Západočeská univerzita v Plzni

Fakulta aplikovaných věd

Katedra informatiky a výpočetní techniky

Bakalářská práce

Vytváření citačních sítí z bibliografických dat

Plzeň, 2013 Radek Bouda

Prohlášení

Prohlašuji, že jsem bakalářskou práci vypracoval samostatně a výhradně s použitím citovaných pramenů.

V Plzni dne 18. 6. 2013, Radek Bouda

Abstract

Creation of citation networks from bibliographic data

Scopus Data Miner, ACM Data Harvester and Google Scholar Grabber are three different tools which can be used for simple creation of citation networks.

This bachelor thesis deals with creating citation networks using these tools. It describes their maintenance and their implementation into meta-browser. It also describes creation of universal XML structure of outputs of these tools.

It describes how to create a citation network and which conditions must be stand. The target of this thesis is to create the biggest and the most up-to-date citation network.

Vytváření citačních sítí z bibliografických dat

Scopus Data Miner, ACM Data Harvester a Google Scholar Grabber jsou tři rozdílné nástroje, kterými lze jednoduše vytvářet citační sítě.

Tato bakalářská práce se zabývá vytvářením citačních sítí těmito nástroji. Popisuje problém jejich údržby a jejich implementaci do meta-vyhledávače. Také se zabývá vytvořením univerzální XML struktury pro výstupy z těchto nástrojů.

Popisuje, jak postupovat a jaké podmínky dodržet při vytváření citační sítě. Cílem práce je vytvořit co největší a nejaktuálnější citační síť.

Poděkování

Tímto děkuji Ing. Daliboru Fialovi, Ph.D. za vedení mé bakalářské práce, zvláště pak za podnětné připomínky, rady a čas, který mi věnoval.

Obsah

1 Úvod

Cílem projektu je využít dostupné nástroje k získávání bibliografických dat z webových služeb, z nichž se budou následně vytvářet citační sítě. Aktuálně jsou k dispozici nástroje k získávání dat ze služeb ACM Digital Library, Google Scholar a Scopus.

Jedním z cílů bude ověřit funkčnost těchto tří nástrojů a následně je implementovat do jedné aplikace. Tato aplikace bude mít jednotné uživatelské rozhraní pro všechny výše uvedené nástroje a umožňovat jednoduché a účinné ovládání.

Veškeré výstupy jmenovaných nástrojů jsou do souborů formátu XML. Dalším z cílů tedy bude strukturu XML sjednotit pro lepší orientaci a budoucí využití.

Záměrem práce je vytvořit XML soubor s co největší citační sítí pomocí těchto nástrojů.

2 Nástroje pro získávání dat z webových bibliografických databází

Nástroje a aplikace popsané v této kapitole jsou detailněji popsány v dokumentech [1], [2] a [3].

2.1 Google Scholar

Scholar je webová bibliografická databáze vytvořená firmou Google. Nástroj pro získávání dat z této databáze se nazývá Google Scholar Grabber. Tento nástroj však kvůli omezenosti filtrace Googlem nabízí velmi málo kritérií ke specifikaci vyhledávaných výsledků. Patří mezi ně například vyhledávání pouze v titulku, omezení výsledků rokem jejich publikace nebo vyhledávání podle jména autora. Srovnáme-li tyto možnosti filtrace s možnostmi filtrace u jiných nástrojů, zjistíme, že ostatní nástroje mají daleko širší možnosti.

Ačkoliv je práce s webovým rozhraním Scholaru jednoduchá a pohodlná, problémem je, že Google usilovně bojuje proti robotickým přístupům k jeho serverům. To se projevuje tak, že v případě neobvykle velké aktivity z nějaké sítě začne webová stránka při dalším přístupu z této sítě vyžadovat opsání kontrolního textu CAPTCHA[1]. V případě neplatného pokusu o projití CAPTCHA testem Google síť, ze které se snažíme přistupovat, „odřízne“ úplně – vrací chybu 503[2]. Řešení tohoto problému není obtížné, ale časově náročné. Efektivním řešením se zdá být nastavení hodinové přestávky po 150 přístupech, mezi nimiž je vteřinová prodleva. Z těchto konstant lze jednoduše určit, že při pokusu o stažení 1500 článků bude aplikace min. 9 hodin nečinná. Tento čas je nevyužitý a tento postup je neefektivní. Naneštěstí, jediný možný.

Ačkoliv databáze Googlu je rozsáhlá (využívá i ACM Digital Library a SciVerse Scopus), kvůli nárokům aplikace na čas potřebný k získání dat, se souhlasem vedoucího práce, po implementaci do meta-aplikace se Scholarem dále pracovat nebudeme a ve výsledku ho ani nevyužijeme.

Uživatelské rozhraní nástroje Google Scholar Grabber je ukázáno v příloze B.

[1] Program, který chrání webové stránky proti robotům generováním testů, kterými lidé projdou ale současné počítačové programy ne. [10]
[2] 503 – Služba nedostupná (*Service Unavailable*) je chybový kód HTTP protokolu. [11]

2.2 ACM Digital Library

Společnost ACM prostřednictvím webové služby ACM Digital Library poskytuje nejobsáhlejší kolekci full-textových článků a bibliografických záznamů, které se týkají oblasti výpočetní techniky a informačních technologií. Kromě této služby společnost nabízí ještě službu ACM GUIDE, která taktéž poskytuje bibliografické informace, ty jsou však velmi často neúplné.

Implementace nástroje získávajícího bibliografická data z této webové aplikace je vskutku kvalitní. Nástroj se nazývá ACM Data Harvester a už v základu umí získávat data jak z databáze Digital Library tak z databáze GUIDE. To může být velmi užitečné, protože ačkoliv databáze GUIDE obsahuje neúplné bibliografické informace, obsahuje především citace, což je pro nás důležité. Kromě této výhody má ACM Data Harvester ze jmenovaných nástrojů navíc nejširší možnosti filtrování výsledků. Umožňuje vyhledávat pomocí klíčových slov, roku vydání nebo autora/editora. Kromě těchto možností můžeme zvolit třeba sponzora, místo konání konference, ISBN/ISSN nebo afiliaci.

Ačkoliv ACM Data Harvester nabízí mnoho možností k filtraci výsledků, což může být užitečné, jeho databáze je oproti databázi Scopusu vcelku omezená. Se souhlasem vedoucího práce tedy ACM Data Harvester do meta-aplikace implementujeme, ale dále ho nevyužijeme.

Uživatelské rozhraní nástroje ACM Data Harvester je ukázáno v příloze C.

2.3 SciVerse Scopus

Webová aplikace nakladatelské firmy Elsevier je považována za nejrozsáhlejší databázi bibliografických dat. Obsahuje záznamy ze zdravotních, fyzikálních, sociálních, humanitních věd a věd o živé přírodě.

Od ostatních služeb se liší tím, že je třeba mít pro práci s ní zaplacenou licenci, což třeba například Google Scholar nevyžaduje. Nástroj pro získávání dat z ní, Scopus Data Miner, s tímto faktem počítá. Jelikož Západočeská univerzita v Plzni má licenci zaplacenou a přístup je přes školní síť funkční, lze problém s licencí částečně vyřešit

připojením přes univerzitní VPN. To samozřejmě vyžaduje Orion konto, které však musí mít každý student i zaměstnanec.

Scopus Data Miner v základu nabízí relativně hodně možností filtrování výsledků, méně však než ACM Data Harvester. Umožňuje například vyhledávat podle titulku, abstraktu nebo autora. Dále nabízí možnost omezit publikace rokem vydání a zároveň vybrat pouze určité typy dokumentů. Také umožňuje filtrovat výsledky pouze z určité vědecké oblasti.

Scopus Data Miner se svými možnostmi filtrace se tedy pohybuje někde mezi dvěma výše zmíněnými nástroji. Množstvím dat však převyšuje obě výše zmíněné databáze. Se souhlasem vedoucího práce tedy bude středem našeho zájmu a z této databáze se budeme snažit vytvořit citační síť.

Uživatelské rozhraní nástroje Scopus Data Miner je ukázáno v příloze A.

2.4 Společný nedostatek nástrojů pro získávání dat

Všechny tři z výše zmíněných nástrojů, pomocí kterých získáváme data z webových databází, trpí jedním význačným nedostatkem, který naneštěstí velmi ztěžuje naší snahu o vytvoření co největší citační sítě.

Všechny tyto nástroje přistupují k webovým aplikacím prostřednictvím protokolu HTTP. Aplikace se „maskují“ jako webový prohlížeč a stahují celé zdrojové kódy jednotlivých webových stránek. Stažený zdrojový kód je následně parsován[3] a jsou z něj fyzicky získána žádaná data.

Data se vyhledávají v elementech zdrojového kódu podle jednoznačně nastavených konstant. Z toho plyne náchylnost ke změnám zdrojového kódu. Autorům webových stránek nic nebrání ve změně zdrojových kódů a je jasné, že i jejich sebemenší změna může způsobit částečnou či úplnou nefunkčnost daného nástroje.

Je tedy nutné tyto nástroje neustále udržovat. Protože ke změnám webů a tedy i zdrojových kódů dochází nepravidelně, je těžké udržovat všechny 3 nástroje najednou a soustředíme se tedy pouze na jeden nástroj – Scopus Data Miner. Bohužel vzhledem

[3] Parsován – strojově zpracován nástrojem nazývaným parser

k tomu, že tyto webové aplikace neposkytují žádné, případně nepoužitelné API, není aktuálně jiný jednoduchý způsob, jak se k datům dostat.

3 XML

Detaily jazyka XML jsou blíže popsány na webu [4] a v dokumentu [5].

3.1 Popis jazyka XML

XML je zkratka pro „*eXtensible Markup Language*“, což můžeme přeložit jako rozšiřitelný značkovací jazyk. Primárně je to jazyk, vytvořený k ukládání a přenosu dat. XML tedy popisuje, jaká data jsou, co obsahují a to nezávisle na platformě. Nepopisuje už, jak data vypadají, a samo o sobě nevykonává žádnou činnost.

3.2 Příklad souboru .xml

Pro další popis XML použiji jako příklad následující xml soubor:

```
<schuzky>
	<schuzka>
		<misto>Plzeň, kavárna na náměstí</misto>
		<osoba>Petr Novák</osoba>
	</schuzka>
	<schuzka>
		<misto>Plzeň, zasedání na Borech</misto>
		<osoba>Jan Navrátil</osoba>
	</schuzka>
</schuzky>
```

3.3 Pouze strukturovaná data

Soubor .xml je textový soubor, zobrazitelný a upravitelný v libovolném textovém editoru. Po otevření takového souboru například v poznámkovém bloku vidíme, že XML splňuje všechny náležitosti, které jsou zmíněny výše.

Dokument jako takový vůbec nepopisuje, jak mají data vypadat. Nepopisuje, jestli je text tučný nebo psaný kurzívou. Zároveň se nejedná o spustitelný soubor, který by mohl sám o sobě vykonávat nějakou činnost. Při tomto všem však po zobrazení dokumentu vidíme, že obsahuje dvě schůzky. Jedna z nich je v Plzni na náměstí s Petrem Novákem a druhá v Plzni na Borech s Janem Navrátilem. Tím se dostáváme k největší výhodě formátu XML. Po otevření dokumentu v textovém editoru i bez použití konkrétní aplikace, k níž by měl být tento soubor přidružen (v našem případě nějaký plánovač schůzek), vidíme, jaká data přesně soubor obsahuje, a můžeme si domyslet jejich pravděpodobný význam.

3.4 Značky

Slovo rozšiřitelný v definici zkratky XML si můžeme představit například jako možnost vytvářet si v dokumentu vlastní značky. Značky nejsou nikde předdefinované. Z toho vyplývá, že každý dokument, jeho struktura a vzhled jeho značek je plně v rukách autora dokumentu.

3.4.1 Formát značek v XML dokumentech

Při definování značek v XML se musíme řídit následujícími pravidly:

- Lze používat libovolná písmena, číslice a znaky „-“ (pomlčka), „_“ (podtržítko) a „.“ (tečka)
- Značka nesmí začínat číslicí, interpunkčním znaménkem nebo znaky „xml“
- Značky jsou case-sensitive[4]

Velmi důležité při definování značek v XML je zachování významovosti značek. Nesmíme zapomenout, že ačkoliv značky „pouze“ určují strukturu XML dokumentu, zároveň nám umožňují určit význam dat. Jde o to, že pojmenujeme-li značku „<o>“, bude značka sice správně, ale nelze podle ní určit druh či význam obsahu. Použitím takovýchto značek bychom se naprosto dobrovolně ochudili o jednu ze základních výhod formátu XML.

