowledge. Cognitive Science 2:129-153.
[A05] Yeap, W.K. 1988. Towards a computational theory of cognitive maps.
```

Číslice s tečkou

```
1. Kuipers, B. 1978. Modeling spatial knowledge. Cognitive Science 2:129-153.
2. Yeap, W.K. 1988. Towards a computational theory of cognitive maps.
```

Římská číslice

```
I Kuipers, B. 1978. Modeling spatial knowledge. Cognitive Science 2:129-153.
II Yeap, W.K. 1988. Towards a computational theory of cognitive maps.
```

Číslice v závorce

```
(1) Kuipers, B. 1978. Modeling spatial knowledge. Cognitive Science 2:129-153.
(2) Yeap, W.K. 1988. Towards a computational theory of cognitive maps.
```

Obrázek 3.10: Příklady bibliografických citací s různými oddělovači

Oddělení referencí bez oddělovače

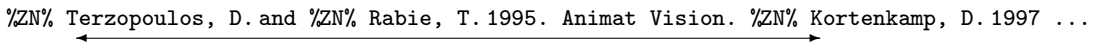
Častý je také výskyt bibliografických citací, které od sebe nejsou nijak jednoznačně odděleny. V článku jsou každá na novém řádku, ale na to se program nemůže spolehnout. Proto jsou do textu referencí dány značky, podle kterých jsou od sebe jednotlivé reference odděleny.

Každá bibliografická citace by měla obsahovat jméno autora, proto je opět využito slovníku křestních jmen a před každé nalezené křestní jméno je přidána značka. Také je vytvořena předloha, která charakterizuje počátek reference v případě, kdy jsou například křestní jména zkrácena pouze na iniciálu nebo začínají příjmením. Další značky jsou vloženy před každý text, který vyhovuje této předloze. Biliografická citace by tedy zpravidla měla obsahovat následující údaje:

- číselný údaj (rok vydání, strany apod.)

- značku na začátku
- tečku a za ní značku na konci

Podle těchto kritérií jsou poté od sebe odděleny jednotlivé reference, jak znázorňuje obrázek 3.11.

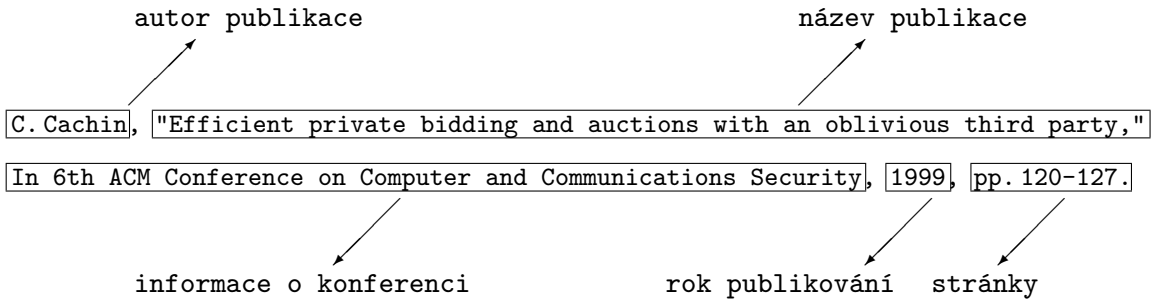


Obrázek 3.11: Ukázka oddělení referencí pomocí značek

Strukturování bibliografických citací

Aby bylo možné data z referencí lépe použít, musí se tyto citace strukturovat, tzn. oddělit od sebe jednotlivé údaje. Je zde však problém v jednotném formátu oddělování informací. Proto jsou nejprve nalezeny informace o počtu stran, roku vydání, konferencích, editorech a odkazech. Tyto entity jsou nahrazeny značkou pro oddělení. Poté se vyhledají jména autorů.

Jelikož všechny informace, které se zpravidla nacházejí na začátku bibliografické citace, jsou již vyjmuty, zbývá zde ještě název daného článku. Ten je vyhledán až po první oddělovač (buď vložená značka, čárka nebo tečka). Není-li název nalezen pomocí značky, nemusí být vždy úplný, poněvadž v sobě může obsahovat i čárky (případně tečky). Ve většině případů je to ale lepší řešení, protože jinak by se jako název mohl považovat celý zbytek bibliografické citace, což by znehodnotilo i ostatní informace.



Obrázek 3.12: Ukázka strukturování bibliografických citací

Nalezení odkazů na bibliografické citace

Odkazem na bibliografickou citaci je v tomto případě myšlena věta, která se odvolává na dílo uvedené v seznamu použité literatury. Vyhledání se provádí tak, že celý text článku je rozdělen na věty, ve kterých je následně vyhledáván odkaz. Tento odkaz bývá například oddělovač, kterým jsou odděleny jednotlivé reference. To záleží na normě, podle které je článek napsán. Program také vyhledává odkazy, které obsahují jméno autora a případně i rok vydání díla. Příklady odkazů na bibliografické citace jsou v obrázku 3.13.

```
Some of them (Briscoe and Carroll, 1993; Inui et al., 1997) treat information on
contexts, but the contextual information is derived only from a structure to which
the parser is trying to assign a likelihood value.

