VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

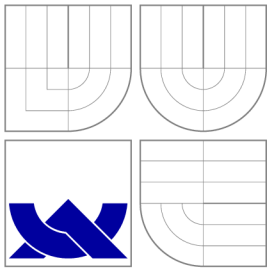
POROVNÁNÍ ÚLOŽIŠŤ PRO RDF DATA

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

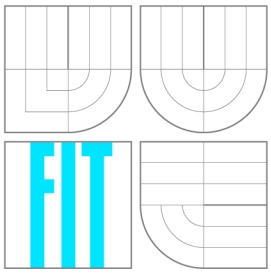
AUTOR PRÁCE
AUTHOR

JAKUB CHATRNÝ

BRNO 2015



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

POROVNÁNÍ ÚLOŽIŠŤ PRO RDF DATA

COMPARISON RDF DATABASES

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

JAKUB CHATRNÝ

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. MARTIN MILIČKA

BRNO 2015

Abstrakt

Cílem práce je porovnání několika úložišť pro RDF data a to Blazegraph, Jena TDB a Stardog. Návrh testů vychází z již existujících benchmarků. U navržených testovacích sad je monitorován čas jejich provedení a počet výsledků, které vrací. Mezi testovaná kritéria úložišť také patří čas potřebný k vytvoření persistentního úložiště a jeho velikost. Všechna testovaná úložiště i samotný program pro testování, jsou implementovány v jazyce Java.

Abstract

The goal of this thesis is compare of several RDF stores Blazegraph, Jena TDB and Stardog. Design of test suites is based on existings benchmarks. In the designed test suites are monitored execution time and number of results. The testing criteria of stores also include the time required to create persistent store and its size. All tested stores and the program for testing itself are implemented in Java.

Klíčová slova

RDF, OWL, SPARQL, sémantický web, RDF úložiště, Java, BigData, Blazegraph, Apache Jena, Sesame Jena TDB, Stardog

Keywords

RDF, OWL, SPARQL, semantic web, RDF store, Java, BigData, Blazegraph, Apache Jena, Sesame Jena TDB, Stardog

Citace

Jakub Chatrný: Porovnání úložišť pro RDF data, bakalářská práce, Brno, FIT VUT v Brně, 2015

Porovnání úložišť pro RDF data

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Martina Miličky.

.....
Jakub Chatrný
18. května 2015

Poděkování

Tímto bych rád poděkoval Ing. Martinu Miličkovi za pomoc a odborné vedení mé práce. Martinu Veselému a Bc. Františku Němci za korekce textu a zpětnou vazbu.

Obsah

Kapitola 1

Úvod

World Wide Web je bez pochyb jedním z největších fenoménů naší doby. Jeho popularita spočívá především v rychlém přístupu k velkému množství informací.

Současná forma webu je založená na dokumentech. Pokud chceme přidat nějakou informaci, vytvoříme dokument, do něj informaci vložíme a kdokoliv se pak může na náš dokument odkazovat. Webová informace je organickou entitou, která roste zájmem a energií jejích podporovatelů [1]. Množství informací na webu je ohromující a neustále roste. S tím přichází ovšem řada problémů, například čím dál tím těžší nalezení relevantních informací.

Seznámení s tématikou sémantického webu je popsáno v kapitole 2. Data sémantického webu jsou modelovány pomocí RDF nebo některým z jeho rozšíření RDF(s) a OWL. Data zapsat pomocí známých serializačních formátů XML, JSON a dalších. Stejně jako u relačních databází dochází k úpravě a čtení dat pomocí dotazovacího jazyka. RDF úložiště často používají jako dotazovací jazyk SPARQL.

Účelem této práce je porovnání úložišť sémantického webu – RDF úložišť. V současnosti existuje celá řada RDF úložišť implementována v různých programovacích jazycích. Vybrané úložiště a jejich aplikační rozhraní jsou popsány v kapitole 3.

Návrhem testů se zabývá kapitola 4. Pro otestování úložišť byly vybrány tři sady testovacích dat a bylo vytvořeno šest sad testovacích dotazů.

Testy byly provedeny testovací aplikací implementované v Javě, jejíž tvorba byla součástí této práce. Kapitola 7 popisuje nástroje použité k tvorbě aplikace, použití specifických funkcí a náčrt struktury programu. Popis ovládání aplikace je uveden v příloze B. Popis požadavků pro běh aplikace je uveden na cd viz. příloha C.

Při vyhodnocovaní výsledků je hodnocena rychlost vytvoření persistentního úložiště, jeho velikost a rychlost provedení dotazů jednotlivých testovacích sad. Údaje o výsledcích testů jsou součástí kapitoly 5.

Vyhodnocení testů je popsáno v kapitole 6. Kapitola vyhodnocuje, jak si vedly jednotlivé úložiště. Na základě provedených testů je zde doporučeno, kde testované úložiště použít.

Kapitola 2

Seznámení s tématikou sémantického webu

Tato kapitola popisuje základy o tom, co je to sémantický web a o nástrojích, které ho umožňují tvořit. Jako zdroj informací pro tuto kapitolu posloužila kniha *Semantic Web for the Working Ontologist: Effective Modeling in RDFS and OWL* [1], která je eesenciální pro lepší pochopení dané problematiky. Velmi užitečným zdrojem se ukázali také stránky konzorcia W3C [15] a prezentační snímky předmětu *Internetové aplikace* Fakulty informačních technologií VUT v Brně [6]. Jako alternativní zdroj informací lze, také použít akademické publikace Ing. Jaroslava Dytrycha [5], Petra Matulíka a Tomáše Pitnera [7].

2.1 Historie webu

V roce 1980 byl Tim Berns-Lee, oxfordský absolvent oboru fyziky, zaměstnán v Evropské organizaci pro jaderný výzkum známé jako CERN. Během svého tamního působení si všiml, že se mezi místními výzkumnými týmy nachází značné množství neorganizovaných informací. Jako řešení tohoto problému vytvořil program, který nazval ENQUIRE. Ten vytvářel vztahy mezi těmito informacemi. Tento program je považován jako předek současného webu. Koncem osmdesátých letech minulého století představil projekt, který měl myšlenku sdílení dat ještě rozšířit. O pár let později byl tento projekt zveřejněn skrze internet a World Wide Web se zrodil.

