Katedra informatiky
Přírodovědecká fakulta
Univerzita Palackého v Olomouci

BAKALÁŘSKÁ PRÁCE

Poloautomatické získávání formálních kontextů z obsahu Wikipedie

2017

Jakub Pátek

Vedoucí práce: Mgr. Petr Osička, Ph.D.

Studijní obor: Aplikovaná informatika, prezenční forma

Bibliografické údaje

Autor:	Jakub Pátek
Název práce:	Poloautomatické získávání formálních kontextů z obsahu Wikipedie
Typ práce:	bakalářská práce
Pracoviště:	Katedra informatiky, Přírodovědecká fakulta, Univerzita Palackého v Olomouci
Rok obhajoby:	2017
Studijní obor:	Aplikovaná informatika, prezenční forma
Vedoucí práce:	Mgr. Petr Osička, Ph.D.
Počet stran:	32
Přílohy:	1 CD/DVD
Jazyk práce:	český

Bibliograhic info

Author:	Jakub Pátek
Title:	Semiautomatic mining of formal contexts from Wikipedia content
Thesis type:	bachelor thesis
Department:	Department of Computer Science, Faculty of Science, Palacký University Olomouc
Year of defense:	2017
Study field:	Applied Computer Science, full-time form
Supervisor:	Mgr. Petr Osička, Ph.D.
Page count:	32
Supplements:	1 CD/DVD
Thesis language:	Czech

Anotace

Program WikiExtractor slouží k poloautomatickému získávání tabulkových dat z obsahu Wikipedie. K článkům je možno přistupovat jak v online režimu (díky MediaWiki API), tak za pomoci výpisů v XML souborech. Úvod práce pojednává o datech - jaké typy dat je možné získat. Další část se věnuje Wikipedii a seznamuje s některými prvky jejího značkovacího jazyka. Závěrečná část textu popisuje strukturu programu, použité technologie a implementaci jeho důležitých částí.

Synopsis

WikiExtractor is used for semi-automatic retrieval of table data from Wikipedia content. The articles can be accessed both in online mode (via the MediaWiki API) and with the help of extracts in XML files. The introduction discusses the data - what types of data can be obtained. The next section is dedicated to Wikipedia and introduces some elements of its markup language. The final part of the text describes the structure of the program, the technology used and the implementation of its important parts.

Klíčová slova: formální konceptualní analýza; formální kontext; Wikipedie; infobox; XML; parsování; regulární výraz

Keywords: formal concept analysis; formal context; Wikipedia; infobox; XML; parsing; regular expression

Děkuji Mgr. Petru Osičkovi, Ph.D. za vedení, ochotu a pomoc při zpracování této práce.

Místopřísežně prohlašuji, že jsem celou práci včetně příloh vypracoval/a samostatně a za použití pouze zdrojů citovaných v textu práce a uvedených v seznamu literatury.

datum odevzdání práce podpis autora

Obsah

Seznam obrázků

Seznam tabulek

Seznam vět

Seznam zdrojových kódů

1 Úvod

Primárním smyslem práce bylo vytvoření multiplatformní aplikace pro získávání tabulkových dat - vícehodnotových formálních kontextů ze zvolených článků Wikipedie. Sekundárním cílem bylo poté pomocí škálování převést tyto kontexty na jim odpovídající formální kontexty s bivalentními logickými atributy, tedy na tabulku binárních dat. Získaná data, ať již vícehodnotová či binární, je pak možno použít k další analýze.

Wikipedie je největší webovou encyklopedií a tedy největším zdrojem volně šiřitelných informací. Ačkoliv její obsah není vždy zcela důvěryhodný - autory jednotlivých článků jsou dobrovolní přispěvatelé ze všech koutů světa, jedná se oblíbený nástroj k získávání informací. Její nevýhodou je často nejednotný styl článků - některé jsou rozsáhlé, jiné naopak velmi stručné a neúplné. Úlohou aplikace je hledání souvislostí mezi jednotlivými články na základě jejich obsahu a získání komplexních dat - WikiExtractor je tedy první existující aplikací svého druhu.

Formální konceptuální analýza (FKA) je metodou analýzy tabulkových dat. Přestože aplikace používá prostředky formální konceptuální analýzy, jejím hlavním úkolem je získávání dat ve formátu použitelném pro tuto analýzu. Díky systematické struktuře Wikipedie je uživateli nabídnuta možnost v několika krocích snadno a rychle přistupovat k souhrnným informacím z nejrůznějších odvětví, ať již jde o taxonomické údaje živočichů či rostlin, schopnosti komiksových superhrdinů či vlastnosti chemických sloučenin. Se získanými daty může uživatel dále nakládat dle svých potřeb.

2 Data

Ještě než přejdeme k popisu syntaxe značkovacího jazyka Wikipedie a použitých algoritmů, upřesníme si, jaký typ dat od aplikace požadujeme. Jak již bylo řečeno, výstupem jsou tabulková data, a to v textové podobě. Tabulkovými daty jsou myšleny tabulky, kde řádky představují jednotlivé objekty (v tomto případě to jsou články Wikipedie) a sloupce jejich atributy. Jednotlivé položky těchto tabulek nám tedy říkají, že objekt x má atribut y s hodnotou *I(x, y)*, kde I je relace mezi x a y (viz tabulka 1). [1] Tyto položky se označují jako formální kontexty.

Tabulka 1: Tabulková data

	y_1	$\cdots$	y_j	$\cdots$	y_l
x_1			$\vdots$		
$\vdots$			$\vdots$		
x_i		$\cdots$	$I(x_i, y_j)$	$\cdots$	
$\vdots$			$\vdots$		
x_k			$\vdots$		

2.1 Vícehodnotové kontexty

Pokud atributy těchto tabulek nabývají i jiných hodnot než 0 a 1, mluvíme o tzv. vícehodnotových kontextech. Definice zní následovně:

Definice 1 (Vícehodnotový kontext)
Vícehodnotový kontext je čtveřice $\langle X, Y, W, I \rangle$, kde $I \subseteq X \times Y \times W$ je ternární relace taková, že pokud $\langle x, y, v \rangle \in I$ a $\langle x, y, w \rangle \in I$, pak $v = w$. [1]

Tabulky vícehodnotových kontextů jsou tedy primárním výstupem aplikace - obsahují data přesně v takové formě, v jaké jsou na Wikipedii uloženy. Příkladem výstupu pro několik zvolených článků by mohla být následující tabulka 2. Pořadí sloupců je nepodstatné. Dále je potřeba mít na vědomí, že články by měly mít co nejvíce společných atributů (v příkladu je uvedeno několik filmů - a ty společné atributy jistě mají). V opačném případě se v tabulce mohou často vyskytovat nespecifikované hodnoty a hledání kontextů by poté ztrácelo svůj význam.