We used mobility traces from the WiFi access network of Dartmouth college [12]
to model the movements of users in our simulations.
```

Obrázek 3.13: Příklady odkazů na bibliografické citace

3.3 Určování důvěryhodnosti extrahovaných metadat

Nezanedbatelnou součástí extrakce metadat je určení jejich důvěryhodnosti. Toho využívá systém *ReReSearch* (viz kapitola 3.6.1) při duplicitních informacích z více zdrojů. Podle hodnoty důvěryhodnosti poté rozhodne, který údaj je správný a uloží jej do databáze. Tato hodnota se pohybuje v rozmezí **0–100%**. Údaje s důvěryhodností blížící se k nule jsou doporučeny pouze jako doplňující informace k nedůležitým položkám v databázi. Vyjímku tvoří přiřazené e-maily, které mohou být i při nízké důvěryhodnosti správné.

Určení důvěryhodnosti pro jednotlivé údaje se liší. Extrakce většiny metainformací používá k nalezení potřebných údajů různé metody a postupy. Právě podle těchto metod může být stanovena míra důvěryhodnosti. Někdy se také vypočítává podle počtu dalších možných hodnot nebo je určena podle úspěšnosti v experimentu. Při určování důvěryhodností je také zohledněno ruční srovnání programu `clanky2meta.py` se správnými metadaty a serverem `CiteSeerX` (viz kapitola 4.3.1) popsané v kapitole 4.3.3. V tomto srovnání se například ukázalo, že program je úspěšnější při extrakci názvu článku než při hledání jmen jeho autorů, proto jsou obecně důvěryhodnosti názvu článku určovány na větší hodnoty.

- **název článku**
  - pokud je název článku uvozen řetězcem „Title:“, není důvod pochybovat o správnosti nalezené informace a důvěryhodnost je stanovena na 100
  - v ostatních případech je důvěryhodnost nalezeného názvu článku stanovena na 80 podle dosažené úspěšnosti ve srovnání v kapitole 4.3.3
  - pokud je celý název napsán velkými písmeny, je k důvěryhodnosti připsáno 20, pokud bylo třeba název článku zkrátit, je odečteno 10
- **autoři článku**
  - důvěryhodnost extrahovaných jmen autorů je stanovena podle dosažených výsledků srovnání z kapitoly 4.3.3 na 50
  - pokud je jméno nalezeno z e-mailu, má experimentálně určenou důvěryhodnost na 90
  - je-li jméno nalezeno mezi jinými dříve extrahovanými jmény, je jeho důvěryhodnost experimentálně určena na 80
- **přiřazení e-mailu k autorovi**
  - důvěryhodnost správného přiřazení e-mailu je dána procentuálním výskytem podřetězců

  - jsou-li počty nalezených jmen autorů a e-mailů stejné, je k důvěryhodnosti připočteno až 50

- **přiřazení adres k autorovi**
  - pokud je adresa přiřazena podle značky u jména autora, je důvěryhodnost podle metody určena na 80
  - pokud je přiřazení provedeno podle pozice v hlavičce, je důvěryhodnost podle metody určena 60
  - pokud je ke všem autorům přiřazena jedna společná adresa, je důvěryhodnost experimentálně určena na 50

- **abstrakt**
  - důvěryhodnost nalezeného abstraktu je určena na 85 podle dosažených výsledků srovnání v kapitole 4.3.3
  - pokud je abstrakt řádně ukončen tečkou a dvěma volnými řádky, je důvěryhodnost zvýšena až o 20
  - pokud není řádně ukončen tečkou, je důvěryhodnost snížena až o 20
  - při zkrácení abstraktu je od důvěryhodnosti odečteno až 10

- **kapitoly**
  - názvy kapitol extrahované z obsahu:
    * extrakce názvů kapitol z nepoškozených obsahů bývá bezchybná, proto je důvěryhodnost stanovena na 100
  - názvy kapitol uvozené arabskou číslicí:
    * při uvození arabskou číslicí a nalezení názvu kapitoly v seznamu běžných názvů kapitol je skoro jisté, že se jedná o správnou informaci, a důvěryhodnost je proto stanovena na 100
    * ostatní kapitoly mají důvěryhodnost stanovenu podle vzorce 3.1, kde *pocet* je počet nenalezených kapitol od poslední nalezené

$$80 - 10 * pocet \tag{3.1}$$

    * podkapitoly jsou extrahovány s důvěryhodností určenou na 70
  - názvy kapitol uvozené římskou číslicí:
    * důvěryhodnost je stanovena na 70
  - ničím neuvozené názvy kapitol:
    * názvy kapitol bez uvození mají důvěryhodnost stanovenu na 10, protože zde bývá hodně nepřesností a chybných informací

- **klíčová slova**
  - důvěryhodnost nalezených klíčových slov je 100

- **bibliografické citace**
  - celá reference má určenou důvěryhodnost podle způsobu nalezení, tzn. podle uvozující značky
    - hranaté a kulaté závorky 80
    - závorka zprava, římské číslo a číslice s tečkou 70
  - důvěryhodnost je snižována, pokud je nutno pro nalezení použít méně přesný formát citace
  - pokud je reference nalezena pomocí značky vytvořené ze jmen autorů, tak je důvěryhodnost 70
  - důvěryhodnost dílčích informací je různá:
    - text bibliografické citace má stejnou důvěryhodnost jako celá citace
    - odkaz na bibliografickou citaci má důvěryhodnost 90, pokud je nalezen podle značky reference a přiřazen k ní, jinak má 70
    - název citovaného článku má důvěryhodnost 90, pokud je označen (např. uvozovkami), jinak 70
    - autoři mají důvěryhodnost stanovenu na 70
    - rok vydání má důvěryhodnost 70
    - informace o konferenci důvěryhodnost určenou na 70
    - důvěryhodnost nalezeného URL odkazu je 70
    - stránky mají také důvěryhodnost 70

3.4 Dodatečné informace

Program obsahuje velké množství mnohdy složitých regulárních výrazů. Někdy se stává, že při procházení dlouhého či špatného textu (obsahujícího nestadndardní znaky) se program na dlouhou dobu zasekne, pokud je tento text zpracováván za pomocí takového složitého regulárního výrazu. Proto je do programu zabudován časovač nastavený na 5 minut, který při případném záseku ukončí zpracovávání daného článku a pokračuje dalšími. Tento časový údaj byl experimentálně změřen při zpracování článků z `CiteSeer`^{X}. Ukázalo se, že článek zpracovávaný příliš dlouho neobsahuje hodnotné informace nebo je znehodnocen převodem z původního formátu. Program zapisuje průběh extrakce do logovacích souborů. Lze z nich vyčíst soubory, které se nepovedlo zpracovat, a také časy zpracování jednotlivých souborů.

3.5 Návod k použití programu clanky2meta.py

3.5.1 Vstupní omezení a výstupy programu

Program na vstupu přijímá pouze textové soubory. V případě spuštění programu na celou složku jsou v této složce vyhledávány textové soubory ke zpracování. Výstupní `XML` soubory s metadaty budou vytvořeny do výstupní složky určené parametrem.

Při svém běhu program vypisuje právě prováděné operace na standartní výstup společně s časem spuštění a dobou zpracování souboru (viz obrázek 3.14). Při výskytu nějaké chyby (timeout, mnoho špatných znaků v souboru atp.) je také vypsána informace.

