Získávání informací z HTML dokumentů pomocí pokročilých data miningových metod

Advanced Data Mining Methods from HTML Documents

Bc. Michal Pohlídal

Diplomová práce
2012

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
akademický rok: 2011/2012

ZADÁNÍ DIPLOMOVÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Bc. Michal POHLÍDAL**
Osobní číslo:	**A10880**
Studijní program:	**N 3902 Inženýrská informatika**
Studijní obor:	**Informační technologie**
Téma práce:	**Získávání informací z HTML dokumentů pomocí pokročilých data miningových metod.**

Zásady pro vypracování:

1. **Definujte předpokládané výstupy práce.**
2. **Proveďte literární rešerši pokročilých data miningových metod zaměřených na prostředí HTML kódu.**
3. **Formou projektu navrhněte způsob a aplikaci data miningových metod pro řešení tématu.**
4. **Realizujte zvolené řešení.**
5. **Proveďte jeho vyhodnocení.**

Rozsah diplomové práce:

Rozsah příloh:

Forma zpracování diplomové práce: **tištěná/elektronická**

Seznam odborné literatury:

1. **HAN, Jiawei a Micheline KAMBER. Data Mining: Concepts and Techniques. Second Edition. San Francisco: Morgan Kaufmann Publishers, 2006, 770 s. ISBN 1-55860-901-6.**
2. **FELDMAN, Ronen a James SANGER. The Text Mining Handbook: Advanced Approaches in Analyzing Unstructured Data. New York: Cambridge University Press, 2007, 410 s. ISBN 0-521-83657-3.**
3. **BERRY, Michael W. (editor) a Jacob KOGAN (editor). Text Mining: Applications and Theory. Chichester: Wiley, 2010, 207 s. ISBN 978-0-470-74982-1.**
4. **BERKA, Petr. Dobývání znalostí z databází. 1. vydání. Praha: Academia, 2003, 366 s. ISBN 80-200-1062-9.**
5. **ZENDULKA, Jaroslav, Vladimír BARTÍK, Roman LUKÁŠ a Ivana RUDOLFOVÁ. Získávání znalostí z databází. Brno, 2009. Studijní opora. FIT VUT v Brně.**
6. **LIU, Bing. Web data mining: exploring hyperlinks, contents, and usage data. Berlin: Springer, 2007, 532 s. ISBN 978-354-0378-815.**

Vedoucí diplomové práce: **doc. Mgr. Roman Jašek, Ph.D.**
Ústav informatiky a umělé inteligence

Datum zadání diplomové práce: **24. února 2012**

Termín odevzdání diplomové práce: **21. května 2012**

Ve Zlíně dne 24. února 2012

L.S.

prof. Ing. Vladimír Vašek, CSc.
děkan

doc. Mgr. Roman Jašek, Ph.D.
ředitel ústavu

ABSTRAKT

Tato diplomová práce se zabývá získáváním informací z HTML dokumentů. V práci jsou vysvětleny postupy a metody dolování dat a jsou zde popsány rozdíly mezi jednotlivými typy HTML stránek z pohledu získávání dat. Dále práce obsahuje návrh a realizaci vytvořené aplikace pro získávání informací z HTML dokumentů. Součástí práce je také prezentace dosažených výsledků a srovnání s dalšími dostupnými nástroji pro získávání dat. Zmíněny jsou také možnosti využití a přínosy data miningu v praxi.

Klíčová slova: data mining, metody dolování dat, extrakce informace, získávání znalostí, HTML dokumenty, jazyk C#

ABSTRACT

This master thesis deals with the information extraction from HTML documents. In thesis are explains procedures and methods of data mining and describes the differences between the types of HTML pages from the perspective of data mining. The thesis includes design and implementation of developed application for the information extraction from HTML documents. The thesis also contains presentation of the results and comparison with other available tools for data mining. Mentioned are also possible uses and benefits of data mining in practice.

Keywords: data mining, data mining methods, information extraction, knowledge discovery, HTML documents, C# language

PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu mé diplomové práce doc. Mgr. Romanu Jaškovi, Ph.D. za jeho vedení, cenné rady a konzultace, které mi ochotně poskytoval.

Prohlašuji, že

- beru na vědomí, že odevzdáním diplomové/bakalářské práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že diplomová/bakalářská práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk diplomové/bakalářské práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně a jeden výtisk bude uložen u vedoucího práce;
- byl/a jsem seznámen/a s tím, že na moji diplomovou/bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – diplomovou/bakalářskou práci nebo poskytnout licenci k jejímu využití jen s předchozím písemným souhlasem Univerzity Tomáše Bati ve Zlíně, která je oprávněna v takovém případě ode mne požadovat přiměřený příspěvek na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše);
- beru na vědomí, že pokud bylo k vypracování diplomové/bakalářské práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky diplomové/bakalářské práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem diplomové/bakalářské práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji,

- že jsem na diplomové práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně …….

podpis diplomanta

OBSAH

ÚVOD

V dnešní době máme k dispozici velké množství dokumentů, zejména na internetu, které v sobě skrývají zajímavé informace. Z těchto dat je stále více obtížné získat vhodné fragmenty obsahující požadované informace. Usnadnění tohoto procesu vyhledávání a snazší zpracování informací si klade za cíl cílené vyhledávání. Metody, které se zabývají extrakcí informací z těchto zdrojů, se musí umět vypořádat s velkým množstvím dat a efektivně se zbavovat všech balastních informací, které nejsou podstatné.

Cílem mé diplomové práce je navrhnout aplikaci, která z HTML dokumentu získá uživatelem požadované informace a spočítá četnost slov. Aplikace nejprve vyčistí vstupní dokument od všech HTML značek, skriptů, komentářů a dalších dat, která jsou pro další zpracování nepotřebná. Tímto aplikace získá pouze textovou informaci, ze které dle uživatelem specifikovaných výrazů získá požadované informace a četnost slov.

Vhodným využitím navržené aplikace je její začlenění do nějakého většího projektu, kde by sloužila jako zdroj a příprava dat pro další zpracování. Aplikaci je ale také možné použít samostatně.

Získaná data je možné použít téměř ve všech oblastech, jako například v bankovnictví, marketingu, zdravotnictví, ve státní sféře a dalších.

Diplomová práce je členěna celkem do osmi kapitol. První čtyři kapitoly se zabývají teoretickou částí, ve které jsou popsány a vysvětleny základní pojmy a principy, se kterými bylo třeba seznámit se pro řešení navrhované aplikace. Zbývající kapitoly popisují praktické provedení aplikace.

V první kapitole se zabývám popisem a rozdíly mezi jednotlivými typy webových stránek. Dále zde uvádím jazyky, které se pro vytváření webových stránek používají nejčastěji.

Ve druhé kapitole se věnuji HTML dokumentům. Je zde popsána jejich struktura, která je pro získávání informací důležitá z důvodu správného získání dat. Dále je zde popsán zápis HTML tagů a zmíněny jsou také speciální značky.

Třetí kapitola se věnuje programovacímu jazyku C#, ve kterém jsem navrženou aplikaci realizoval. Popsána je jeho historie, vlastnosti a zmíněny jsou také vývojová prostředí, která se pro vývoj aplikací v jazyce C# používají.

V následující čtvrté kapitole popisuji obecně data mining. Je zde popsán proces získávání znalostí. Vysvětleny jsou také jednotlivé metody dolování dat. Závěr kapitoly tvoří využití dolování dat.

Praktická část diplomové práce začíná pátou kapitolou, ve které je popsán návrh aplikace a obecné cíle. Zmíněny jsou možné problémy při návrhu aplikace, se kterými se můžeme setkat. Součástí této kapitoly je také popis způsobů odstranění nepotřebných značek a možností získání dalších užitečných informací.

Samotnou realizací navržené aplikace se zabývá šestá kapitola. Ta se skládá z několika částí, kde popisuji přípravu a vyčištění vstupních dat. Dále je popsán způsob získání informací dle uživatelem zadaného výrazu a je popsána četnost slov. Důležitou částí je i popis uživatelského rozhraní vytvořené aplikace.

Sedmá kapitola se zabývá vyhodnocením realizovaného řešení a návrhy na další vylepšení aplikace.

Práce obsahuje také porovnání navržené aplikace s dalšími dostupnými nástroji, které je obsahem poslední osmé kapitoly.

V závěru práce jsou shrnuty dosažené výsledky, přínosy a je navrženo možné budoucí rozšíření práce.

I. TEORETICKÁ ČÁST

1 TYPY A JAZYKY WEBOVÝCH STRÁNEK

Téměř všechny dokumenty se v dnešní době přesouvají na internet. S přibývajícím množstvím informací je tak stále složitější se dostat jen k těm informacím, které hledáme. Na internetu totiž nejsou jen podstatné informace, ale je tam také velké množství nepodstatných a zbytečných informací. Dolování dat tak získává na větší důležitosti.

1.1 Statické stránky

Jak již název napovídá, jedná se o stránky, které se zobrazují tak, jak byly napsány. Získání dat z tohoto typu stránek je nejjednodušší, protože nemusíme řešit, jak se daná stránka zobrazí. Většinou se jedná o jednoduché prezentace s minimálním rozsahem. Obsahují převážně textové informace, obrázky, tabulky a hypertextové odkazy, kterými jsou navzájem propojeny.

1.1.1 Jazyk HTML

Jde o značkovací jazyk pro hypertext, který je jedním z mnoha jazyků pro vytváření webových stránek založený na SGML. Pomocí HTML určujeme, jaký význam bude mít která část HTML stránky, to znamená, že rozčleníme text na jednotlivé části. Označíme, co jsou nadpisy, kde jsou odstavce, tabulky a další.

Jazyk HTML umožňuje formátovat vzhled výsledné stránky. Toto formátování se provádí pomocí stylů – formátovacího jazyka CSS. Tyto styly se mohou zapisovat přímo k jednotlivým značkám v HTML kódu, ale nejčastější formou je zápis do samostatného souboru s uvedením odkazu na tento soubor v HTML. Tím je možné tuto sadu stylů použít ve více HTML souborech současně. Případné změny stylu se nemusí provádět v každé stránce zvlášť, ale stačí upravit jeden CSS soubor.

První definici HTML vytvořil Tim Berners-Lee v roce 1991. První verze byla označena jako HTML 0.9. Tím, jak se zvyšovaly požadavky uživatelů, a vývojáři prohlížečů přidávali další prvky do HTML, tak se tento jazyk rychle vyvíjel. Poslední vývojová verze HTML 4.01 byla vydána v roce 1999. Tímto vývoj HTML sice skončil, ale používá se dále. Jeho nástupcem je jazyk XHTML vyvinutý mezinárodním konsorciem W3C. [1]

1.1.2 Jazyk XHTML

Jazyk XHTML je nástupce HTML založený na XML. Rozdílů oproti HTML je celá řada. Na začátku dokumentu se nachází deklarace XML, následuje povinná deklarace typu dokumentu DTD a kořenový element html obsahuje atribut xmlns, který definuje jmenný prostor a jazyk dokumentu.

Všechny tagy i jejich atributy musíme psát s malými písmeny, protože jazyk XHTML je case-sensitive, to znamená, že záleží na velikosti písmen. Například tagy `<title>` a `<TITLE>` jsou tedy rozdílné. Jednotlivé atributy tagů musí být v uvozovkách, všechny tagy musí být párové, případně u nepárových se musí ukončit lomítkem a nesmí se navzájem křížit. [1]

1.2 Dynamické stránky

Na rozdíl od statických stránek, se obsah dynamických stránek generuje. To znamená, že se dle času, přihlášeného uživatele, kontextu nebo jiných podmínek vygeneruje obsah stránky.

Zdrojový kód dynamických stránek obsahuje také kód programovacích a skriptovacích jazyků. Tyto kódy se následně buďto na straně serveru, nebo na straně klienta zpracují, čímž dojde k nahrazení těchto kódů zpracovaným výsledkem. Uživateli se zobrazí až výsledná stránka.

Z pohledu získávání dat je situace u dynamických stránek mnohem komplikovanější než u statických. Z důvodu generování obsahu můžeme ze stejné stránky získat pokaždé jiný výstup. Pro získání dat z dynamických stránek je potřeba vzít v potaz parametry, které ovlivňují výstup. Podrobněji se způsobu získání textu věnuje kapitola Data mining.

V následujících podkapitolách popíši rozdíly v generování obsahu na straně klienta a straně serveru. Popíši také jazyky, které se pro vytváření dynamických stránek používají nejčastěji.

1.2.1 Generování obsahu stránek na straně klienta

Jde o změnu obsahu stránky dle událostí, které uživatel vyvolává klávesnicí nebo myší. Na základě těchto událostí se na stránce generují změny, jako například změna obrázků v galerii, rozbalení hlavní nabídky menu s animací a zvukem, zobrazení

kalendáře, změna zobrazeného textu dle volby uživatele, změna stylu stránky a další. Nejčastěji se zde využívá JavaScript.

Získání dat z těchto stránek je obtížné, protože k některým datům se není možné dostat jinak, než vyvoláním patřičné události pro jejich zobrazení. Taková data nejsou součástí dané stránky, ale jsou generována ze zdrojového kódu JavaScriptu. Někdy jsou ale data součástí HTML dokumentu a na základě událostí uživatele se za pomocí JavaScriptu zobrazí. V tomto případě se data dají bez větších problémů získat.

1.2.2 Generování obsahu stránek na straně serveru

Pro zobrazení stránek tohoto typu je nutné stránky nejprve přeložit programem běžícím na serveru. Nejedná se přímo o HTML dokumenty, ale o soubory se zdrojovými kódy jazyka, ve kterém byly napsány, a teprve po přeložení se získá HTML dokument. Mezi jazyky, které se pro psaní těchto stránek používají nejčastěji, patří PHP, ASP, ASP.NET, Pearl a JSP.

Tyto jazyky často využívají CGI, což je protokol propojující externí aplikace a webový server. Tím je možné skript z webové stránky odeslat pro zpracování externí aplikaci, která serveru vrátí výsledek v podobě statické stránky. Tato stránka je následně zobrazena uživateli. Výjimkou jsou jazyky ASP.NET a JSP, které CGI nevyužívají, protože jeho koncept mají v sobě již zahrnutý.