Roku 1994 bylo založeno konzorcium W3C (World Wide Web Consortium), které vytváří standardy a dohlíží nad dalším vývojem webu. Jedním ze spoluzakladatelů tohoto konzorcia byl i samotný tvůrce webu Tim Berns-Lee. V roce 2001 zveřejnil časopis Sciencific American článek [3], ve kterém se Berns-Lee spolu s Jimem Hendlerem a Orou Lassilou zamýšlí nad tím, jak by mohl web vypadat v budoucnu. V článku je zmíněno několik futuristických vizí, kde webové aplikace plní automaticky úkony za člověka, jako je například organizace kalendáře a jiné. K tomu aby bylo takovéto aplikace možné vytvořit, je nutné současný web vylepšit. A tak vznikla myšlenka *sémantického webu.*

Na sémantickém webu jsou v samém centru pozornosti samotná data, nikoli dokumenty, které data obsahují. Tato myšlenka umožňuje vyřešit problém nekonzistence dat. Data jsou modelována tak, že je možné je používat jako zdroje.

2.2 Sémantický web

Sémantický web si klade za cíl zlepšit konzistenci a dostupnost webových dat. Jeho tvůrci se zároveň zamýšlí, jak by měl web vypadat v budoucnu. Měl by umožnit tvorbu chytrých aplikací, které dokážou vykonávat různé úkony automaticky, bez přispění člověka. Mnoho takovýchto webových aplikací již existuje. Webové vyhledávače, které dokáží získat výsledky, které působí intuitivně. Internetové obchody dokážou mapovat vaše nakupovací zvyky a dle nich vám nabízet zboží, které by se vám mohlo líbit. Všechny tyto aplikace si vytváří vlastní infrastrukturu podobnou myšlence sémantického webu. Ale pokud je přístup k těmto informacím možný, tak pouze přes nejednotné rozhraní. Sémantický web se tento snaží problém odstranit a vytvořit jednotné rozhraní pro přístup ke zdrojům informací na webu. Staví na několika myšlenkách:

- Vytvoření formátu, který umožní integraci dat z různých zdrojů.
- Vytvoření prostředků pro zaznamenávání, jak se data váží k objektům reálného světa.
- Web je otevřený komukoli. Kdokoli může přispět do celku, tak aby to bylo přístupné komukoli. (Tato myšlenka je součástí původního webu, ale je velmi důležité, aby zůstala dodržena)

K „přetvoření“ současného webu na web sémantický je třeba přispění tvůrců webových aplikací tak, aby umožňovali čerpat veřejné informace přímo z jejich zdrojů. Ačkoli myšlenka sémantického webu vznikla již před více než dekádou, rozšíření sémantického webu je prozatím velmi pomalé. Jeho tvůrci ovšem doufají, že čím více lidí se zapojí a změní svou datovou strukturu, tím více bude perspektivní pro ostatní konvertovat své data do RDF. Tento jev začarovaného kruhu se nazývá *network effect* a byl důvodem, proč došlo k popularizaci původního webu.

Jedním z příkladů poukazujících na problém současné formy webu, zmínili Dean Allemang a Jim Hendler [1] ve své publikaci.

> *Představte si, že se rozhodnete v rámci vzdělávání zúčastnit kurzu. Kurz probíhá několik dní ve školícím centru, které se nachází ve vzdáleném městě. Budete tedy potřebovat ubytování. Podíváte se na webové stránky školícího centra, kde naleznete odkaz na hotelový řetězec. S tímto řetězcem máte již zkušenosti, proto se rozhodnete zde ubytovat. Podíváte se proto na stránky tohoto řetězce a k vašemu překvapení zjistíte, že daná pobočka již neexistuje.*

Tento problém je způsoben ne-konzistencí dat. Informace o hotelové pobočce se nacházela na obou webech, ale jelikož mezi touto informací nebyla žádná vazba, tak odstranění informace na jedné straně neovlivnilo druhou stranu. Problematika konzistence dat je jedna z věcí, které by měl řešit právě sémantický web. Informace o pobočce by byla uložena pouze na straně řetězce a obsahovala by jednoznačný identifikátor (URI). Pomocí toho identifikátoru by byl zobrazen její obsah na obou webových stránkách. Pokud by tedy došlo ke změně informace, projevila by se na obou webových stránkách.

2.3 RDF - Resource Description Framework

RDF je základním frameworkem, na kterém je založen zbytek sémantického webu. Hlavním účelem RDF je reprezentovat data tak, aby je mohli zpracovat jiné stroje a přitom nedošlo ke

ztrátě významu. Data modelována pomocí RDF se v kontextu sémantického webu nazývají zdroje (anglicky resources). Zdroje jsou popsány pomocí jednoduchých vlastností a hodnot. Modelování je založeno na základním prvku, který se nazývá RDF trojice a ta umožňuje mezi dvěma libovolnými entitami vytvořit vazbu. Trojice se mohou libovolně vázat mezi sebou a vytvářet modely obsahující milióny trojic. Takto zapsané informace lze reprezentovat jako graf.

2.3.1 RDF trojice

Trojici lze chápat jako relaci mezi dvěma entitami reálného světa. Skládá se ze **subjektu** - **predikátu** - **objektu**.

Například v tvrzení:

Shakespeare je autorem Richard III.

„Shakespeare“ je subjektem, predikát představuje „je autorem“ a objektem je „Richard III“. V tomto příkladu si lze povšimnout, že jak subjekt, tak objekt může vyjadřovat více významů. Richard III je zároveň hrou, kterou Shakespeare napsal a historickou postavou. Pro odlišení těchto ne-jednoznačností se v RDF používají jmenné prostory *namespace*.

Příklad by tedy s přidáním patřičných namespace vypadal, takto:

lit:Shakespeare lit:autor lit:RichardIII.