```
(2010, 4, 21, 19, 10, 33, 2, 111, 1)
Zpracovavam soubor 'mnt{*}data2{*}clanky{*}data{*}ab{*}abf947341c62b775d52bc4a3444eb2029d47b9e2.txt':
        Upravuji vstupni text.....              hotovo
        Hledam hlavicku.....                    hotovo
        Hledam nazvy kapitol.....               hotovo
        Hledam odkazy.....                      hotovo
        Hledam titulek.....                     hotovo
        Hledam klicova slova.....               hotovo
        Hledam emaily.....                      hotovo
        Hledam autory.....                      hotovo
        Prirazuji emaily.....                   hotovo
        Hledam adresy.....                      hotovo
        Prirazuji adresy.....                   hotovo
        Hledam abstrakt.....                    hotovo
        Hledam reference.....                   hotovo
        Hledam citace.....                      hotovo
        Hledam Related Work                     hotovo
        Vytvarim XML soubor.....                hotovo
Soubor 'mnt{*}data2{*}clanky{*}data{*}ab{*}abf947341c62b775d52bc4a3444eb2029d47b9e2.txt' uspesne zpracovan za 1.42s
```

Obrázek 3.14: Výstup programu clanky2meta.py

3.5.2 Hardware a software

Požadavky ke spuštění

Program je doporučeno spouštět v operačním systému `Linux` s nainstalovaným `Pythonem` ve verzi **2.6.2.** a optimalizační knihovnou `Psyco` (viz kapitola 3.1.2).

Použitý hardware

Hardware použitý při vývoji a testování programu zobrazuje tabulka 3.5.

	Hardware	Software
Vývoj	AMD Turion 64 X2 1,59 GHz, 2 GB RAM	Fedora 12 64bit Linux
Testování	2xDual Core AMD Opteron 2220, 16 GB RAM	CentOS 5.4 64bit Linux

Tabulka 3.5: Hardware použitý k vývoji a testování programu clanky2meta.py

3.5.3 Spuštění programu

Program `clanky2meta.py` se spouští s parametry, které určují, zda-li se bude zpracovávat jeden nebo více textových souborů. V případě jednoho souboru se používá přepínač `-f` následovaný jeho adresou. Je-li potřeba extrahovat metadata z více souborů, používá se přepínač `-d`, za kterým musí být adresa složky obsahující textové soubory. Přepínačem `-o` musí být také specifikována adresa výstupní složky pro XML soubory s nalezenými metadaty. Posledním přepínačem `-l` se určuje složka pro logovací soubory, které informují o průběhu extrakce.

Příklady spuštění

- pro jeden soubor:
  `./clanky2meta.py -f vstupni_soubor.txt -o vystupni_slozka -l logy`

- pro celou složku:
  `./clanky2meta.py -d vstupni_slozka -o vystupni_slozka -l logy`

3.6 Použití programu clanky2meta.py v systému ReReSearch

Vytvořený program `clanky2meta.py` je určen jako součást projektu *ReReSearch*. Ten vzniká v Ústavu počítačové grafiky a multimédií na Fakultě informačních technologií VUT v Brně. Je využíván *podsystémem pro automatické zpracování lokálních dokumentů*.

3.6.1 Projekt ReReSearch

Jedná se o rozsáhlý projekt, který má za cíl vytvořit systém poskytující přehled vědeckých prací z různých oblastí. Systém analyzuje data o výzkumnících, vědeckých institucích a týmech, vydavatelích, knihách a časopisech, konferencích, vědeckých seminářích, programových výborech, projektech a soutěžích, tématech, trendech atd. Informace získává z internetu a lokální databáze. Informace o tomto projektu jsou čerpány z [7].

3.6.2 Podsystém pro automatické zpracování lokálních dokumentů

Tento podsystém má za úkol automatizovaně zpracovávat dodané vědecké články, které mohou být v podobě `pdf`, `ps`, `doc`[6] nebo `rtf`[7] souborů. Podsystém je převede na text, vzniklé textové soubory uloží do společného uložiště a spustí na nich extrakci metainformací programem `clanky2meta.py`. Vzniklé XML soubory poté předá dalšímu modulu k následnému zpracování, například k nahrání dat do databáze.

[6] Formát souborů programu Microsoft Word pro uchování textu
[7] Rich Text Format - platformově nezávislý formát souborů pro uložení textu a grafiky [13]

Kapitola 4

Srovnání s jinými systémy

Každý program prochází při svém vývoji řadou testů, které pomáhají mimo jiné nalézt chyby a zlepšit jeho výsledky. Důležitou součástí testování programu `clanky2meta.py` bylo srovnávání extrahovaných metadat s jinými systémy. To pomohlo odhalit slabiny v určitých extrakcích a umožnilo jejich alespoň částečné zlepšení.

4.1 Srovnání se serverem arXiv.org

Prvním prováděným srovnáním bylo ruční porovnávání padesáti článků, které bylo možno nalézt na serveru `arXiv.org`[1]. Ten k nim totiž nabízí také některá jejich metadata. Výsledek srovnání znázorňuje tabulka 4.1. Při srovnávání byly tolerovány případné zkomoleniny či chybějící předložky v případech, kdy bylo evidentní, že se jedná o správnou informaci znehodnocenou například převodem na text.

Srovnávaná metadata	Shodných článků	Procentuální shoda
Název článku	38	76,00%
Autoři článku	36	72,00%
Abstrakt článku	35	64,00%
Použitá literatura	45	90,00%

Tabulka 4.1: Shoda extrahovaných metainformací s daty ze serveru arXiv.org

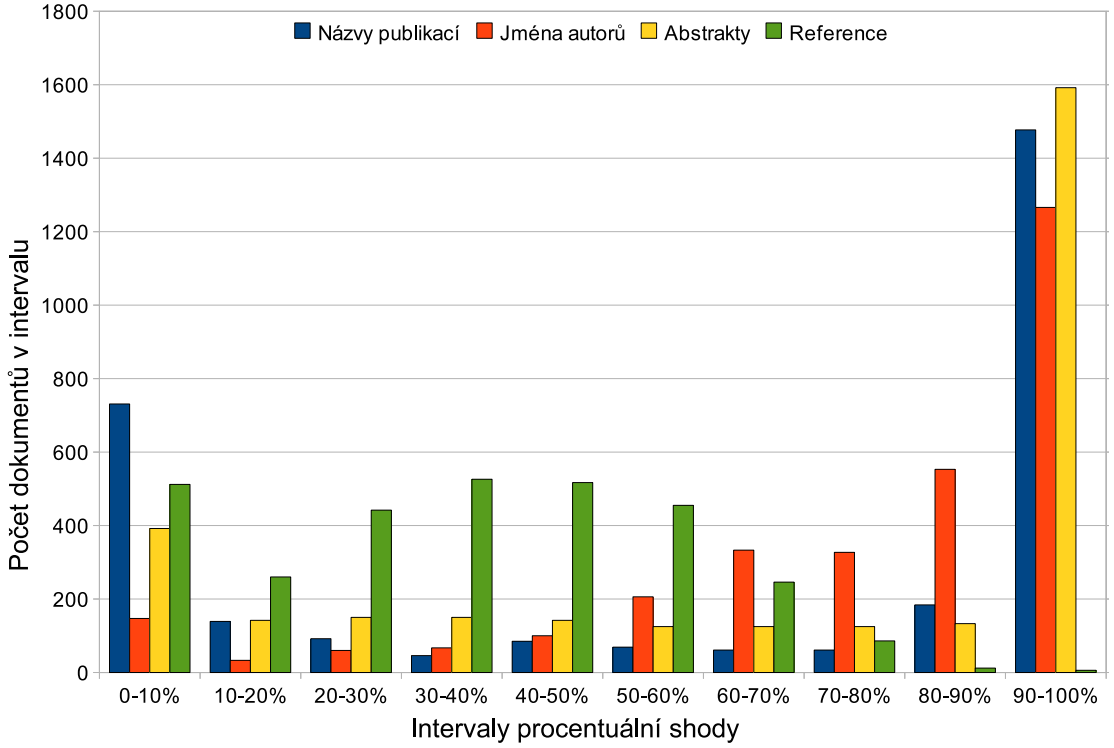
Zjištěné příčiny špatné extrakce některých metainforamcí, podle kterých byl program později upraven:

- **Název článku:** špatné určení některých názvů článku bylo způsobeno tím, že na začátku článku byl přidán název zdrojového serveru, název souboru, datum, apod. Tyto informace byly tedy špatně považovány za název článku.
- **Autoři článku:** některé jméno autora buď chybělo nebo naopak přebývalo
- **Abstrakt článku:** abstrakt byl někdy delší, někdy mu část scházela
- **Použitá literatura:** použitá literatura neukázala v tomto srovnání zásadní nedostatky

[1] http://arxiv.org/

4.2 Srovnání v bakalářské práci Petra Váchy

Srovnáním programu `clanky2meta.py` s jinými systémy se také zabývá bakalářská práce Petra Váchy [10]. Ten ve své práci provádí srovnání se servery `CiteSeer`^{X} (viz kapitola 4.3.1) a `Google Scholar`[2]. Jako vzorek použil **3 045** vědeckých publikací. Následující graf (obrázek 4.1) a tabulka 4.2 jsou převzaty z této bakalářské práce. Program `clanky2meta.py` se zde nazývá `STL`[3].



Obrázek 4.1: Graf úspěšnějších STL výsledků uvedených v bakalářské práci Petra Váchy

Autor používá pro srovnání vlastní algoritmus, který porovnává dvojice slov v řetězcích a zohledňuje i případné iniciály. Samotná shoda je spočtena podle vzorce (4.1).

$$shoda = \frac{pomer\_shodnych\_dvojic * 2}{pocet\_vsech\_slov} * 100 \tag{4.1}$$

Srovnávaná metadata	Úspěšnost extrakce
Název článku	63,30%
Autoři článku	77,00%
Abstrakt článku	68,67%
Použitá literatura	34,57%

Tabulka 4.2: Výsledky srovnání dosažené v bakalářské práci Petra Váchy

[2] http://scholar.google.cz/
[3] Script Tomáše Lokaje

Hodnoty v této tabulce znamenají průměr dosažených lepších hodnot programu `clanky2meta.py` v porovnání s `Google Scholar` a `CiteSeer`^{X}. Toto srovnání bylo provedeno v ranné fázi vývoje programu, od té doby byl několikrát upravován k dosažení lepších výsledků.

4.3 Srovnání extrahovaných metadat se CiteSeer^X

Dalším prováděným srovnáním bylo srovnání s metadaty k článkům staženým ze serveru *CiteSeer*^{X}.

4.3.1 Co je to CiteSeer^X?

CiteSeer^X[4] je nová generace digitální knihovny **CiteSeer**[5]. Jde o systém vyvinutý v roce 1997 vědci v NEC Research Institute [9]. Jeho autory jsou Steve Lawrence, Lee Giles a Kurt Bollarcker. Knihovna využívá Autonomous Citation Indexing[6]. To znamená, že prohledává web a vyhledává v něm vědecké články. Ty následně zpracovává a informace z nich ukládá do své databáze. Díky tomu je například možné publikace vyhledávat, zkoumat počty jejich citací a procházet použitou literaturu, ze které tato díla čerpala.

Po zhruba desetiletém provozu však CiteSeer začal přesahovat kapacity své původní architektury [14]. Obsahoval okolo **150 000** dokumentů a denně obsluhoval přes **1,5 miliónu** dotazů. Na základě předešlých poznatků a potřeb byl vytvořen systém **CiteSeer**^{X}. Ten dnes dokáže indexovat skoro **1,6 miliónu** dokumentů.

4.3.2 Výsledek srovnání

Srovnání bylo prováděné na **69 537** článcích. Dostupnými metainformacemi ke staženým článkům ze serveru `CiteSeer`^{X} byly:

- název článku
- jména autorů článku
- abstrakt článku

K prohlášení informace za identickou byla potřeba shoda řetězce alespoň 70% pro název a jména autorů, 50% pak pro abstrakt článku. Výsledky toho porovnání zobrazuje tabulka 4.3.

Srovnávaná metadata	Shodných informací	Úspěšnost
Název článku	42 516	61,14%
Autoři článku	27 892	40,11%
Abstrakt článku	40 025	57,56%

Tabulka 4.3: Shoda metadat mezi systémy clanky2meta.py a CiteSeer^X při druhém srovnání

Porovnání však nepřineslo uspokojivá čísla. Při procházení metadat dostupných ze serveru `CiteSeer`^{X} bylo zjištěno, že i tyto informace nejsou vždy správné, a to také

[4] http://citeseerx.ist.psu.edu/
[5] http://citeseer.ist.psu.edu/
[6] Automatické indexování citací

ovlivňuje srovnání těchto dvou systémů. Bylo potřeba získat sadu správných dat, na které by bylo demonstrováno objektivní srovnání. Proto bylo provedeno *srovnání systémů s ručně získanými metadaty*.

4.3.3 Srovnání systémů s ručně získanými metadaty

Během porovnávání metainformací vyextrahovaných programem `clanky2meta.py` s metadaty ze serveru `CiteSeer`^{X} se zjistilo, že také tato data obsahují hodně nepřesností. To bezpochyby ovlivnilo již zmíněné srovnání. Proto bylo ručně zpracováno **1 500** publikací, ze kterých se zaznamenal jejich název, autoři a abstrakt. U některých článků nebylo tyto informace možno určit ani ručně, protože se může prakticky jednat o jakýkoliv text (např. zpráva, u které není uveden žádný přesný název a ani autoři). Tyto články se do srovnání nezapočítávaly, proto bylo výsledné srovnání provedeno na **1 448** publikacích.

Srovnání názvů článků

Při porovnávání názvů článků byla zvolena podmínka shody řetězce minimálně **90%**. Pro zajímavost je uvedena také stoprocentní shoda řetězce. Bylo zjištěno, že při této extrakci je mírně lepší program `clanky2meta` (viz tabulka 4.4).

Zdroj metainformací	Shoda	Správných metainformací	Úspěšnost
`clanky2meta.py`	100%	1 124	77,62%
	90–99%	21	1,45%
`CiteSeer`^{X}	100%	806	55,66%
	90–99%	119	8,22%

Tabulka 4.4: Výsledky ručního porovnání názvů článků

Srovnání jmen autorů článků

Zdroj metainformací	Shoda	Správných metainformací	Úspěšnost
`clanky2meta.py`	100%	442	30,52%
	90–99%	107	7,39%
	80–89%	194	13,40%
`CiteSeer`^{X}	100%	638	44,06%