2.2 Škálování

Pokud chceme získanou tabulku převést na binární data, využijeme škálování. Pod tímto pojmem rozumíme způsob, kterým jsou vícehodnotové atributy převedeny na atributy bivalentní - výsledkem jsou logické hodnoty. Pro přehlednost

Tabulka 2: Vícehodnotové kontexty

film	**režie**	**premiéra**	**stopáž**	**žánr**
2001: A Space Odyssey	Stanley Kubrick	1968	149 min	sci-fi
E.T.	Steven Spielberg	1982	115 min	rodinný
Forrest Gump	Robert Zemeckis	1994	136 min	komedie
Jurassic Park	Steven Spielberg	1993	127 min	sci-fi
The Shining	Stanley Kubrick	1980	119 min	horror
The Toxic Avenger	Lloyd Kaufman	1984	87 min	akční

a srozumitelnost dat je škálován pouze jeden zvolený atribut a výstup tedy odpovídá škále pro daný atribut. Definice pojmu škála je následující:

Definice 2 (Škála)
Škála pro atribut y vícehodnotového kontextu je kontext $S_y = \langle X_y, Y_y, I_y \rangle$, pro který $y(X) \subseteq X_y$ (kde $y(X) = \{y(x) | x \in X\}$). Prvky množin X_y a Y_y se nazývají škálové hodnoty a škálové atributy. [1]

Objekty vytvořené škály jsou shodné s objekty tabulky vícehodnotových kontextů a atributy škály jsou rozkladem množiny všech hodnot atributu, pro který je škála tvořena. Nejjednodušší škálou je tedy rozklad hodnot daného atributu na jednoprvkové množiny. Uvažujme tabulku č. 2. Škálové atributy pro atribut „režie“ by mohly vypadat následovně:

$$\begin{gathered} \text{Je } y_1 \text{ režisérem filmu } x? \\ \vdots \\ \text{Je } y_n \text{ režisérem filmu } x? \end{gathered}$$

$Y = \{y_1, ..., y_n\}$ je tedy množinou režisérů (v našem případě $Y = \{$Stanley Kubrick, Steven Spielberg, Robert Zemeckis, Lloyd Kaufman$\}$). Tabulka č. 3 odpovídá této škále. Stejným způsobem může být škála vytvořena i pro ostatní atributy. Pro podrobnější informace o škálování a dalších aspektech formální konceptuální analýzy odkazuji na [1] a [2].

Tabulka 3: Škála odpovídající atributu „režie“ z tabulky 2

film	y_1	y_2	y_3	y_4
2001: A Space Odyssey	x			
E.T.		x		
Forrest Gump			x	
Jurassic Park		x		
The Shining	x			
The Toxic Avenger				x

3 Wikipedie

3.1 Základní fakta

Wikipedie je největší existující webovou encyklopedií a patří mezi deset nejnavštěvovanějších webů světa. Není komerčním webem a je provozována nadací Wikimedia Foundation. [3] Existuje ve více než 290ti jazykových verzích, z nichž již třináct přesáhlo počet jednoho milionu článků. Zcela nejobsáhlejší je anglická verze se svými téměř 5,5 miliony článků. Česká verze měla v době psaní práce necelých 390 tisíc článků.

Funguje na principu zvaném wiki, což znamená, že kdokoliv může články nejen přidávat, ale i editovat (ať již jde o malou změnu či úplné přepsání). Její obsah je tedy výsledkem přispěvatelů z celého světa. Tato výhoda je zároveň i její nevýhodou, neboť poskytované informace tak mohou být nepřesné nebo dokonce zcela chybné (ať již úmyslně či neúmyslně).

Prvním krokem k vytvoření Wikipedie byly webové stránky nazvané WikiWikiWeb (vytvořené v roce 1995), jejichž obsah mohl upravovat každý jejich návštěvník. Wikipedie (tehdy pouze v anglické verzi) vznikla 15. ledna 2001 jako doplňkový projekt k encyklopedii Nupedia. Jejími zakladately jsou Jimmy Wales a Larry Sanger. Rychle poté vznikaly další jazykové verze, z nich česká byla spuštěna 3. května 2002. Wikipedie dnes běží na softwaru zvaném MediaWiki (napsaném v PHP).[4] Tyto a mnohé další informace jsou dostupné na stránkách projektu.[5][6]

3.2 Jmenné prostory

Každá stránka Wikipedie náleží do určité skupiny. Tyto skupiny stránek se nazývají jmenné prostory (angl. namespaces). Jedná se o důležitou informaci, která udává, o jaký typ stránky se jedná. Nejčastějším jmenným prostorem je hlavní jmenný prostor (Main/Article), pod kterým jsou schovány veškeré články Wikipedie. Zároveň je také jediným jmenným prostorem, jehož název se neobjevuje v názvu stránky. Názvy stránek spadajících do jakéhokoliv jiného jmenného prostoru jsou ve tvaru:

jmenný-prostor:název-stránky

Příkladem takových jmenných prostorů je např. Šablona (Template), Kategorie (Category), Soubor (File), Nápověda (Help), Wikipedista (User) a další. Tyto jmenné prostory jsou společné pro všechny jazykové verze Wikipedie. Kromě těchto existují i specifické prostory pro různé jazykové verze (pro českou Wikipedii např. Portál, Rejstřík). Znalost do jakého jmenného prostoru stránka náleží je pro činnost aplikace velmi důležitá - klíčovými typy jsou články, kategorie a šablony. Tabulka 4 zobrazuje lokální názvy a číselné konstanty hlavních jmenných prostorů. Kompletní přehled je dostupný na [7] a [8].

Tabulka 4: Lokální názvy a číselné konstanty hlavních jmenných prostorů

konstanta	**český název**	**anglický název**
0	(hlavní jmenný prostor)	(Main/Article)
2	Wikipedista	User
4	Wikipedie	Wikipedia
6	Soubor	File
8	MediaWiki	MediaWiki
10	Šablona	Template
12	Nápověda	Help
14	Kategorie	Category

3.3 Formátování

Pro formátování textu stránek používá Wikipedie svůj vlastní značkovací jazyk (wikiznačky) a v menší míře i některé tagy jazyka HTML. Častými wikiznačkami jsou apostrofy (udávají, zda bude uvozený text zobrazen tučně nebo jako kurzíva, případně tučná kurzíva). Text uvozený rovnítky dává najevo, že se bude jednat o nadpis.

3.3.1 Odkazy

Důležitými a velmi často se vyskytujícími wikiznačkami jsou odkazy. Ty mohou být buďto vnitřní - odkazují na článek v rámci Wikipedie, nebo externí - jedná se o URL adresu odkazující na jiné webové stránky. Text, který má být vnitřním odkazem je ohraničen dvojicí hranatých závorek. Odkaz musí být přesným názvem článku, pokud je označovaný text odlišný, vloží se tento text za jméno článku a oddělí se svislou čarou. Syntaxe úplného vnitřního odkazu je tedy:

[[jméno cílového článku|text odkazu]]

Externí odkaz má vícero podob. Nejjednodušším je přímé vložení URL adresy do textu, a to včetně protokolu. V tomto případě se v textu článku zobrazuje celá URL adresa. Druhým způsobem je uzavření textu do jednoduchých hranatých závorek. Odkaz se pak v článku zobrazuje jako číslovaná poznámka v hranatých závorkách. Má-li být k odkazu přidělený jiný popisek, za URL adresu je vložena mezera a text odkazu. Níže je k vidění syntaxe takového odkazu. Znalost formátování odkazů je důležitá pro činnost programu - chceme získat data v čisté textové podobě.