PHP je skriptovací programovací jazyk jehož výstupem je HTML nebo XHTML stránka. Pro použití tohoto jazyka je nutné mít webový server, na kterém se stránky budou překládat. Nejčastěji se používá server Apache. Syntaxe jazyka je inspirovaná jazyky C, Pascal, Pearl a Java. Soubory mají nejčastěji koncovku `.php` a PHP skripty se zapisují mezi značky `<?php` a `?>`. [2]

ASP je skriptovací jazyk od společnosti Microsoft. Pro použití je potřebný webový server IIS. Programovací jazyky se pro ASP používají VBScript a JScript. Soubory mají koncovku `.asp`. Jeho nástupcem je ASP.NET, který je součástí .NET Framework od Microsoftu. Zdrojový kód je zde kompilovaný, čímž jsou aplikace rychlejší. Výsledkem je HTML stránka. [3]

JSP vyvinula společnost Sun Microsystems za účelem vytváření aplikací na straně serveru. Jde o speciální tagy, které se vkládají do HTML stránek. Tagy obsahují zdrojový kód jazyka Java. Server tyto tagy přeloží a uživateli zobrazí výsledek. Tím se vytváří

dynamický obsah stránek. JSP je vhodné použít v případě, že stránka je z větší části statická a jen některé její části se dynamicky mění. Jednotlivé části stránky, statická a dynamická, se tak přehledně oddělí. Výhodou JSP je, že používá programovací jazyk Java a není tak nutné se učit další programovací jazyk kvůli webovým stránkám. Java je multiplatformní jazyk a tak zde není omezení na operační systém nebo server. [4]

Podobně jako u získávání textu ze stránek generovaných na straně klienta, je i zde situace složitá. Získat lze vždy jen tu část, která je serverem vrácena, nedostaneme se tak ke všem datům. Nicméně v některých případech, kdy jsou požadavky na zobrazení generovaného obsahu předávány za pomocí parametrů v adrese stránky, je tak možné tato data získat. To, která data se nám zobrazí, ovlivňuje také identifikace zařízení, ze kterého k datům přistupujeme. To znamená, že se na počítači mohou zobrazit jiná data, než na mobilním zařízení. I toto má tedy vliv na získaná data.

1.3 Speciální typy stránek

Do této skupiny stránek řadíme například stránky vytvořené pomocí Adobe Flash. Textové informace obsažené ve Flash animacích se ze zdrojového kódu HTML dokumentu získat nedají, protože je zde uvedena pouze cesta k Flash animaci. Z tohoto umístění je při zobrazení stránky animace přehrána pomocí přehrávače Adobe Flash Player.

2 HTML DOKUMENTY

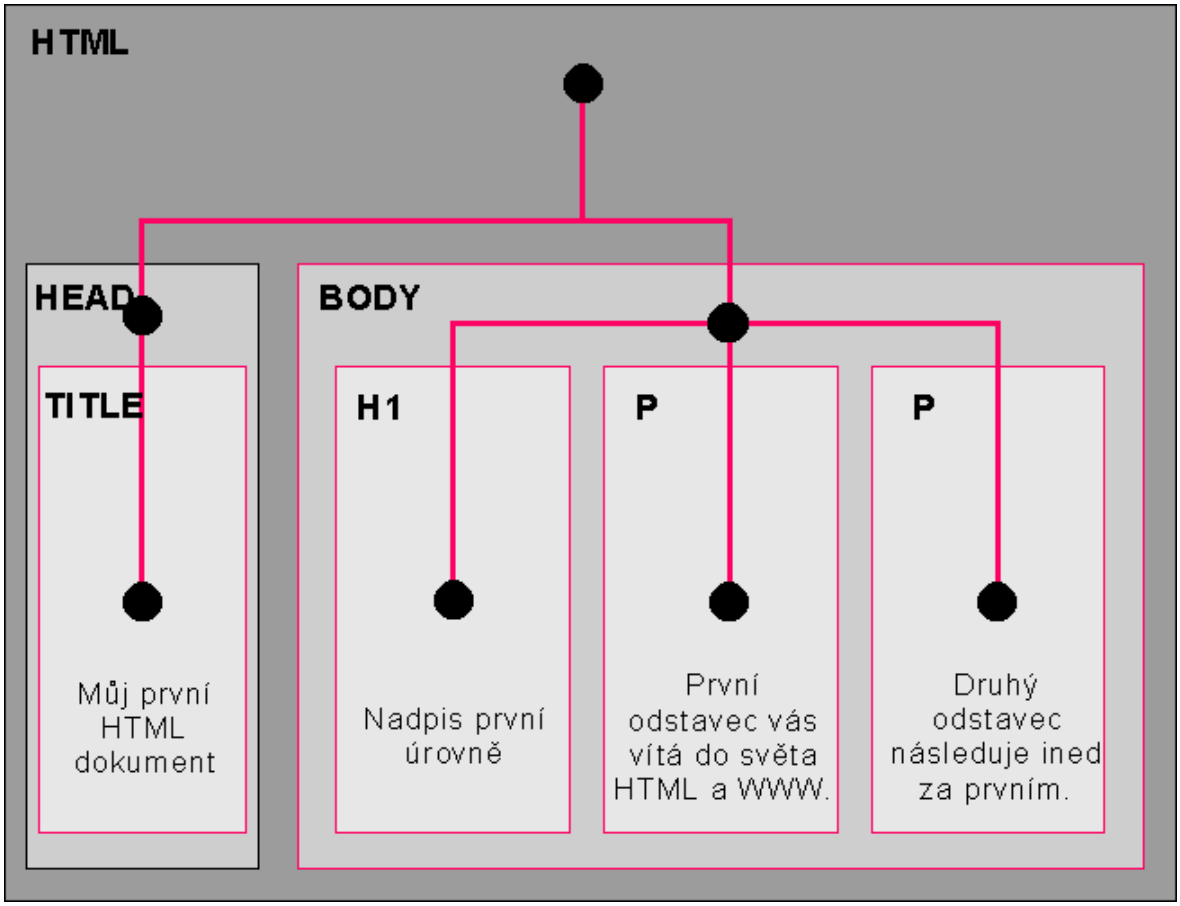
V této kapitole se věnuji HTML dokumentům, jejich struktuře a zápisu kódu. Znalost správné struktury dokumentu spolu se zápisem jeho kódu je pro správné získání dat velmi důležitá.

2.1 Struktura HTML dokumentů

Dokument HTML obsahuje stromovou hierarchii jednotlivých elementů. Na začátku je uvedena deklarace typu dokumentu. Nejvyšší úrovní je element HTML, který obsahuje dva elementy – HEAD (hlavička) a BODY (tělo dokumentu).

Hlavička obsahuje definici typu obsahu, znakové sady a stylu, dále obsahuje titulek stránky a jiné informace. Tělo dokumentu obsahuje elementy definující samotný obsah stránky. [5]

Na obrázku 1 je uvedena ukázka uspořádání HTML dokumentu. Hlavička obsahuje titulek stránky a tělo se skládá z nadpisu a dvou odstavců.



Obr. 1. Hierarchické uspořádání HTML dokumentu. [5]

2.2 Zápis HTML tagů

Základem HTML dokumentů jsou tagy, které dělíme na párové a nepárové. Párové tagy se skládají z dvojice tagů, počátečního a ukončovacího. Ukončovací tag se od počátečního liší tím, že obsahuje navíc lomítko, které nám říká, že tento tag je ukončovací. Tím je vymezena část, která představuje například nadpis, začátek a konec odstavce či bloku, nebo třeba odkaz. Párové tagy se v HTML i XHTML zapisují stejným způsobem. Párové tagy vypadají následovně:

```
<h1>Nadpis první úrovně</h1>

<p>Odstavec</p>

<div>Blok</div>

<a href="www.fai.utb.cz">Odkaz na web fakulty</a>
```

Nepárové tagy neobsahují ukončovací, ale pouze počáteční tagy. Patří sem například tagy pro obrázek, přechod na nový řádek nebo vodorovnou čáru. Na rozdíl od HTML, se v XHTML nepárové tagy zapisují mírně odlišně. Před ukončovací úhlovou závorkou je navíc vložená mezera a lomítko. Nepárové tagy vypadají následovně:

```
<img src="obrazek.jpg" alt="Titulek obrázku" />

<br />

<hr />
```

Pro získávání dat, jsou párové tagy složitější, protože není možné odebrat vše od počátečního po koncový tag. Tímto bychom přišli o důležitou část, která obsahuje text. Je proto nutné část mezi tagy ponechat, správně identifikovat a odstranit pouze počáteční a koncové tagy. U nepárových tagů se tagy odstraňují celé, protože netvoří pár a text není umístěn mezi tímto párem.

2.3 Speciální značky

V HTML kódu se mohou vyskytnout také speciální značky, které se z textu neodstraňují, ale naopak se nahrazují znaky, které zastupují. Tyto značky se používají z důvodu správného zobrazení některých znaků, vkládání pevné mezery a další. Těchto speciálních značek je velké množství.

Mezi nejčastěji používanou speciální značku patří ` ` zastupující pevnou mezeru. Dále se často používají značky `©` a `®` pro zobrazení znaku copyrightu a registrované ochranné známky.

Odstraněním těchto značek bychom přišli o data, která do textu patří. Internetový prohlížeč tyto značky automaticky nahrazuje znaky, které zastupují. Nicméně při dolování dat k tomuto nahrazení nedojde, protože program získá data tak, jak jsou napsána, tedy ve své původní podobě. Je tedy vhodné tyto značky nahradit správnými znaky, aby uživatel viděl získaná data ve správném formátu.

3 JAZYK C#

Aplikaci realizující navržené řešení dolování dat, jsem se rozhodl naprogramovat v programovacím jazyce C#. Jedná se o moderní objektově orientovaný programovací jazyk.

3.1 Historie

Programovací jazyk C# vytvořila společnost Microsoft spolu s platformou .NET a je založený na jazycích C++ a Java. Verze 1.0 měla premiéru v roce 2001 spolu s .NET Frameworkem 1.0. Obsahovala základní podporu objektového programování.

Další verze 2.0, vydaná koncem roku 2005, přinesla několik důležitých nových prvků, včetně generických typů, iterátorů, anonymních metod a dalších.

Koncem roku 2007 byla vydána verze 3.0 včetně .NET Frameworku 3.5 a Visual Studia 2008. Tato verze přidala například rozšiřující metody, lambda výrazy a hlavně technologii LINQ.

Aktuální verze 4.0 vydaná v dubnu 2010 se zaměřuje na zlepšení interoperability s ostatními jazyky, technologiemi a spolupráci s dynamickými aspekty programování. Dále pak podpora pro pojmenované a volitelné argumenty. Spolu se C# 4.0 byl vydán také .NET Framework 4.0, který obsahuje řadu přídavků z nichž jsou nejvýznamnější třídy a typy tvořící knihovnu TPL. Pomocí této knihovny je možné vytvářet vysoce škálovatelné aplikace, které snadno využijí potenciálu vícejádrových procesorů. Byla také rozšířena podpora pro webové služby. Vydána byla také nová verze vývojového prostředí Visual Studio 2010. [6]

3.2 Vlastnosti jazyka

Jazyk C# je case-sensitive, čistě objektově orientovaný, podporuje vývoj založený na komponentách a má přímou podporu základních konstrukcí – vlastností, metod a událostí. Podobně jako jazyk Java obsahuje pouze jednoduchou dědičnost s možností násobné implementace rozhraní. Je zde automatická správa paměti, zpracovávání chyb pomocí výjimek a spolupráce s existujícími komponentami. Správné uvolňování zdrojů aplikace obstarává garbage collector. Jazyk C# integruje verzování komponent, XML dokumentaci

a zajišťuje typovou bezpečnost. Podporuje metadata, atributy, delegáta, přetěžování operátorů, konverze a další.

Jazyk C# zajišťuje zpětnou kompatibilitu se stávajícím kódem jak na binární úrovni, tak i na zdrojové. Většina uvedených vlastností vychází přímo z .NET Frameworku. [6]

3.3 Vývojová prostředí

Vývojových prostředí pro jazyk C# je více. Oficiální vývojové prostředí je Visual Studio od společnosti Microsoft. Toto prostředí nabízí několik edic, z nichž základní edice Express je k dispozici zdarma. Tato edice je určena zejména začátečníkům a příležitostným programátorům. Další edice jsou Professional, Premium a nejvyšší, ale také nejdražší je edice Ultimate. Studenti mají možnost získat, pokud je jejich škola zapojena do programu Microsoft Developer Network Academic Alliance, toto vývojové prostředí zcela zdarma v libovolné edici, včetně nejvyšší edice Ultimate.

Dalším vývojovým prostředím je Turbo C# Explorer od společnosti Borland. K Open Source nástrojům patří například SharpDevelop a MonoDevelop.

Pro naprogramování aplikace realizující navržené řešení dolování dat jsem využil vývojového prostředí Visual Studio 2010 od společnosti Microsoft.

4 DATA MINING

Pojem data mining (dolování dat) je znám již od 60. let minulého století, kdy se objevil spolu s rozvojem výpočetní techniky. Jednalo se o využívání regresní analýzy a první rozhodovací stromy. Od té doby prošel data mining značným vývojem.

Jedná se o objevování souvislostí v datech, které jsou skryté, za pomocí analýzy dat. Někdy se tato činnost označuje také jako dobývání znalostí z databází, případně jako prediktivní analýza. Data mining je schopen generalizace, to znamená, že nalezené závislosti by měly být obecné a platné pro všechna data stejného charakteru.

Pro dolování dat se využívají metody – algoritmy, které automaticky získávají data z dokumentů, databází, souborových systémů a jiných datových zdrojů. Tyto metody jsou netriviální, založené na statistice a mnohdy obsahují prvky umělé inteligence a strojového učení. Všechny metody mají společný cíl, kterým je prezentace získaných dat v co možná nejsrozumitelnější podobě pro uživatele. [7], [8]

K data miningu patří také text mining (dolování dat z textu). Jeho časté uplatnění je například při získávání dat z textových dokumentů, analýzách elektronické pošty, odhalování spamu a další. Dále k data miningu patří Web mining, který se zabývá získáváním dat z webových stránek.