	90–99%	121	8,36%
	80–89%	156	11,77%

Tabulka 4.5: Výsledky ručního porovnání jmen autorů článků

Srovnání bere v potaz i možnost zapsání jména zkráceně pomocí iniciály a byla zde zvolena trojí hranice potřebné shody:

- shoda větší než **90%** odpovídá správnému nalezení všech jmen autorů
- shoda **80–90%** odpovídá při výskytu tří a více autorů buď jedné chybějící části jména autora nebo celému autorovi

Tabulka 4.5 dokazuje, že extrakce autorů článků je úspěšnější v systému `CiteSeer`^{X}.

Srovnání abstraktů článku

Zdroj metainformací	Shoda	Správných metainformací	Úspěšnost
`clanky2meta.py`	100%	673	46,48%
	90–99%	573	39,57%
`CiteSeer`^{X}	100%	392	27,07%
	90–99%	882	60,91%

Tabulka 4.6: Výsledky ručního porovnání abstraktů článků

Před porovnáním abstraktů byly všechny abstrakty zkráceny na stejnou délku, protože v datech z `CiteSeer`^{X} může být jen jejich část. Bylo ukázáno, že extrakce abstraktů je u obou zdrojů podobně úspěšná (viz tabulka 4.6).

Kapitola 5

Statistiky běhu programu, jeho nedostatky a možná zlepšení

Tato kapitola ukazuje výsledky, kterých dosáhl program `clanky2meta.py` při testovacím běhu k použití v systému `ReReSearch`. Dále shrnuje nedostatky programu a navrhuje možná zlepšení.

5.1 Testovací běh pro systém ReReSearch

Díky požadavkům vycházejícím z projektu `ReReSearch` byl program `clanky2meta.py` testován na velkém množství článků, které pocházely ze serveru *CiteSeer*^{X} (viz kapitola 4.3.1). Těchto **668 458** dokumentů bylo převedeno na text a poté na nich byla provedena extrakce metadat. Úspěšně se podařilo zpracovat **99,73%** textových dokumentů. Statistické údaje o tomto průběhu shrnuje tabulka 5.1. Údaje s hvězičkou byly spočteny pouze na úspěšně zpracovaných souborech. Chyba `Overflow Error` nastává v případě, kdy je jako vstupní řetězec pro složitý regulární výraz poslán text, který je příliš dlouhý nebo obsahuje samé nestandardní znaky. `Timeout` nastane po pěti minutách, které jsou povoleny jako maximální doba zpracování jednoho souboru.

Údaj		**Změřená hodnota**
Celkový počet zpracovávaných souborů		668 458
Počet úspěšně zpracovaných souborů		666 620
Počet zpracování skončených s chybou	`Overflow Error`	1 452
	`Timeout`	386
Průměrná doba zpracování jednoho souboru*		1,72 s
Nejkratší doba zpracování jednoho souboru*		0,04 s
Nejdelší doba zpracování jednoho souboru*		300,00 s
Průměrná velikost jednoho zpracovávaného souboru*		62,23 kB
Nejmenší velikost jednoho zpracovávaného souboru*		0,37 kB
Největší velikost jednoho zpracovávaného souboru*		6 787,98 kB

Tabulka 5.1: Statistické údaje o běhu programu.

Následující tabulky zobrazují rozložení časových a velikostních údajů do intervalů. Tabulka 5.2 dokazuje, že více jak **70 %** souborů je zpracováno do dvou sekund, mnoho

z nich dokonce do půl sekundy. Horní hranice 300 s je dána časovačem, který povoluje zpracovávání jednoho souboru maximálně 5 minut.

Časový interval [s]	Počet souborů	Počet procent
$(0-0,5>$	238 218	35,64%
$(0,5-1,0>$	115 752	17,32%
$(1,0-1,5>$	10 152	1,52%
$(1,5-2,0>$	125 795	18,82%
$(2,0-2,5>$	33 275	4,98%
$(2,5-3,0>$	61 367	9,18%
$(3,0-10,0>$	20 704	3,10%
$(10,0-300,0>$	61 357	9,18%

Tabulka 5.2: Rozložení doby zpracování do časových intervalů.

Z tabulky 5.3 lze zase vyčíst, že většina souborů má velikost od 40ti kB do 120ti kB.

Velikostní interval [kB]	Počet souborů	Počet procent
$(0-20>$	12 890	1,93%
$(20-40>$	8 477	1,27%
$(40-60>$	128 026	19,15%
$(60-80>$	251 177	37,58%
$(80-100>$	56 449	8,44%
$(100-120>$	135 479	20,27%
$(120-140>$	24 121	3,60%
$(140-6\,788>$	50 001	7,48%

Tabulka 5.3: Rozložení velikostí souborů do velikostních intervalů.

5.2 Nedostatky a možná zlepšení programu clanky2meta.py

Příčiny nepřesností ve výsledcích hledání metadat lze spatřovat v několika rovinách. Prvotní problémy nastávají ve struktuře původního dokumentu, kdy nemusí být jednotlivé údaje jednoznačně určeny pro strojové zpracování. Další problémy se přidávají při převodu souboru do textového formátu, jak již bylo několikrát zmíněno. Program má potom problémy najít celý obsah jisté informace, či může naopak její náplň rozšířit o nesprávná data. To se nejčastěji děje při hledání jmen autorů, protože program nemůže vědět, jak dlouhé jméno jeden autor má, jestli už náhodou zpracovávané slovo není část jména dalšího autora. Částečně by se tento problém dal eliminovat vytvořením jakéhosi filtru, který by filtroval soubory na vstupu programu. Při stahování publikací z internetu se totiž ne vždy stáhnou opravdové články, ale často také zprávy (reporty) či prezentace, které nemusí a mnohdy ani neobsahují vyhledávané metainformace. Jelikož je program založen na principu, aby pokud možno vždy nějaké informace našel, tak může dojít ke zbytečnému zanesení nesprávných dat do databáze. Příkladem takového špatného vstupního souboru může být soubor obsahující hlavičku zobrazenou v obrázku 5.1. Je zde ukázán jak problém nevhodného typu článku pro zpracování, tak znehodnocení informací rozházením textu

při převodu. Neobsahuje totiž klasickou strukturu **vědeckého článku**, která by měla postupně obsahovat jeho název, jména autorů, abstrakt, úvodní kapitolu atp.