[URL text odkazu]

```
1 [[Berlín]] je hlavní město [[Německo|Německa]]
2 [http://www.google.com/ vyhledávač Google]
```

Zdrojový kód 1: Vnitřní a externí odkazy

3.3.2 Seznamy

Dalším často se vyskytujícím aspektem textu jsou odrážkové nebo číslované seznamy. Odrážkové seznamy jsou tvořeny pomocí symbolu *, číslované pak pomocí symbolu #. Alternativním způsobem zápisu je použití HTML tagů nebo speciálních šablon. Zdrojový kód 2 obsahuje příklad odrážkového seznamu v zápisu pomocí wikiznaček.

```
1 * Polozka 1
2 * Polozka 2
3 ** Sub-polozka 2 a)
4 *** Sub-polozka 2 a) 1.
5 **** Sub-polozka 2 a) 1. i)
6 **** Sub-polozka 2 a) 1. ii)
7 ** Sub-polozka 2 b)
8 * Polozka 3
```

Zdrojový kód 2: Odrážkový seznam

3.3.3 Šablony

Klíčovou a naprosto zásadní částí aplikace je znalost formátování šablon. Šablonou rozumíme speciální druh stránky, jejíž obsah se vkládá do jiných článků. Díky nim je možno do stránek jednoduše vkládat stejný či podobný obsah. Šablona je do kódu článku vložena pomocí jejího názvu uzavřeného ve dvojitých složených závorkách. Kromě názvu mohou obsahovat i další parametry, které jsou od sebe navzájem odděleny pomocí svislého lomítka.

Existuje velké množství šablon, které jsou používány k různým účelům. Obecně se dají rozdělit do několika skupin: [9]

- šablony tvořící běžnou součást článku - Tyto šablony jsou součástí každého rozsáhlejšího článku. Jsou to především infoboxy, šablony pro rozcestníky, skupiny článků, citace a některé další.
- tématické šablony - Jedná se o šablony specifické pro různé kategorie článků. Většinou jde o specifické infoboxy (např. infobox - hudební umělec, taxobox (pro taxonomické údaje) atd.). Dále se jedná o různé typy tabulek, které jsou v kontextu s daným tématem.

- údržba, šablony vyjadřující stav článku - Zde patří např. šablony pro úpravy článků, pahýly[1], údržbu, přesměrování a další.
- jiné - Šablony, používané pouze na stránkách mimo hlavní jmenný prostor nebo interně (např. uživatelské, pro datum a čas, autorská práva).

Nejpodstatnější šablonou pro nás jsou infoboxy - odtud jsou získávána data. Tyto šablony bývají umístěny v pravé horní části článků a nesou základní informace o daném tématu. Obvykle mívají formu předpřipravené tabulky, která je společná pro více souvisejících článků. Data jsou ve formě atribut-hodnota. Jako dodatek je třeba uvést, že u infoboxů s automaticky generovanými parametry často nebývají vyplněny všechny jejich hodnoty. Předpis šablony pro zobrazení infoboxu vypadá následovně.

{{Infobox - téma článku
| název 1. parametru = hodnota
| název 2. parametru = hodnota
...
| název n. parametru = hodnota
}}

Struktura infoboxu může být mnohdy velmi složitá, neboť se v něm mohou vyskytovat nejrůznější prvky značkovacího jazyka Wikipedie, ať již jde o odkazy, seznamy či vnořené šablony, a to dokonce přímo další infoboxy. Na obrázku 1 je k vidění část infoboxu tak, jak jej můžeme spatřit na Wikipedii.

2001: Vesmírná odysea	
Původní název	*2001: A Space Odyssey*
Země	USA Spojené království
Jazyk	angličtina
Délka	141 min.
Žánr	Sci-Fi / Dobrodružný
Předloha	Hlídka
Scénář	Stanley Kubrick a Arthur C. Clarke
Režie	Stanley Kubrick

Obrázek 1: Infobox. Zdroj: [10]

Zbývá dodat, že šablon existuje opravdu velké množství a ke každé je dostupná rozsáhlá dokumentace. Také wikiznaček byl uveden pouze malý zlomek, tyto jsou však nejdůležitější pro parsování zdrojových textů článků. Samotný

[1]Označení pro neúplný článek, který neposkytuje ani základní informace.

```
{{Infobox - film
 | film = 2001: Vesmírná odysea
 | obrázek =
 | originál = 2001: A Space Odyssey
 | žánr = Sci-Fi / Dobrodružný
 | režie = [[Stanley Kubrick]]
 | produkce = Stanley Kubrick
 | scénář = [[Stanley Kubrick]]<br />a [[Arthur C. Clarke]]
 | hrají = [[Keir Dullea]]<br />[[Gary Lockwood]]<br />[[William
    Sylvester]]<br />[[Daniel Richter]]
 | hudba =
 | kamera = [[Geoffrey Unsworth]]
 | střih = [[Ray Lovejoy]]
 | distribuce = [[MGM]]
 | premiéra = [[6. duben|6. dubna]] [[1968]]
 | délka = 141 min.
 | země = {{flagicon|USA}} [[Spojené státy americké|USA]]<br />
    {{flagicon|Spojené království}} [[Spojené království]]
 | jazyk = [[angličtina]]
 | rozpočet = 10 500 000 $
 | tržby = 193 600 000 $
 | imdb = 0062622
 | čsfd = 5393
}}
```

Zdrojový kód 3: Infobox

proces parsování je popsán v další kapitole. Popis formátování textu Wikipedie je dostupný na [11].

3.4 Export dat

Wikipedie nabízí svůj obsah zdarma ke stažení, a to ve vícero formátech. [12] Nejpoužívanější jsou výpisy v XML souborech. XML je značkovacím jazykem, který slouží především pro ukládání a transport dat. Tyto soubory (tzv. dumpy) jsou zdrojem dat pro naši aplikaci v offline módu. Dumpy všech jazykových verzí Wikipedie je možno stáhnout na webu Wikimedia Downloads. [13] Jelikož se obsah Wikipedie neustále a rychle zvětšuje, dumpy jsou aktualizovány dvakrát měsíčně. Tyto soubory jsou dostupné v několika více či méně rozsáhlých verzích, z nichž my budeme používat ten se suffixem „pages-articles“, které obsahují pouze aktuální verze článků a průvodní informace o stránce. Diskusní a uživatelské stránky nejsou pro aplikaci potřebné. Alternativně je taktéž možné exportovat do XML pouze vybrané stránky (či stránky zvolených kategorií). [14]

Ke každé verzi dumpu existuje také množství různých databázových souborů (obvykle jde o různé seznamy, statistiky, použité odkazy apod.). Nás zajímá soubor se suffixem „categorylinks“. Tento soubor obsahuje informace o členství dané stránky v daných kategoriích, čehož budeme využívat při vyhledávání a

```xml
<page>
  <title>2001: Vesmírná odysea (film)</title>
  <ns>0</ns>
  <id>43303</id>
  <revision>
    <id>14714407</id>
    <parentid>14389806</parentid>
    <timestamp>2017-03-04T04:27:20Z</timestamp>
    <contributor>
      <username>Bazi</username>
      <id>39262</id>
    </contributor>
    <model>wikitext</model>
    <format>text/x-wiki</format>
    <text xml:space="preserve">Zdrojový text článku</text>
    <sha1>25jackcsil5z5udbzqudvbmz3igd8gd</sha1>
  </revision>
</page>
```

Zdrojový kód 4: Struktura článku v XML

procházení kategorií. Tabulka 5 zobrazuje strukturu této databáze. Podstatnými údaji pro nás jsou položky cl_from, cl_to a cl_type.

Tabulka 5: Struktura databáze categorylinks

field	**type**	**null**	**key**
cl_from	int(8) unsigned	no	primary
cl_to	varbinary(255)	no	primary
cl_sortkey	varbinary(230)	no	
cl_timestamp	timestamp	no	
cl_sortkey_prefix	varbinary(255)	no	
cl_collation	varbinary(32)	no	multiple
cl_type	enum('page','subcat','file')	no	

4 Získávání formálních kontextů

V předchozích kapitolách bylo vysvětleno jaká data budeme od aplikace požadovat a s jakými daty budeme pracovat. Tato část se věnuje programu samotnému - jsou zde popsány použité technologie, struktura programu a především jeho implementace. Pro zachování přiměřenosti rozsahu i lepší přehlednost jsou v zobrazených zdrojových kódech vypuštěny nepodstatné části původního kódu (jako aktualizace komponent GUI, odchytávání výjimek apod.).