4.1 Text mining

Dolování dat v textech má za úkol identifikovat a analyzovat užitečné informace v textech, které jsou pro uživatele zajímavé. Cílem dolování v textech je usnadnění vyhledávání informací a dalšího zpracování. Takto získané informace mohou vést ke zvýšení ziskovosti, efektivity práce a další.

Text mining je činnost, která má dvě fáze. V první fázi dojde k předzpracování, kdy se vstupní data převádí do podoby, která se následně používá pro další zpracování. To znamená, že se ze vstupu získá pouze text a ostatní části jako například obrázky jsou vynechány. U textu se může ponechat jeho struktura, která může pomoci dalšímu zpracování. Ve druhé fázi dochází k získávání znalostí za pomocí analýzy získaných dat. Dále dochází k zařazení dokumentu do kategorie, získání abstraktu a další.

Každý dokument je jiný, a tak jeho kvalitní předzpracování je klíčové pro získání hodnotného výsledku. Předzpracování dokumentů ovlivňuje nejvíce jazyk dokumentu,

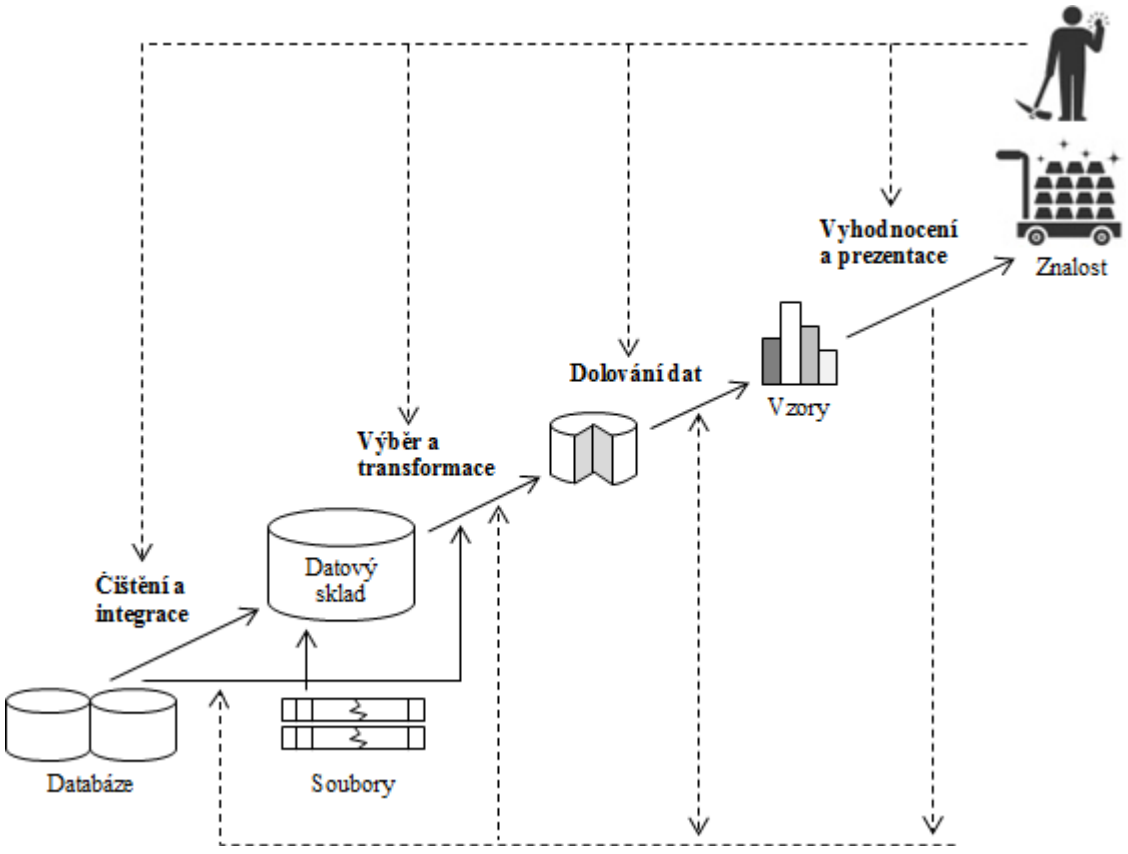
protože každý jazyk má jinou syntaxi. Dále pak znaková sada dokumentu, jeho formát (například MS Word, PDF, HTML), kategorie (noviny, vědecké články, webové stránky, aj.), požadovaná úroveň a oblast zájmu získaných dat. [9], [10]

4.2 Web mining

Web obsahuje velké množství dat, které se neustále zvětšuje. Nalézt v tomto množství dat požadované informace je čím dál tím složitější. Zde nám pomůže web mining, který se zabývá dolováním dat v prostředí webu. Dále se z těchto dat získávají zajímavé informace. Jedná se například o nejrůznější statistické údaje, jaké stránky uživatelé navštěvují a v jaké posloupnosti, co nejčastěji na internetu hledají a další zajímavé informace. [11]

4.3 Proces získávání znalostí

Proces získávání znalostí má několik fází, které se mohou v určitých cyklech opakovat. Celý postup je znázorněn na obrázku 2.



Obr. 2. Proces získávání znalostí [7]

Fáze čištění a integrace provede vypořádání se s chybějícími daty, vyřeší možné nekonzistence a spojí data pocházející z různých zdrojů. Výsledek se uloží do datového skladu. Dále se ve fázi výběru a transformace vyberou jen relevantní data, která se převedou do potřebného tvaru pro dolování.

Následuje hlavní fáze celého procesu – dolování dat, ve které se za použití patřičných metod dolují data. Poslední fáze vyhodnocení a prezentace hodnotí ze získaných dat části, které dle užitečnosti nejvíce odpovídají požadovaným znalostem. Poté se prezentují získané znalosti uživateli.

4.4 Metody dolování dat

Metod dolování dat je velké množství. Jejich rozdíly spočívají v tom, jakým způsobem reprezentují hledané znalosti, jak jsou srozumitelné pro uživatele, jak složité shluky dokáží reprezentovat, jak jsou efektivní při klasifikaci nových případů a pro jaký typ dat jsou vhodné. Z tohoto důvodu je pro získání požadovaného výsledku nejdůležitější správný výběr metody.

Mezi často používané metody dolování dat patří klasifikace, predikce, shlukování, asociace, vzorkování a další.

4.4.1 Klasifikace

Podle [12] „Klasifikace je obecně proces, který umožní přiřazovat data na základě jejich vlastností do jistých tříd, kterých je konečný počet.“.

Dokument se třídí do předem definovaných cílových tříd, do kterých jsou data předkládána a automaticky zařazována. Výsledky se porovnávají se správnými a celý proces se opakuje. Ukončen je ve chvíli, kdy se dosáhne požadovaných výsledků a dokumenty se řadí do správných tříd.

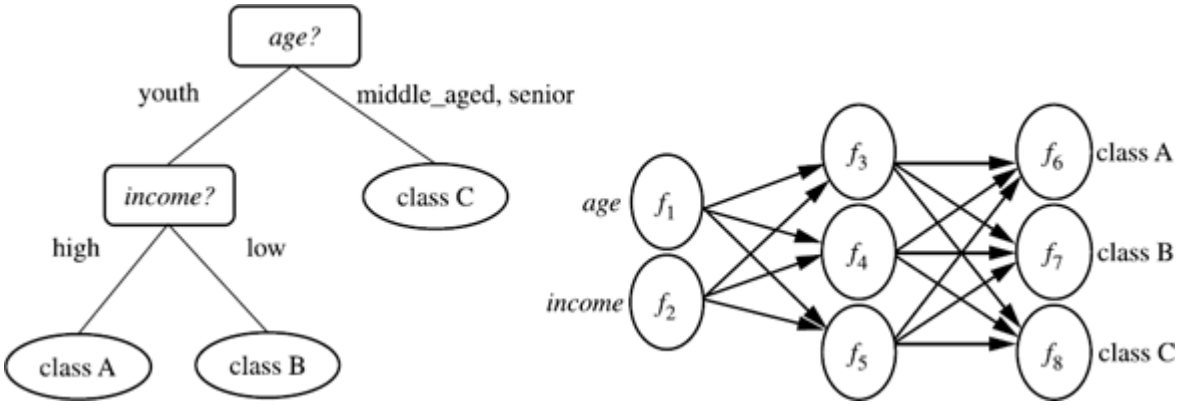
Rozlišujeme klasifikaci pomocí rozhodovacích stromů, Bayesovskou klasifikaci, klasifikaci pomocí neuronové sítě Backpropagation a další.

Rozhodovací stromy jsou stromovou strukturou, ve které se data postupně rozdělují na menší a menší podmnožiny, dokud v těchto podmnožinách nebude převládat jedna třída. Do ní jsou data zařazena.

U Bayeovské klasifikace se podle statistických metod určí největší pravděpodobnost, ze všech tříd, že daný vzorek dat do této třídy patří a bude do ní zařazen.

Neuronová síť Backpropagation je tvořena několika vrstvami neuronů. Do první vrstvy se přivedou vstupní data a neurony je šíří do dalších, tzv. skrytých, vrstev. Poslední – výstupní vrstva obsahuje třídu, do které mají být data zařazena.

Obrázek 3 zobrazuje ukázku klasifikační metody za pomocí rozhodovacího stromu a neuronové sítě.



Obr. 3. Klasifikace pomocí rozhodovacího stromu a neuronové sítě [7]

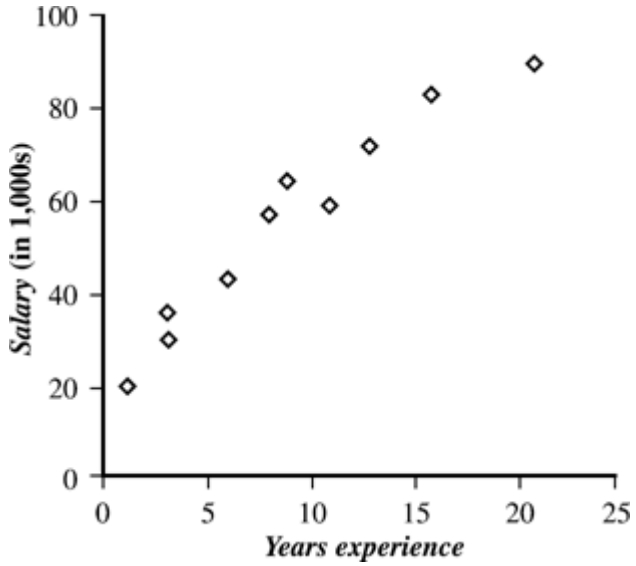
4.4.2 Predikce

Podle [12] „Predikce je proces, který umožní přiřazovat datům hodnoty, které mají obecně spojitý charakter.“.

Dle zadané množiny vstupních hodnot a jim odpovídajících výstupních hodnot se hledá nejpravděpodobnější hodnota výstupu pro předem neznámé kombinace vstupních hodnot.

Nejznámějšími metodami predikce jsou lineární jednoduchá a vícenásobná regrese. Regrese je statistická metoda, jejímž úkolem je pro dané vstupní hodnoty odhadnout neznámé spojité výstupní hodnoty.

Lineární jednoduchá regrese očekává data jako kombinace vždy jedné vstupní a jedné výstupní hodnoty. Výstup je možné zobrazit v podobě grafu, kde na ose X jsou vstupní hodnoty a na ose Y jsou výstupní hodnoty. Jejich výsledkem bude přímka. Toto je znázorněno na obrázku 4, kde je vstupem délka praxe v letech a výstupem výše ročního platu v tisících dolarů.



Obr. 4. Lineární jednoduchá regrese [7]

U vícenásobné regrese se očekávají data jako kombinace všech možných vstupních hodnot pro jednu výstupní. Podle těchto kombinací se bude předpovídat nejpravděpodobnější výstupní hodnota.

4.4.3 Shlukování

Podle [12] „Shlukování je proces rozdělování objektů do tříd (clusterů) na základě podobnosti objektů. Jednotlivé třídy jsou potom tvořeny objekty, které jsou si navzájem hodně podobné, a zároveň nejsou příliš podobné objektům z jiných tříd. Podobnost objektů se posuzuje na základě hodnot jednotlivých atributů objektů a často se využívají tzv. vzdálenostní funkce. V mnoha aplikacích potom můžeme objekty, které patří do stejné třídy zpracovávat hromadně.“.

Počet tříd může být dopředu znám. Shlukování může být využíváno i jako příprava dat pro klasifikaci, která s vytvořenými třídami dále pracuje.

Mezi metody používané u shlukování patří metody založené na rozdělování, hierarchii, mřížce a další.

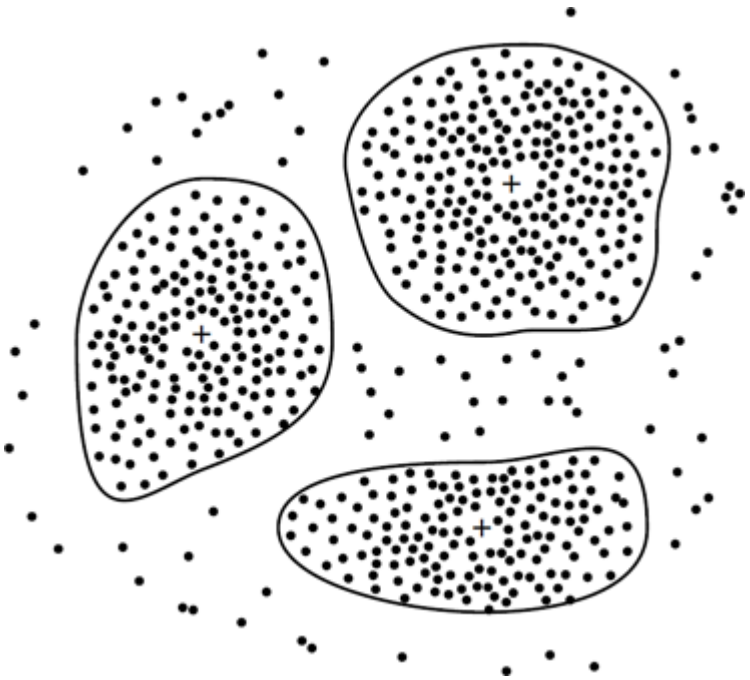
U metody založené na rozdělování musíme znát počet tříd, do kterých se budou objekty rozdělovat. Vždy musí platit, že počet objektů je větší než počet tříd, protože každý objekt může patřit pouze jedné třídě a zároveň každá třída musí mít minimálně jeden objekt. Na začátku se vyberou objekty, které jsou nejvíce podobné daným třídám a ty následně reprezentují. Zbylé prvky se za pomoci vzdálenostní funkce rozdělí do těchto tříd.