3.4.2 Správný zápis struktury značek

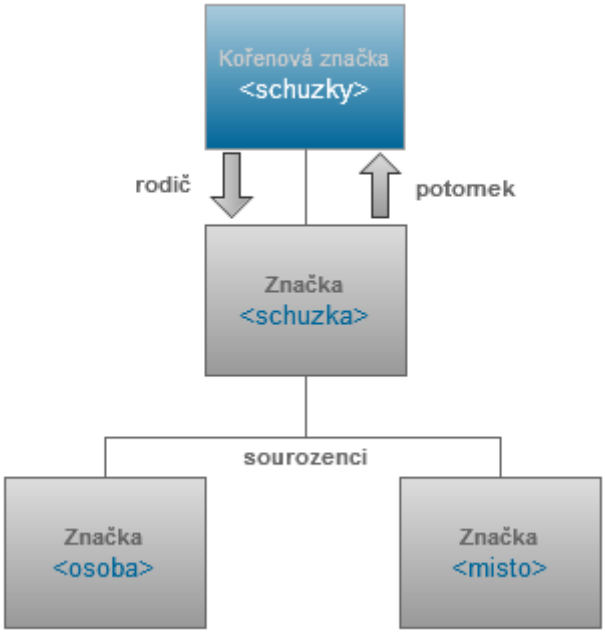
Při zápisu značek se opět snažíme dodržovat několik jednoduchých pravidel:

- Jedna značka je kořenová
- Ostatní značky obsahují buď další značky nebo data
- Značky skládáme do stromové struktury
- Každý potomek má nejvýše jednoho rodiče
- Každá značka je buď prázdná („<prazdnaZnacka />“), nebo má svou ukončující značku

[4] Case-sensitive: výraz pro situaci kdy záleží na velikosti písmen – zde značky „schuzka“ a „Schuzka“ jsou dvě různé značky

Když se podíváme do příkladu .xml souboru v kapitole 3.2, vidíme, že soubor splňuje všechny zmíněné požadavky. Kořenová značka je „schuzky“. Značky jsou poskládané do stromové struktury a nejsou překřížené.

Vzhledem k tomu, že značky skládáme do stromové struktury, velmi často se používají názvy jako „potomek“ (*child*), „rodič“ (*parent*), „sourozenec“ (*sibling*). Tyto vztahy jsou vysvětleny v Obrázek 1 - Vztahy v XML dokumentu.



Obrázek 1 - Vztahy v XML dokumentu

3.4.3 Atributy

Každá značka v dokumentu může mít volitelně 0 až N atributů (občas také nazývaných parametrů). Atribut a hodnota je nerozdělitelná, k sobě patřící dvojice. Jsou spojeny znakem „=“ (rovná se) a hodnota je uzavřena mezi znaky „"“ (uvozovky). Každý atribut je jedinečný (nesmí se opakovat). V množině dvojic atributů a hodnot nezáleží na pořadí.

Možné zápisy tedy jsou, například:

```
<misto mesto="Plzeň">Kavárna na náměstí</misto>

<misto mesto="Plzeň" popis="Kavárna na náměstí" />
```

Je vždy k zamyšlení a někdy úplně jednoduché rozhodnout, který zápis použít. Tyto dva zápisy dohromady se zápisem v ukázkovém .xml souboru v kapitole 3.2 totiž popisují

stejnou informaci třemi různými způsoby. Nejvhodnější je podle mě použít první, ze zmíněných v této kapitole. Hodnota parametru „mesto“ může původně vycházet z nějakého výčtu, a bližší popis už může být individuální. U druhého zde zmíněného už to tak jisté není. Způsob zapsání v ukázkovém případě bychom museli zpracovat, pokud bychom chtěli zjistit pouze město.

Atributy mají několik nevýhod, kvůli kterým se doporučuje je nepoužívat. Mezi ty nejzákladnější z nich patří:

- Atributy nemohou obsahovat, na rozdíl od značek, více hodnot
- Atributy nemohou obsahovat, na rozdíl od značek, stromové struktury
- Atributy nejsou jednoduše rozšiřitelné pro budoucí změny ve struktuře

Pro data se tedy doporučuje používat značky. Atributy se pak doporučuje používat pro informace, které nejsou k datům relevantní. To můžou být například celočíselné identifikátory, pokud bychom chtěli například schůzky v našem ukázkovém .xml souboru v kapitole 3.2 číslovat nebo počítat.

3.5 Datový obsah značek

V předchozích kapitolách bylo zmíněno, že značky mohou obsahovat další značky, atributy a text (ačkoliv značkám obsahujícím text i další značky se snažíme vyhnout). Jistá omezení však existují i pro samotný text, tedy datový obsah dokumentu.

3.5.1 Speciální znaky

Mezi znaky, které se v dokumentu nesmí vyskytnout, patří „<“ (menší než), „>“ (větší než), „&“ (ampersand), „‘“ (apostrof), „"“ (uvozovky). Tyto znaky je dobrým zvykem nahrazovat předdefinovanými entitami v XML. Dobrým zvykem, protože pouze znaky „<“ (menší než) a „&“ (ampersand) jsou v XML striktně zakázané. Popis jednotlivých entit viz Tabulka 1 - Entity v XML.

Znak	Entita
< (menší než)	<
> (větší než)	>
& (ampersand)	&
‘ (apostrof)	'
" (uvozovky)	"

Tabulka 1 - Entity v XML

3.5.2 Zdrojové kódy a jiné speciality

Pokud chceme, nebo z nějakého důvodu musíme do XML začlenit např. části zdrojových kódů, je vhodné je umístit do bloku nazývaného „CDATA“.

Text v elementech obecně patří do množiny „PCDATA“. Tato zkratka znamená „*parsed character data*“, tedy „parsovaná znaková data“. Parser zpracovávající dokument musí zpracovat i tato textová data, protože obsahem značek mohou být další značky, které musí najít a zpracovat. Oproti tomu, „CDATA“ je část souboru, která obsahuje neparsovaná data a parser tento blok ignoruje. Blok „CDATA“ může být zapsán například jako v následující ukázce.

```
<duleziteFunkce>
      <funkce>
            <nazev>jeMensi</nazev>
            <zdroj>
            <![CDATA[
            boolean jeMensi(prvni, druhy){
                  if(prvni < druhy) {
                        return true;
                  }
                  else {
                        return false;
                  }
            }
            ]]>
            </zdroj>
      </funkce>
</duleziteFunkce>
```

V ukázce je vidět, že při použití bloku CDATA nemusíme nahrazovat nepovolené znaky entitami (viz kapitola 3.5.1), protože parser blok neprochází a vnímá jej jako textový celek. Jediné, co nesmí blok CDATA obsahovat je řetězec „]]>“. Díky tomu ani není možné použití vnořených bloků CDATA.

3.6 Kontrola a validace XML dokumentů

Pokud chceme kontrolovat/validovat XML dokumenty, nabízí se několik možností. Z pravidla při každém parsování dochází ke kontrole, jestli je dokument tzv. „*well-formed*“ neboli správně strukturovaný. To je základním předpokladem, aby mohl být dokument vůbec parsován. Pokud vyžadujeme nějakou důslednější kontrolu dokumentu, můžeme dokument validovat tzv. „proti DTD[5]“ nebo „proti schémovému souboru“. Oba dva způsoby mají své výhody i nevýhody.

3.6.1 Well-formed dokument

Správně strukturovaný dokument musí splňovat následující podmínky:

- Dokument musí mít kořenovou značku
- Značky musí být správně ukončeny
- Značky jsou case-sensitive
- Značky musí být správně zanořené
- Hodnoty XML atributů musí být v uvozovkách

Pouze v případě, že je dokument správně strukturovaný, může být dokument i validní, ať už proti DTD nebo proti schémovému souboru.

3.6.2 Validace proti DTD

DTD je obecně užívaná zkratka pro definici typu dokumentu. Vymezuje, jaké značky se mohou v daném dokumentu vyskytnout, jak mohou být uspořádány a zanořeny, jaké mohou mít atributy a jaké jsou jejich typy. Výhodou DTD je, že je velmi jednoduché. Nevýhody jsou však mnohem závažnější. DTD samotné není XML dokument, a umí pouze minimálně kontrolovat data v dokumentu. Proto se dnes při validaci XML používá spíše druhá možnost.

3.6.3 Validace proti schémovému souboru

Vytvoření schémového souboru (obvykle s příponou XSD[6]) je náročnější než vytvoření dokumentu DTD a také je vytvořený dokument „neúhledný“ – pro definici relativně mála informací je třeba hodně textu. Oproti tomu výhodou však je, že schémový

[5] DTD – zkratka pro *Document Type Definition* – definice typu dokumentu
[6] XSD – zkratka *Xml Schema Definition* – definice XML schématu

soubor je sám o sobě XML dokumentem a navíc nám umožňuje do značné míry i kontrolovat data, což je velmi důležité a užitečné.

Ve schématu můžeme definovat vlastní datové typy, přičemž většinou využíváme možnosti udělat restrikci již hotových. Prakticky tedy použijeme např. celočíselný datový typ a omezíme hodnoty na aktuálně potřebný rozsah.

Abychom mohli XML soubor validovat proti XSD schématu, je nutné mít nainstalovaný v počítači software, který validaci podporuje. Mezi takové validátory patří například xerces [6].

Příklad XML souboru (test.xml):

```
<?xml version="1.0" encoding="UTF-8"?>
<deti xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:noNamespaceSchemaLocation="test.xsd">
      <dite>
            <jmeno>Petr</jmeno>
            <prijmeni>Vomacka</prijmeni>
            <vek>13</vek>
      </dite>
</deti>
```

Příklad XSD (test.xsd) souboru, kterým lze ověřit správnost předchozího XML souboru:

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
      <xs:simpleType name="jmenoType">
            <xs:restriction base="xs:string">
            </xs:restriction>
      </xs:simpleType>

      <xs:simpleType name="prijmeniType">
            <xs:restriction base="xs:string">
            </xs:restriction>
      </xs:simpleType>

      <xs:simpleType name="vekType">
            <xs:restriction base="xs:nonNegativeInteger">
                  <xs:maxInclusive value="18"/>
            </xs:restriction>
      </xs:simpleType>

      <xs:complexType name="diteType">
            <xs:sequence>
            <xs:element name="jmeno" type="jmenoType"/>
            <xs:element name="prijmeni" type="prijmeniType"/>
            <xs:element name="vek" type="vekType"/>
            </xs:sequence>
      </xs:complexType>

      <xs:complexType name="detiType">
            <xs:sequence>
```

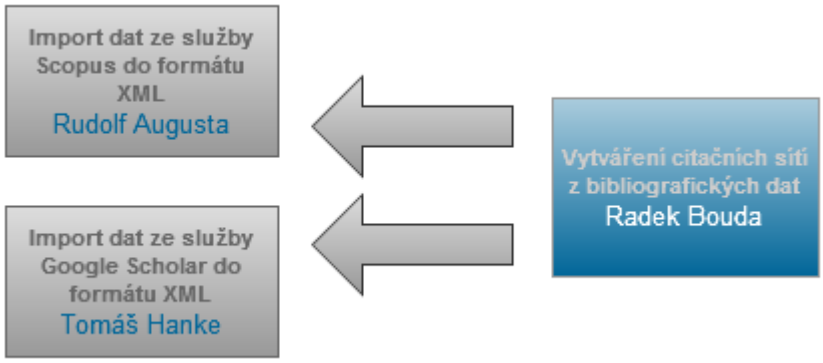
```
                <xs:element name="dite" type="diteType"
               maxOccurs="unbounded"/>
          </xs:sequence>
     </xs:complexType>