4.1 Základní údaje

WikiExtractor je multiplatformní aplikací sloužící k získávání formálních kontextů z obsahu Wikipedie. Je napsán v jazyce Java. Ta je objektově orientovaným programovacím jazykem (a zároveň počítačovou platformou), vyvinutým v roce 1995 firmou Sun MicroSystems. Ke stažení je dostupný na [15] (aktuální verzí byla Java 8). Komunikace s programem je umožněna za pomoci grafického uživatelského rozhraní v JavaFX. Kromě standardních komponent bylo použito i prvků z projektu ControlsFX (konkrétně třídy CheckListView). [16] Pro vytváření formulářů bylo použito jazyka FXML.

Program funguje ve dvou režimech - online a offline módu. V online módu je využito knihovny wiki-java. [17] Jedná se o framework, který komunikuje s MediaWiki API a obstarává mnoho důležitých funkcí programu, jako např. získávání členů dané kategorie, získávání šablon a textu daného článku či např. zjištění, zda daná stránka existuje. V offline režimu si tyto funkce obstaráme sami. Jako zdroj dat slouží XML soubory a databáze categorylinks, jež byly představeny v minulé kapitole. K získání dat vede několik kroků, které jsou společné pro oba režimy. Posledním (avšak již volitelným) krokem je vytvoření škály pro zvolený atribut a tedy získání tabulky binárních dat.

4.2 Parsování XML souborů

Ještě před tím, než přejdeme k prvnímu kroku vedoucímu k získání dat, připravíme si XML soubor s články do požadované podoby. Jelikož práce s XML není příliš praktická (soubor by musel být parsován při každém čtení) a vyhledávání v něm by bylo obtížné (potřebujeme získat text článků), vyparsujeme XML soubor ještě před jeho užíváním. Výsledkem jsou dva nové soubory - čistý textový soubor, ve kterém jsou uloženy pouze obsahy článků z XML souboru a databáze, jež nese informace o stránkách v textovém souboru (název stránky, id, zda je či není kategorií a především pak číslo bajtu, na kterém začíná text stránky v textovém souboru). Původní XML soubor pak již není potřebný. Výhodou navíc je fakt, že výsledný textový soubor je znatelně menší než původní XML (jsou totiž odstraněny všechny tagy a ponechán je pouze obsah článků). U testovaných souborů `simplewiki.xml` (558 MB) a `cswiki.xml` (2 599 MB) byla výsledná velikost textových souborů 342 MB a 1 930 MB, jde tedy o opravdu podstatný

rozdíl. Velikost výsledných databázových souborů je již vcelku zanedbatelná (6,9 MB a 26,6 MB pro zmiňované vstupní soubory).

Tabulka 6: Struktura databáze k odpovídajícímu textovému souboru

field	type	null	key
page_id	int	no	primary
page_name	text	no	
is_category	boolean	no	
position	int	no	

Postup parsování XML souboru je následující. Jako první je vytvořen databázový soubor, který obsahuje tabulku 6. Dále je vytvořen textový soubor, který nese obsahy stránek. Pomocí API bliki [18] je spuštěn proces, který iteruje přes každou stránku uloženou v XML. Pokud je stránka článkem nebo kategorií, je její obsah uložen do textového souboru a příslušná data jsou vložena do zmíněné databáze. V opačném případě je stránka ignorována. Na závěr je do databáze přidána „zarážka“ - to aby nedošlo k chybě při získávání textu posledního článku v databázi - jeho pozice v textovém souboru je vždy dána rozmezím bajtů označujících počátek téhož článku a dalšího následujícího. Zbývá dodat, že názvy stránek jsou uloženy bez prefixu jmenných prostorů a to z toho důvodu, že názvy stránek v tabulce categorylinks (viz 5) tyto prefixy také neobsahují (a není to ani potřeba, neboť tyto informace máme již uloženy v databázi).

4.3 Výběr kategorií

Prvním krokem směřujícím k získání dat je výběr článků. Jelikož chceme, aby články byly nějakým způsobem související (potřebujeme, aby jejich infoboxy měly co nejvíce společných atributů), jsou vybírány zejména články uvnitř dané kategorie (případně kategorií). Prvotním vstupem je tedy název požadované kategorie. Algoritmus výběru kategorie je velmi jednoduchý: zjistíme, zda zadaný vstup je kategorií, a pokud ano, je vrácen její název. Pokud ne, zjistíme zda alespoň existuje článek zadaného názvu - a pokud ano, je vrácen seznam kategorií, ve kterých se článek nachází. Implementace algoritmu pro online a offline režim je dosti odlišná.

Zdrojový kód 5 představuje metodu pro získávání kategorií v online režimu. Pro zjištění, zda kategorie opravdu existuje je třeba k zadanému vstupu přidat prefix jmenného prostoru pro kategorii (ten je získán na řádku 7 - je totiž odlišný pro každou jazykovou verzi Wikipedie, zatímco jeho konstanta je pro všechny stejná, viz tabulka 4). Dále se postupuje dle zmíněného algoritmu. Na řádku 17 je naopak tento prefix odebrán (není to estetické a ani potřeba).

V offline je verzi je nejprve poslán jednoduchý dotaz na databázi vzniklou vyparsováním XML souboru. Pokud vstup není kategorií, avšak článek existuje,

```
1  private final int CATEGORY_NAMESPACE = 14;
2  private Wiki wiki; // trida z knihovny wiki-java
3
4  public List<String> queryCategoriesOnline(String title) {
5
6     List<String> categories = new ArrayList<>();
7     String categoryIdentifier = wiki.namespaceIdentifier(
          CATEGORY_NAMESPACE);
8     String categoryTitle = categoryIdentifier + ":" + title;
9
10    String[] array = {categoryTitle, title};
11    boolean pageExists[] = wiki.exists(array);
12
13    if (pageExists[0]) {
14       categories.add(title);
15    } else if (pageExists[1]) {
16       for (String category : wiki.getCategories(title)) {
17          categories.add(category.replace(categoryIdentifier + ":", "
                "));
18       }
19    }
20    return categories;
21 }
```

Zdrojový kód 5: Získání kategorií v online režimu

je třeba najít jemu odpovídající kategorie v databázi categorylinks. Tento proces zobrazuje kód 6. Nejprve je zaslán dotaz na získání id článku (řádek 11). Pokud byl článek nalezen, je poslán dotaz na získání kategorií z databáze categorylinks (řádek 21). Cyklus na řádku 22 přidává nalezené kategorie do seznamu kategorií. Na řádku 24 jsou pro korektní zobrazování názvů stránek nahrazeny podtržítka mezerou.