Dále se objekty mezi třídami přesouvají tak, aby podobnost objektů v každé třídě byla maximální a zároveň byla podobnost mezi různými třídami minimální.

Principem hierarchické metody je rozklad dané množiny objektů. Rozlišujeme postup zdola-nahoru, kde se každý objekt umístí do vlastní třídy, které se následně slučují. Dále postup shora-dolů, kde se všechny objekty umístí do jedné třídy a ta se postupně rozděluje. Uživatel si může určit počet tříd, do kterých budou objekty rozděleny, tím se určí podmínka pro ukončení. Pokud není určen počet tříd, tak metoda končí spojením nebo rozdělením všech tříd. Rozdělování nebo spojování tříd probíhá podle vzdálenosti mezi jednotlivými shluky. Nevýhodou této metody je, že jednou sloučené nebo rozdělené třídy již nelze opětovně rozdělit nebo spojit.

Základem metody založené na mřížce je mřížka, která tvoří konečný počet buněk. Nad touto mřížkou probíhají všechny operace shlukování. Metoda vyniká svojí rychlostí, nezáleží na počtu objektů ale jen na počtu buněk mřížky.

Ukázka výsledku shlukování je uvedena na obrázku 5. Jsou zde znázorněny tři shluky, jejichž střed je vyznačen křížkem. Body mimo vyznačené shluky jsou tak zvané odlehlé hodnoty.



Obr. 5. Výsledek shlukování [7]

4.4.4 Asociace

Jde o metodu, která na základě asociačních pravidel zjišťuje pravděpodobnost výskytu více položek společně. Tohoto se často využívá v marketingu a příkladem jsou prodejní analýzy, kdy se zkoumají zvyklosti nakupování jednotlivých předmětů. Například zda si zákazník koupí spolu se zbožím A i zboží B.

4.4.5 Vzorkování

Jedná se o jednu ze základních technik dolování dat, která umožňuje získat výsledek v rozumném čase. Tohoto je dosaženo náhodným výběrem, čímž se zmenší objem zpracovávaných dat a zrychlí se výpočty. Jedná se tak o jednodušší způsob vzorkování s menší přesností. Pro dosažení vyšší přesnosti výsledku slouží složitější metody vzorkování, například výběr stejného počtu záznamů daného typu. Tímto se zmenší objem zpracovávaných dat, ale zachová se požadovaná přesnost.

4.5 Využití dolování dat

Dolování dat se dá využít téměř ve všech oblastech. Například v marketingu, ve výrobě, v investicích, ve zdravotní péči, při detekci podvodů, ve vědě a další. Uživatelům pomáhá při získávání informací zpracováváním rozsáhlých textů, poskytuje statistiky společných výskytů určitých slov, frází a pojmů.

Za pomoci klasifikace lze například vytipovat zákazníky, kteří budou reagovat na konkrétní reklamní nabídku a tu jim následně zaslat. Tímto dojde ke snížení nákladů na reklamní kampaň.

Pomocí shlukování lze vytipovat vysoce ziskové zákazníky, na které se je možné následně zaměřit.

Dalším příkladem data miningu je analýza nákupů, bez ohledu na způsob prodeje. Tím je možné zjistit, co si zákazníci nejčastěji kupují a na toto zboží se více zaměřit, uspořádat zboží v regálech, doplnit nabídku elektronického obchodu a další.

Je možné také hledat různé anomálie, jako jsou pojistné podvody, daňové podvody, podvody v sociální sféře a další. Na základě takto získaných dat lze provádět následná opatření.

Data mining se využívá také v bankovnictví, kdy na základě informací o uživateli banka rozhodne například o přidělení úvěru. Využití nalézá také v pojišťovnictví, zdravotnictví, státní správě, telekomunikacích a ve službách.

Jak je vidět, tak využití data miningu je opravdu velké. V dnešní době, kdy informace přibývají čím dál tím rychleji a je potřeba je zpracovávat v co nejkratších časech s co nejlepšími výsledky, má využití dolování dat velké uplatnění.

II. PRAKTICKÁ ČÁST

5 NÁVRH APLIKACE

Při navrhování každé aplikace je důležité správně určit cíle, kterých se při návrhu aplikace budeme držet, porovnat jednotlivé možnosti návrhu a zaměřit se na možné problémy, které mohou nastat.

5.1 Obecné cíle

Hlavním cílem vytvořené aplikace byl snadný způsob získání požadovaných dat ve správné podobě s co nejmenším množstvím balastních informací.

Dále jsem se zaměřil na vytvoření maximálně přehledného uživatelského prostředí s jednoduchým ovládáním tak, aby i neznalý uživatel snadno a rychle pochopil, jak se aplikace ovládá.

Důležitou částí je také informovat uživatele o vzniklých chybách nebo opomenutích za pomocí informačních a chybových hlášek. Ty se snaží přehledným a stručným způsobem popsat vzniklý problém. Uživatel je tak informován o vzniklé situaci a je mu sděleno, jak má dále postupovat.

5.2 Možné problémy

Při návrhu aplikace je nutné vzít v potaz faktory, které ovlivňují nejen vstupní, ale i výstupní data a zvážit jednotlivé možnosti řešení.

Nejčastějším faktorem ovlivňujícím vstupní data, je správné určení znakové sady dokumentu tak, aby získaný text byl bez problémů čitelný. Znakových sad je velké množství, proto se může jejím chybným určením stát, že se některé znaky budou chybně zobrazovat a budou tak ztěžovat nebo dokonce znemožňovat čtení získaných informací.

Dalším faktorem ovlivňujícím vstup je identifikace klienta. Pro získání HTML dokumentu z internetové adresy se aplikace serveru identifikuje a ten dle této identifikace odešle data. V dnešní době se vytvářejí stránky, které mají odlišnou podobu pro různá zařízení. Tohoto se využívá například pro vytvoření jednodušší verze stránek pro mobilní telefony, tablety nebo jiná přenosná zařízení. Stránky tak mají podobu upravenou pro snazší čtení na menším displeji, menší velikost pro rychlejší načítání apod. Může se tak stát, že aplikace obdrží data v jiné podobě, než v jaké jsou k dispozici za pomocí běžného internetového prohlížeče.

Internetové stránky běžně obsahují více jazykových verzí. Často je výchozí jazyková verze stránek nastavována dle identifikace klienta a aplikace tak může obdržet stránku v jiném jazyce. Možným řešením je doplnění zadané internetové adresy o parametr, který označuje požadovaný jazyk dle formátu použitého na dané stránce.

Faktor, který ovlivňuje jak vstupní, tak i výstupní data je například nekorektní formát HTML dokumentu. To může být způsobeno například poškozením souboru, kdy část jeho struktury chybí. Tím pádem je výstup z aplikace ovlivněn schopností aplikace vypořádat se s takto poškozeným vstupním dokumentem. Při návrhu aplikace jsem se proto zaměřil také na správné získávání dat i při chybějící části struktury dokumentu, chybějících HTML značkách nebo jiných poškozeních zdrojového dokumentu.

Protože HTML dokument může mít jednotlivé části zapsané různým způsobem, kdy například jednotlivé odstavce jsou odděleny pouze použitým stylem, tak odebráním formátovacích značek dojde ke spojení výstupu v jeden blok textu. Toto jsem se v aplikaci rozhodl ošetřit, a tak jsou ještě před odstraněním formátovacích značek do textu vloženy prvky, které způsobí, že výsledek bude správně formátován.

5.3 Odstranění nepotřebných značek

Každý HTML dokument obsahuje značky, které formátují obsah dokumentu, vkládají různé skripty, animace nebo jiné prvky. V dokumentu mohou být také komentáře ke zdrojovému kódu, které nejsou běžným prohlížením dokumentu v internetovém prohlížeči viditelné a slouží pouze pro vývojáře, aby věděl, k čemu která část kódu patří. Je proto vhodné odstranit také tyto komentáře.

Způsobů odstranění těchto značek je několik. Nejpomalejším způsobem je odstraňování značek dle jejich názvu. Tímto způsobem se dokument prochází pro každou značku zvlášť. Vzhledem k tomu, že těchto značek je velké množství (kolem 100 značek), tak tento proces trvá delší dobu.

O něco rychlejším způsobem je odstraňování značek dle jejich počátečního písmene, to znamená, že se dokument prochází a při nalezení zadaného písmene se zkontroluje, jestli toto písmeno tvoří HTML značku nebo ne. Pokud ano, tak je z dokumentu odstraněno. Tento proces je o něco rychlejší, protože písmen anglické abecedy, kterými značky mohou začínat je 20. Ani tento způsob není ale z hlediska rychlosti optimální.

Nejlepším způsobem je postup, kdy se použijí regulární výrazy, kterými se specifikuje formát HTML značek a dokument se prochází pouze jednou. Vše co odpovídá zadanému regulárnímu výrazu je z dokumentu odstraněno. Tento postup je sice nejrychlejší, ale zároveň je nejsložitější na implementaci. Je totiž důležité správně specifikovat zadaný výraz, aby se nestalo, že z dokumentu budou odstraněny i ty informace, které jsou pro uživatele podstatné.

Ve své aplikaci jsem se rozhodl využít způsobu odstraňování HTML značek za pomocí regulárních výrazů. Tento postup považuji za nejvhodnější.

5.4 Získání dalších užitečných informací

Při návrhu vytvářené aplikace jsem se zaměřil také na možnosti získání dalších užitečných informací ze vstupního HTML dokumentu. Rozhodl jsem se toto řešit způsobem, kdy si uživatel vybere HTML prvky, ze kterých chce informace získat a zadá požadovaný výraz pro získání informací.

Získané informace bude možné dále zpracovávat v jiné aplikaci nebo je uložit do souboru pro pozdější použití.

Podrobněji se způsobu a možnostem získání dalších užitečných informací věnuje kapitola 6.3.

6 REALIZACE NAVRŽENÉ APLIKACE

Zdrojová část vytvořené aplikace se skládá z celkem pěti tříd. Hlavní třídou je `MainWindow`, která obsahuje základní okno programu a metody, které se zde používají, včetně metod pro získávání dat z HTML dokumentů a požadovaných informací dle uživatelem zadaných podmínek.

Dalšími důležitými třídami jsou `OccurrenceWindow`, která se stará o vykreslování grafu četnosti slov a třída `SettingsWindow` starající se o nastavení programu. Poslední dvě třídy `ProgressWindow` a `AboutWindow` mají na starosti okna zobrazující postup aplikace a informace o programu.

6.1 Příprava dat

Po zadání vstupního HTML dokumentu a stisknutí tlačítka Start se spustí metoda `loadPage`, která si nejprve ověří, že zadaný vstupní dokument existuje, případně že zadaná internetová adresa je ve správném tvaru, a pokud je vše v pořádku, tak jeho obsah načte do proměnné pro další zpracování. Pokud dokument nebyl nalezen, nebo nastala chyba například při připojení k serveru, je uživatel informován zprávou. Při získávání dat se použije znaková sada, kterou uživatel nastavil v nastavení aplikace. Pokud uživatel zvolil automatickou detekci znakové sady, tak se ji aplikace při načítání dokumentu pokusí ze zdrojového kódu načíst a použít. K tomuto se použije metoda `getCharset`. Jestliže nebyla znaková sada nalezena, je použita znaková sada UTF-8.

Pokud je v nastavení aplikace zvoleno načítání titulku stránky, tak se použije metoda `getTitle`, která ve zdrojových datech vyhledá HTML tag obsahující titulek a v případě úspěchu jej uloží do proměnné, ze které je poté zobrazen v programu. V opačném případě se do proměnné uloží prázdný řetězec.

Další metoda, která se zavolá, je `getBody`. Tato metoda ze zdrojových dat odebere vše, kromě těla dokumentu. Tělo obsahuje ta data, která nás v dokumentu zajímají. Pokud není tělo nalezeno, je uživatel informován o nekorektním HTML formátu a proces je ukončen. V případě, že uživatel v nastavení aplikace zvolil možnost načítání vstupního dokumentu i v případě nekorektního formátu, je metoda ukončena a aplikace pokračuje dalšími metodami pro vyčištění dokumentu, získání požadovaných informací dle zadaného výrazu a spočítání četnosti slov.

6.2 Vyčištění dat

Připravená data je zapotřebí vyčistit od HTML značek, různých skriptů a dalších nepotřebných formátovacích prvků. Některé značky je naopak vhodné nahradit jinými, které výsledek zformátují do uživatelsky přívětivé formy. Jde například o nahrazení speciálních značek správným znakem, aby uživatel nepřišel o část informace, dále o zachování odstavců a další.

O toto se postará metoda nazvaná `removeTags`. Tato metoda postupně nahrazuje HTML značky, které se nebudou odstraňovat, značkami pro odsazení na nový řádek, pevnou mezerou, znaky pro copyright, registrovanou obchodní známku, úhlové závorky, ampersand a jiné. Dále se oddělí jednotlivé odstavce tak, aby se po odstranění HTML značek text nespojil v jeden celek a byla zachována původní struktura. V závěru metody se odstraní skripty a jejich komentáře. Kopie takto upravených dat se uloží do zvláštní proměnné pro pozdější použití, viz kapitola 6.3.