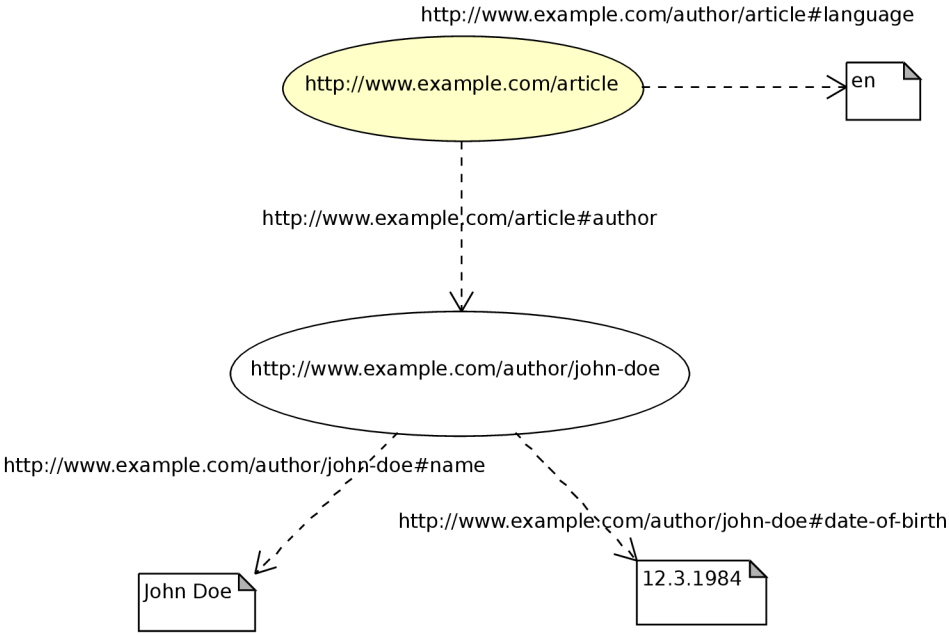
K identifikaci každé entity trojice se využívá mechaniky převzaté z původního webu a to sice webových identifikátorů. Tyto identifikátory odpovídají specifikaci URI (*Uniform Resource Identifiers*). V roce 2005 byl také zveřejněn standard pro mezinárodní identifikátory IRI (*Internationalized Resource Identifier*), které mohou obsahovat znaky mimo latinskou abecedu.

Zde je příklad RDF grafu, který by mohl reprezentovat reálnou aplikaci:

2.3.2 Serializační nástroje pro zápis RDF

RDF samotné slouží jen pro popis sémantiky dat a neobsahuje žádnou syntaxi. Zápis RDF je možné provést několika způsoby.

- **RDF\XML** – Tento způsob syntaxe byl navržen spolu se samotným RDF. Je stále používán, ačkoli jeho alternativy jsou preferovány díky lepší čitelnosti pro člověka.
- **N-TRIPLES** – řádkově založený serilizační formát. Využívá nejednodušího možného zápisu RDF dat. Trojice je zapsána sekvenčně za sebou v pořadí subjekt, predikát, objekt. Jednotlivé části trojice jsou oděleny mezerou. Přičemž každá RDF trojice je umístněna na novém řádku ukončeným tečkou.
- **Trutle** – jedná se o jednu z populárnějších variant zápisu RDF. Gramatika této syntaxe je podmnožinou SPARQL, které využívá velké množství RDF úložišť.
- **JSON-LD** – Javascript Object Notation for Linked Data. Tato syntaxe převádí trojice do serializačního formátu JSON, který je ve sféře vývoje webových aplikací také populární.



Obrázek 2.1: Příklad jednoduchého RDF grafu

2.4 RDF Schéma

RDF Schéma je rozšíření, které dovoluje definovat vlastní třídy, podtřídy, vlastnosti a binární relace. Zjednodušeně lze říci, že přidává do RDF typový systém. Při tvorbě tohoto typového systému byla brána inspirace převážně z objektově orientovaných jazyků, jako je Java. RDF Schéma umožňuje vícenásobnou dědičnost a hierarchickou organizaci tříd a relací.

Pokud aplikace umožňuje RDF, je možné použít i RDF Schéma. To je možné, díky tomu, že slovní zásoba zapsaná v RDF Schéma, je regulérním RDF grafem. Tento mechanismus dokonce umožňuje komukoli redefinovat RDF, stejně jako data v RDF.

2.5 OWL – Web Ontology Language

Jak již název napovídá, jedná se o ontologický jazyk. Ontologii je možné definovat jako doslovný a formalizovaný popis určité problematiky, v tomto případě sémantického webu.

Účelem OWL je rozšířit možnosti sémantického webu. Vznikl jako revize DAML+OIL a snaží se využít zkušeností získaných při návrhu tohoto ontologického jazyka. Oproti RDF a RDF Schema také přidává terminologii pro popis vlastností a tříd, jako jsou vztahy mezi třídami, kardinality a jiné.

OWL nabízí tři podjazyky určené pro použití specifických komunit uživatelů.

- **OWL Lite** – zjednodušená verze umožňuje především klasifikaci hiearchií jednoduchých omezení (např. omezení kardinality).
- **OWL DL** – obsahuje všechny jazykové konstrukce, ale mohou být použity pouze v rámci určitých omezení. Zaručuje výpočetní úplnost.
- **OWL Full** – umožňuje maximální kompatibilitu s RDF. Dále také rozšiřuje význam předdefinované slovní zásoby (jak RDF tak OWL). Ovšem výpočetní úplnost není zaručena.

2.6 SPARQL – Simple Protocol and RDF Query Language

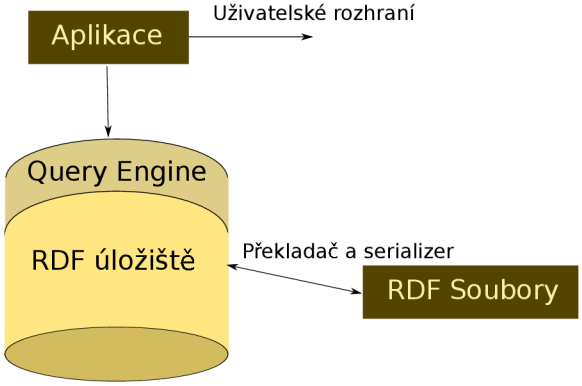
SPARQL [9] je dotazovacím jazykem nad daty uloženými v RDF. Konstrukce a syntaxe se podobá jazyku SQL. Obsahuje řadu analytických funkcí pro konjunkci, disjunkci, řazení a agregaci. Jak již bylo zmíněno v kapitole 2.3.1, pro RDF trojice lze použít jmenné prostory. SPARQL syntaxe je umožňuje v dotazu definovat a následně užít pomocí klíčového slova `PREFIX`. Využití jmenných prostor souvisí s identifikací RDF trojice, která je blíže popsané v kapitole 2.3.1.

2.7 Architektura aplikace sémantického webu

V předchozí kapitole byla popsána myšlenka sémantického webu a nástrojů, kterými lze modelovat data. Tato kapitola stručně představí, z jakých částí se aplikace sémantického webu skládá.