Uživateli je dále nabídnuta možnost zúžení výběru kategorií pomocí výběru jejich subkategorií (ve stylu jednoduchého souborového manažeru). Je tedy umožněn průchod stromem kategorií, ale pouze od výchozí kategorie (a zpět) - pokud bychom chtěli postupovat opačným směrem, nevěděli bychom, z které kategorie jsme se dostali k oné výchozí. Implementace výběru subkategorií zvolené kategorie pro online verzi je podobná kódu 5 s tím rozdílem, že již nemusíme kontrolovat, zda kategorie skutečně existuje a místo kategorií, ve kterých je ona kategorie umístěna, hledáme její členy (jen ty, které jsou taky kategoriemi). V offline verzi musíme nejprve získat id stránek z databáze categorylinks. Tomuto dotazu odpovídá první řádek zdrojového kódu 7. Jelikož chceme jako výsledek názvy subkategorií, musíme vyhledat odpovídající názvy dle získaných id v databázi vzniklé vyparsováním XML souboru. Tento dotaz je k vidění na druhém řádku kódu. Alternativní možností pro offline režim je pak vyhledávání kategorií zcela přeskočit a rovnou přejít k následujícímu kroku (bez potřeby databáze categorylinks).

```
1  private final String TABLE_NAME;
2  private final String SQL_FIND_PAGE = "SELECT page_name, page_id FROM
       " + TABLE_NAME + " WHERE page_name = ?";
3  private Connection dumpDatabaseConnection;
4  private Connection mysqlConnection;
5
6  private List<String> queryCategoriesFromMySQL(String title) {
7
8     List<String> categories = new ArrayList<>();
9     PreparedStatement pstmt = dumpDatabaseConnection.prepareStatement
          (SQL_FIND_PAGE);
10    pstmt.setString(1, title);
11    ResultSet rs = pstmt.executeQuery();
12    int id;
13
14    if (rs.next()) {
15       id = rs.getInt("page_id");
16       mysqlConnection = mysqlDataSource.getConnection();
17       String sql = "SELECT cl_to FROM data WHERE cl_from = ?";
18       pstmt = mysqlConnection.prepareStatement(sql);
19       pstmt.setInt(1, id);
20
21       ResultSet results = pstmt.executeQuery();
22       while (results.next()) {
23          byte[] varbinary = (byte[]) results.getBytes("cl_to");
24          categories.add(new String(varbinary).replace("_", " "));
25       }
26       mysqlConnection.close();
27    }
28    return categories;
29 }
```

Zdrojový kód 6: Získání kategorií článku v offline režimu

4.4 Výběr článků

Jakmile máme k dispozici vybrané kategorie, můžeme přejít k dalšímu kroku - a to k získání článků, které jsou v oněch kategoriích umístěny. Zde jsou nabídnuty dvě možnosti - jednak můžeme požadovat pouze ty články, které jsou skutečně členy dané kategorie a jednak můžeme kategorií procházet do hloubky - tedy získáme i členy všech jejich subkategorií libovolného zanoření. Jestliže nejsme se získanými články spokojení (některý nám tam chybí), můžeme v tomto kroku přidávat články i ručně.

Implementace pro online verzi je znázorněna v kódu 8. Vstupem je množina zvolených kategorií. Poté stačí pro každou z těchto kategorií zavolat metodu pro získání jejich členů (tentokrát článků - viz atribut ARTICLE_NAMESPACE) a kontrolovat, zda nebyl přesažen nastavený limit pro počet článků (některé kategorie jich mohou obsahovat obrovské množství). Návratovou hodnotou je množina,

```
1 SELECT cl_from FROM category_links WHERE cl_to = title AND cl_type =
      'subcat';
2 SELECT page_name FROM data WHERE page_id IN (...); /* id stranek z
    predchoziho dotazu */
```

Zdrojový kód 7: Dotazy pro získání subkategorií v offline režimu

neboť není vyloučeno, že se jeden a ten samý článek může vyskytovat ve vícero kategoriích.

V offline režimu je třeba podobně jako u hledání subkategorií nejprve získat id článků vybraných kategorií z tabulky categorylinks a poté k nim najít odpovídající názvy v databázi vzniklé vyparsováním XML souboru. Tento dotaz je podobný tomu z prvního řádku v kódu 7, pouze s tím rozdílem, že potřebujeme získat články a ne kategorie. Dotaz má tedy podobu: `SELECT cl_from FROM categorylinks WHERE cl_to = title AND cl_type = 'subcat'LIMIT ?;` (kde otazník je roven konstantě omezující počet získaných členů). Druhý dotaz pro získání názvů článů je zcela totožný s dotazem na druhém řádku odkazovaného kódu.

Pokud chceme procházet kategorií do hloubky, tedy zahrnout i členy všech jejich subkategorií, jsou tyto metody rekurzivně volány pro všechny subkategorie zvolených kategorií. To je společné pro online i offline režim. Poslední možností je výběr všech členů bez ohledu na kategorii (pokud byl výběr kategorií přeskočen). Tato možnost je funkční pouze v offline režimu a smysl má jen tehdy, pokud jsou v dumpu články stejné kategorie (či alespoň nějakým způsobem souvisejících kategorií). Dotaz pro získání článků pak odpovídá prosté selekci všech článků z databáze vzniklé vyparsováním XML souboru, omezené limitem pro maximální počet článků.

4.5 Výběr šablon

Dalším krokem směřujícím k získání dat je získání šablon zvolených článků. I přesto, že už máme vybrány články, které by měly být nějakým způsobem související, nemáme nijak zaručeno, že objekty, o kterých články pojednávají, jsou stejného typu. Jako příklad si uveďme např. kategorii film - články uvnitř této kategorie nemusí pojednávat pouze o filmech, ale např. i o osobách souvisejících s filmem (režiséři, herci, atd.). V tomto kroku jsou navíc i získávány a ukládány do paměti texty článků - implementace následujících kroků je pak totožná pro online i offline režim.

Požadovaným typem šablon jsou infoboxy - odtud jsou čerpána data. Pro každý zvolený článek tedy potřebujeme získat seznam jeho infoboxů (může jich být více, ale i žádný), ostatní šablony nás (prozatím) nezajímají. Vybírány tak jsou pouze ty šablony, které ve svém názvu obsahují podřetězec „box“. Ne vždy je totiž infobox pojmenován jako infobox - existují i specifické infoboxy, jako

```
1  private final int ARTICLE_NAMESPACE = 0;
2  private int MAX_MEMBERS;
3  private Wiki wiki;
4
5  public Set<String> queryMembersOnline(Set<String> categories) {
6
7     Set<String> members = new LinkedHashSet<>();
8     String[] categoryMembers;
9
10    for (String category : categories) {
11
12       if (members.size() < MAX_MEMBERS) {
13
14          categoryMembers = wiki.getCategoryMembers(category,
               ARTICLE_NAMESPACE));
15          for (String member : categoryMembers) {
16
17             if (members.size() < MAX_MEMBERS) {
18                members.add(member);
19             } else {
20                break;
21             }
22          }
23       } else {
24          break;
25       }
26    }
27    return members;
28 }
```

Zdrojový kód 8: Získání členů kategorie v online režimu

např. taxobox, superherobox atd. Výsledkem je asociativní pole, kde je ke každé infoboxové šabloně přiřazen seznam článků, ve kterých se tato šablona vyskytuje. Zde je nutno podotknout, že tento způsob nemusí být zcela funkční pro všechny jazykové verze Wikipedie - např. pokud jsou názvy infoboxů na japonské verzi uvedeny japonskými znaky, tak zřejmě podřetězec „box“ neobsahují. Možným řešením by poté bylo zadání názvu požadovaných infoboxových šablon, pro nás to ale není podstatné. Dále je třeba mít na mysli, že mezi získanými šablonami se mohou objevit i jiné než infoboxy (např. navbox, sidebox a jiné). To nám ale nevadí - vybíráme pouze ty šablony, které skutečně požadujeme.

Zdrojový kód implementace v online režimu již uvádět nebudu - je opět jednoduchý a podobný předchozím uvedeným. Stačí zavolat metodu pro získání šablon dané stránky a zjistit, zda obsahuje zmíněný podřetězec („box“). Z názvu infoboxových šablon je poté ještě odstraněn prefix jmenného prostoru pro šablonu.