Nyní je zavolána metoda `removeHTMLandSpaces`, která provede pomocí regulárního výrazu odstranění nejen všech HTML značek, ale také všech HTML komentářů. Je zde použita volba, která ignoruje velikost písmen, proto je jedno, jak jsou značky zapsány. Použitý regulární výraz má následující podobu:

```
<[a-z|/|!--]+(.|\n)*?>
```

Tento výraz hledá HTML značky dle jejich specifického tvaru. Ten začíná levou úhlovou závorkou, dále následují buďto libovolné znaky abecedy, lomítko, nebo vykřičník se dvěma pomlčkami, označující HTML komentář. Dále mohou, ale také nemusí následovat libovolné znaky nebo nový řádek. Jakmile bude nalezena pravá úhlová závorka, je HTML značka ukončena. Tímto způsobem regulární výraz vymezí HTML značku, která je následně nahrazena mezerou. Ta je zde proto, že někdy není mezi HTML značkou a dalším textem mezera. Odstraněním této značky by se předešlý a následující text spojil a byl by tak hůře čitelný. Proces nahrazování HTML značek mezerou se opakuje až do dosažení konce vstupních dat.

Následně se odstraní přebytečné mezery a prázdné řádky. Ponechány jsou pouze případné oddělující řádky jednotlivých odstavců nebo jiných částí textu. Tím jsou data vyčištěna a zobrazena uživateli v hlavním textovém poli aplikace. Je možné zobrazit četnost slov, zadat výraz a získat požadované informace nebo výsledek uložit do souboru.

6.3 Získání informací dle zadaného výrazu

Důležitou funkcí aplikace je možnost zadat výrazy, které nás zajímají. Na jejich základě budou ze získaného textu zobrazeny pouze ty informace, které tomuto výrazu odpovídají. Pokud chceme například z internetové stránky věnující se sportu získat informace, které obsahují hokej nebo fotbal, zadáme požadovaný výraz s logickým operátorem OR a aplikace nám z textu zobrazí ty části, které se věnují hokeji nebo fotbalu. Takto získaná data je možné za použití dalších nástrojů dále zpracovávat.

Tento úkol provádí metoda `getInformation`. Na začátku se provedou nutné úpravy, které uživatelem zadaný výraz převedou do podoby regulárního výrazu. Za pomocí tohoto výrazu se budou následně získávat požadované informace. Tvar vytvořeného regulárního výrazu je například pro výše zmíněný výraz hokej nebo fotbal následující:

```
hokej|fotbal
```

Dále se dle zvolených HTML značek vytvoří druhý regulární výraz, na jehož základě jsou pak při procházení textu identifikovány uživatelem zvolené HTML značky. Pokud byly uživatelem vybrány HTML značky například pro odstavce a nadpisy, bude mít vytvořený regulární výraz tuto podobu:

```
(<p(.|\n)*?</p>)|(<h([1-6]){1}(.|\n)*?</h([1-6]){1}>)
```

Po těchto potřebných přípravách se začne se samotným získáváním požadovaných informací. Zde se jako zdroj dat využije již dříve vytvořená kopie upravených vstupních dat, viz kapitola 6.2. Tato data je nutné použít z důvodu, že obsahují HTML značky, podle kterých se určují prvky, ze kterých se mají požadované informace získat. Celý proces probíhá tak, že na základě vytvořeného regulárního výrazu pro HTML značky se najde první odpovídající prvek, který je prohledán na uživatelem zadaný výraz. Pokud prvek obsahuje zadaný výraz, je obsah celého prvku uložen do proměnné a celý proces se opakuje pro další nalezený prvek.

Jakmile jsou zkontrolována všechna vstupní data, je na získaný výstup zavolána metoda `removeHTMLandSpaces`, která odstraní HTML značky. Ty byly potřeba pouze pro identifikaci prvků pro získání informací. Také jsou odstraněny přebytečné mezery a prázdné řádky. Výsledek je zobrazen uživateli.

Je zde také možnost výraz upravit a získat jiné zajímavé informace, nebo se vrátit k původnímu získanému textu z HTML dokumentu bez nutnosti jej znovu zpracovávat.

6.4 Četnost slov

Četnost slov je ve vytvořené aplikaci implementována tak, že je možné, počítat nejen četnost slov v získaných datech, ale uživatel si může také specifikovat vlastní slova nebo výrazy, které se budou počítat. Toto je možné zvolit v nastavení programu. Podrobněji jsou jednotlivé možnosti nastavení četnosti popsány v kapitole 6.5.2.

Pro četnost je v aplikaci použita kolekce typu slovník. Jedná se o datovou strukturu, uchovávající jednotlivé prvky v podobě dvojic, které obsahují klíč a hodnotu. Funguje to tak, že pod unikátním klíčem je uložena hodnota. Unikátní klíč zde zastupuje slova nebo výrazy a hodnota jejich počet.

Metoda pro počítání četnosti všech slov se jmenuje `countOccurrence`. Jednotlivá slova jsou vyhledávána za pomocí regulárního výrazu, který vyhledává libovolná malá nebo velká písmena, číslice a podtržítka. Ve výrazu je místo znaku `x` zadáno číslo, které určuje minimální délku vyhledávaného slova. Regulární výraz má tvar:

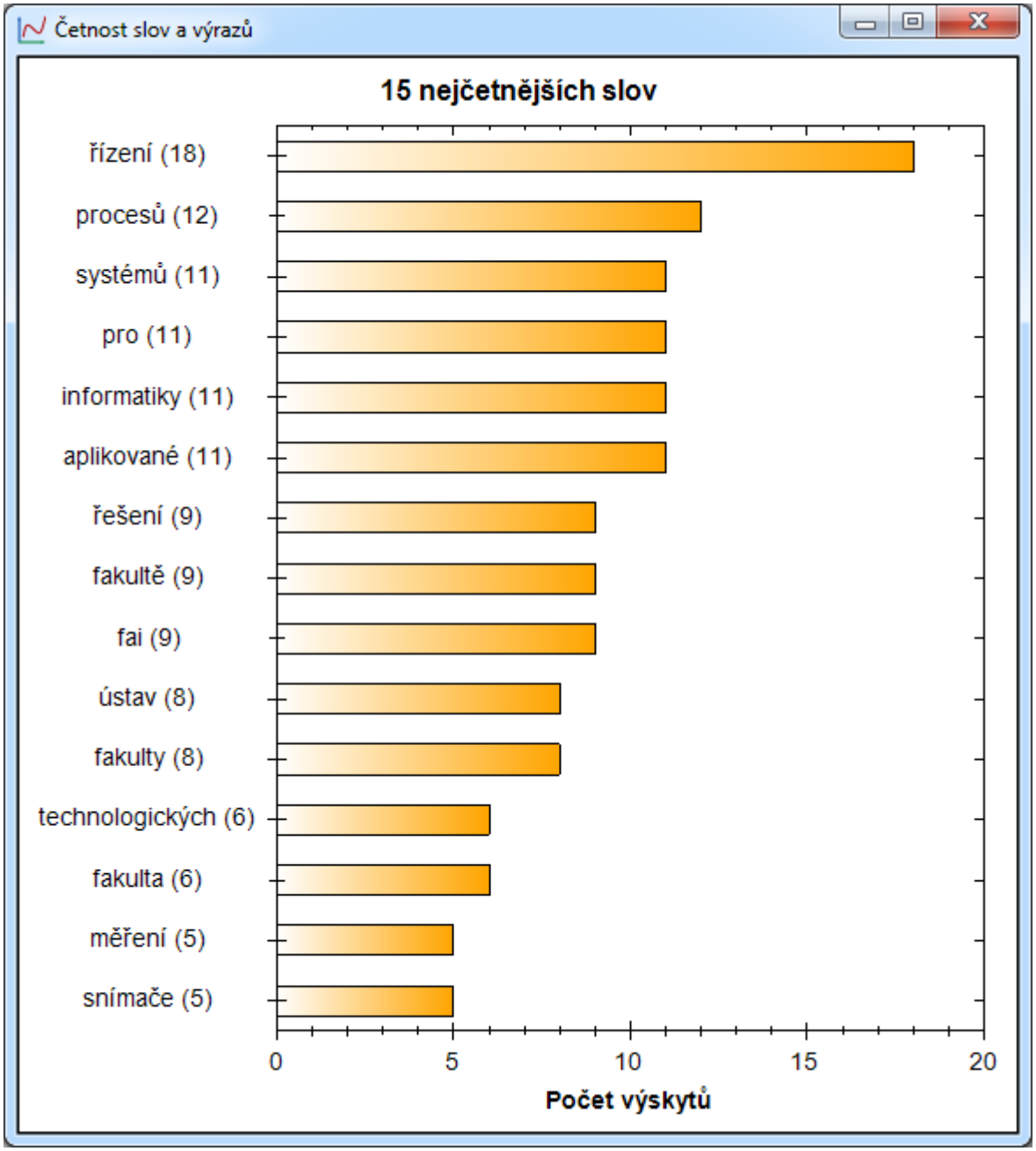
```
\w{x,}
```

Vždy se provede kontrola, zda nalezené slovo již není ve slovníku obsaženo. Pokud ano, tak je jeho hodnota zvýšena o 1, pokud ne, tak je do slovníku uloženo a je mu nastavena hodnota 1. Takto se projde celý text.

Pro druhý způsob, kdy se počítají uživatelem definovaná slova nebo výrazy, se používá metoda `countOccurrenceUser`. Protože hledaná slova nebo výrazy pro četnost jsou uživatelem definovány, není potřeba použít regulární výraz. Metoda vyhledá a spočítá všechny výskyty zadaného slova nebo výrazu a výsledek uloží do slovníku. Je tak nutné, pro každé zadané slovo nebo výraz, tuto metodu zavolat zvlášť.

Jakmile jsou všechna slova spočítána, tak se slovník seřadí dle počtů jednotlivých slov a grafický výsledek je zobrazen uživateli v novém okně. Každé slovo je na samostatném řádku grafu včetně slov, která mají stejnou četnost. Stejná slova tedy nejsou seskupena. K tomuto řešení jsem se rozhodl z důvodu zachování přehlednosti grafu.

Na obrázku 6 je vidět grafické zobrazení četnosti slov ze stránky Fakulty aplikované informatiky UTB Zlín, z části O fakultě – Profil fakulty. Levá část grafu obsahuje jednotlivá slova spolu s jejich počtem v textu, který je uveden v závorce. V nastavení programu si je možné nastavit barvu grafu, viz kapitola 6.5.2.



Obr. 6. Grafické zobrazení četnosti slov

Graf je možné za pomocí kolečka myší zoomovat. Po stisku pravého tlačítka myši v okně grafu, se uživateli zobrazí kontextové menu, které obsahuje několik možností. Je zde kopie grafu do schránky, uložení grafu jako obrázek do některého z nabízených grafických formátů, možnost nastavení stránky pro tisk a samotný tisk grafu. K dalším možnostem patří například možnost zrušit navolený zoom nebo nastavit výchozí měřítko.

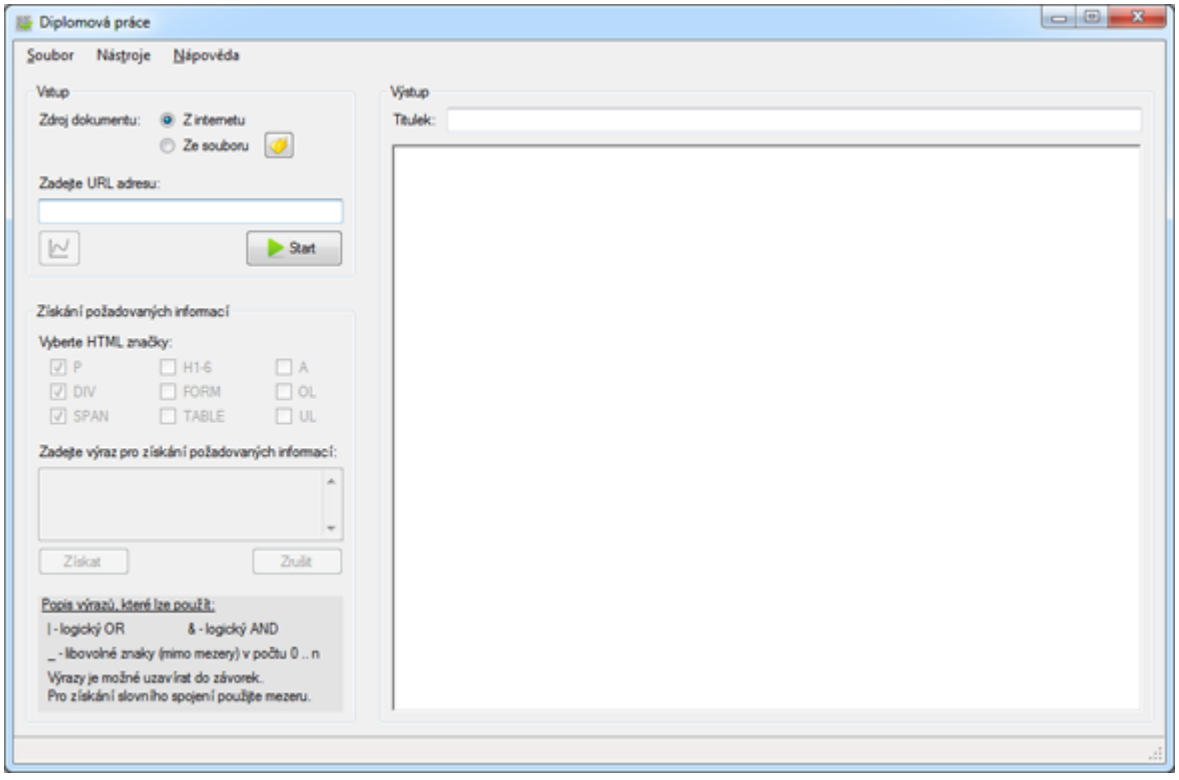
Protože vývojové prostředí Visual Studio neobsahuje komponenty pro zobrazení grafů, použil jsem knihovnu ZedGraph. Jedná o open-source projekt.

6.5 Uživatelské rozhraní aplikace

Uživatelské rozhraní vytvořené aplikace jsem se snažil koncipovat tak, aby bylo přehledné, jednoduché a snadné na pochopení. Vše je v aplikaci pěkně a srozumitelně popsané. S obsluhou aplikace by tak neměl mít problém ani neznalý uživatel, protože uspořádání jednotlivých komponent je rozvržené tak, aby uživatele navádělo, jak má postupovat při ovládání aplikace.