- **RDF serializér/překladač** – k serializaci RDF slouží například RDF/XML, turtle nebo JSON-LD (viz. předchozí kapitola) 2.3.2. RDF serializér převádí RDF data reprezentovaná trojicemi do serializačního formátu vyjadřujícího jejich obsah. RDF překladač provádí opačnou proceduru, serializované data převádí zpět do RDF data modelu.
- **RDF uložiště** (RDF store, nebo také triple store) – je databází určenou pro ukládání a načítání trojic z RDF modelů. Disponuje všemi nástroji relačních databází, navíc dokáže spojovat (anglicky merge) více zdrojů dat v souladu s RDF standardem.
- **RDF Query engine/jazyk** – úzce spjatý s RDF úložištěm. Spolu s RDF query jazykem poskytuje možnost přístupu k datům. Mezi populární RDF query enginy patří například ARQ. Jedním z nejpoužívanějších jazyků je SPARQL, které bylo vytvořen W3C.
- **Aplikace** – využívá všech předchozích nástrojů k tomu, aby data byla prezentována uživateli. Volba programovacího jazyka se nejčastěji odvíjí od implementačního jazyka RDF úložiště.

V dnešní době již existují implementace RDF úložišť v jazycích jako C, C++, Java, C#, Perl, PHP, Python a další. Seznam implementací těchto úložišť lze nalézt na wikipedii [16] Pro lepší představu architektury RDF aplikace může posloužit následující obrázek.



Obrázek 2.2: Architektura aplikace využívající RDF úložiště

Kapitola 3

Testované RDF úložiště

V této kapitole je představeno testované úložiště a to Blazegraph [12], Jena TDB [10], Stardog [14]. Dále frameworky obsahující API pro tvorbu aplikace.

Všechny testované úložiště jsou implementována v jazyce Java, Podporují modelovaní dat pomocí RDF, RDF(s) a OWL. Úložiště dále umožňují podporu dotazování pomocí SPARQL.

- **Blazegraph**[1] - je vysoko výkonnostní open-source databáze podporující RDF data model. Podporuje dvě možná API a to Sesame a Bluprints.
- **Jena TDB** – nativní úložiště frameworku open-source Apache Jena. Může být použito jako vysoko výkonnostní RDF úložiště na jednom zařízení.
- **Stardog** – a výkonná komerční RDF databáze. Je možné užít omezenou community verzi. Omezení se především týkají maximálně deseti databází a maximálně 25 milionu trojic na databázi. Více o licenci lze nalézt na webu projektu [13].

3.1 Apache Jena

Apache Jena, dále jen jako Jena, je open-source framework pro tvorbu aplikací sémantického webu. Základní architektura Jeny je v podstatě shodná s architekturou sémantické aplikace z předchozí kapitoly 2.7. Aplikace využívající framework komunikuje s vrstvou, která obsahuje několik API. A to sice pro modelování RDF, serializaci/překlad RDF, API vykonávající dotazy nad existujícími modely a vrstvu umožňující tvorbu vlastních ontologií.

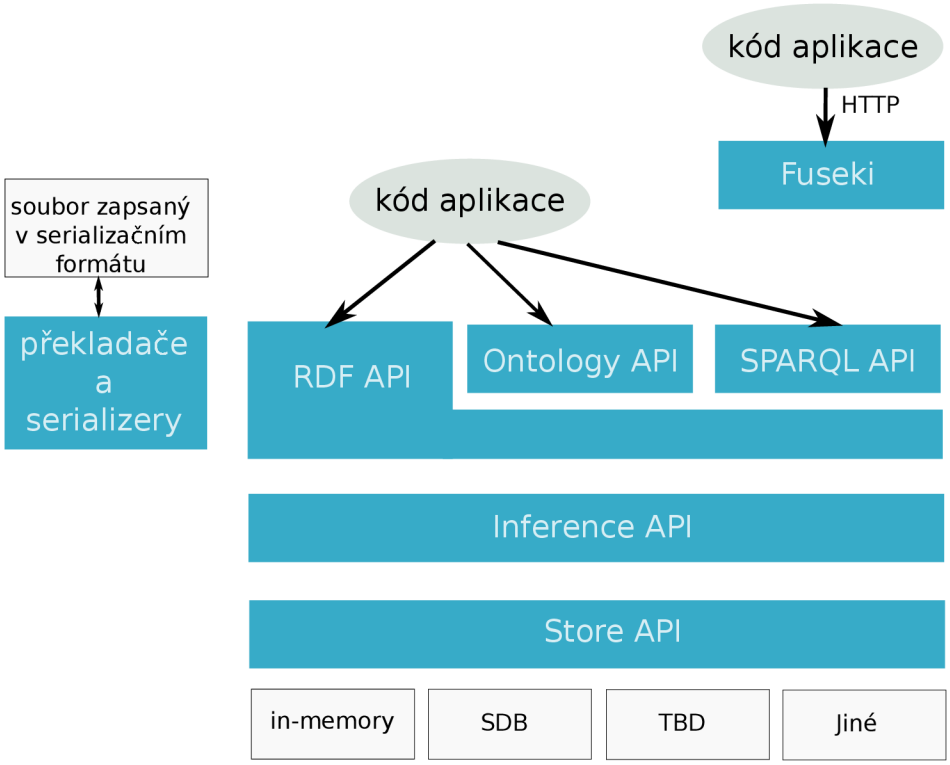
3.1.1 RDF Serializace a Překladač

Funkce pro serializaci, případně pro překlad serializovaného souboru, je zahrnuta v *RDF API*. Mezi podporované serializační formáty patří RDF\XML a Trutle.

3.1.2 RDF Query engine

Jako *query engine* je použit ARQ, který podporuje *SPARQL query language*. Přístup k ARQ zprostředkovává *Store API*, které podporuje následující možnosti uložení dat na fyzické úložiště:

[1] Původním názvem Blazegraphu je Bigdata. Přejmenovaní projektu proběhlo počátkem roku 2015.



Obrázek 3.1: Zjednodušená architektura Apache Jena. Převzato z [11]

- **In-memory** – data jsou uložena v operační paměti.
- **SDB** – ukládá data do SQL databáze na pevný disk.
- **TDB** – vysoko výkonnostní nativní RDF úložiště na pevný disk. Používá tuple index.
- **Ostatní** – Jena umožňuje vytvořit vlastní adaptér pro jiná úložiště, jako například LDAP.