```
U.S. Department of Justice
DE PA

T OF JU MEN S RT

CE TI

Office of Juvenile Justice and Delinquency Prevention
IJ J O F OJJ DP B RO J US TIC E P

YOUTH GANG

S G OVC RA MS

Office of Justice Programs

N BJ A C E I OF F

Office of Juvenile Justice and Delinquency Prevention
The Office of Juvenile Justice and Delinquency Prevention (OJJDP) was established...
```

Obrázek 5.1: Příklad špatné hlavičky v textu

Z hlavičky jsou pak programem extrahovány následující údaje, které jsou určitě chybné:

- **Název článku:**
  - *DE PA*
- **Jména autorů:**
  - *Justice Programs*
  - *OF JU*
  - *OVC RA*
- **Abstrakt:**
  - nenalezen

Slabou stránkou programu `clanky2meta.py` je také zpracování textů v jiném než anglickém jazyce. To je dáno jednak diakritikou v některých jazycích (viz kapitola 3.2.1), tak také pojmenováním klíčových termínů pro extrakci, jako je například abstrakt, názvy kapitol nebo dokonce použitá literatura. To vše může být nazváno například v rodném jazyce autora a program potom tyto informace nedokáže nalézt. K tomu by se musela vytvořit databáze všech možných pojmů v různých jazycích, což by mimojiné vedlo ke zpomalení celého procesu extrakce. Tabulka 5.4 popisuje situaci, při které dochází k několikanásobnému znehodnocení jména autora obsahujícího znaky s diakritikou. Z autora se jménem **Johan Kåhrström** se tak stane **Johan Khrstrom**, který je poté například vložen do databáze, takže zpracovávaný článek může být připsán někomu jinému.

Výskyt	Jméno autora	Popis problému
Původní článek v PDF	`Johan Kåhrström`	jméno obsahuje `ö` a `å`
Převedený textový soubor	`Johan K:hrström`	při převodu došlo ke znehodnocení znaku `å`
Extrahovaná metadata v XML	`Johan Khstrom`	při zpracování programem došlo ke ztrátě znehodnoceného znaku `å` a k převodu znaku `ö` na `o`

Tabulka 5.4: Příklad znehodnocení autorova jména obsahujícího diakritiku během procesu získávání metadat

Ke zlepšení výsledků programu `clanky2meta.py` by bylo později možné využít například databázi systému `ReReSearch`. Z ní by bylo možné získávat například aktuálnější slovníky křestních jmen či seznam běžných názvů kapitol. Nejdříve by ale bylo třeba všechna data důkladně kontrolovat a dodávat do programu jen ta, u kterých je stoprocentní jistota jejich správnosti. Mohlo by se totiž stát, že by program `clanky2meta.py` špatně našel autorovo jméno a to by bylo uloženo do databáze. Bez ověření toho, že křestní jméno autora opravdu existuje, by se totiž při pozdějších extrakcích vyhledávala i jména autorů s právě takovým křestním jménem a docházelo by k nekontrolovanému šíření špatných dat celou databází. Systém `ReResearch` obsahuje více extračních modulů vyhledávajících metadata z jiných zdrojů a jinými způsoby. Proto by jednou z možností kontroly existence takového křestního jména mohlo být jeho případné nalezení také jinými moduly, případně by se taková křestní jména musela potvrzovat uživatelem systému.

Kapitola 6

Závěr

Cíl této bakalářské práce, tj. vytvoření systému k extrakci metadat z vědeckých článků, se podařilo splnit. Byl vytvořen program `clanky2meta.py`, který dokáže z textových souborů získat určité informace a uložit je ve formátu vhodném k dalšímu zpracování. K tomu byl zvolen formát `XML`. Byla také zohledněna potřeba zpracování velkého množství dat, proto bylo v programu mimo jiné použito optimalizační knihovny `Psyco` a byl zabudován časovač.

Bylo provedeno několik srovnání, které odhalily slabé stránky extrakce určitých informací. Částečně se podařilo některé chyby odstranit, ale i přesto nejsou dosahované výsledky stoprocentní. Nutno podotknout, že se jedná o netriviální operace nad daty s nejednotnou strukturou, a ani srovnávaný systém `CiteSeer`^{X} není v tomto ohledu dokonalý.

Ke zlepšení výsledků extrakce metadat z vědeckých článků programem `clanky2meta.py` by v budoucnu bylo možné použití filtru k vyčlenění článků, které opravdu má smysl zpracovávat, tzn. dokumenty dodržující strukturu vědeckého článku, které nejsou příliš znehodnoceny převodem do textového formátu. Dále se nabízí vyřešení problému s texty nepsanými v anglickém jazyce. Takových dokumentů se sice v testovaných souborech vyskytlo mizivé množství, ale i tak mohou v některých anglických publikacích být hledané informace psány jazykem cizím, resp. můžou být použity národní znaky s diakritikou.

Program vytvořený v rámci této práce je určen jako součást systému `ReReSearch`, kde bude získávat informace z vědeckých článků, které budou vkládány do databáze. S tímto systémem by program mohl v budoucnu více komunikovat nejen jako informační zdroj, ale také z tohoto systému získávat aktuální data potřebná k vyhledání učitých informací, například křestní jména autorů.

Literatura

[1] Hruška, T.: Pojem informačního systému, Data, Procesy, Transakce - studijní opora. 2008.

[2] Humphreys, K.; Demetriou, G.; Gaizauskas, R.: Two Applications of Information Extraction to Biological Science Journal Articles: Enzyme Interactions and Protein Structures.

[3] Menoušek, J.: Jak (ne)napsat článek pro odborný časopis? [online]. *Ikaros*, ročník 6, č. 9, 2002 [cit. 2010-04-19].

[4] Petránek, T.: *Extrakce bibliografických informací z textu*. Diplomová práce, Masarykova univerzita, Brno, 2006.

[5] Polák, M.: *Návrh a implementace systému pro extrakci informací*. Diplomová práce, Masarykova univerzita, Brno, 2009.

[6] Pospíšil, J.; Nemrava, M.: Dolování dat a jeho aplikace. 2006.

[7] Smrž, P.: ReReSearch [online]. `https://merlin.fit.vutbr.cz/nlp-wiki/index.php/ReReSearch`, [cit. 2010-04-13], interní dokumentace.

[8] Tan, A.-H.: Text Mining: The state of the art and the challenges.