6.5.1 Základní okno aplikace

Na obrázku 7 je znázorněno rozvržení základního okna vytvořené aplikace. Z tohoto obrázku je patrné, že aplikace je rozvržena přehledně a uživatel dle srozumitelných popisků pozná, k čemu co slouží. Ovládání je jednoduché a zvládne jej i začátečník.



Obr. 7. Rozvržení základního okna vytvořené aplikace

V horní části aplikace je umístěna hlavní nabídka, která se skládá ze tří základních částí. První je nabídka Soubor, která obsahuje základní operace, jako je založení nového dokumentu, kdy dojde ke smazání všech získaných dat z okna programu s upozorněním na neuložená data. Dále je zde otevření HTML souboru pro získání dat, uložení získaných informací do souboru a ukončení aplikace. Druhou nabídkou jsou Nástroje, odkud je

možné zobrazit četnost slov a otevřít nastavení aplikace. Poslední nabídkou je Nápověda, která obsahuje odkaz pro zobrazení informací o programu.

Hlavní část okna aplikace je přehledně rozdělena do několika částí. Základní částí je část Vstup, ve které se definuje umístění vstupního dokumentu s daty, se kterým se v programu bude následně pracovat. Je možné zadat internetovou adresu nebo otevřít HTML dokument z počítače, USB Flash paměti nebo jiného umístění. Aplikace podporuje také zabezpečené internetové adresy za pomocí protokolu HTTPS. Po zadání adresy se stisknutím klávesy Enter nebo tlačítka Start aplikace spustí.

Další částí je část Výstup, která obsahuje pole titulek a velké textové pole pro zobrazení výsledku. Do titulku se načte titulek HTML dokumentu a do textového pole je načten získaný obsah HTML dokumentu bez formátovacích značek. Také se zde zobrazuje výsledek podle výrazu, který uživatel zadal v části filtrování.

Poslední část je pojmenovaná Získání požadovaných informací a obsahuje ovládací prvky pro nastavení parametrů získávání informací pro zadaný výraz. Nejprve je nutné zvolit HTML prvky, ve kterých se budou informace vyhledávat. Pokud nás tedy například zajímají nadpisy v dokumentu, tak zvolíme možnost H1-6, čímž se zobrazí pouze ty nadpisy, které odpovídají zadanému výrazu. Dále se zadá požadovaný výraz. Ve výrazu se mohou používat logické operace OR a AND, které se zapisují pomocí znaků | a &. Pomocí logického OR je možné získat informace, obsahující alespoň jeden ze zadaných výrazů a pomocí logického AND se získávají informace, které obsahují oba výrazy současně. Tyto výrazy je možné také uzavírat do závorek a vytvořit tak složený výraz. Je možné zadat znak podtržítka pro nahrazení libovolných znaků. Tím se docílí, že zadáním například `t_t` dojde k vyhledávání slov, která začínají a končí písmenem t, to znamená text, test, trest, apod. Pokud ve výrazu zadáme slovo bez mezery na konci, je to stejné jako bychom toto slovo zadali s podtržítkem na konci. Aplikace prohledává data dle zadaného výrazu, který může být nalezen i v části jiného slova. Toto je možné omezit zadáním mezery, nebo jiného znaku, například tečky, čárky apod.

Tlačítkem Zrušit v části Získání požadovaných informací je možné zrušit zobrazení informací dle zadaného výrazu a zobrazit původní získaná data ze vstupního HTML dokumentu. Původní data se v aplikaci uchovávají v samostatné proměnné, tím jejich zobrazení nevyžaduje opětovné získávání.

Po spuštění aplikace se zobrazí okno s indikátorem průběhu a tlačítkem Stop. Proces získávání dat je umístěn v samostatném vlákně, proto je možné jej za pomocí tlačítka Stop přerušit. Proces je většinou vykonán velmi rychle, a tak se možnost jej přerušit hodí spíše pro případy, kdy by zdrojový dokument obsahoval příliš velké množství dat.

6.5.2 Nastavení aplikace

Nastavení aplikace obsahuje dvě části. První část se věnuje obecným možnostem nastavení programu a druhá se věnuje četnosti slov.

Nejdůležitější možností části obecné je nastavení výchozí znakové sady pro vstupní HTML dokumenty. Na začátku seznamu jsou umístěny znakové sady nejčastěji používané u nás a dále následují ostatní sady seřazené podle abecedy. Seznam obsahuje nejrůznější znakové sady používané po celém světě. Na výběr je také automatická detekce, kdy se aplikace snaží znakovou sadu načíst ze zdrojového kódu HTML dokumentu.

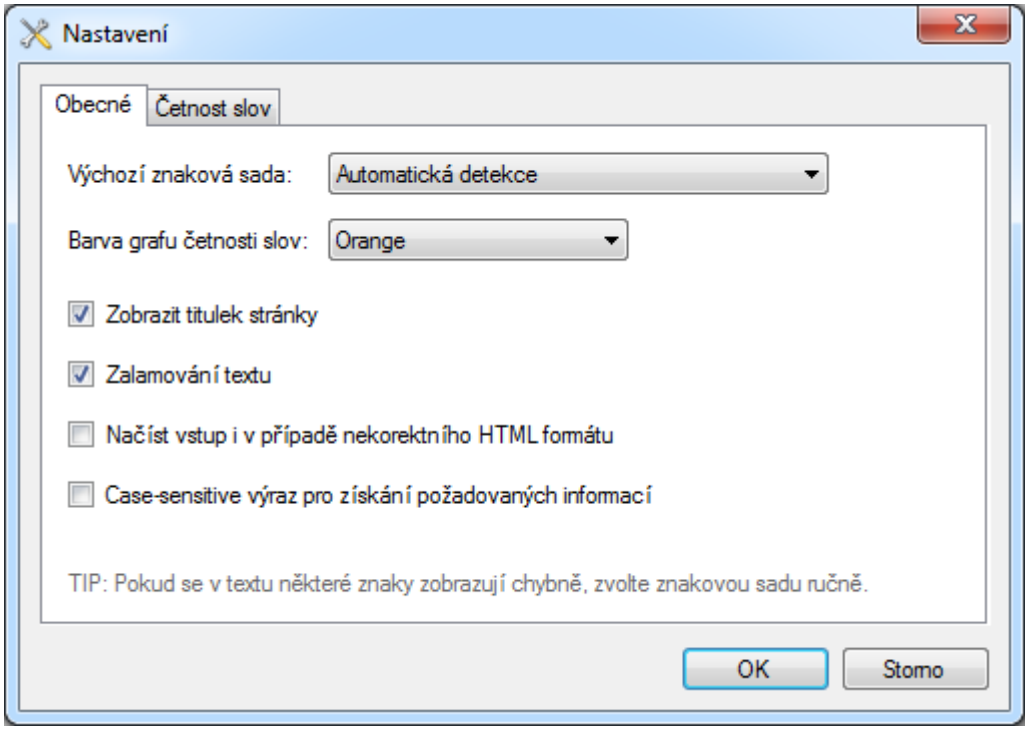
Je možné si zde nastavit také barvu grafu četnosti slov. Seznam obsahuje velké množství barev a každý uživatel si tak jistě vybere barvu, která mu nejvíce vyhovuje.

Možnosti zobrazit titulek stránky a zalamování textu jsou ve výchozím stavu povolené. Titulek se v aplikaci zobrazuje nad textovým polem pro zobrazení výsledku a je možné jej v případě potřeby vypnout. Výsledný text zobrazený v textovém poli je zalomen dle šířky pole tak, aby byl uživatelem pohodlně čitelný. Vypnutím této volby se text nebude zalamovat a zobrazí se, v případě potřeby, vodorovný posuvník.

Další možností obecného nastavení je možnost načíst vstup i v případě nekorektního HTML formátu. Pokud tak máme vstupní HTML dokument ve špatném formátu a jsme aplikaci na tuto skutečnost upozorněni, tak pomocí této volby je možné načíst a zpracovat i takovýto dokument.

Poslední možností je volba case-sensitive výraz pro získání požadovaných informací, která ovlivňuje získávání informací. Pokud je tato volba povolena, tak program zohledňuje velikost písmen v zadaném výrazu. Tato volba je vhodná například tehdy, když potřebujeme vyhledat přesně zadaný výraz.

Obrázek 8 znázorňuje rozložení nastavení jednotlivých komponent obecné části nastavení programu.



Obr. 8. Nastavení aplikace – Obecné

Druhou částí nastavení aplikace je nastavení četnosti slov. Je možné zde povolit nebo zakázat počítání četnosti slov a zvolit, zda se budou počítat všechna slova obsažená v získaném textu vstupního HTML dokumentu, nebo jen ta, která jsou uživatelem zadaná.

V první možnosti počítání všech slov si je možné nastavit počet nejčetnějších slov v intervalu od 1 do 15 slov. Dle nastaveného počtu bude následně v grafu omezen počet zobrazených slov. Dále je možné nastavit minimální délku slova. Toto nastavení je vhodné pokud například nechceme počítat spojky nebo příliš krátká slova.

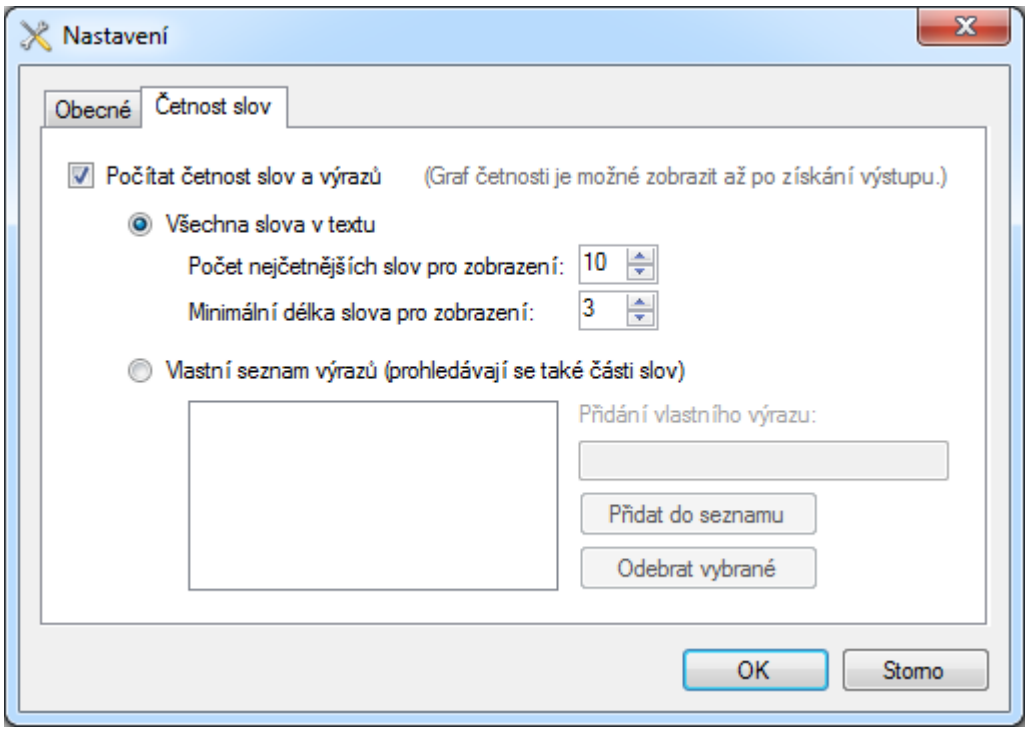
Druhou možností nastavení je počítání četnosti zadaných výrazů. Dle zadaných slov bude v získaném textu vstupního HTML dokumentu spočítána jejich četnost. Podporována jsou také slovní spojení. Slova nebo slovní spojení se zadají do textového pole a potvrdí stisknutím tlačítka přidat do seznamu.

Po přidání výrazu je seznam seřazen podle abecedy. Na případné duplicitní záznamy je uživatel upozorněn formou informační zprávy. Pro odebrání slova ze seznamu jej vybereme myší a stiskneme tlačítko odebrat vybrané. Odebrat je možné také více slov současně. Možností výběru více slov v seznamu je několik. Například pomocí tažení myší, dále pomocí kombinace tlačítka SHIFT a myši nebo tlačítka CTRL a myši.

Vlastností tohoto způsobu je, že zadané výrazy jsou počítány také v částech slov, to znamená, že pokud zadáme například slovo text, budou počítána i slova textem, textu, textová a další.

Po nastavení požadovaných možností se stiskem tlačítka OK změny uloží. Tlačítkem Storno se všechny provedené změny zruší.

Rozložení jednotlivých komponent nastavení četnosti slov je zobrazeno na obrázku 9.