3.1.3 Aplikační rozhraní

Aplikační rozhraní se skládá z následujících podčástí:

- **RDF API** – slouží pro přístup k trojicím a grafům.
- **Ontology API** – poskytuje tvorby vlastních ontologií pomocí OWL a RDFS.
- **SPARQL API** – umožňuje provádět dotazy nad již existujícími modely a trojicemi.
- **Fuseki** – je SPARQL server, který umožňuje přístup a úpravu dat vzdáleně skrze HTTP.

3.2 Sesame

Sesame je open-source framework napsaný v Javě umožňující ukládat RDF data a provádět nad nimi dotazy. Vedle SPAQL Sesame podporuje SeRQL dotazovací jazyk.

3.2.1 RDF Serializer a překladač

Mezi serializační formáty, které Sesame podporuje patří TriG, BinaryRDF, TriX, N-Triples, N-Quads, N3, RDF/XML, RDF/JSON a Turtle.

3.2.2 RDF Query engine

U Sesame je jako *query engine* použit SeRQL, který podporuje stejnojmenný dotazovací jazyk a samozřejmě i standardní *SPARQL query language*. Engine je součástí Sail API (The Storage And Inference Layer) společně s inferenční vrstvou patří mezi módy, které jsou dostupné pro ukládání dat:

- **MemoryStore** – data jsou uložena v operační paměti
- **NativeStore** – ukládá data na pevný disk.

3.2.3 Aplikační rozhraní

Aplikační rozhraní je realizováno skrze Repository API, které má několik implementací:

- **SailRepository** – slouží pro lokální přístup k RDF modelům.
- **HTTPRepository** – umožňuje transparentní *client-server* komunikaci přes HTTP.

Kapitola 4

Navržené Testy

Tato kapitola se zabývá návrhem samotných testů pro vybraná RDF úložiště. Při návrhu testů bylo čerpáno z několika existujících benchmarků a to sice DBSPB [8], BSBM [4] a *RDF Store Benchmarks with DBPedia* [2].

Při tvorbě testovací sady bylo třeba dbát na licenční omezení jednoho z testovaných úložišť a to Stardogu. Zde licence dovoluje maximálně 25 milionů trojic na databázi.

4.1 Existující Benchmarky

Součástí této kapitoly je stručný souhrn existujících benchmarků, které měří výkon RDF úložišť.

4.1.1 DBSPB - DBpedia SPARQL Benchmark

DBSPB je SPARQL benchmark, který aplikuje metodiku query-log miningu, query clusteringu a analýzy SPARQL vlastností. Tento benchmark využívá jako datavou sadu RDF data DBpedie a rovněž dotazy, které byly nad touto databází v minulosti často prováděny. DBSPB vytvořil tým z oddělení informatiky Univerzity v Leipzigu. Členi týmu, kteří se na DBSPB podíleli jsou Mohamed Morsey, Jens Lehmann, Soren Auer, and Axel-Cyrille Ngonga Ngomo.

4.1.2 BSBM - Berlin SPARQL Benchmark

Tento benchmark v roce 2009 vytvořila dvojice výzkumníků Prof. Dr. Christian Bizer a Andreas Schultz z univerzity Freie Universität Berlin. Testovací data BSBM simulují prostředí elektronického obchodování a jsou generována pomocí nástroje, který byl vytvořen přímo pro účely benchmarku.

4.1.3 RDF Store Benchmarks with DBPedia

BSBM navazuje na diplomovou práci Christiana Beckera RDF Store Benchmarks with DBPedia [2]. Becker porovnává několik RDF úložišť (OpenLink Virtuoso, Postgre SDB a Sesame).

4.2 Testovací sada dat

Testová sada dat (v angličtině Dataset) se skládá z několika vzorků, které pochází ze zdroje DBpedia 2014[1]. Všechny užité testové sady dat byly staženy v serializačním formátu *N-Triples*, více o formátu je popsáno v kapitole 2.3.2.

V případě informačních boxů verze z roku 2014 přesahuje počet trojic omezení některých testovaných úložišť. Proto byla použita straší verze DBPedie z roku 2007[2], kterou ve své práci použil Christian Becker 4.1.3.

- **DBPedia – Info Boxes** – Obsahuje informační boxy wikipedie popisující stručně základní informace uvedené u článků. Databáze obsahuje značné množství trojic, které se váží ke stejnému subjektu, proto nad touto sadou lze vytvářet složitější dotazy.

Tabulka 4.1: Specifikace Info boxes

Název souboru	infoboxes-fixed.nt
Počet trojic	15 472 623
Velikost souboru	2 169 999 894 Bajtů

- **DBPedia – Person data** – databáze obsahuje základní informace (jméno, základní popis, datum narození a datum úmrtí) o osobách uvedených na wikipedii. Datová sada navzdory relativně malému objemu dat (950MB), má necelých 8 milionů trojic.

Tabulka 4.2: Specifikace datastetu *Person data*

Název souboru	persondata_en.nt
Počet trojic	7 889 574
Velikost souboru	951 901 286 Bajtů

- **DBPedia – Geo coordinates** – tato databáze obsahuje zeměpisné souřadnice míst uvedených na wikipedii. Tato sada obsahující relativně malý počet trojic. Je určen pro otestovaní složitějších dotazů nad menším objemem dat.

Tabulka 4.3: Specifikace datasetu *Geo coordinates*

Název souboru	short_abstracts_en.nt
Počet trojic	447 517
Velikost souboru	66 503 903 bajtů Bajtů

Nad těmito daty jsou provedeny dotazy pomocí jazyka SPARQL, který byl popsán v kapitole 2.6. U dotazů bude jedním z faktorů stupeň zanoření trojice.

[1] Datové sady Short abstracts, Person data lze stáhnout na adrese http://wiki.dbpedia.org/Downloads2014.

[2] DBPedia -Info boxes je dostupné http://wifo5-03.informatik.uni-mannheim.de/benchmarks-200801/#queries.

4.3 Testovací sady dotazů

Většina moderních databázových úložišť si databázové dotazy ukládá do cache paměti, proto se každá sada skládá z několika dotazů, kde jsou zaměněny některé objekty nebo subjekty RDF trojic.

Dotazy navržené v této kapitole obsahují proměnné %VALUE%, které je třeba nahradit příslušnými hodnotami, které jsou uvedeny v popisu testovací sady.

4.3.1 Testovací sada 1

Tato testovací sada je založena na jednoduchém dotazu, který vrací zeměpisnou šířku a délku několika měst. Brno, Bratislava, Paříž, Berlín a Toronto jsou města použita v testu. V dotazu je třeba nahradit %VALUE% za Brno, Bratislava, Paris, Berlin, Toronto.