     <xs:element name="deti" type="detiType"/>
</xs:schema>
```

Pokud chceme validovat výše uvedený XML soubor proti popsanému XSD souboru pomocí validátoru xerces, můžeme to provést příkazem *xerces –v –s test.xml*. Validátor skončí s výpisem: *moje.xml: 227 ms (5 elems, 1 attrs, 0 spaces, 32 chars)*, což značí, že validace proběhla v pořádku.

4 Citační sítě

Citační síť je, z hlediska teorie grafů, orientovaným grafem. Vrcholy v grafu většinou představují jednotlivé dokumenty. Orientované hrany mezi nimi pak představují citační vazby. Publikace B, která ve svých referencích odkazuje na publikaci A, je počátečním vrcholem orientované hrany. Publikace A je vrcholem koncovým. Příklad grafické reprezentace je znázorněn v obrázku Obrázek 2 - Příklad malé citační sítě. V této práci se však nebudeme zabývat vizualizací citační sítě a naše citační síť bude reprezentována XML souborem.



Obrázek 2 - Příklad malé citační sítě

Z vizualizace je vidět, že v tomto dokumentu se navazuje na dvě práce „Import dat ze služby Scopus do formátu XML“ a „Import dat ze služby Google Scholar do formátu XML“. Z grafu je tedy mimo jiné i částečně vidět směr, kterým se daný výzkum a vývoj ubírá. V případě, že by citující dokument zůstal tak, jak je, ale citovaným dokumentům citační vztahy přibývaly, dalo by se předpokládat, že výzkum tímto směrem se ukázal být z nějakého důvodu špatný nebo nevhodný.

Spolu s vývojem jdou samozřejmě vědci, kteří za svými výzkumy a vědeckými pracemi stojí. Množství citačních vztahů dané publikace tedy ovlivňuje i postavení jednotlivých vědců, protože je evidentní, že vědec, jehož publikace má velké množství citačních vztahů, vytváří velký přínos pro vědu.

4.1 Existující citační rejstříky

Na internetu je online dostupných několik citačních rejstříků. Liší se rozsáhlostí, zaměřením a většina z nich vyžaduje placený přístup [7].

4.1.1 Web of Science

Mezi nejdůležitější (nejen) citační rejstříky dostupné online patří bezpochyby Web of Science. Web of Science je soubor vysoce kvalitních databází s informacemi o článcích, jejich autorech, obsahu, referencích, citovanosti a edičních údajích. Obsahuje bibliografické záznamy včetně abstraktů a v případě předplacení dokonce plné texty. Skládá se z pěti samostatných citačních rejstříků:

- *Science Citation Index Expanded* (*SCI EXPANDED*) – sleduje citace ve vědeckých časopisech ze 150 oborů přírodních a technických věd od roku 1900
- *Social Science Citation Index* (*SSCI*) – sleduje citace ve vědeckých časopisech z 55 oborů společenských věd od roku 1900
- *Art & Humanities Citation Index* (*A&HCI*) – sleduje citace ve vědeckých časopisech z oborů humanitních věd a uměnovědy od roku 1975
- *Conference Proceedings Citation Index – Science* (*CPCI-S*) – sleduje citace v konferenční literatuře ze všech oborů přírodních a technických věd
- *Conference Proceedings Citation Index – Social Science & Humanities* (*CPCI-SSH*) – sleduje citace v konferenční literatuře ze všech oborů společenských a humanitních věd a uměnovědy

Conference Proceedings citační rejstříky indexují více než 148000 konferencí z 256 vědních oborů od roku 1990.

4.1.2 Journal Citation Reports

Journal Citation Reports je každoročně vydávaná jedinečná databáze, která představuje soupis citačních údajů obsažených v databázi Web of Science za každý rok. Mezi základními informacemi o vědeckých časopisech lze najít například počty vydaných článků a kategorie, do kterých spadají. V detailních informacích o citacích nalezneme například počty citací ze stejného časopisu, popřípadě jiných časopisů nebo počty citací ze stejného časopisu v daném roce a mnoho dalších. Součástí jsou také metriky získané z těchto dat nazývané *Impact Factor*, *Immediacy Index*, *citing half life* a *cited half life*.

JCR je bráno jako doplněk databáze Web of Science a je hlavním zdrojem pro získávání metrických ukazatelů jako například *Impact factor*[7], *Immediacy index*[8] a další.

4.1.3 SciVerse Scopus

Neměli bychom opomenout Scopus, jako jednu z největších polytematických databází. Blíže je však již popsán v kapitole 2.3.

4.2 Využití citačních sítí

Vzhledem k tomu, jaká data nám poskytují, se dají citační sítě použít k mnoha účelům. Mezi ty nejzásadnější z nich patří:

- Inovační krok ve výzkumu
- Současné vývojové tendence
- Nejvýznamnější autority
- Vliv jednotlivých článků na současný výzkum v oboru
- Citovanost jednotlivých autorů a institucí

Inovační krok ve výzkumu a současné vývojové tendence jsou dvě velmi blízké vlastnosti. V rámci těchto informací nám citační síť ukazuje, které téma je nejvíce rozvíjeno, kam výzkum směřuje a jak je inovační krok veliký. Lze z nich například zjistit, jak rychle výzkum probíhá, odhadnout dobu dalšího pokroku nebo částečně „předvídat“ výsledky již zahájeného výzkumu. Je evidentní, že čím širší oblast vědy do citační sítě zahrneme, tím přesnější dostaneme výsledky.

Nejvýznamnější autority jsou pro nás rozhodně jedna z nejdůležitějších informací, kterou nám může citační síť poskytnout. Jedná se o pohled na citační síť, který nám pomůže určit, kdo je nejvýznamnějším odborníkem na oblast, v rámci které máme vytvořenou citační síť. V tomto kontextu významnost a citovanost přímo souvisí, lze tedy zjednodušeně říci, že čím je autor citovanější, tím je pro danou vědní oblast významnější. Nemusí však vždy jít pouze o autory. S autory se zpravidla pojí i jejich

[7] Poměr počtu citací, které byly zaznamenány v hodnoceném roce na všechny články publikované v daném časopise za předchozí dva roky, k celkovému počtu všech těchto článků [8]
[8] Index bezprostřední odezvy ukazuje, jak často jsou články vydané v daném roce citovány ještě v témže roce [8]

afiliace[9] a lze tedy kromě významnosti autorů v dané vědní oblasti, zkoumat i významnost jednotlivých afiliací. Tento údaj však může být velmi snadno zkreslený, protože někteří autoři mají více než jednu afiliaci. Tyto afiliace jsou s autory neoddělitelně spojeny, přičemž se může stát, že autorův výzkum probíhá v rámci jen podmnožiny ze všech, kterých je daný autor součástí.

Citovanost jednotlivých autorů a institucí je tou nejzákladnější informací poskytovanou citační sítí. Jelikož lze počet citací velmi snadno zjistit, lze jej také velmi jednoduše využít k zjišťování dalších informací, které nám citační síť přímo neposkytuje. Jednou z takových informací je například již okrajově zmíněný *Impact Factor*.

4.3 Impact Factor

Impact Factor (obecně označovaný IF) je z jistého úhlu pohledu brán jako míra kvality vědeckých časopisů. Často bývá používán pro určení významnosti vědeckého časopisu v rámci dané vědní oblasti. IF je každoročně počítán pro časopisy, které jsou indexovány v Journal Citation Reports (JCR), viz kapitola 4.1.2. Data v JCR jsou počítána na základě databáze Web of Science, viz kapitola 4.1.1.

4.3.1 Výpočet IF

Výpočet IF [8] provedeme s použitím následujících proměnných:

- X = počet, kolikrát byly články publikované v letech 2010 a 2011 citovány[10] v roce 2012
- Y = počet citovatelných článků publikovaných časopisem v letech 2010 a 2011

$$IF_{2012} = \frac{X}{Y}$$

Impact Factor za rok 2012, v rovnici zastoupený proměnnou IF_{2012}, může být samozřejmě zveřejněn až v roce 2013.

4.3.2 Nedostatky míry Impact Factor

[9] Ačkoliv je *Impact Factor* dobře vymyšleným způsobem jak hodnotit jednotlivé časopisy, věrohodnost této míry bohužel závisí na jednom velmi důležitém a špatně

[9] Afiliace je v tomto kontextu synonymem pro instituci
[10] Počítají se citace z časopisů, které jsou sledovány Journal Citation Reports.

ověřitelném faktu. Základem věrohodného *Impact Factor* je při psaní vědecké publikace dodržovat jistou etiku při tvorbě citací. Jsou známy případy, kdy docházelo ke změnám v citacích takovým způsobem, aby je nebylo možné jednoduše ověřit. Častějším problémem bývají stavy, kdy se neuvádějí použité zdroje nebo v horším případě, jsou uvedeny zdroje, které však ve skutečnosti při práci nikdo nepoužil.

Všechny výše zmíněné problémy jsou velkým nedostatkem. Nejen, že dochází k znepřesnění hodnoty *Impact Factor*, ale očividně také dochází ke zkreslení většiny informací, které můžeme získat z citační sítě.

5 Úprava nástrojů pro získávání dat

Jak již bylo řečeno v kapitole 2.4, veškeré nástroje pro získávání dat do formátu XML, které v této práci používáme, je třeba neustále udržovat kvůli nepředvídatelným změnám ve struktuře webových stránek jednotlivých služeb, z nichž jsou data získávána.

Při prvním pokusu o získání dat z webových služeb byl bezproblémově použitelný pouze nástroj ACM Data Harvester pro získávání dat z webů ACM Digital Library a ACM Guide. Nástroje Google Scholar Grabber pro získávání z webu Google Scholar i Scopus Data Miner pro získávání z webu Scopus vyžadovaly jistou sérii oprav, které zajistily jejich původní funkčnost.

5.1 Oprava nástroje Google Scholar Grabber

Ačkoliv nástroj Google Scholar Grabber vždy skončil svou činnost úspěšně, a nezobrazovala se žádná chybová hlášení, ve výstupním souboru se občas vyskytovala špatná a většinou však žádná data. Tento problém mohl vzniknout z několika příčin, případně jejich kombinací:

- Změna adresy, na které se daný dokument nachází (spíše nepravděpodobné)
- Odstranění dané informace z informací o dokumentu (velmi nepravděpodobné)
- Změna značky/atributu obsahující daná data (velmi pravděpodobné)

První dvě možnosti jsou velmi nepravděpodobné. Odebrání dané informace by obecně nemělo smysl, pokud již byla dříve poskytována, ačkoliv lze vymyslet případ, kdy by ke skrytí části dat mohlo dojít (např. změna licence díky které smí či nesmí být data poskytována). Díky kontrolním výpisům, na které weby Google Scholar Grabber přistupuje, lze první možnost naprosto bezpečně vyloučit z úvah.

Při analýze třetí možnosti byla třetí možnost potvrzena. Došlo k tomu, že Google do webové stránky přidal další styly, což zásadně ovlivnilo HTML elementy. Původní aplikace počítala s tím, že element obsahující informace o publikaci má atribut „class“ právě a pouze s hodnotou „gs_r“. Jelikož však Google do svého kódu přidal další atributy, HTML parser, který vrací všechny atributy a hodnoty jako jeden řetězec nepracoval správně. Očekával totiž řetězec „class=gs_r“. Tento řetězec je však v „nové

verzi“ právě díky novým atributům rozšířenější a „class=gs_r“ je v novém řetězci pouze podřetězcem, nikoliv celým řetězcem. Podle toho se odvíjely programové opravy.

Ještě před implementací do meta-vyhledávače (viz dále) se tedy měnil zdrojový kód, který se v současnosti nachází ve třídě scholar.HtmlParser, na řádku 158:

```
Původní zdrojový kód:
if(atr.toString().equalsIgnoreCase("class=gs_r")){
…
}

Opravený zdrojový kód:
if(atr.toString().toLowerCase().contains("class=gs_r")){
…
}
```

Z opravy je vidět, že se nejednalo o žádný vážnější zásah do zdrojového kódu nástroje. O to více je zde však vidět, jak moc jsou tyto nástroje náchylné na změny ve struktuře webových stránek.

5.2 Oprava nástroje Scopus Data Miner

Ani nástroj Scopus Data Miner při našem prvním testování neprováděl správnou činnost. Ačkoliv byl problém označený „*Problem with client protocol*“ odstraněný spuštěním z univerzitní sítě ZČU, stále se objevovaly problémy další.

5.2.1 Opravy základní funkčnosti

Nefunkčnost nástroje se v tomto případě projevovala tak, že činnost nástroje končila chybou „*nullPointerException*“ nebo podobnými. Podrobnějším zkoumáním bylo zjištěno, že parser zpracovávající vyhledané výsledky nedostává absolutně žádný zdrojový kód, který by mohl být zpracován. Bylo několik možností, které způsobily tento stav:

- Změnila se jména stránek, na kterých se zobrazují vyhledané výsledky (spíše nepravděpodobné)
- Ve vyhledávacím formuláři se změnily identifikátory formulářových prvků (pravděpodobné)
- HTTP server neumí při současném nastavení klienta zpracovat požadavek (pravděpodobné)
- Jiný, předem neodhadnutelný problém

Jelikož Scopus Data Miner poskytuje v kontrolních výpisech konkrétní adresy URL, ke kterým přistupuje, lze snadno ověřit, že veškeré webové adresy, ke kterým nástroj přistupuje, jsou správné. Následovalo tedy ověření správnosti identifikátorů všech formulářových polí. Všechny identifikátory formulářových polí byly i po několikátém ověření správné, ačkoliv situaci ztěžoval fakt, že identifikátorů je velmi mnoho. Při ověřování třetí možnosti z předchozího výčtu bylo postupováno tak, že jsme se pokusili stejně nastaveným HTTP klientem přistoupit k jiné URL adrese. Úspěšně.

Pravá příčina problému byla odhalena náhodným testem, kdy takto nastavený klient nebyl schopen získat ani zdrojový kód stránky, standardně zobrazující vyhledávací formulář, tj. úvodní stránku Scopusu. Po několika dalších testech se ukázalo, že Scopus není schopen pracovat bez podpory *cookies* na straně klienta.

Při doplňování podpory *cookies* daného klienta jsme použili třídu *DefaultCookieStore* z balíku *org.apache.http.impl.client*. Instance této třídy se ukázala pro tyto účely postačující a nebylo třeba její implementaci nějak měnit či rozšiřovat. Přidaný zdrojový kód se projevil v třídě *scopus.app.DocumentParser* na řádku 62:

```
private static BasicCookieStore cookies = new BasicCookieStore();
```

Tato instance byla nadefinována statická, aby bylo zajištěno používání vždy jednoho úložiště *cookies*.

Jednotné úložiště *cookies* je důležité, protože se z nějakého důvodu v kódu vyskytuje více HTTP klientů[11] (ačkoliv to není nutné). Jeden ze záměrů byl tuto komplikovanost odstranit použitím jednoho HTTP klienta, ale samotné přepsání celé této třídy by nakonec bylo tak komplikované, že od záměru bylo nakonec upuštěno a všem HTTP klientům nastavené úložiště *cookies*. Toto doplnění zdrojového kódu můžeme nalézt ve stejné třídě např. na řádcích 134 a 135:

```
DefaultHttpClient httpClient = new DefaultHttpClient();
httpClient.setCookieStore(cookies);
```

Po dokončení této úpravy se podařilo získat zdrojový kód webové stránky, vždy ale až té, která se měla načíst jako druhá. To je způsobeno tím, že při prvním přístupu se

[11] Instancí tříd *DefaultHttpClient*

načtou *cookies* do úložiště a až druhý přístup zobrazí požadovaný kód. Tento nedostatek byl vyřešen jednoduše tím, že ihned na začátku činnosti nástroj provede jeden „slepý“ přístup na web Scopusu, jehož jediným účelem je načtení *cookies* do úložiště.

Zdrojový kód, který HTTP server vrátil, však použitý parser stále nebyl schopný zpracovat. Problém byl zjištěn v tom, že data se vrátila s kompresí, kterou parser neuměl zpracovat. Byl tedy zakomentován[12] řádek 96 ve třídě *scopus.app.DocumentParser*, díky čemuž se zdrojový kód vrátí bez komprese:

```
//httpMessage.addHeader("Accept-Encoding", "gzip, deflate");
```

Po výše zmíněných úpravách byla základní funkčnost Scopus Data Mineru opět obnovena.

5.2.2 Chyba s protokolem HTTPS

Při prvních pokusech o vytvoření citační sítě bylo zjištěno, že nástroj nedokáže stáhnout bibliografické informace o větším množství publikací. Řádově se jednalo o stovky publikací, což je pro získávání dalších informací z citační sítě nedostačující. Po bližším zkoumání bylo zjištěno, že HTTP klient neumí přistupovat k dokumentům, na které je odkazováno pomocí protokolu HTTPS a stahování publikací skončí výjimkou *„nullPointerException“*.

HTTPS je zabezpečená varianta protokolu HTTP. Data přenášená pomocí protokolu HTTPS jsou šifrována použitím SSL. Pro činnost naší aplikace je důležitý pouze fakt, že přistoupíme-li k datům na Scopusu, která jsou odkazována pomocí protokolu HTTPS, pomocí protokolu HTTP, tak dostaneme stejný výsledek, který však náš parser bez problému zpracuje. Přepis adresy zajišťuje zdrojový kód na řádcích 127 až 129 ve třídě *scopus.app.DocumentParser*:

```
if(url.contains("https://")){
      url = url.replace("https://", "http://");
}
```

Po přidání této opravy se podařilo vytvořit citační sítě o velikostech 1000 až 5000 uzlů.

[12] Celý řádek byl označen jako komentář, což způsobí, že daný příkaz není procesorem vůbec proveden.

5.2.3 Změna řazení výsledků

Při dalších pokusech o vytvoření citační sítě bylo zjištěno, že se nedaří vytvořit prakticky žádné zanoření. I větší množství dokumentů v nulté generaci nezvyšovalo celkový počet výsledků. Zde bylo mylným předpokladem, že nástroj funguje nesprávně. Scopus výsledky vyhledávání řadí standardně podle relevantnosti, což se v reálné situaci, kdy chceme co nejvíce citované publikace, ukázalo jako nevhodné.

Jednoznačným řešením je tedy seřadit výsledky podle počtu citací. Ačkoliv je to v prostředí webového prohlížeče prostřednictvím formulářového prvku možné, programově se to nezdařilo. Programová úprava je přidána v třídě *scopus.app.DocumentParser* na řádcích 123 až 126:

```
if(url.contains("sort=plf-f")){
      url = url.replace("sort=plf-f", "sort=cp-f");
      url = url.concat("&ss=cp-f");
}
```

V ukázce zdrojového kódu řazení *„plf-f“* značí řazení podle relevantnosti a *„cp-f“* řazení podle počtu citací. Po této programové úpravě se podaří dosáhnout většího zanoření, co se týče počtu generací výsledků. Celkový počet vyhledaných výsledků se však příliš nezvýšil (řádově o desítky až stovky publikací).

5.2.4 Cyklické přístupy ke Scopusu

V průběhu pokusů o vytvoření co největší citační sítě se objevil problém s výpadkem konektivity. Při současném programování si nástroj nedokázal poradit ani se sebemenším výpadkem. I méně než vteřinový výpadek ukončil okamžitě činnost nástroje, který skončil výjimkou.

Tuto výjimku způsobovala metoda *execute()* kterou má programově každý HTTP klient. Metoda výjimku přímo vyvolávala v případě, že se jí nepodařilo dokončit činnost. V případě úspěšného dokončení činnosti vrací HTML stránku. Tato výkonná část v metodě *getResponse(String url)* byla tedy pozměněna způsobem, že nyní výjimky neukončují činnost programu, ale je na ně pouze upozorněno. Získávání HTML stránky ze serveru probíhá v nekonečném cyklu, který se ukončí až v momentě úspěšného načtení stránky. Nekonečný cyklus je kvůli příliš častým přístupům k webu

pozastavován na dobu *searchDelay*, která je shodná s dobou mezi jakýmikoli jinými přístupy ke Scopusu. Upravený zdrojový kód je na řádkách 139 až 151:

```
do{
	try {
		Thread.sleep(searchDelay);
		response = httpClient.execute(httpGet);
		break;
	} catch (ClientProtocolException e) {
		System.out.println("chyba s klientskym protokolem - zkousim
				znovu!");
	} catch (IOException e) {
		System.out.println("chyba I/O - zkousim znovu!");
	} catch (InterruptedException e) {
		System.out.println("chyba v preruseni - zkousim znovu!");
	}
}while(true);
```

Tato programová úprava se projevila jako velmi důležitá. Po jejím provedení se podařilo vytvořit citační síť o více než patnácti tisících uzlech.

5.3 Optimalizace zdrojových kódů

Jak již bylo napsáno dříve, se souhlasem vedoucího práce se v této práci nebudeme zabývat službami ACM Digital Library/GUIDE a Google Scholar, protože se jeví oproti službě Scopus jako méně perspektivní.

Při analýze zdrojového kódu nástroje Scopus Data Miner nebyl nalezen úsek kódu, který by výrazněji zatěžoval operační systém svými nároky na operační paměť nebo čas procesoru. Nebyla zde tedy provedena žádná optimalizace, protože se ukázalo, že to není třeba.

6 Vytvoření meta-vyhledávače

Jedním z vedlejších cílů práce bylo vytvoření meta-vyhledávače, který by se dal používat k vytváření citačních sítí. V současnosti každý z nástrojů, zmíněný v kapitole 2, umožňuje automaticky vytvářet citační síť z dat, která získá. Není to však vždy úplně jednoduché, přímočaré, intuitivní a už vůbec ne univerzální. Nejsou zde však jen programové nedostatky (rozdílnosti) ale hlavně také uživatelské, jako například nejednotnost uživatelského rozhraní nebo dokonce nejednotnost jazyka uživatelského rozhraní. Nutno podotknout, že veškeré zmíněné nedostatky platí pro všechny tři nástroje jako celek.

Výše zmíněné nedostatky jsou postačujícím důvodem k potřebě vytvořit meta-vyhledávač, který je bude co nejlépe řešit. Zjištěné nedostatky byly stanoveny jako některé z jednotlivých cílů při vytváření meta-vyhledávače:

- Jedno uživatelské rozhraní pro ovládání 3 různých nástrojů
- Jeden jazyk uživatelského rozhraní
- Jednoduchost a intuitivnost uživatelského rozhraní
- Monitorování stavu jednotlivých nástrojů z jedné aplikace
- Parametrické vyhledávání
- Možnost používat pouze vybrané nástroje

6.1 Metody implementace

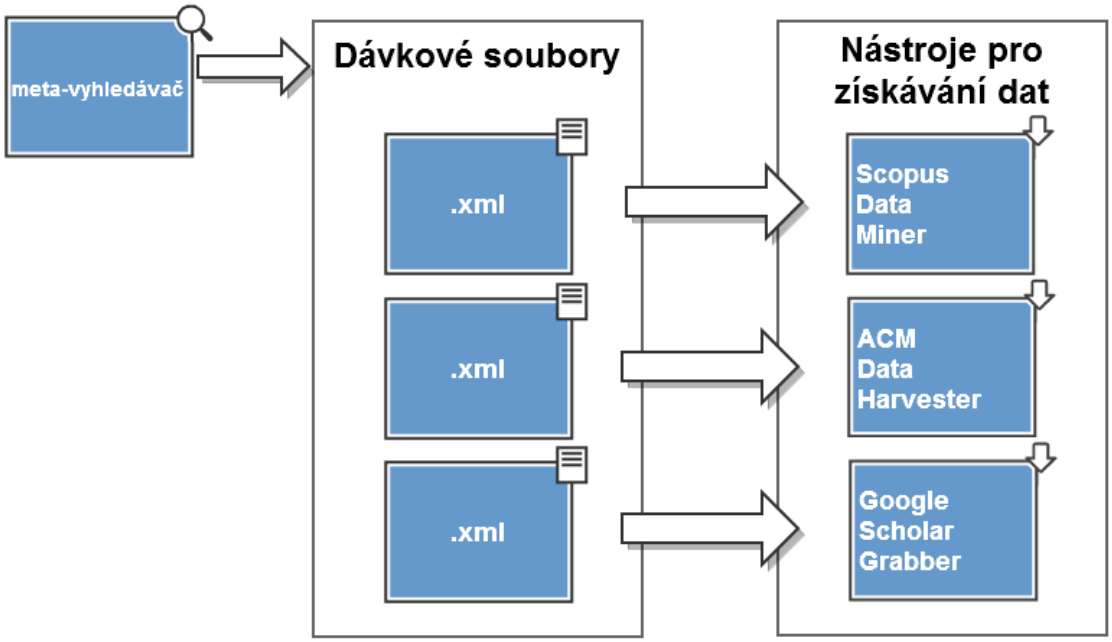
Při vytváření meta-vyhledávače bylo možné použít několik metod, jak postupovat. Pro naplnění stanovených cílů bylo možné postupovat dvěma způsoby:

- Generátor dávkového souboru
- Přímá implementace zdrojových kódů

6.1.1 Generátor dávkového souboru

Princip tohoto návrhu spočívá v tom, že se vytvoří aplikace, kde se v jednotném uživatelském rozhraní vyberou parametry vyhledávání pro všechny nástroje, které chceme použít. Z těch se následně vygenerují soubory (např.: textové XML soubory), které budou obsahovat data pro řízení činnosti jednotlivých nástrojů. Každý z nástrojů by pak bylo nutné upravit tak, aby bylo možné řídit jejich činnost právě tímto

souborem. V praxi to může vypadat jako načtení hodnot ze souboru do jednotlivých polí grafického uživatelského prostředí nebo spouštění nástroje z příkazové řádky se speciálním parametrem. Princip funkce je naznačen na následujícím obrázku.



Obrázek 3 - Schéma generátoru dávkových souborů

Tento způsob by byl vcelku efektivní. Vyřešily by se veškeré nedostatky grafického uživatelského prostředí (na úrovni meta-aplikace by bylo s požadovanými vlastnostmi a na úrovni nástrojů by se o něj nebylo třeba starat – bylo by vyplněné z dávkového souboru). Jednotlivé nástroje by mohly (na rozdíl od druhého způsobu, viz dále) běžet nezávisle na sobě, což by bylo rozhodně přínosem. Touto metodou však nelze jednoduše dosáhnout jednotného spouštění a monitorování jednotlivých nástrojů a proto jsme jej nepoužili.

6.1.2 Přímá implementace zdrojových kódů

Tato varianta je možná jen díky tomu, že máme k dispozici zdrojové kódy jednotlivých nástrojů. Metoda je založena na principu implementace zdrojových kódů nástrojů přímo jako součásti meta-aplikace. Jelikož jsou všechny nástroje implementovány v programovacím jazyce Java, tak i meta-aplikace bude implementována v tomto jazyce.

Tímto způsobem můžeme velmi jednoduše dosáhnout stanovených cílů. Nově navržené uživatelské rozhraní bude splňovat naše požadavky, a data z něj se převedou do stávajících uživatelských rozhraní nástrojů programově, což přinese uživatelskou jednoduchost. Navíc je díky přímému „propojení“ možné jednoduše zasílat zprávy o aktuálním stavu jednotlivých nástrojů meta-aplikaci, díky čemuž dosáhneme požadavku možnosti monitorovat všechny nástroje z jednoho místa. Jediná nevyhnutelná nevýhoda je, že jednotlivé nástroje pro stahování dat přestanou být nezávislé. Vzhledem k tomu, že je každý z nástrojů závislý pouze na hlavním okně meta-aplikace, ale ne na jiném nástroji, nezpůsobí tato závislost nástrojů žádné velké škody. Při chybě ve vlákně meta-aplikace by pochopitelně nedokončil svou činnost žádný z právě pracujících nástrojů. Pravděpodobnost chyby ve vlákně meta-aplikace je však velmi nízká. Pravděpodobnost chyby je vzhledem k více závislostem a náročnější činnosti vyšší ve vláknech jednotlivých nástrojů. Tyto chyby však neovlivní činnost ostatních vláken, pokud nedojde k závažnější chybě, která by například zastavila činnost celého virtuálního stroje Javy. Tento způsob jsme vybrali pro implementaci z důvodu evidentně nižšího počtu nedostatků.

6.2 Postup implementace uživatelského rozhraní

6.2.1 Návrh uživatelského rozhraní

Návrh kvalitního uživatelského rozhraní je v implementaci klíčový. Grafické uživatelské rozhraní meta-aplikace by mělo umožňovat následující:

- Základní nastavení jednotlivých nástrojů
- Pokročilá nastavení jednotlivých nástrojů
- Nastavení samotného meta-vyhledávače
- Jednoduché monitorování jednotlivých nástrojů

Jelikož bylo cílem, aby uživatelské rozhraní bylo jednoduché, postupovali jsme také podle toho. Nejjednodušší volbou byla možnost, poskytnout uživateli všechny informace najednou, v případě že jich nebude mnoho, a nebude třeba je třídit více, než je uvedeno v bodech výše. Protože se předpokládá, že meta-aplikace bude obsahovat pouze nastavení a monitorování, není důvod, aby byly jednotlivé položky např. skryty v menu. V praktickém používání by to totiž znamenalo jen zbytečné zdržování.

Hlavní okno meta-aplikace bude tedy obsahovat veškeré informace. Jednotlivé informace budou umístěny do rovnoměrně rozmístěných bloků, tvořících dohromady čtvercové pole 4x4.

V prvním bloku se budou nacházet základní nastavení jednotlivých nástrojů. Patří mezi ně například vyhledávaný řetězec, počet generací a počet záznamů v nulté generaci. V druhém bloku se budou nacházet možnosti bližší specifikace vyhledávaných výsledků (bližší popis viz níže). Třetí blok bude obsahovat nastavení přímo meta-aplikace. V praxi to znamená výběr nástrojů, které se mají pro vyhledávání použít. Poslední blok bude umožňovat spuštění vyhledávání a jednoduché monitorování průběhu činnosti jednotlivých nástrojů. Je zde ještě jeden blok se speciálním významem, ten bude popsán dále.

6.2.2 Základní nastavení nástrojů

V první, levé horní části okna, nazvané „Vyhledávání“ bude možnost nastavit ty jediné nutné parametry, které jsou třeba ke spuštění jednotlivých nástrojů. Patří mezi ně vyhledávaný text, počet záznamů v nulté generaci a počet generací. Užitečnou vlastností je výběr cílové složky, do které se budou ukládat stažené výsledky. Z důvodu přímé provázanosti s hledaným textem je zde i speciální pole, kterým lze specifikovat, kde se daný text bude hledat. Mezi možnosti bude patřit volba všude, pouze v titulku nebo pouze v abstraktu.

6.2.3 Specifikace vyhledávaných výsledků

V druhém, pravém horním bloku bude možnost specifikovat parametry vyhledávání. Každý z nástrojů pro získávání dat umožňuje nějaké bližší specifikace vyhledaných výsledků. To nám umožňuje získat přesnější výsledky bližší našemu zájmu a vyloučit nežádoucí.

Problém nastává při určování parametrů, které bude možné z hlavního okna meta-aplikace zadávat. Množina parametrů, které lze specifikovat, je v každém nástroji pro získávání dat jiná. Při vybírání parametrů jsou dvě možnosti:

- Bude možno nastavovat parametry, vybrané sjednocením množin všech parametrů jednotlivých nástrojů

- Bude možno nastavovat parametry, vybrané průnikem množin všech parametrů jednotlivých nástrojů

Obě tyto možnosti mají své přednosti a nedostatky. Pokud vezmeme v úvahu první variantu, bude v hlavním okně meta-aplikace možné nastavit velmi mnoho parametrů. Může to být výhodné z hlediska „nastavování z jednoho místa“ apod. Nepříjemná situace však vznikne hned poté, kdy bude nutno uživatele nějakým způsobem upozornit, že parametr, který nastavil, pro určité nástroje použít lze ale pro určité naopak ne.

Druhá zmíněná varianta je schůdnější. Výsledek dopadne pravděpodobně tak, že v hlavním okně meta-aplikace nebude mnoho nastavení, ale uživatel bude přesně vědět, že zvolené parametry, které vybral, budou platit pro všechny použité nástroje. V případě, že se uživatel rozhodne specifikovat vyhledávání více, bude to muset udělat ručně přímo v nastavení nástroje, který to umožňuje. Ruční konfigurace jednotlivých nástrojů je umožněna (viz níže).

Výběr jednotlivých parametrů vyhledávání na základě textu z předchozího odstavce je vidět v tabulce 2. Průnik množin je označen zeleně.

ACM Data Harvester	Scopus Data Miner	Google Scholar Grabber
Jméno autora, editora	Jméno autora	Jméno autora
Klíčová slova	Klíčová slova	Rok publikace (rozsah)
Rok publikace (rozsah)	Typ dokumentu	
Vydavatel	Výběr věd	
Rok a místo konference	Rok publikace (rozsah)	
ISBN/ISSN		
Afiliace		

Tabulka 2 - Parametry vyhledávání jednotlivých nástrojů

Jak je vidět z Tabulky 2, Google Scholar Grabber svým způsobem „omezuje“ ostatní nástroje. Princip tohoto výběru parametrů do možností meta-vyhledávače však spočívá v myšlence nabídnout ty „nejvlivnější“ parametry. Mezi „nejvlivnější“ parametry nepatří například volba vydavatele, protože jí umožňuje pouze ACM Data Harvester. Bylo by však krátkozraké tento parametr odstranit-znehodnotit. To, že nebude tento

parametr v meta-vyhledávači zobrazen, nebude znamenat, že jej nelze „ručně“ nastavit.

Tohoto „ručního“ nastavení bude docíleno dvojicí komponent. První komponenta bude zaškrtávací políčko, jehož zatržení bude rozhodovat o tom, jestli se mají vybrané nástroje spustit ihned po spuštění z okna meta-vyhledávače. Pokud bude zatržené (ve výchozím nastavení ano), po spuštění vyhledávání v meta-vyhledávači se zadaná data z meta-vyhledávače předají jednotlivým nástrojům a zahájí se jejich činnost. V opačném případě se zadaná data nástrojům předají taktéž, ale jednotlivé nástroje se nespustí. V obou případech se však aktivují tlačítka, umožňující přístup k nástrojům, které byly zvoleny k vyhledávání. V případě, že nástroje nebyly spuštěny, lze díky těmto tlačítkům dokončit nastavení (se všemi možnostmi pro daný nástroj) nástrojů a odděleně je spustit. V případě, že jsou již nástroje spuštěny, lze přístup k nim využít k detailnímu sledování jejich činnosti, protože meta-vyhledávač poskytuje pouze základní informace o činnosti jednotlivých nástrojů.

6.2.4 Ostatní bloky rozhraní meta-vyhledávače

Ve třetím, levém spodním bloku, se bude nacházet nastavení meta-vyhledávače samotného. Mezi jediné volby meta-vyhledávače patří výběr vyhledávacích nástrojů, které se mají použít a možnost automatického zahájení vyhledávání ihned po jejich spuštění.

Ve čtvrtém, pravém spodním bloku, se bude vyskytovat část umožňující sledování činnosti jednotlivých nástrojů. Není cílem popisovat činnost přehnaně detailně. Z popisu však bude možno poznat, zdali je nástroj spuštěn, provádí činnost, nebo jak jeho činnost skončila (úspěch nebo chyba). V tomto bloku se bude také nacházet tlačítko ke spuštění zvolených nástrojů.

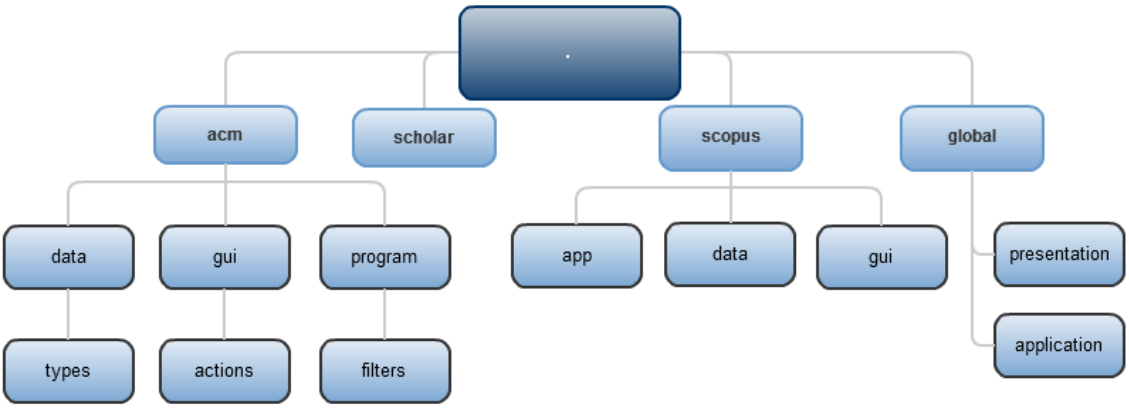
Meta-vyhledávač byl pojmenován Bibliographic Data Miner (BDM) a náhled jeho realizovaného uživatelského prostředí lze vidět v příloze D.

6.3 Programová implementace

Jelikož meta-vyhledávač jako takový vykonává minimum operací, bude se tato podkapitola zaměřovat spíše na programové propojení nástrojů pro získávání dat s meta-vyhledávačem.

6.3.1 Organizace balíků

Protože jednotlivých tříd je velmi mnoho a v rámci jednoho balíku by bylo těžké se v nich orientovat, využijeme toho, že zdrojové kódy původních nástrojů již jsou nějakým způsobem v balících organizovány. Je žádoucí, aby struktura balíků jednotlivých nástrojů byla co nejméně měněna (z důvodu minima úprav pro správnou funkci) a proto zůstanou z tohoto hlediska původní třídy prakticky netknuté. Jedinou změnou je vytvoření nového balíku pro každý nástroj, do kterého se pak přesunou původní balíky daného nástroje. Nakonec se přidá balík *global* obsahující třídy pro práci meta-vyhledávače. Struktura balíků je znázorněna pomocí stromu v obrázku č. 4.



Obrázek 4 - Struktura balíků

6.3.2 Třída GOkno

Třída *GOkno*, v balíku *global.presentation,* je hlavní třídou, vytvářející uživatelské rozhraní meta-vyhledávače. Instance této třídy vlastní kromě obvyklých atributů navíc několik metod a atributů, které jsou klíčové pro správnou komunikaci mezi meta-vyhledávačem a nástroji.

Nejdůležitější pro tuto komunikaci jsou metody zajišťující předání meta-vyhledávači informaci o činnosti nástroje. Tyto metody mají následující signatury:

```
public void setACMStatus(String status, String barva);
public void setScholarStatus(String status, String barva);
public void setScopusStatus(String status, String barva);
```

Každá z metod vyžaduje dva řetězcové parametry. První parametr je status, tedy zpráva, která se má meta-vyhledávači předat. Druhým parametrem je barva, kterou se má daná zpráva zobrazit. Standardní hodnota je „black“ pro černý text, je však záměrem různým typům zpráv přiřadit různou barvu. Současně je zvoleno: zelenou pro spuštění a úspěšné dokončení, modrou pro činnost, červenou pro chyby nebo oranžovou pro setrvávání nástroje v čekací smyčce.

6.3.3 Úpravy jednotlivých nástrojů

Aby mohly jednotlivé nástroje komunikovat s hlavním oknem meta-vyhledávače, bylo nutné vždy přidat třídě, která danému nástroji vytváří uživatelské rozhraní, atribut typu *GOkno*. Tento atribut je zpočátku nastaven na hodnotu *null*. Nastavuje se až v novém konstruktoru, který oproti původnímu obsahuje navíc formální parametr, kde se předává odkaz na instanci hlavního okna meta-vyhledávače.

V těle tohoto nového konstruktoru se kromě kódu z původního konstruktoru vykonává navíc ještě další kód, který zajišťuje (pomocí reference na instanci meta-vyhledávače) přepsání hodnot zvolených v meta-vyhledávači do polí uživatelského rozhraní daného nástroje. Je programově zajištěno, aby přepsání hodnot proběhlo do správných polí.

Po přepsání všech dat z meta-vyhledávače do polí nástroje, je třeba ověřit, jestli se má nástroj také okamžitě spustit. To se ověřuje metodou *boolean stahovatBezVyzvy(),* kterou vlastní instance třídy *GOkno*. V případě, že metoda vrátí hodnotu *true*, započne se činnost nástroje voláním metody *void doClick()*, kterou vlastní všechny instance třídy *JButton,* tedy tlačítka pro spuštění činnosti nástroje. Při použití metody *doClick()* proběhne vše stejně, jako kdyby uživatel klikl na tlačítko myší.

Protože každý z nástrojů je jiný, tak i kód každého konstruktoru se pochopitelně liší. Jde však většinou o různá jména identifikátorů, principiální význam rozšířeného konstruktoru je ale vždy stejný.

Pro sledování činnosti jednotlivých nástrojů bylo ještě třeba projít hlavní metody jednotlivých nástrojů a doplnit systémová hlášení. Nejdůležitější výpisy, tj.: číslo

aktuálně stahovaného dokumentu/generace, chyby, ukončení, se kromě výstupu na konzoli musí ještě vypisovat do meta-vyhledávače. K tomu slouží vhodně použité metody *setXYZStatus(String status, String barva)*, popsané výše.

7 Sjednocení formátu XML

Každý ze zmiňovaných nástrojů umožňuje export dat do formátu XML. Struktura XML souborů se však liší, navíc z každé služby lze získat jiná data (například v jiné míře podrobnosti). Pro zjednodušení další práce – zpracování, vizualizace apod., je třeba strukturu těchto dat sjednotit, to znamená navrhnout novou, univerzální XML strukturu.

Vzhledem k požadavku na univerzálnost je rozhodnuto, že nová XML struktura bude muset zvládnout pojmout všechna data, která dokážou poskytnout všechny nástroje dohromady. To však bude ve výsledku znamenat, že některé značky XML struktury zůstanou nevyužité. Ačkoliv to za následek bude mít větší objem souboru, může se stát, že v budoucnu – po případném rozšíření nástrojů – bude možné tyto nevyužité značky využít.

7.1 Data potřebná k uchování

Z dat stahovaných jednotlivými nástroji vyplynulo, že je třeba, aby struktura nového XML souboru dokázala pojmout o každém dokumentu následující informace:

- Identifikátor, číslo generace a pořadové číslo
- Název dokumentu
- Jméno autora (celé jméno, křestní, 2. křestní, příjmení)
- Všechny afiliace autora a jejich počet
- Identifikátory publikací, které danou publikaci citují a jejich počet
- Typ zdroje (např.: kniha, článek)
- Název časopisu, ve kterém článek vyšel (pro časopisy)
- Název konference (pro sborníky)
- Rok publikování
- Ročník a číslo (pro časopisy)
- Číslo stránky / rozsah stránek, kde se článek nachází
- Země, město a měsíc konání (pro konference)
- ISBN a ISSN
- Poznámky, které by mohly být užitečné, ale nedají se zařadit jinam

Kořenový element XML souboru bude obsahovat jednotlivé publikace. Pro jednoduchou orientaci se budeme snažit o vnořování jen tam, kde je to opravdu nezbytně nutné. Taktéž se budeme snažit minimálně používat atributy, protože používání značek umožňuje do budoucna snazší rozšíření v případě potřeby.

Konkrétní struktura univerzálního XML souboru je ukázána v příloze F.

7.2 Programové úpravy pro sjednocení XML

Výhodou procesu úprav bylo, že všechny tři nástroje pro získávání dat používají pro zápis do souboru technologii StAX. Tato technologie se využívá velmi jednoduše, a proto pro změnu výsledného souboru stačilo zaměnit pořadí příkazů, maximálně změnit, aby se daná data zapisovala do značky, místo hodnoty atributu.

Změny byly provedeny v následujících metodách:

- V metodě *save(PublicationType pub)*, ve třídě acm.data.Output, řádek 62 a v metodách vnořených
- V metodě vytvorXML(), ve třídě scholar.StAX, řádek 88
- V metodě printDocumentToXmlFile(Document document), ve třídě scopus.data.XmlFileStax, řádek 79 a ve vnořených metodách

8 Vytvoření citační sítě

Každý z nástrojů vytváří citační síť automaticky, to znamená, že bez jakéhokoliv přičinění či speciálního nastavení uchovává ve výstupním XML souboru informaci o citačních vazbách. Jelikož si dokumenty procházené na webu lze z počátku představit jako strom (s duplicitami), můžeme i dokumenty stahovat dvěma metodami, známými z teorie grafů:

- Procházení do šířky
- Procházení do hloubky

Jelikož nikdy přesně nevíme, kolik dokumentů získáme nebo kolik je jich dostupných[13], nelze způsob procházení přímo změnit. Spíše se můžeme snažit ho dosáhnout – přiblížit se k němu. Například při procházení do šířky by bylo nutné stáhnout celou nultou generaci výsledků. Pomineme-li fakt, že toto nastavení žádný z nástrojů neumožňuje, tak ani nemusí být všechny dokumenty vyhovující vyhledávání dostupné.

Jediná možnost, jak toto procházení ovlivnit je tedy pomocí parametrů „počet dokumentů v generaci“ a „počet generací“. Se zvyšujícím se počtem dokumentů v generaci a snižujícím se počtem generací se blížíme k procházení do šířky. V opačném případě pak k procházení do hloubky.

8.1 Vyhledávané fráze

Vzhledem k našemu cíli, vytvořit co největší citační síť publikací ze služby Scopus, bylo nutné zvolit vhodné fráze, které se budou vyhledávat (a také kde se budou vyhledávat). Vybrali jsme, že se bude vyhledávat v titulku, abstraktu a klíčových slovech (výchozí nastavení Scopusu). Jako vhodné fráze se ukázaly jednoduché výrazy. Používali jsme výrazy „*science*“, pro který online Scopus našel více než 2 miliony dokumentů a „*internet*“ s necelými třemi sty tisíci dokumenty.

[13] Například Scopus omezuje zobrazení pouze 2000 výsledků. Toto číslo se však může také bez upozornění změnit.

8.2 Podmínky pro vyhledávání

Jak již bylo napsáno, citační síť se pro publikace vytváří automaticky. To však situaci zjednodušuje jen málo. Pro úspěšné vyhledávání je třeba zajistit ještě několik následujících podmínek:

- Nepřetržitý provoz stroje, na kterém je spuštěno vyhledávání
- Připojení ze školní sítě ZČU nebo sítě, která má zaplacený přístup ke Scopusu
- Nepřetržité spojení k webu Scopusu

První bod, vznikající z důvodu nemožnosti navázat na předchozí vyhledávání, lze relativně snadno splnit. Větší potíže způsobuje druhý bod. Standardně se využívá připojení pomocí VPN. VPN tunel však smí být otevřený pouze 12 hodin, poté se musí obnovit a tím dojde ke kolizi s bodem třetím. My jsme pro svou práci využili virtuální server, který pro naše potřeby zařídila Katedra informatiky a výpočetní techniky.