Obr. 9. Nastavení aplikace – Četnost slov

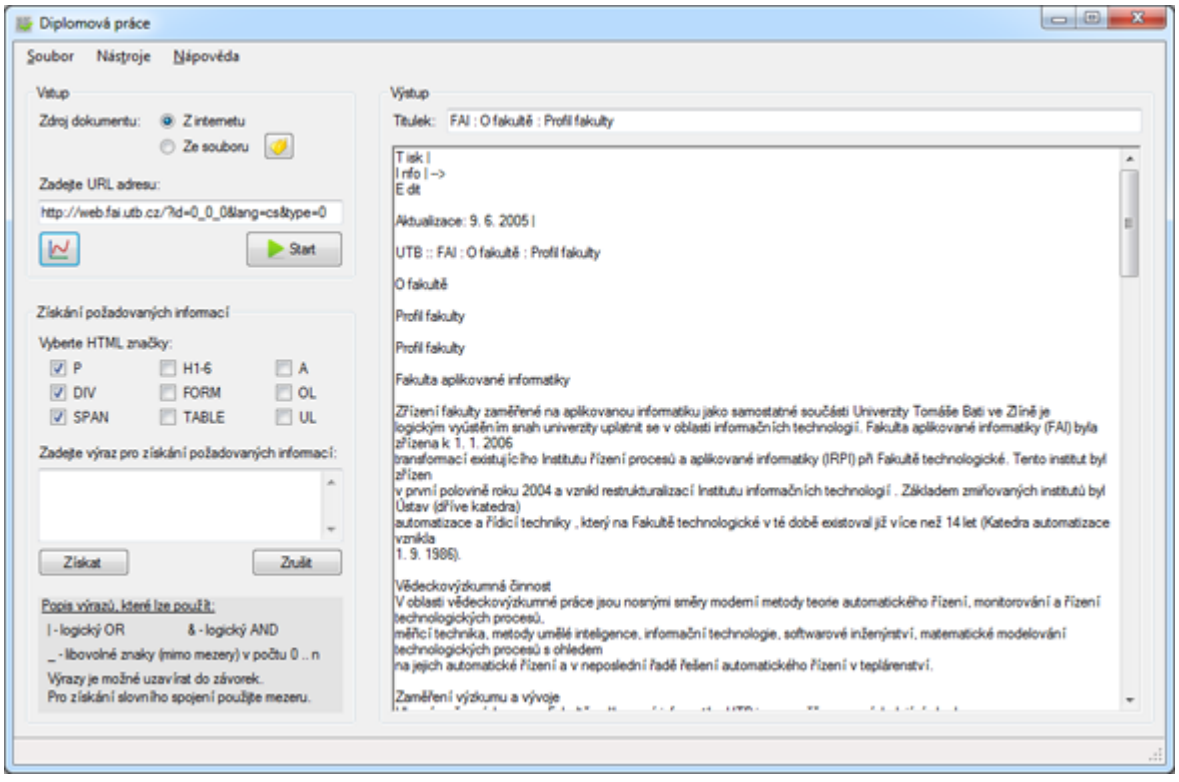
7 VYHODNOCENÍ ŘEŠENÍ

Důležitou částí práce je vyhodnocení řešení vytvořené aplikace a jejího textového výstupu. Jak je popsáno v následujících podkapitolách, tak výstupy aplikace jsou celkem dobré. Najdou se ale i případy, kdy získaný výstup není úplně ideální. I tyto případy jsou popsány a zdůvodněny.

7.1 Textový výstup HTML dokumentu

Pro ověření správnosti výstupu vytvořené aplikace jsem jako vstup použil stránku Fakulty aplikované informatiky UTB Zlín, část O fakultě – Profil fakulty.

Na obrázku 10 je znázorněn získaný textový výstup z této stránky ve vytvořené aplikaci. Text je úplný, bez HTML značek nebo jiných prvků, které do textu nepatří. Aplikace se snaží výsledek zarovnávat do odstavců tak, aby byly získané informace pro uživatele zobrazeny v přehlednější podobě, ale ne vždy je výsledné zarovnání optimální. Odstavce jsou navzájem oddělené, ale text v nich je často rozdělený na více řádcích. Toto je způsobené zarovnáním zdrojového kódu HTML dokumentu. Případná úprava zarovnání odstavců by mohla být námětem na další vylepšení aplikace.



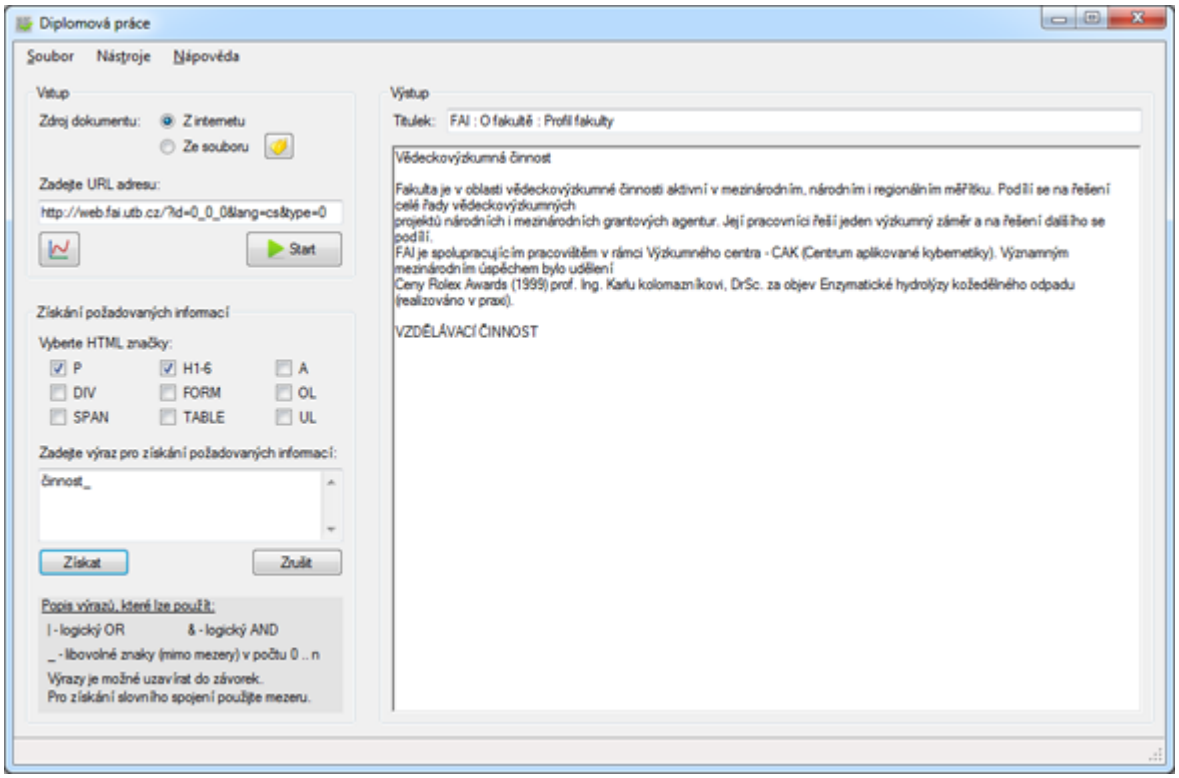
Obr. 10. Výstup aplikace

7.2 Získané informace dle zadaného výrazu

K získání požadovaných informací ze vstupního dokumentu jsem zvolil HTML značky pro odstavce a nadpisy a zadal následující výraz:

```
činnost_
```

Získaný výsledek je zobrazen na obrázku 11. Je zde vidět, že z dokumentu byly získány dva nadpisy a jeden odstavec odpovídající zadanému výrazu. Při porovnání zobrazeného výsledku s obsahem stránky v internetovém prohlížeči jsem zjistil, že výsledek je správný a daná stránka neobsahuje více informací se zadaným výrazem.



Obr. 11. Získané informace dle zadaného výrazu

Výsledky ale nejsou vždy optimální. Zjistil jsem, že pro některé výrazy, aplikace získá také ta data, která zadaný výraz neobsahují. To může být způsobeno strukturou HTML dokumentu, kde mohou být jednotlivé prvky různě zanořovány. Aplikace při procházení dat hledá vždy dvojice prvků – počáteční a koncové. V nich poté vyhledává požadované informace. Může se tak stát, že vlivem zanoření prvků bude označena dvojice prvků, které k sobě přímo nepatří a ty tak budou obsahovat větší rozsah dat.

Lepší detekce prvků je námětem na budoucí vylepšení aplikace. Návrhům na vylepšení se věnuje následující kapitola 7.3.

7.3 Návrhy na vylepšení

V každé aplikaci se najdou věci, které se dají vylepšit. Ani tato aplikace není výjimkou a návrhů na její vylepšení se zde najde několik.

Jedním z vylepšení by mohla být možnost uživatelské volby identifikace pro internetové servery. Nestalo by se tak, že uživatel obdrží výstup ze zadané internetové adresy například pro mobilní zařízení, ve kterém budou některé požadované informace chybět.

Dalším vylepšením by mohla být možnost specifikovat jazykovou verzi stránek, kterou aplikace od serveru obdrží. Tato funkce je ale složitá na implementaci, protože volba jazyka se na každé internetové stránce provádí jiným způsobem. Jednotný způsob identifikace se bohužel nepoužívá a záleží tak pouze na vývojáři webu, jak si toto naprogramuje. Výsledek by nemusel být funkční na všech stránkách.

Jak již bylo v práci zmíněno, v aplikaci je implementováno nahrazování jen některých speciálních značek z jejich velkého množství. Možností na vylepšení by tak mohla být implementace editovatelného seznamu v nastavení programu, kde by byly zadány nejčastěji používané značky. Další by si uživatel mohl přidat dle potřeby.

Při získávání dalších informací se data získávají dle zadaných HTML značek. Ty je možné různě zanořovat a tak se může stát, že program chybně vyhodnotí začátek a konec značky. Vylepšením je lepší detekce značek například způsobem, kdy si program nejprve vytvoří strukturu značek v celém dokumentu a následně je začne prohledávat od nejvíce zanořené až po nejméně zanořenou značku.

Dále by aplikace mohla umožňovat zadání většího množství HTML značek pro získání informací. Tento seznam by opět mohl být v nastavení aplikace v podobě seznamu s možností editace. Také více možností jejich zadání by bylo užitečné. Uživatel by si tak mohl specifikovat, že chce informace získat například pouze z nadpisů první a druhé úrovně, ostatní úrovně nadpisů by se ignorovaly.

V části zadání výrazu pro získání informací by možným vylepšením mohlo být větší množství doplňujících prvků pro zadání, například možnost zadání negace pro vyloučení určitého slova a další.

Vylepšení je možné provést i v možnostech četnosti slov. V nastavení aplikace by mohla být volba, která by povolila nebo zakázala prohledávání částí slov při počítání

četnosti uživatelem zadaných slov. Mohla by zde být také možnost definovat slova, která by se do četnosti nezapočítávala. Zde by uživatel mohl zadat například různé spojky nebo slova, která se do četnosti nehodí. Přidat by se také mohla možnost pro seskupování slov se stejnou četností.

Velkým vylepšením by byla také možnost, kdy by program procházel nejen zadanou HTML stránku, ale také všechny další stránky na které vedou odkazy. Zde by mohla být možnost určení úrovně, do jaké by se program přes odkazy zanořoval. Rozsah získaných informací dle zadaného výrazu by tak byl mnohem větší a pro uživatele zajímavější.

Možností vylepšení je opravdu mnoho. Jednotlivá vylepšení jsou často individuální dle potřeb uživatele. Žádná aplikace nebude nikdy přesně dle potřeb uživatele, protože každý uživatel je jiný a má odlišné potřeby a přání. Vhodnou kombinací použitých funkcí v aplikaci se dá vyhovět většině základních požadavků uživatelů.

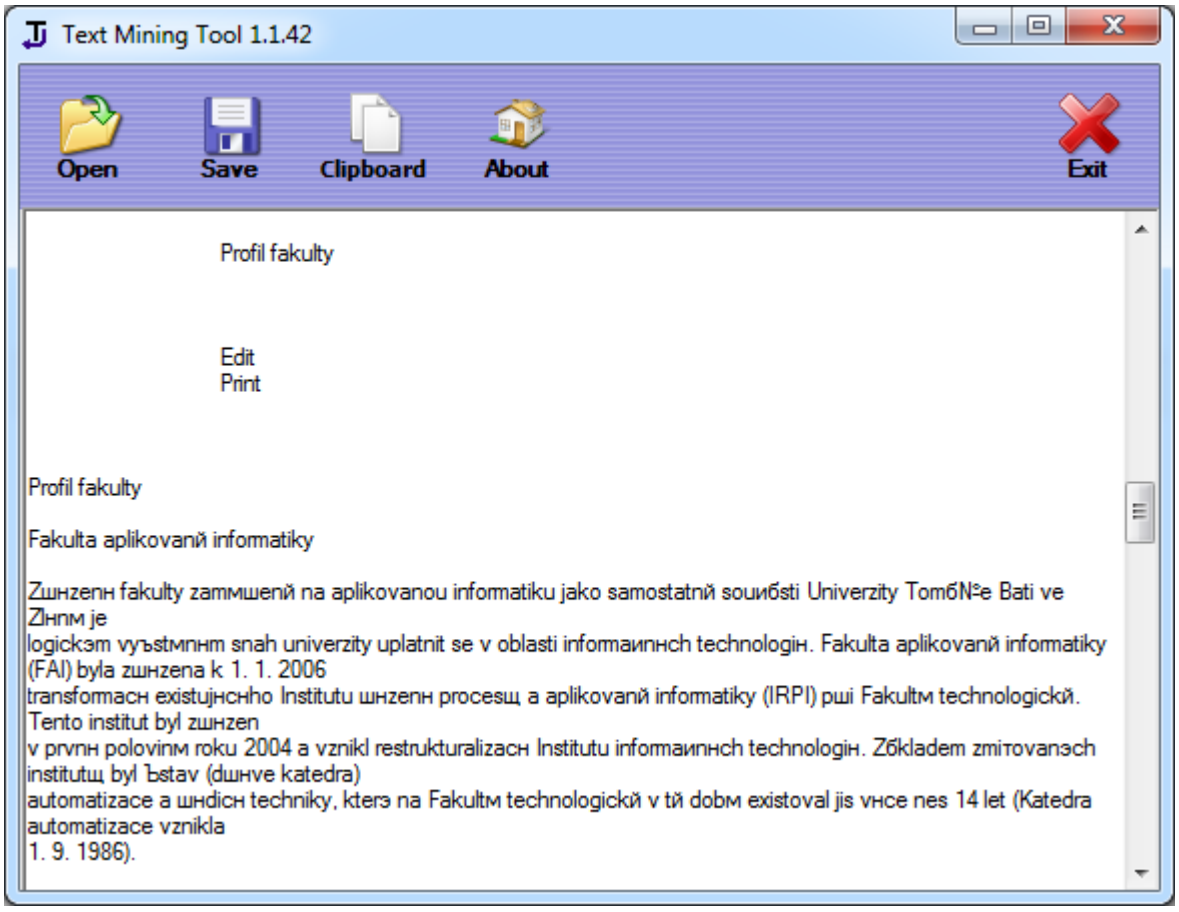
8 POROVNÁNÍ S DALŠÍMI NÁSTROJI

Vytvořenou aplikaci jsem porovnal se dvěma dalšími aplikacemi. Těchto aplikací je více, některé jsou volně šiřitelné a jiné jsou placené. Zaměřil jsem se nejen na kvalitu výstupu, ale také na uživatelské prostředí a jeho ovládání. Aplikace, se kterými jsem vytvořenou aplikaci porovnával, jsou podrobněji popsány v následujících podkapitolách.