[9] Župa, P.: CiteSeer - Scientific Literature DL. 2006, esej do předmětu Digitální knihovny.

[10] Vácha, P.: *Extrakce metadat z vědeckých článků*. Bakalářská práce, FIT VUT v Brně, Brno, 2009.

[11] Voltr, J.: Odborný článek - jak na něj [online]. `http://fyzsem.fjfi.cvut.cz/2008-2009/Leto09/html/publish/paper.html`, [cit. 2010-04-18].

[12] Weingartner, P.: A First Guide to PostScript [online]. `http://www.tailrecursive.org/postscript/postscript.html`, 2006 [rev. 2006-24-02], [cit. 2010-28-04].

[13] WWW stránky: Rich Text Format (RTF) Specification, version 1.6 [online]. `http://msdn.microsoft.com/en-us/library/aa140277%28office.10%29.aspx`, 1999 [cit. 2010-28-04].

[14] WWW stránky: About CiteSeerX [online].
`http://citeseerx.ist.psu.edu/about/site`, [cit. 2010-04-15].

[15] WWW stránky: Psyco [online].
`http://psyco.sourceforge.net/introduction.html`, [rev. 2010-03-8],
[cit. 2010-04-02].

Příloha A

Obsah CD

Obsah jednotlivých složek na přiloženém CD:

- `bp` - obsahuje elektronickou verzi této práce ve formátu `PDF`
  - `source` - zdrojové LaTeXové soubory a obrázky
- `clanky2meta` - vytvořený program `clanky2meta.py` s manuálem
  - `moduly` - moduly programu
  - `slovniky` - použité slovníky (například křestních jmen)
- `files` - ukázkové soubory
  - `input_txt` - vstupní textové soubory
  - `output_xml` - výstupní `XML` soubory s extrahovanými metadaty
- `plakat` - plakát stručně prezentující tuto práci

Příloha B

Ukázka vstupního textového souboru

Composition methods in the presence of small parameters
Robert I. McLachlan February 4, 2003
1991 Mathematics Subject Classification. 65L05, 7008. Keywords and phrases. Composition methods, initial value problems, splitting methods, symplectic integrators, solar system.

Abstract
We derive numerical methods for arbitrary small perturbations of exactly solvable differential equations. The methods, based in one instance on Gaussian quadrature, are symplectic if the system is Hamiltonian and are asymptotically more accurate than previously known methods.

1

Introduction

Composition methods are numerical integrators for ordinary differential equations which compose certain elementary flows to build an approximation of high order. For a vector field X with flow exp(tX) (i.e., x = X(x) x(t) = exp(tX)(x(0))), a popular special case is when X can be split as X=A+B where the vector fields A and B can both be integrated explicitly. The advantage of doing this is that geometric properties of the true flow are easily preserved: if X, A, and B are Hamiltonian then the flow and the method are symplectic; if divergence free, then volume preserving, and so on. It is annoying, however, that these special properties of X are never taken advantage of in deriving the composition. In fact, it is shown in [6] that these methods are equivalent to those formed by composing an arbitrary first-order method and its inverse, so it seems that even the fact that one is solving the constituent vector fields A and B exactly is not used. Here we consider one case in which this information can be used to advantage. Suppose the system is a small perturbation of one that can be solved exactly --i.e., X is near-integrable--and is split accordingly: X = A + B.

We now have two small parameters, and the time step t. Although for convergence ...

... *zkráceno* ...

5

Examples

If a system is only weakly nonlinear, a standard second-order numerical method methods will have global truncation errors of O(t2). Solving the linear part exactly in a splitting method reduces the errors to O(t2); using the (2s,2) methods introduced here will lead to errors of O(2t2+t2s). For small t the error is thus reduced by an extra factor of ; for large t the error decreases more rapidly, like t2s , the switchover taking place at t 1/(2s-2) . In general it should pay to use larger s as decreases, although of course there is no universal criterion here. An example is the Boussinesq partial differential equation ...

... *zkráceno* ...

9

References [1] Burstein, S. Z., and A. A. Mirin, Third order difference methods for hyperbolic equations, J. Comp. Phys. 5 (1970), 547571. [2] Gjaja, I., A. J. Dragt, and D. T. Abell, A comparison of methods for longterm tracking using symplectic maps, preprint, 1993. [3] Goldman, D., and T. J. Kaper, N th-order operator splitting schemes and nonreversible systems, Center for Fluid Mechanics Turbulence and Computation publication 9316, 1993. [4] Hairer, E., S. P. Nřrsett, and G. Wanner, Solving Ordinary Differential Equations I, 2nd ed., Springer-Verlag, Berlin New York, 1993. [5] Koseleff, P.-V., Relations among formal Lie series and construction of symplectic integrators, in Applied Algebra, Algebraic Algorithms and Errorcorrecting Codes, AAECC10 (Puerto Rico, 1993), G. Cohen, T. Mora, and O. Moreno, eds., Springer, Berlin New York, 1993. [6] McLachlan, R. I., On the numerical integration of ordinary differential equations by symmetric
composition methods, SIAM J. Sci. Comp. (1994), to appear. [7] Saha, P., and S. Tremaine, Symplectic integrators for solar system dynamics, Astron. J. 104 (1992), 16331640. [8] Wisdom, J., The origin of the Kirkwood gaps: a mapping for the asteroidal motion near the 3/1 commensurability, Astron. J. 87 (1982), 577593. [9] Wisdom J., and M. Holman, Symplectic maps for the N-body problem, Astron. J. 102 (1991), 15281538. [10] Wisdom, J., M. Holman,
and J.Touma, Symplectic correctors, preprint. [11]Yoshida, H., Construction of higher order symplectic integrators, Phys. Letters A 150 (1990), 262268.

Příloha C

Ukázka výstupního XML souboru