Třetí bod je samostatný problém, který jsme se snažili řešit v kapitole 5.2.4. Ačkoliv se toto řešení v praxi ukázalo jako velmi užitečné, rozhodně není klíčovým řešením tohoto problému. Navíc, občasně se objevující a průběhu vyhledávání nedeterministicky vznikající, chyby v balících *org.apache.http.** jsou bohužel špatně dokumentované a proto se špatně opravují.

Z výše zmíněných důvodů skončila všechna stahování dat chybou. Míra úspěšnosti zde přímo úměrně závisí na naší schopnosti zajistit splnění výše zmíněných bodů.

9 Závěr

Cílem této práce bylo upravit a opravit existující nástroje pro získávání dat z webových služeb, implementovat je do meta-vyhledávače a vytvořit pomocí nich co největší citační síť.

Ačkoliv jsme pro vytvoření citační sítě nevyužili veškeré zmíněné nástroje, všechny nástroje byly před implementací do meta-vyhledávače plně funkční. Po jednotlivých opravách byla úspěšně provedena implementace do meta-vyhledávače, čímž byl splněn i nepsaný cíl „zjednodušit vytváření citačních sítí“.

Bohužel v průběhu činnosti došlo opět ke změnám ve zdrojových kódech webů a tak aktuálně nejsou všechny nástroje zcela funkční. Všechny nástroje jsou však správně implementovány do meta-vyhledávače. Aktuální stav samotných nástrojů je zhruba následující:

- ACM Data Harvester – nefunkční
- Scopus Data Miner – funkční, kromě afiliací
- Google Scholar Grabber – funkční, chybné časové konstanty

Ke změně webu Scopus došlo bohužel v průběhu vytváření citační sítě, takže Scopus Data Miner získává afiliace autorů správně pouze v případě, má-li autor jen jednu afiliaci. Google Scholar Grabber je funkční, nicméně jsme zpozorovali, že Google je nyní ještě „citlivější“ na robotické přístupy. Bylo by proto vhodné tyto konstanty oproti konstantám zmíněným v dokumentu [1] ještě zvýšit.

Při pokusech o vytvoření co největší citační sítě byl následující požadavek:

- Vyhledávaná fráze: „internet“
- Počet dokumentů v nulté generaci: 1
- Počet generací: 3
- Využívané nástroje: Scopus Data Miner

V tomto nejúspěšnějším pokusu se podařilo vytvořit citační síť, jejíž poslední dokument je v druhé generaci a má pořadové číslo 16 329. Je to dostatečně rozsáhlá síť pro další činnost, ačkoliv je naneštěstí ochuzena o některé afiliace, protože v průběhu jejího

vytváření došlo ke změně struktury webové stránky. Její vytvoření trvalo necelých 5 dní. Je uložena na přiloženém disku v adresáři „out“, v souboru s názvem „vystup_scopus_16329.xml“. Ve stejném adresáři jsou uloženy i další (menší) citační sítě.

Jako další rozšíření této práce vidíme opravení jednotlivých nástrojů podle zmíněných problémů a vizualizaci vytvořených citačních sítí. Také by bylo vhodné zvážit výměnu knihoven, pomocí kterých se připojují nástroje k webovým službám, za jiné a stabilnější.

Reference

1. **Hanke, Tomáš.** *Import dat ze služby Google Scholar do XML.* Plzeň : Západočeská univerzita v Plzni, 2012. Bakalářská práce.

2. **Krupička, Jan.** *Import dat ze služby ACM DL do formátu XML.* Plzeň : Západočeská univerzita v Plzni, 2012.

3. **Augusta, Rudolf.** *Import dat ze služby Scopus do formátu XML.* Plzeň : Západočeská univerzita v Plzni, 2012.

4. **Refsnes Data.** XML Tutorial. *W3Schools Online Web Tutorials.* [Online] 2013. [Citace: 10. Březen 2013.] http://www.w3schools.com/xml/.

5. **Herout, Pavel.** *Přednášky z JXT.* Plzeň : Západočeská univerzita v Plzni, 2012.

6. **The Apache Software Foundation.** The Apache Xerces Project - xerces.apache.org. *The Apache Xerces Project.* [Online] 2013. [Citace: 13. Březen 2013.] http://xerces.apache.org/.

7. **Vysoká škola ekonomická.** Citační rejstříky. *Econlib.* [Online] 2013. [Citace: 15. 04 2013.] http://www.econlib.cz/veda/citacni-rejstriky.html.

8. —. Citační index a impakt faktor | VŠE. [Online] Vysoká škola ekonomická, 20. Srpen 2008. [Citace: 15. Duben 2013.] http://www.vse.cz/obecne/impactfk.php3.

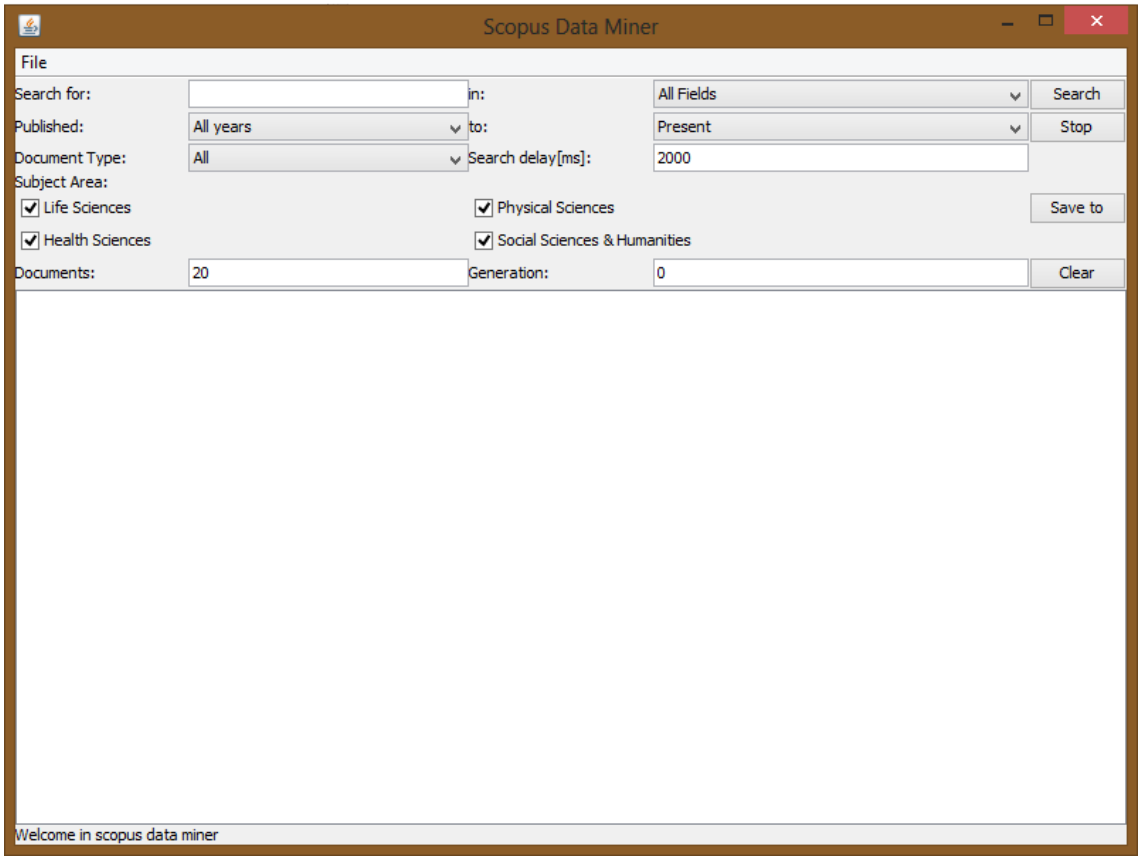
9. **Boldiš, Petr.** Věda.cz: Web of Science a JCR. *Věda.cz.* [Online] Věda.cz, 10. Listopad 2003. [Citace: 18. 03 2013.] http://www.veda.cz/article.do?articleId=8886.

10. **University, Carnegie Mellon.** The Official CAPTCHA Site. *The Official CAPTCHA Site.* [Online] Carnegie Mellon University, 2010. [Citace: 21. Únor 2013.] http://www.captcha.net/.

11. **Fielding, R., a další.** HTTP/1.1: Status Code Definitions. *World Wide Web Consortium (W3C).* [Online] Červen 1999. [Citace: 24. Únor 2013.] http://www.w3.org/Protocols/rfc2616/rfc2616-sec10.html.

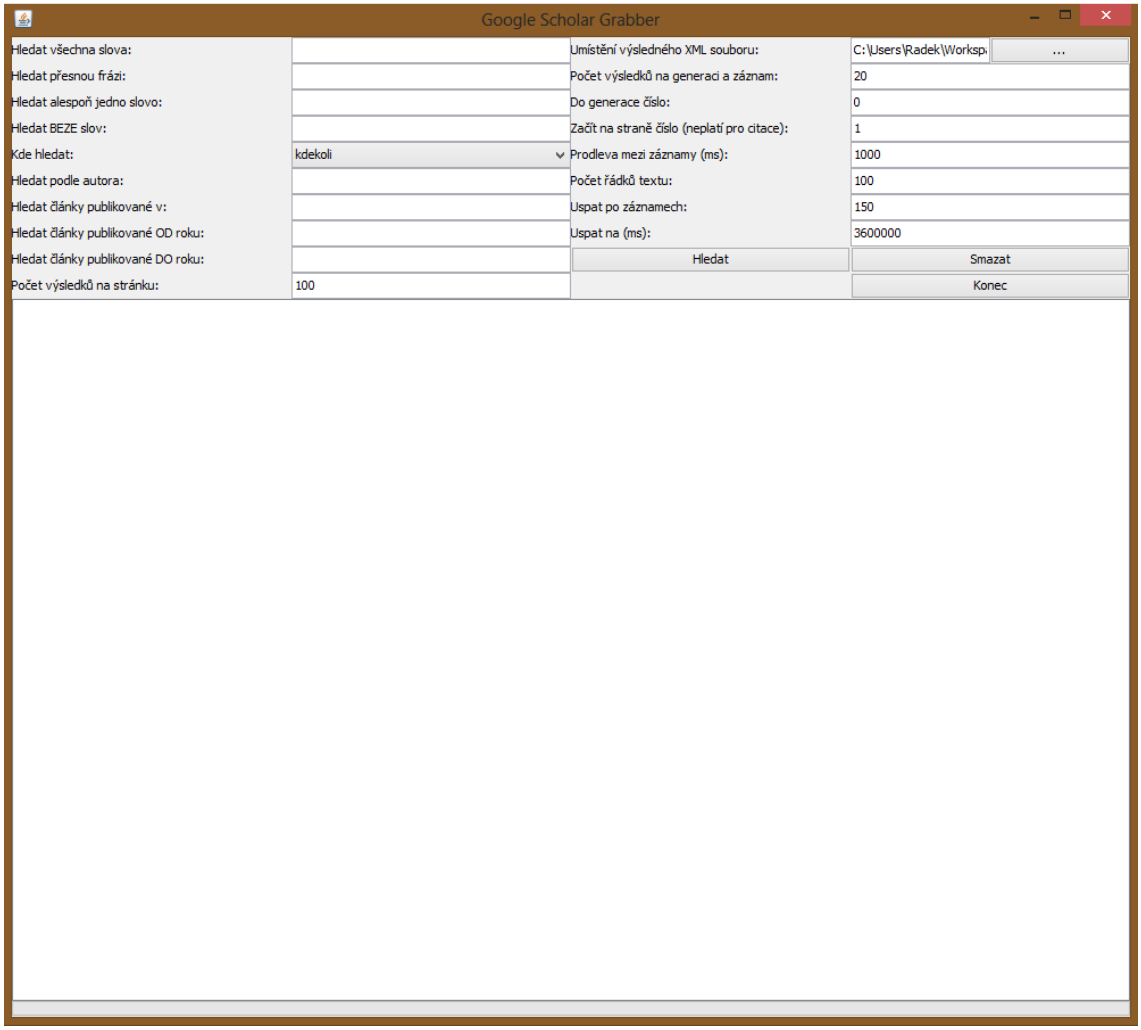
Přílohy

Příloha A Rozhraní nástroje Scopus Data Miner



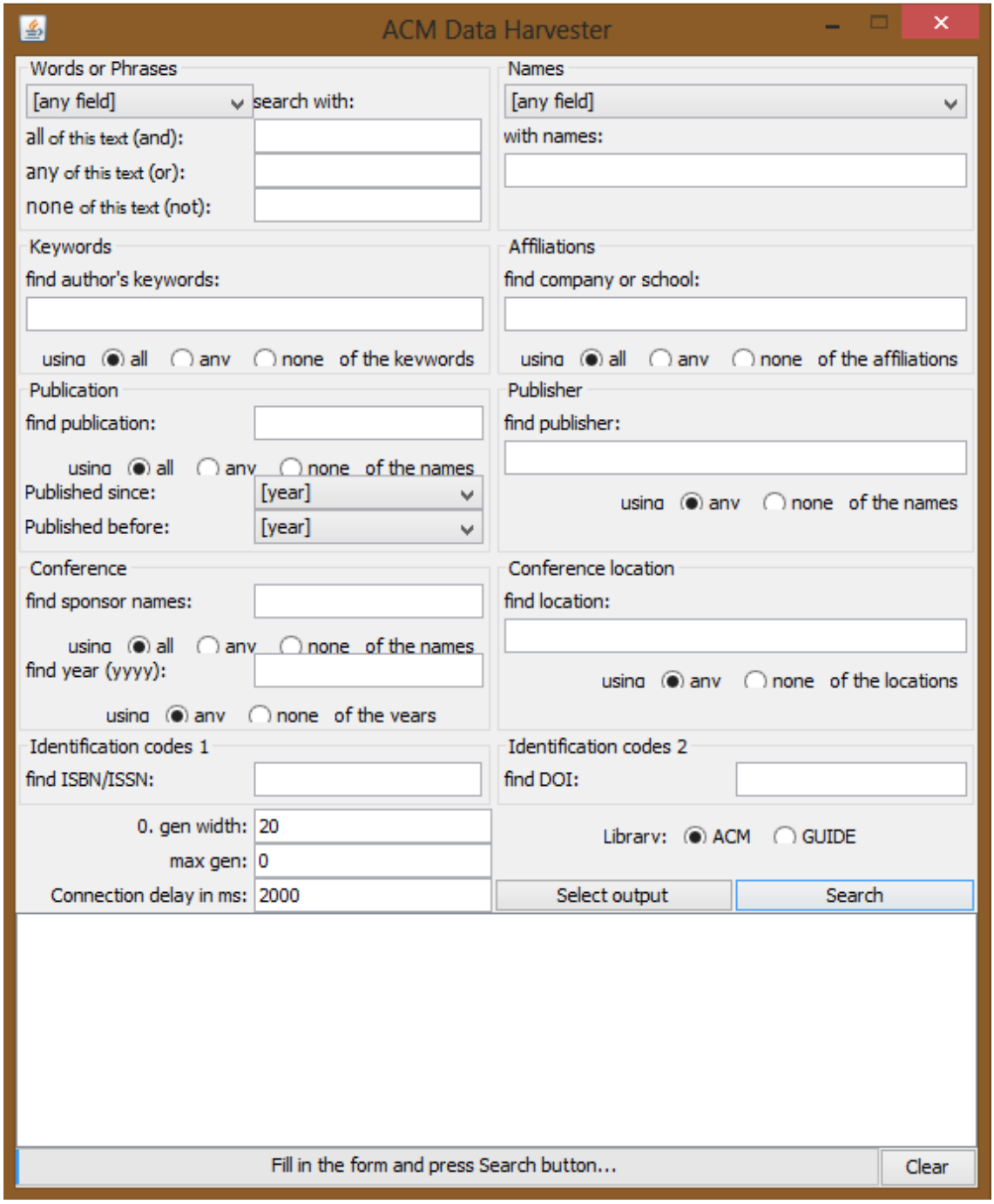
Obrázek 5 - Uživatelské rozhraní nástroje Scopus Data Miner

Příloha B Rozhraní nástroje Google Scholar Grabber



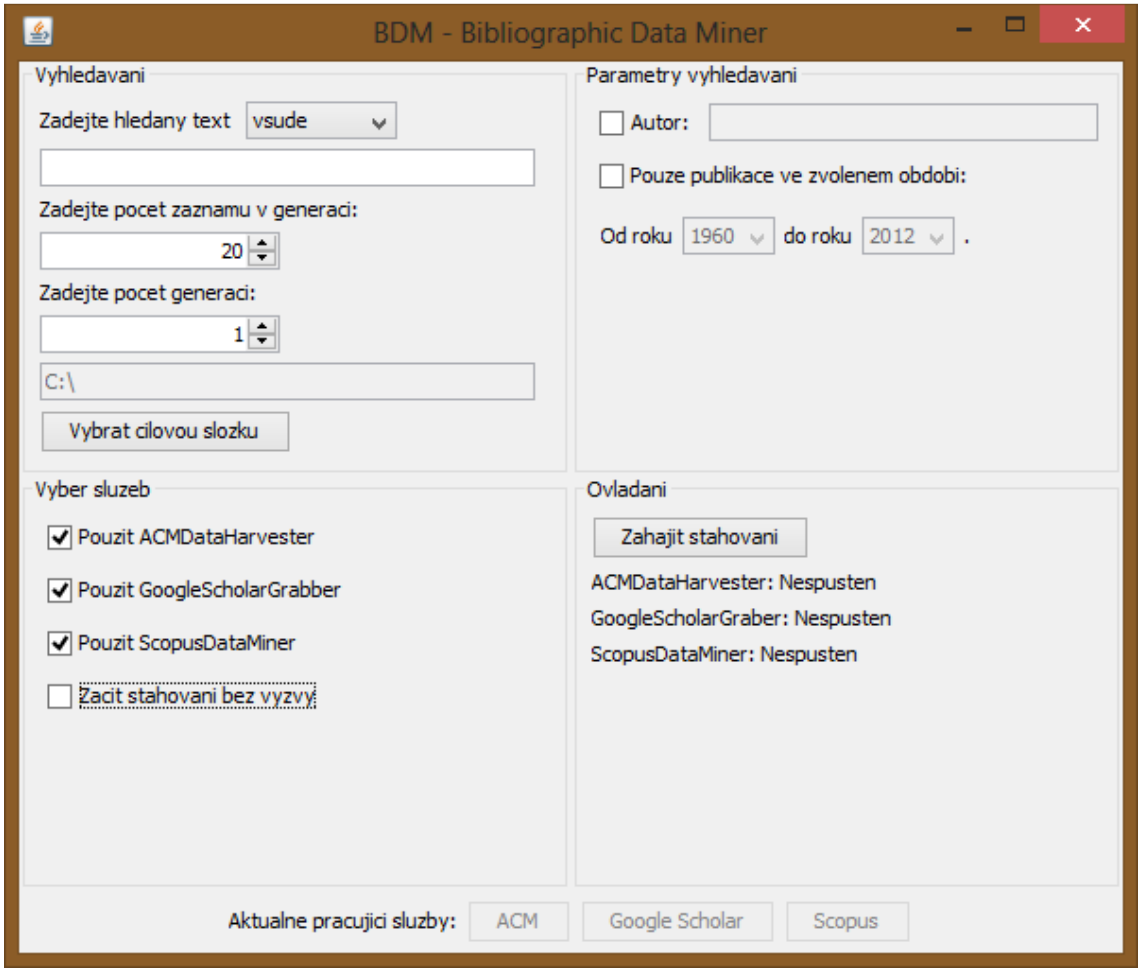
Obrázek 6 - Uživatelské rozhraní nástroje Google Scholar Grabber

Příloha C Rozhraní nástroje ACM Data Harvester



Obrázek 7 - Uživatelské rozhraní nástroje ACM Data Harvester

Příloha D Rozhraní meta-vyhledávače BDM



Obrázek 8 - Uživatelské rozhraní meta-vyhledávače Bibliographic Data Miner

Příloha E Uživatelský manuál meta-vyhledávače BDM

E.1 Spuštění

Kompletní meta-vyhledávač Bibliographic Data Miner (BDM) je uložen na přiloženém disku, v adresáři „bin“. Spustitelný *.jar* soubor se jmenuje „bdm.jar“ a při správně nastavených systémových asociacích může být spuštěn dvojklikem na ikonu.

Druhou (doporučovanou) možností je spustit BDM z příkazové řádky příkazem:

- *java -jar bdm.jar*

Vzhledem k tomu, že program generuje mnoho textu do výstupu příkazové řádky, je vhodné přesměrovat výstup do souboru následujícím způsobem:

- *java -jar bdm.jar > vystup.txt*

Pro spuštění a správnou funkci meta-vyhledávače jsou vyžadovány veškeré knihovny obsažené v podadresáři „lib“ a nainstalované JRE – prostředí pro podporu aplikací Javy.

E.2 Zahájení vyhledávání

Po spuštění se zobrazí hlavní okno meta-vyhledávače, viz Příloha D. Nejdůležitější nastavení jsou v levém horním bloku, kde zadejte vyhledávanou frázi a vyberte místo, kde se má daná fráze vyhledávat. Po zvolení cílového adresáře zvolte počet dokumentů v nulté generaci a počet generací. Pokud preferujete vyhledávání do šířky, zvolte vyšší počet dokumentů na menší počet generací. Pokud do hloubky, volte nižší počet dokumentů oproti počtu generací.

Upozornění: 1 generace = pouze nultá generace!

Pravý horní blok použijte pouze v případě, že chcete specifikovat výsledky vyhledávání.

Levý dolní blok slouží k výběru nástrojů, které chcete použít a určení, zdali mají nástroje zahájit vyhledávání dokumentů okamžitě po stisknutí tlačítka „Zahajit stahovani“ v pravém dolním bloku. To je takzvané „stahování bez výzvy“ – stahování začne automaticky.

V případě, že jste zatrhli stahování bez výzvy a spustili vyhledávání, zvolené nástroje již běží a ve spodní části okna se aktivovala tlačítka těch nástrojů, které jste zvolili. Pomocí těchto tlačítek je možné činnost jednotlivých nástrojů detailně sledovat. Jednotlivá okna nástrojů je možno kliknutím na křížek v pravém horním rohu skrýt.

V případě, že jste nezatrhli stahování bez výzvy a spustili vyhledávání, zvolené nástroje se pouze aktivují, což se projeví aktivací tlačítek ve spodní části meta-vyhledávače. Nyní je možno po kliknutí na tato tlačítka jednotlivé nástroje dodatečně nastavit a poté je nutné je spustit – každý nástroj zvlášť. Volby z meta-vyhledávače jsou do jednotlivých nástrojů přeneseny a není třeba je přepisovat.

Postupy konfigurace a ruční spuštění jednotlivých nástrojů jsou popsány v dokumentech [1], [2] a [3].

V průběhu činnosti jsou data ukládána do adresáře zvoleného v meta-vyhledávači, do souborů, podle názvu služby, ze které jsou získána:

- vystup_scopus.xml
- vystup_acm.xml
- vystup_scholar.xml

Všechny důležité zprávy z činnosti jednotlivých nástrojů jsou vypisovány do pravého dolního bloku v meta-vyhledávači.

Příloha F Navržená struktura univerzálního XML souboru