Podstatně složitější je implementace v offline režimu. Jelikož nemáme nikde dostupné informace o šablonách daného článku, musíme jej vyhledat přímo v

textu článku. Prvním krokem je tedy získání textu článku z dumpu. Tuto akci zobrazuje zdrojový kód 9. Prvně je třeba získat id řádku a pozici článku - z databáze vzniklé parsováním XML souboru (SQL dotaz je na řádku 2). Tato pozice je výchozí pozicí pro hledání článku v dumpu (je to číslo bajtu, na kterém článek začíná). Dále je potřeba získat pozici následujícího článku (dotaz na řádku 3). Rozdílem těchto pozic (řádek 21) získáme délku článku v bajtech. Poté tedy stačí načíst onen počet bajtů od výchozí pozice a převést jej do textového řetězce (řádky 24 - 26).

```
1  private final String SQL_SELECT_ARTICLE = "SELECT rowid, * FROM " +
       TABLE_NAME + " WHERE page_name = ? and is_category = ?";
2  private final String SQL_NEXT_POSITION = "SELECT position FROM " +
       TABLE_NAME + " WHERE rowid = ?";
3  private RandomAccessFile dump;
4
5  private String queryPageText(String title) {
6
7     PreparedStatement pstmt = dumpDatabaseConnection.prepareStatement
          (SQL_SELECT_ARTICLE);
8     pstmt.setString(1, title);
9     pstmt.setBoolean(2, false);
10    ResultSet rs = pstmt.executeQuery();
11    String content = "";
12
13    if (rs.next()) {
14       int rowId = rs.getInt("rowid");
15       pstmt = dumpDatabaseConnection.prepareStatement(
             SQL_NEXT_POSITION);
16       pstmt.setInt(1, rowId + 1);
17       ResultSet nextRs = pstmt.executeQuery();
18
19       long startPosition = rs.getLong("position");
20       long nextPosition = nextRs.getLong("position");
21       int length = (int) (nextPosition - startPosition);
22       byte bytes[] = new byte[length];
23
24       dump.seek(startPosition);
25       dump.read(bytes);
26       content = new String(bytes);
27    }
28    return content;
29 }
```

Zdrojový kód 9: Získání textu článku v offline režimu

Druhým krokem je nalezení infoboxů uvnitř textu článku (pro připomenutí formátu infoboxu odkazuji na zdrojový kód 3). K tomu poslouží následující regulární výraz: `"\\{{2}\\s*(\\p{L}+box[^}|<]*)"`. Hledáme tedy takový podřetězec textu, který začíná dvěma složenými závorkami (začátek šablony). Výraz

`\\s*` bere v potaz, že se mezi závorkami a názvem šablony mohou objevit bílé znaky. Předpis šablony totiž nebývá vždy dodržen úplně přesně a tyto znaky se tam skutečně objevují. Část `\\p{L}+box` má za úkol najít text (název infoboxu), jež obsahuje podřetězec „box“. Poslední část regulárního výrazu zajistí ukončení řetězce - nechceme získat celý zbytek textu nebo infoboxu, ale pouze jeho název. Výsledkem je pak výraz uvedený v kulatých závorkách.

4.6 Výběr atributů a hodnot

V tomto kroku již nezáleží na tom, zda jsme v online či offline režimu, vše potřebné (šablony a texty článků) už máme připravené. Data jsme již tedy vlastně získali, odteď se budeme zabývat jen jejich úpravou do požadované podoby. Cílem tohoto kroku je získat pro každý článek asociativní pole, jehož klíči jsou názvy atributů a prvky jejich hodnoty. Pokud článek obsahuje infoboxů více, je více i těchto polí - kontext uvažujeme vždy jen v rámci jedné šablony (infoboxy by mohly mít např. stejné atributy a pak by docházelo k problémům).

Prvním krůčkem je vytažení infoboxů a jejich obsahu z textu článku. Jelikož víme, které články obsahují které infoboxy, vyhledáváme pouze ty infoboxy, které se v textu skutečně vyskytují. Nejjednodušším řešením je nalezení indexu podřetězce s názvem infoboxu a dopočítání příslušného počtu závorek (stále se jedná o text ve wikiznačkách) iterací přes jednotlivé znaky. Regulární výraz by byl v tomto případě složitý a v mnohých případech by nebyl ani možný, neboť infobox může obsahovat další šablony, které mohou obsahovat také další šablony atd.

Jakmile máme připravený zdrojový text infoboxu, můžeme se pustit do jeho parsování. Tato část je zřejmě nejkomplikovanější částí programu. Prvně z infoboxu odstraníme nepotřebné údaje. Zvlášť problematickou částí jsou vnořené šablony. Jednou z nejčastěji se vyskytujících je šablona pro citaci. Tyto jsou z textu automaticky odstraňovány - nejsou potřebné a nemají žádnou přímou souvislost s daty v infoboxu. Jejich výskyt v textu je rozpoznán dle přítomnosti tagu `<ref>`, případně `<references />`. Regulární výraz pro nalezení těchto tagů a jejich obsahu odpovídá prvnímu řádku zdrojového kódu 10.

```
1 "<ref.*?>?(.*?\\n?)*<?/(ref)?>"
2 "\\{{2}[^{}]*\\}{2}"
3 "<!--(.*?)-->"
```

Zdrojový kód 10: Regulární výrazy pro odstraňování nepotřebných částí infoboxu

U ostatních šablon záleží na uživateli, zda bude chtít jejich obsah zahrnout do výsledných dat. Standardně jsou z infoboxů odstraňovány - ve většině případů nenesou příliš užitečná data (obvykle jde o popisy použitých souborů na stránce apod.), existují ale i šablony např. pro datum narození. Další jejich nevýhodou je, že často obsahují více parametrů ve formě atribut-hodnota (podobně jako

infobox). Pokud by se tedy v hodnotě infoboxového atributu nacházela šablona, mohli bychom získat jako hodnotu atributu data ve formě dalších atributů a hodnot, což není příliš žádoucí. Regulární výraz pro rozpoznání šablony v textu je uveden na řádku 2. V podstatě stačí najít cokoliv, co je obklopeno dvojitými složenými závorkami. Levá složená závorka v části `[^{}]*` zajistí, že jsou korektně odstraňovány i vnořené šablony (začne se totiž od té nejvíce zanořené). Víceřádkové šablony jsou z textu odstraňovány vždy. Vnořené infoboxy taktéž, to nám ale nijak nevadí - každý infobox stránky je totiž zpracováván samostatně. Dalším odstraňovaným a nepotřebným elementem jsou komentáře. Jejich podobu ukazuje regulární výraz na řádku 3.

Dalším krůčkem je získání seznamu atributů a jejich hodnot (zatím společně). Pro připomenutí formátování infoboxu znovu odkazuji na zdrojový kód 3. Nejjednodušším způsobem by bylo rozdělení textu infoboxu dle svislých lomítek - těmi jsou rozděleny jednotlivé atributy. To by ale bylo možné pouze v případě, že bychom z textu odstranili všechny šablony (uživatel má možnost jednořádkové šablony ponechat) a interní odkazy. Jelikož i jednořádkové šablony mohou obsahovat více parametrů, které jsou rozděleny také pomocí svislých lomítek, získali bychom nekonzistentní data. Je tedy potřeba vymyslet sofistikovanější způsob. Můžeme se spolehnout na to, že každému atributu a jeho hodnotě je v infoboxu vymezen samostatný řádek (případně více řádků). Budeme tedy infobox procházet řádek po řádku. U každého řádku je kontrolováno, zda jde o nový atribut či zda jde o pokračování jeho hodnoty (ty mohou být víceřádkové). Pokud je po odstranění bílých znaků řádek začíná jakýmkoli jiným znakem než svislým lomítkem, jedná se o pokračování hodnoty a řádek připojíme k tomu předchozímu. Tím získáme seznam atributů a jejich hodnot.