Query 1

```
PREFIX prop: <http://dbpedia.org/property/>
PREFIX res: <http://dbpedia.org/resource/>

SELECT ?subject
(group_concat(distinct ?cityLatd ?cityLatm; separator = ",")
AS ?cityLat)
(group_concat(distinct ?cityLongd ?cityLongm; separator = ",")
AS ?cityLong)
WHERE {
   res:%VALUE% prop:latd ?cityLatd .
   res:%VALUE% prop:latm ?cityLatm .
   res:%VALUE% prop:longd ?cityLongd .
   res:%VALUE% prop:longm ?cityLongm .
}
```

Cílem této sady je otestovat rychlost provedení triviálního dotazu nad velkým vzorkem dat.

4.3.2 Testovací sada 2

Dotaz, ze kterého testovací sada vychází, zjišťuje místo narození vůdců státu dle daného hlavního města.

Dotaz nejprve nalezne stát, ve kterém se dané město nachází. Následně vyhledá vůdce (premiéra, prezidenta nebo monarchy) a nakonec vybere místo jeho narození. Města použitá v této testovací sadě jsou Paříž, Berlín, Madrid, Řím a Londýn. V dotazu je třeba nahradit %VALUE% za Paris, Berlin, Madrid, Rome a London.

Query 2

```
PREFIX res: <http://dbpedia.org/resource/>
PREFIX prop: <http://dbpedia.org/property/>

SELECT ?leader ?prop ?salary WHERE {
   ?country prop:capital res:%VALUE%.
   ?country prop:leaderName ?leader .
   ?leader prop:placeOfBirth ?placeOfBirth .
}
```

Tento dotaz již obsahuje zanoření RDF trojic.

4.3.3 Testovací sada 3

Účelem dotazu je vybrat všechny filmy daného režiséra, následně vybrat všechny uvedené herce, kteří v nich účinkovali. Nakonec u těchto herců vybere všechny filmy, v kterých hráli. V dotazu je třeba nahradit %VALUE% za Stanley_Kubrick, Martin_Scorsese, Tim_Burton, Jim_Jarmusch, Danny_Boyle.

Query 3

```
PREFIX prop: <http://dbpedia.org/property/>
PREFIX res: <http://dbpedia.org/resource/>

SELECT ?actor ?movie2 WHERE {
        ?movie prop:director res:%VALUE% .
        ?movie prop:starring ?actor.
        ?movie2 prop:starring ?actor .
}
```

Tento dotaz má za úkol otestovat dobu provedení při větším počtu výsledků.

4.3.4 Testovací sada 4

Cílem dotazu je vybrat jména umělců daného zaměření, kteří se narodili a zemřeli ve 20. století. Dotaz obsahuje čtyři filtry, které jsou aplikované na velké množství výsledků. Je třeba dodat, že daný umělec musí mít zapsáno datum narození i datum úmrtí ve formátu xsd:dateTime, jinak je potencionální nález ignorován. Použitá zaměření umělců v této testovací sadě jsou básník, herec, muzikant, režisér a malíř. V dotazu je třeba nahradit %VALUE% za poet, actor, musician, director, pointer.

Query 4

```
PREFIX ont: <http://dbpedia.org/ontology/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX purl: <http://purl.org/dc/elements/1.1/>

SELECT ?name ?description ?birthDate ?deathDate WHERE {
 ?person purl:description ?description
 FILTER(str(?description)="%VALUE%").
 ?person foaf:name ?name
 FILTER (langMatches(lang(?name),"en")) .
 ?person ont:birthDate ?birthDate .
 ?person ont:deathDate ?deathDate .
 FILTER (
  xsd:dateTime(?birthDate) >= "1900-01-01T00:00:00"^^xsd:dateTime
  &&
  xsd:dateTime(?deathDate) <= "1999-12-30T00:00:00"^^xsd:dateTime
 )
}
```

4.3.5 Testovací sada 5

Dotaz vrací významné lokace, poblíž daného města, ve vzdálenosti kratší než 100km. Tato testovací sada očekává vyšší počet výsledků. V dotazu je třeba nahradit %VALUE% za Moscow, New_Delhi, Tokyo, Helsinki, Stockholm.

Query 5

```
PREFIX geo: <http://www.w3.org/2003/01/geo/wgs84_pos#>
PREFIX res: <http://dbpedia.org/resource/>

SELECT ?loc ?locLat ?LocLong WHERE {
   res:%VALUE% geo:lat ?cityLat .
   res:%VALUE% geo:long ?cityLong .
   ?loc geo:lat ?locLat .
   ?loc geo:long ?locLong .
 FILTER (
     ?locLat < ?cityLat + 0.9009009009 &&
     ?locLong > ?cityLong − 0.9009009009 &&
     ?locLat > ?cityLat − 0.9009009009 &&
     ?locLong < ?cityLong + 0.9009009009
 )
}
```

4.3.6 Testovací sada 6

Tato testovací sada vychází z dotazů prvních čtyř testovacích sad. Tyto dotazy jsou zobecněny – subjekt ani objekt zde není specifikován.

Dotaz z první testovací sady vrací zeměpisnou polohu všech měst uvedených v datasetu, druhý dotaz nalezne místo narození všech vůdců, třetí nalezne filmy herců, kteří hráli ve filmu libovolného režiséra a čtvrtý osobu libovolného popisu, která se narodila ve 20. století.

Dotaz páté testovací sady není součástí. V případě zobecnění tohoto dotazu by množství výsledků mohl přesáhnout 100 000 000 (počet trojic × průměrný počet výsledků v páté testovací sadě).

Kapitola 5

Výsledky testů

V této kapitole jsou prezentovány výsledky jednotlivých testovacích sad. Součástí zhodnocení výsledků jsou také časy inicializace úložiště, jeho velikost na pevném disku a rychlost provedení inicializačního dotazu.

Každý z dotazů testovací sady byl proveden třikrát. Hodnoty zaznamenané v této kapitole jsou výsledkem jejich aritmetického průměru. Pro každou testovací sadu byli samostatně inicializovány úložiště s příslušným datasetem.

U všech testovaných úložišť dochází k jejich úplné inicializaci až během prvního dotazu, proto byl před spuštěním každé z testovacích sad vykonán *inicializační dotaz*.

Inicializační dotaz

```
SELECT * WHERE {
        ?s ?o ?p
} LIMIT 20
```

Tento jednoduchý dotaz vybere prvních dvacet RDF trojic.