```
<publications>
	<publication id="e1" generation="0" number="1">
		<title></title>
		<authors count="1">
			<author>
				<fullname></fullname>
				<firstname></firstname>
				<middlename></middlename>
				<lastname></lastname>
				<affiliations count="1">
					<affiliation></affiliation>
				</affiliations>
			</author>
		</authors>
		<citedBy count="1">
			<id></id>
		</citedBy>
		<sourceType></sourceType>
		<journal/>
		<conferenceName/>
		<year></year>
		<publisher></publisher>
		<volume></volume>
		<number></number>
		<pages></pages>
		<country/>
		<city/>
		<month/>
		<isbn/>
		<issn/>
		<misc></misc>
	</publication>
</publications>
```

Ve výše ukázané struktuře platí:

- „publications“ je kořenová značka
- „id“ je identifikátor dokumentu v daném systému, „number“ je pořadové číslo dokumentu
- „misc“ může obsahovat libovolné informace o dokumentu
- Značka „publications“ může obsahovat více značek „publication“

Příloha G Ukázka výstupu

V následující ukázce je vidět část výstupního souboru, který reprezentuje malou citační síť vytvořenou nástrojem Scopus Data Miner.

```
<publication id="2-s2.0-37549039510" generation="0" number="2">
	<title>A short history of SHELX</title>
	<authors count="1">
		<author>
			<fullname>Sheldrick, G.M.</fullname>
			<firstname></firstname>
			<middlename></middlename>
			<lastname></lastname>
			<affiliations count="1">
				<affiliation>Department of Structural
Chemistry, University of Goettingen, Tammannstrasse 4, D-37077
Goettingen, Germany</affiliation>
			</affiliations>
		</author>
	</authors>
	<citedBy count="20">
		<id>2-s2.0-84875893372</id>
		<id>2-s2.0-84875746004</id>
		<id>2-s2.0-84875583314</id>
		<id>2-s2.0-84875959935</id>
		<id>2-s2.0-84876163313</id>
		<id>2-s2.0-84875063084</id>
		<id>2-s2.0-84874586494</id>
		<id>2-s2.0-84874521203</id>
		<id>2-s2.0-84875704832</id>
		<id>2-s2.0-84875792475</id>
		<id>2-s2.0-84875862906</id>
		<id>2-s2.0-84875802969</id>
		<id>2-s2.0-84875789398</id>
		<id>2-s2.0-84875868699</id>
		<id>2-s2.0-84875796657</id>
		<id>2-s2.0-84875770939</id>
		<id>2-s2.0-84875760823</id>
		<id>2-s2.0-84875802234</id>
		<id>2-s2.0-84875769840</id>
		<id>2-s2.0-84875831292</id>
	</citedBy>
	<sourceType>unknown</sourceType>
	<journal/>
	<conferenceName/>
	<year>2007</year>
	<publisher>International Union of Crystallography</publisher>
	<volume>64</volume>
	<number>1</number>
	<pages>112-122</pages>
	<country/>
	<city/>
	<month/>
	<isbn/>
	<issn/>
	<misc>Acta Crystallographica Section A: Foundations of
Crystallography</misc>
</publication>
```