Konečně se dostáváme k převodu dat do podoby, jaká byla představena v úvodní kapitole. K dispozici máme seznam textových řetězců s atributy a jejich hodnotami. Tyto řetězce pak stačí rozdělit podle rovnítka na atribut a hodnotu a každý z takto vzniklých řetězců očistit od nepotřebných elementů značkovacího jazyka Wikipedie. S atributy není mnoho práce - stačí odstranit svislé lomítka, ořezat bílé znaky a případné podtržítka nahradit mezerami. Více práce je s jejich hodnotami, neboť mohou obsahovat prakticky jakékoli wikiznačky. Aby byla výsledná data čitelná, pokusíme se wikiznačky odstranit, alespoň ty nejčastější. Využívat k tomu budeme regulárních výrazů.

Jelikož nechceme vždy řetězec daného vzoru zcela odstranit (např. u odkazů a šablon chceme získat jejich obsah), řetězec odpovídající zadanému vzoru často nahradíme některou ze skupin znaků ve vzoru. K tomu slouží algoritmus ve zdrojovém kódu 11. Vstupem je řetězec představující hodnotu atributu, regulární výraz a index skupiny znaků v regulárním výrazu. Cyklus na řádku 6 pak nahradí všechny nalezené podřetězce splňující vzor požadovanou skupinou znaků. V následujícím textu si uvedeme regulární výrazy (viz zdrojový kód 12), pomocí nichž upravíme elementy značkovacího jazyka Wikipedie, jejichž formátování jsme si uvedli v minulé kapitole, do požadované podoby.

Řádek 1 odpovídá regulárnímu výrazu pro odstranění apostrofů. Pro nás ne-

```
1  private String cleanWithRegex(String value, String regex, int group)
       {
2
3     Pattern p = Pattern.compile(regex);
4     Matcher m = p.matcher(value);
5
6     while (m.find()) {
7        value = value.replaceFirst(regex, (m.group(group)));
8     }
9     return value;
10 }
```

Zdrojový kód 11: Čištění řetězce pomocí regulárních výrazů

mají žádný význam, ať se již jedná pouze o text uvozený v apostrofech či speciální případ formátování (tučné písmo, kurzíva). Dalšími nepotřebnými částmi textu jsou tagy. Ty jsou z textu odstraněny všechny. Na řádku 2 je k vidění odpovídající regulární výraz. Speciálním případem je `<br>` (odřádkování), jenž je nahrazeno znakem pro nový řádek. Regulární výraz na řádku 3 bere v potaz HTML5 i jeho starší podobu (`<br />`). Pokud řádek textu začíná jednou či několika hvězdičkami, tak víme, že jde o označení položky odrážkového seznamu. O nalezení a odstranění těchto znaků se stará výraz na řádku 4.

```
1  "\\'+(.*?)\\'+"
2  "<[^>]*>";
3  "\\s*(<br\\s*/>|<br>)\\s*"
4  "\\s*\\*+\\s*"
5  "\\[([^\\s]*)(.*?)\\]"
6  "\\[{2}([^|\\]]*\\|)?(.*?)\\]{2}";
7  "\\{{2}([^|\\}]*\\|)?([^{}]*)\\}{2}"
```

Zdrojový kód 12: Regulární výrazy pro úpravu atributových hodnot

Poněkud složitější situací jsou odkazy a šablony. V případě externího odkazu je třeba hledat text uvozený hranatými závorkami. Externí odkazy jsou dvojího typu - buď obsahují pouze URL adresu nebo i její popisek (odděleno mezerou). Zajímají nás pouze URL a o její vytažení se postará výraz na řádku 5. Postačí tedy najít text uzavřený v jednoduchých hranatých závorkách a vyjmout veškerý obsah po první bílý znak (tomu odpovídá první skupina znaků ve výrazu). Pokud se jedná o odkaz interní, text je uzavřen dvěma hranatými závorkami. Jeho obsahem může být opět pouze odkaz (název stránky) nebo i nepovinný parametr - popisek. Ten je oddělen svislým lomítkem. V tomto případě nás naopak zajímá primárně tento parametr, neboť má užší souvislost s textem. Uveďme si příklad. Pokud je atributem rok premiéry filmu, jeho hodnota může být např. 2000. Na ni může být navázán odkaz na stránku, jejíž název je např. „Filmy natočené v roce

2000“. Jelikož chceme získat hodnotu v co nejvíce heslovité podobě, tak budeme požadovat jako výsledek jednoduše rok 2000 a ne název stránky. Z tohoto důvodu jsou interní odkazy v textu nahrazovány jimiž popisky. Odpovídající regulární výraz je k vidění na řádku 6. Vyhledán je tedy text uzavřený dvojitými hranatými závorkami. Pokud se uvnitř vyskytuje svislé lomítko, je veškerý obsah před ním ignorován (včetně lomítka). Požadovaným řetězcem je tedy druhá skupina znaků ve výrazu. Pokud jsou v textu ponechány i šablony, jejich zpracování je téměř totožné s úpravou interních odkazů. Rozdíl je jen v tom, že je vyhledáván text uvnitř dvojitých složených závorek (řádek 7). Druhou změnou je levá složená závorka ve druhé skupině znaků ve výrazu. Ta je (opět) pro ošetření případu, kdy se uvnitř šablony nachází další vnořená šablona - začíná se od nejvíce zanořené.

V tento moment již máme k dispozici pro každý článek a jemu odpovídající šablony asociativní pole s atributy infoboxů a jejich hodnotami - nic nám tedy nebrání vytvořit tabulku s daty ve formě, jakou jsme si uvedli v úvodní kapitole. Došli jsme tak ke kýženému cíli - získali jsme vícehodnotové kontexty. Zbývá snad jen podotknout, že ve zdrojovém kódu infoboxů se teoreticky mohou vyskytnout i jiné wikiznačky, než byly výše uvedeny. Kompletní parsování zdrojového kódu Wikipedie by ovšem bohatě vydalo na samostatnou práci.

4.7 Binarizace dat

Posledním, avšak již volitelným krokem je vytvoření tabulky binárních dat. To je provedeno pomocí škálování, jež bylo představeno v úvodní kapitole. Škáluje se pouze v rámci zvoleného atributu, výsledná tabulka tedy odpovídá škále pro daný atribut. Celý princip je velmi jednoduchý - ke zvolenému atributu je vytvořena množina všech hodnot, které se v rámci tohoto atributu vyskytují. Atributy nové tabulky jsou tak tyto jednotlivé hodnoty. Výsledná tabulka nám ukáže, zda dané objekty splňují dané atributy - výsledek odpovídá binární matici s logickými hodnotami.

Je třeba dodat, že tento způsob není vždy zcela vyhovující a smysl má zejména v případě heslovitých hodnot. Pokud máme např. tabulku, jejíž řádky reprezentují jednotlivé filmy a jedněmi z atributů je režie a žánr, získáme tak užitečné údaje v přehledné formě (podobně jako bylo znázorněno na tabulce 3 v úvodní kapitole o datech). Horší výsledky jsou u číselných údajů - atributy nové tabulky jsou všechna čísla vyskytující se v rámci daného atributu. U složitějších údajů (např. adresy bydlišť) pak toto počínání není příliš užitečné. Často se vyskytujícím problémem je i to, že stejné údaje mohou mít různou formu. Např. stejné datum narození může být reprezentováno jako 12. 12. 2000, 12. prosinec 2000, 12. prosince 2000 nebo i šablonou `{{datum narození| 12 | 12 | 2000}}`. Škálování takových hodnot je poté velmi obtížné a nad rámec této práce.