8.1 Text Mining Tool

První aplikace, kterou jsem vyzkoušel je Text Mining Tool. Tato aplikace je k dispozici zdarma. Jedná se o jednoduchou aplikaci pro získání textu z několika typů souborů. Její uživatelské rozhraní je přehledně uspořádáno a obsahuje jen pár základních funkcí. Data je možné získat pouze ze souborů, proto jsem stránku Profil fakulty ze stránek Fakulty aplikované informatiky UTB Zlín nejprve uložil a poté v aplikaci otevřel.

Získaná data aplikace umožňuje uložit do textového souboru nebo zkopírovat do schránky pro další použití v jiných aplikacích.



Obr. 12. Výstup z aplikace Text Mining Tool

Z výsledku, zobrazeného na obrázku 12, je patrné, že si aplikace neporadí s českými znaky. Problém je způsoben omezenou podporou znakových sad, protože při dalších testech se u stránek s jinou znakovou sadou české znaky zobrazovaly správně. Zobrazený text je navíc rozházený a plný zbytečných mezer a řádků. Číst text v této podobě je nepohodlné.

Aplikace také nenahrazuje speciální značky znaky, které reprezentují, ale všechny HTML značky rovnou odstraňuje. Stává se tak, že v místě kde v textu byla speciální značka pro pevnou mezeru, je text spojen a jeho čitelnost je tím zhoršena.

Vzhledem k výše uvedeným nedostatkům se větší použití této aplikace jeví jako nedostačující. Aplikace je vhodná spíše pro menší a příležitostné použití, kde výše uvedené nedostatky nevadí.

8.2 HTML Text Extractor

Tato placená aplikace firmy ICONICO poskytuje více možností než předchozí testovaná aplikace. Její cena je 19,50 dolaru a výrobce poskytuje také zkušební verzi.

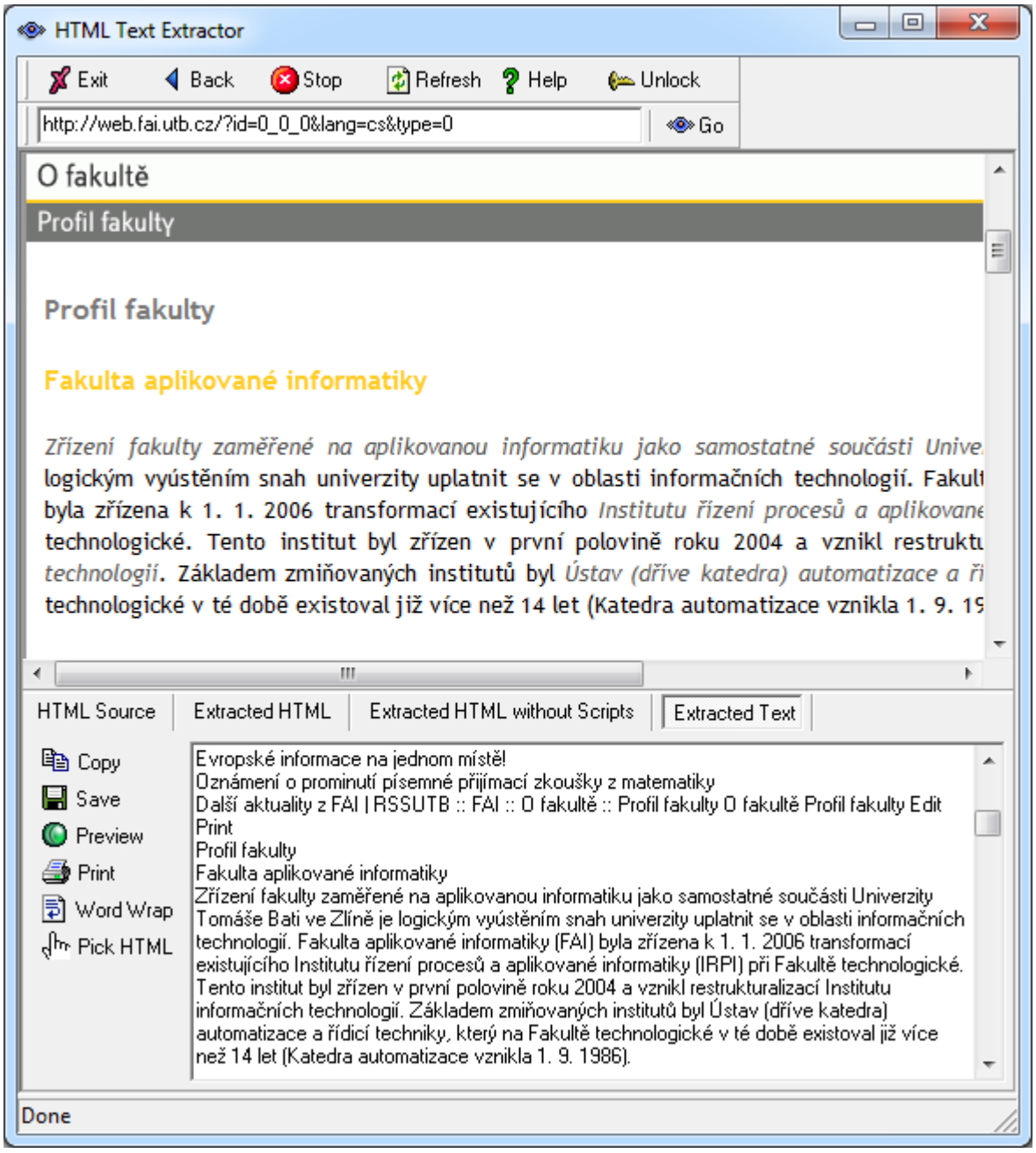
Výsledný text se v této aplikaci získává přímo z webových stránek. Uživatel si může kromě získaného textu zobrazit také zdrojový kód HTML stránky a přehled všech HTML prvků, které byly ze stránky odebrány. Tento přehled je možné zobrazit také bez skriptů.

Uživatelské rozhraní je koncipováno do několika částí. V horní části jsou umístěny ovládací prvky pro zadání adresy, její obnovení nebo zastavení načítání, možnost se vrátit na předchozí adresu, nápověda, zadání registračního klíče a ukončení aplikace. Prostřední část obsahuje velké pole, ve kterém se zobrazuje zadaná stránka. V poslední spodní části se zobrazuje získaný výsledek. Jsou zde také ovládací prvky pro kopírování, ukládání a tisk výsledku, zalamování slov a další. Tyto prvky jsou ve zkušební verzi blokovány a pro jejich použití je nutné aplikaci zakoupit.

Po zadání požadované adresy se stránka zobrazí v integrovaném prohlížeči internetových stránek. To je velkou výhodou, protože stránku je možné ovládat stejně jako v běžném prohlížeči a je možné získat text i z těch částí, které vyžadují přihlášení.

Aplikace se získaný výsledek snaží formátovat do odstavců, ale občas zde zbytečné mezery a prázdné řádky zůstanou. Některá slova jsou spojená, protože aplikace značky odstraňuje. Celkově je ale lépe zpracována a je tak bez větších problémů použitelná.

Na obrázku 13 je znázorněn výstup z aplikace HTML Text Extractor pro stránku Profil fakulty, která byla získána ze stránek Fakulty aplikované informatiky UTB Zlín.



Obr. 13. Výstup z aplikace HTML Text Extractor

ZÁVĚR

Cílem diplomové práce bylo navrhnout a vytvořit aplikaci, která z HTML dokumentu získá pouze textovou část. V získaném textu spočítá četnost slov a dále na základě uživatelem zadaného výrazu získá požadované informace.

K dosažení tohoto cíle bylo zapotřebí se nejprve seznámit s problematikou dolování dat zahrnující postupy, metody a možné problémy, které při dolování dat mohou nastat. Dále bylo nutné se seznámit se strukturou HTML dokumentů, ze kterých se data budou získávat.

Na základě získaných znalostí z dané problematiky jsem navrhl a vytvořil algoritmus pro získání požadovaných informací ze vstupního HTML dokumentu. Tento algoritmus jsem poté použil v realizované aplikaci. Dále jsem implementoval algoritmus pro počítání četnosti výskytu slov.

Vytvořenou aplikaci jsem otestoval na různých typech HTML dokumentů. V práci uvádím výsledky testování na stránce Profil fakulty z webových stránek Fakulty aplikované informatiky UTB Zlín. V poslední kapitole jsem vytvořenou aplikaci porovnal s dalšími dostupnými nástroji a popsal jejich výhody a nevýhody.

Získané informace z vytvořené aplikace nebyly zcela optimální, ale ve většině testovacích případů byly dostatečné a obsahovaly informace odpovídající zadanému výrazu. Vytvořenou aplikaci považuji za funkční a použitelnou pro potřeby běžného uživatele. Pro její větší využití by bylo vhodné ji začlenit do nějakého většího projektu a zaměřit se také na navrhovaná vylepšení, uvedená v kapitole 7.3.

Také výsledky porovnávaných nástrojů obsahovaly nedostatky. K získání ideálních výsledků by byla nejlepší aplikace kombinující vlastnosti a funkce z více aplikací. Volba vhodného nástroje na získávání dat je tak na uživateli, dle jeho potřeb a preferencí.

Hlavním přínosem této práce je osvojení a prohloubení znalostí problematiky data miningu. Pomocí data miningu je možné objevit nové trendy, na jejichž základě lze činit marketingová, obchodní a další důležitá rozhodnutí, která vedou ke zvýšení ziskovosti, efektivity práce a jiné.

ZÁVĚR V ANGLIČTINĚ

The aim of this master thesis was to create an application which extracts only text information from HTML documents. Application in the acquired text counts words frequencies and extract required information by the user specified expression.

To achieve this aim was first necessary to acquaint with data mining procedures, methods and possible problems which can occur. Also was necessary to acquaint with the structure of HTML documents from which the data will be extracted.

Based on the obtained knowledge I have designed and created an algorithm to extract the required information from the input HTML document. This algorithm is used in the implemented application. I also implemented an algorithm for counting words frequencies.

Created application I have tested on different types of HTML documents. In this thesis are presented the results of testing Faculty Profile page from Faculty of Applied Informatics TBU Zlín. I compared the created application with other available tools in the last chapter where I also described their advantages and disadvantages.

Information obtained from the created application was not be completely optimal, but in most test cases are sufficient, and contain information corresponding to the specified expression. I consider the created application as a functional and usable for the user. For its greater use would be appropriate to incorporate it into a larger project and to also focus on the suggested improvements listed in chapter 7.3.

Also the results of the compared data mining tools contained deficiencies. To obtain the ideal results would be the best application combining the features and functionality from multiple applications. Choice of an appropriate application for data mining is on the user, according to his preferences.

The main contribution of this thesis is to acquire knowledge about data mining. With data mining can discover new trends on which can be performs marketing, sales and other important decisions that leads to increase profitability, efficiency and other.

SEZNAM POUŽITÉ LITERATURY

[1] Tvorba WWW stránek. [online]. [cit. 2012-04-12]. Dostupné z: http://www.webtvorba.cz/

[2] KOSEK, Jiří. *PHP - tvorba interaktivních internetových aplikací: podrobný průvodce.* Vyd. 1. Praha: Grada, 1999, 490 s. Průvodce (Grada). ISBN 80-716-9373-1.

[3] WEISSINGER, A. *Active Server Pages v kostce: Pohotová referenční příručka.* 1. vyd. Praha: Computer Press, 1999, 392 s. ISBN 80-722-6199-1.

[4] HALL, Marty. *Java: servlety a stránky JSP.* Praha: Neocortex, 2001, 585 s. ISBN 80-863-3006-0.

[5] KOSEK, Jiří. Vše o WWW. [online]. [cit. 2012-04-12]. Dostupné z: http://www.kosek.cz

[6] SHARP, John. *Microsoft Visual C# 2010: krok za krokem.* Vyd. 1. Brno: Computer Press, 2010, 696 s. ISBN 978-80-251-3147-3.

[7] HAN, Jiawei a Micheline KAMBER. *Data Mining: Concepts and Techniques*. Second Edition. San Francisco: Morgan Kaufmann Publishers, 2006, 770 s. ISBN 1-55860-901-6.

[8] BERKA, Petr. *Dobývání znalostí z databází*. 1. vydání. Praha: Academia, 2003, 366 s. ISBN 80-200-1062-9.

[9] FELDMAN, Ronen a James SANGER. *The Text Mining Handbook: Advanced Approaches in Analyzing Unstructured Data*. New York: Cambridge University Press, 2007, 410 s. ISBN 0-521-83657-3.

[10] BERRY, Michael W. (editor) a Jacob KOGAN (editor). *Text Mining: Applications and Theory*. Chichester: Wiley, 2010, 207 s. ISBN 978-0-470-74982-1.

[11] LIU, Bing. *Web data mining*: *exploring hyperlinks, contents, and usage data*. Berlin: Springer, 2007, 532 s. ISBN 978-354-0378-815.

[12] ZENDULKA, Jaroslav, Vladimír BARTÍK, Roman LUKÁŠ a Ivana RUDOLFOVÁ. *Získávání znalostí z databází*. Brno, 2009. Studijní opora. FIT VUT v Brně.

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

ASP	Active Server Pages
CGI	Common Gateway Interface
CSS	Cascading Style Sheets
DTD	Document Type Definition
HTML	HyperText Markup Language
IIS	Internet Information Services (původně Internet Information Server)
JSP	JavaServer Pages
LINQ	Language Integrated Query
PHP	Hypertext Preprocessor (původně Personal Home Page)
SGML	Standard Generalized Markup Language
TPL	Task Parallel Library
W3C	World Wide Web Consortium
XHTML	Extensible HyperText Markup Language
XML	Extensible Markup Language
XMLNS	Extensible Markup Language NameSpace

SEZNAM OBRÁZKŮ

SEZNAM PŘÍLOH