5.1 Konfigurace

Testy byli prováděny na počítači s následující konfigurací:

- **Procesor:** Intel Core i5-3450 CPU @ 3.10GHz × 4
- **Operační paměť:** Corsair 4GB × 2 @ 1333 MHz
- **Pevné disky:** SSD – LITEONIT L8T-128L6G-HP 128GB, Seagate Barracuda ES2 ST3750330N 750GB
- **Operační systém:** Ubuntu 14.04 64bit
- **Prostředí Java:** Java SE Runtime Environment (build 1.8.0_31-b13)

První uvedený (SSD) disk slouží jako systémový oddíl. Veškeré operace přístupu na disk, které prováděl testovací nástroj (vytvoření persistentního úložiště), byly prováděny na druhém uvedeném disku.

Všechny testované úložiště jsou nastaveny do módu DiskRW, tedy perzistentní úložiště uložené na pevném disku. Všechny testované frameworky umožňují řadu nastavení, které mohou optimalizovat úložiště pro daný způsob užití. Až na vyjímky uvedené níže, nebylo použité žádné dodatečné nastavení.

Blazegraph vyžaduje blíže specifikovat konfiguraci například formou konfiguračního souboru. Konfigurační soubor použitý při testování je součástí přílohy A.

U Stardogu byl z důvodu zrychlení inicializace vypnut ICV (Integrity Constraint Validation) mód, který validuje data uložené v databázi.

5.2 Velikosti persistentních uložišť

Velikosti úložišť pro všechny použité datasety jsou zaznamenáné v tabulkách 5.1, 5.2 a 5.3

Tabulka 5.1: Velikosti úložišť pro dataset *infoboxes*

Testované úložiště	**Velikost datasetu**	**Velikost úložiště**
Jena	2,2GB	2,3GB
Stardog	2,2GB	14,0GB
Blazegraph	2,2GB	1,3GB

Z výsledků uvedených v tabulce 5.1 vyplývá, že nejméně místa na disku zabírá Blazegraph, který si vytváří pouze žurnálovací soubor. Persistentní úložiště vytvořené Jenou je mírně vyšší než samotný dataset, zatímco v případě Stardogu je více jak šesti násobně větší.

Tabulka 5.2: Velikosti úložišť pro dataset *person data*

Testované úložiště	**Velikost datasetu**	**Velikost úložiště**
Jena	952 MB	1,4 GB
Stardog	952 MB	3,3 GB
Blazegraph	952 MB	817,8 MB

Tabulka 5.2 uvádí velikosti úložišť pro druhý dataset. Poměr velikostí je zde podobný jako v případě prvního datasetu.

Tabulka 5.3: Velikosti úložišť pro dataset *geo cordinates*

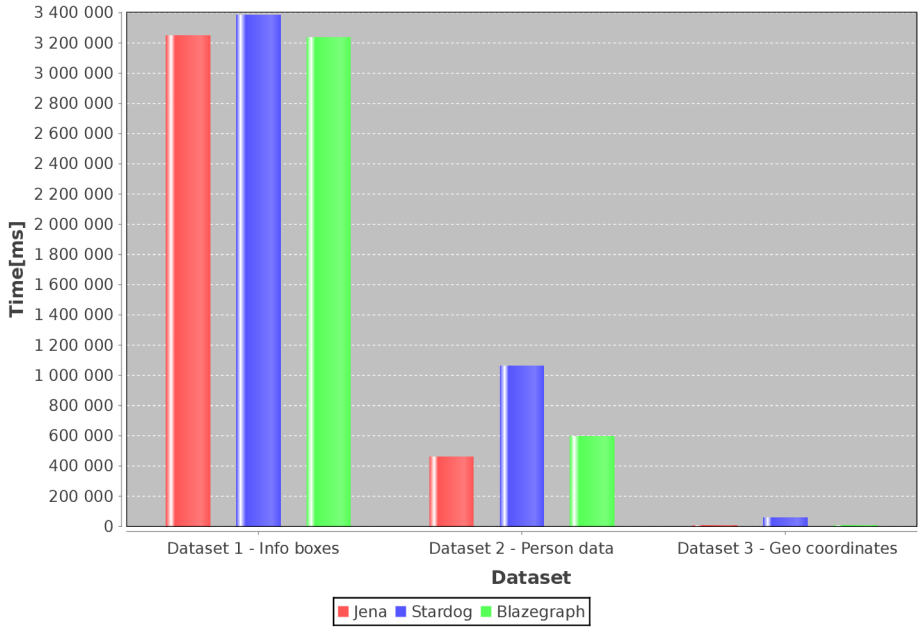
Testované úložiště	**Velikost datasetu**	**Velikost úložiště**
Jena	64 MB	260,3 MB
Stardog	64 MB	100,0 MB
Blazegraph	64 MB	209,7 MB

Dle tabulky 5.3 je zřejmé, že oproti předchozím výsledkům se poměr velikosti úložišť změnil. Starodog pro vytvoření databáze potřebuje nejméně místa. Jena a Blazegraph vyžadují trojnásobek až čtyřnásobek velikosti datasetu.

5.3 Načítací časy uložišť

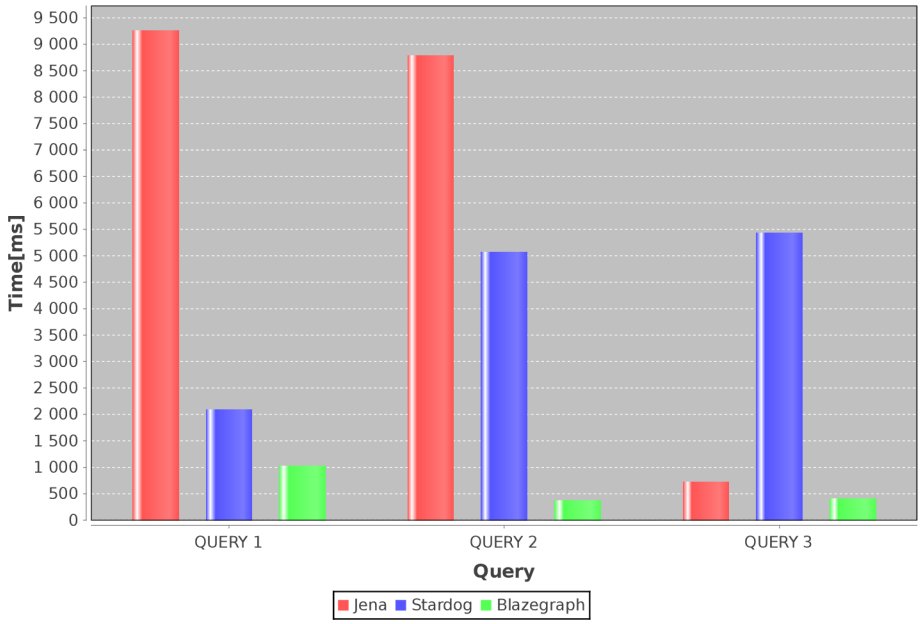
Pro každou testovací sadu dat bylo znovu vytvořeno persistentní úložiště testovaného frameworku. Časy inicializace úložišť byly zaznamenávány stejně, tak čas provedení prvního *inicializačního dotazu*.