Pro lepší výsledky je možno použít některého z existujících konvertorů datových formátů. Jedním z nich je např. Swift - Relational Data Convertor. [19] Získaná data je možno kromě binarizace dále zpracovávat na jiné vícehodnotové kontexty či s nimi provádět další libovolné akce dle potřeb uživatele.

Závěr

Výsledkem práce je aplikace pro získávání tabulkových dat z obsahu Wikipedie. Jejím primárním úkolem je získání vícehodnotových kontextů z infoboxů vybraných článků, sekundárním pak tyto data pomocí škálování převést na data binární. K datům je možno přistupovat jak v online, tak v offline režimu, a to pomocí několika kroků - vyhledání a výběru kategorie (či kategorií), výběru požadovaných článků, šablon a atributů.

Nejproblematičtější částí práce bylo převedení infoboxů do čitelné podoby. Wikipedie používá svůj vlastní značkovací jazyk a proto nebylo možné použít existujících prostředků pro převod HTML do čistého textu. Ačkoliv bylo několik dostupných i pro značkovací jazyk Wikipedie, obvykle nebyly kompletní nebo nevracely text v požadované podobě. Není tedy zcela vyloučeno, že se mohou vyskytnout i články, jejichž data nebudou vrácena v čisté textové podobě či případně neprojdou popsanými algoritmy.

Dalším problémem byly i chyby vyskytující se na straně Wikipedie. Ne u všech článků je dodržena přesná podoba formátování dle dokumentace, což opět opět může vyústit v nekorektní výsledky. Narazil jsem dokonce i na případ, kdy členem kategorie byla téže kategorie, díky čemuž byl znemožněn výběr jejich členů do hloubky - neboť docházelo k nekonečnému procházení kategorie.

V naprosté většině případů byl ovšem průchod všemi kroky bezproblémový a lze tedy říci, že oba úkoly aplikace byly zdárně naplněny.

Conclusions

The result of this thesis is an application for obtaining table data from Wikipedia content. The primary task of the application is to obtain multi-valued contexts from the infoboxs of selected articles, the secondary task is to convert these data to binary data by scaling. Data can be accessed both online and offline, with a few steps - finding and selecting category (or categories), selecting the desired articles, templates, and attributes.

The most problematic part of the work was the conversion of infoboxes into a readable form. Wikipedia uses its own markup language and therefore it was not possible to use existing HTML tools for conversion into pure text. Although there were several tools for Wikipedia markup language available, they were usually not complete or did not return the text in the desired form. So the existence of articles whose data will not be returned in plain text (or alternatively do not pass the described algorithms) is not completely excluded.

Another problems were the errors on the Wikipedia side. Not all articles maintain the exact formatting according to the documentation, which again can result in incorrect results. I even came across a case where a category member was the same category, making it impossible to select their members in depth - because the category was endlessly browsed.

Hovewer, in the vast majority of cases, the passage through all the steps was trouble-free, and so it can be said that the two tasks of the application were successfully fulfilled.

A Obsah přiloženého CD/DVD

bin/
: Program WikiExtractor, spustitelný přímo z CD/DVD. Adresář obsahuje i všechny knihovny a další soubory potřebné pro bezproblémový programu z CD/DVD.

doc/
: Text práce ve formátu PDF, vytvořený s použitím závazného stylu KI PřF UP v Olomouci pro závěrečné práce, včetně všech příloh, a všechny soubory potřebné pro bezproblémové vygenerování PDF dokumentu textu (v ZIP archivu), tj. zdrojový text textu, vložené obrázky, apod.

src/
: Kompletní zdrojové kódy programu.

readme.txt
: Instrukce pro spuštění programu, včetně všech požadavků pro jeho bezproblémový provoz.

Navíc CD/DVD obsahuje:

data/
: Ukázková a testovací data použitá v práci a pro potřeby testování práce při tvorbě posudků a obhajoby práce.

Literatura

[1] BĚLOHLÁVEK, Radim. Konceptuální svazy a formální konceptuální analýza [online]. 19 s. [cit. 2017-07-13].
Dostupné z: http://belohlavek.inf.upol.cz/publications/Bel_Ksfka.pdf

[2] BĚLOHLÁVEK, Radim. Introduction to Formal Concept Analysis [online]. Department of Computer Science, Faculty of Science, Palacký Universtity, Olomouc. 47 s. [cit. 2017-07-14].
Dostupné z: https://phoenix.inf.upol.cz/esf/ucebni/formal.pdf

[3] Wikimedia Foundation [online]. [cit. 2017-07-15]. Dostupné z:
https://wikimediafoundation.org/wiki/Home

[4] MediaWiki [online]. [cit. 2017-07-15]. Dostupné z:
https://www.mediawiki.org/wiki/MediaWiki

[5] Wikipedia [online]. [cit. 2017-07-15]. Dostupné z:
https://en.wikipedia.org/wiki/Wikipedia

[6] Česká Wikipedie [online]. [cit. 2017-07-15]. Dostupné z:
https://cs.wikipedia.org/wiki/Česká_Wikipedie

[7] Wikipedia:Template Namespace [online]. [cit. 2017-07-16]. Dostupné z:
https://en.wikipedia.org/wiki/Wikipedia:Template_namespace

[8] Nápověda:Jmenný prostor [online]. [cit. 2017-07-16]. Dostupné z:
https://cs.wikipedia.org/wiki/N%C3%A1pov%C4%9Bda:Jmenn%C3%BD_prostor

[9] Wikipedie:Šablony [online]. [cit. 2017-07-17]. Dostupné z:
https://cs.wikipedia.org/wiki/Wikipedie:%C5%A0ablony

[10] 2001: Vesmírná odysea [online]. [cit. 2017-07-17]. Dostupné z:
https://cs.wikipedia.org/wiki/2001:_Vesm%C3%ADrn%C3%A1_odysea_(film)

[11] Help:Wiki markup [online]. [cit. 2017-07-18]. Dostupné z:
https://en.wikipedia.org/wiki/Help:Wiki_markup

[12] Wikipedia:Database download [online]. [cit. 2017-07-18]. Dostupné z:
https://en.wikipedia.org/wiki/Wikipedia:Database_download

[13] Wikimedia Downloads [online]. [cit. 2017-07-18]. Dostupné z:
https://dumps.wikimedia.org/

[14] Export Pages [online]. [cit. 2017-07-18]. Dostupné z:
https://en.wikipedia.org/wiki/Special:Export

[15] Free Java Download [online]. [cit. 2017-07-18]. Dostupné z:
https://www.java.com/en/download/

[16] ControlsFX [online]. [cit. 2017-07-18]. Dostupné z: http://fxexperience.com/controlsfx/

[17] MER-C/wiki-java [online]. [cit. 2017-07-18]. Dostupné z: https://github.com/MER-C/wiki-java

[18] KRAMER, Axel. Wiki (info.bliki.wiki) [online]. [cit. 2017-07-18]. Dostupné z: https://bitbucket.org/axelclk/info.bliki.wiki/wiki/Home

[19] NOVÁČEK, Jan. Swift - Relational Data Convertor [online]. [cit. 2017-07-21]. Dostupné z: http://gnovis.github.io/swift/