```
<output>
    <file> aaa10c7c026df51436b1ed9af1bd8ba547da95ec.txt </file>
    <publication>
        <title credibility="80">
            Composition methods in the presence of small parameters
        </title>
        <author credibility="50">
            <full_name> Robert I. McLachlan </full_name>
            <first_name> Robert </first_name>
            <middle_name> I. </middle_name>
            <last_name> McLachlan </last_name>
            <location credibility="50">
                February 4, 2003 1991 Mathematics Subject Classification. 65L05, 7008.
            </location>
        </author>
        <abstract credibility="95">
            We derive numerical methods for arbitrary small perturbations of exactly
            solvable differential equations. The methods, based in one instance on
            Gaussian quadrature, are symplectic if the system is Hamiltonian and are
            asymptotically more accurate than previously known methods.
        </abstract>
        <keyword>
            <keyword credibility="100">
                Composition methods
            </keyword>
            <keyword credibility="100">
                initial value problems
            </keyword>
            <keyword credibility="100">
                splitting methods
            </keyword>
            <keyword credibility="100">
                symplectic integrators
            </keyword>
```

```
    <keyword credibility="100">
        solar system
    </keyword>
</keyword>
<chapter credibility="100">
    <number> 1 </number>
    <name> Introduction </name>
</chapter>
<chapter credibility="60">
    <number> 4 </number>
    <name> Stability </name>
</chapter>
<chapter credibility="100">
    <number> 5 </number>
    <name> Examples </name>
</chapter>
<chapter credibility="100">
    <number> 9 </number>
    <name> References </name>
</chapter>
<citation credibility="60">
    <content credibility="100">
        It's nice that the order (2s, 2) methods have all coefficients positive [1].
    </content>
    <reference credibility="60">
        [1] Burstein, S. Z., and A. A. Mirin, Third order difference methods for
        hyperbolic equations, J. Comp. Phys. 5 (1970), 547571.
    </reference>
    <cited_publication_title credibility="70">
        Third order difference methods for hyperbolic equations
    </cited_publication_title>
    <cited_publication_date credibility="70">
        1970
    </cited_publication_date>
    <name credibility="70">
        <name> Burstein, S. Z. </name>
        <name> A. A. Mirin </name>
    </name>
</citation>
<citation credibility="60">
    <content credibility="100">
        (This anharmonic oscillator is also studied numerically in [2], but a direct
        comparison is not possible.) The primary piece A is a harmonic oscillator
        and the perturbation B is a shear.
    </content>
```

```
            <reference credibility="60">
                [2] Gjaja, I., A. J. Dragt, and D. T. Abell, A comparison of methods for
                longterm tracking using symplectic maps, preprint, 1993.
            </reference>
            <cited_publication_title credibility="70">
                A comparison of methods for longterm tracking using symplectic maps,
                preprint
            </cited_publication_title>
            <cited_publication_date credibility="70">
                1993
            </cited_publication_date>
            <name credibility="70">
                <name> Gjaja, I. </name>
                <name> A. J. Dragt </name>
                <name> D. T. Abell </name>
            </name>
        </citation>
    </publication>
</output>
```