Na grafu 5.1 jsou zaznamenány načítací časy úložišť.



Obrázek 5.1: Výsledky načítacích časů jednotlivých úložišť

Časy provedení inicializačního dotazu jsou uvedeny v grafu 5.2 (pořadí dotazů na grafu odpovídá pořadí datasetů na grafu 5.1).



Obrázek 5.2: Čas provedení inicializačního dotazu

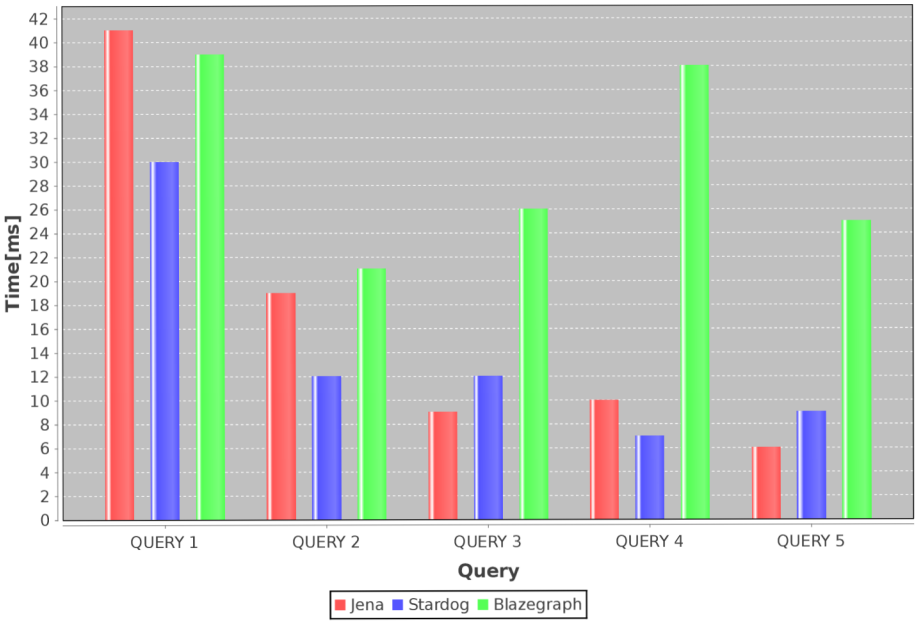
5.4 Výsledky navržených dotazů

Výsledky popsané v této sekci odpovídají výstupu testovacího nástroje, který je součástí této práce. Vstupními hodnotami nástroje jsou dotazy navržené v kapitole 4.3.

5.4.1 Výsledky – Testovací sada 1

Pořadí provedených query odpovídá pořadí proměnných (QUERY 1 - Paris, QUERY 2 - Berlin, QUERY 3 - Madrid, QUERY 4 - Rome, QUERY 5 -London, QUERY 6 - Všechny města).

Všechny dotazy v této testovací sadě vrátily jeden výsledek.



Obrázek 5.3: Graf časů provedení dotazů pro testovací sadu 1

Průměr časů všech dotazů testovací sady:

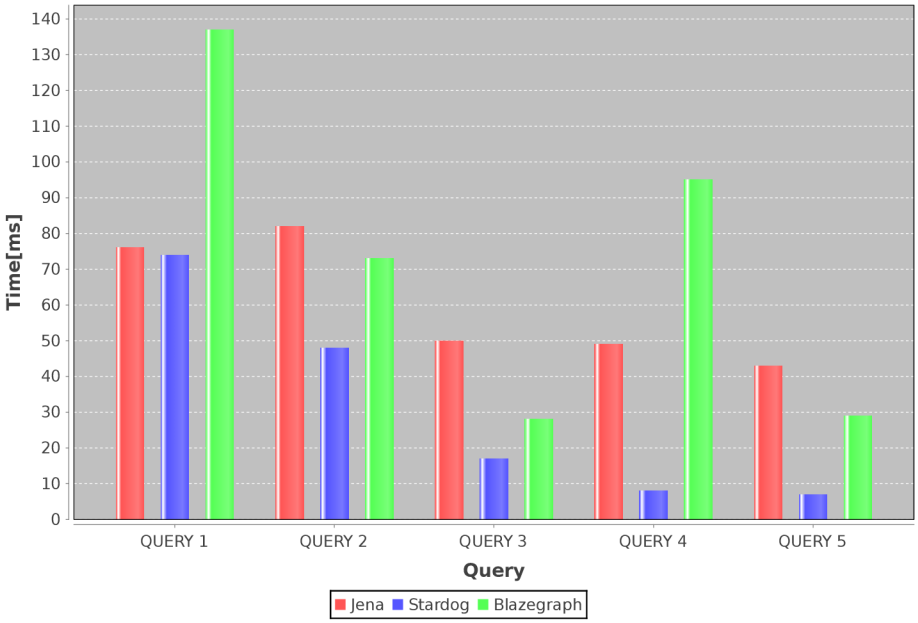
- Jena TDB - 17 ms
- Stardog - 14 ms
- Blazegraph - 29,8 ms

V tomto jednoduchém testu se ukázalo jako nejefektivnější úložiště Stardog a Jena.

5.4.2 Výsledky – Testovací sada 2

Pořadí provedených dotazů odpovídá pořadí proměnných uvedených v návrhu testů. (QUERY 1 - Brno, QUERY 2 - Prague, QUERY 3 - Paris, QUERY 4 - Berlin, QUERY 5 -Torronto).

Všechny dotazy v této testovací sadě vrátily jeden výsledek.



Obrázek 5.4: Graf časů provedení dotazů pro testovací sadu 2

Průměr časů všech dotazů testovací sady:

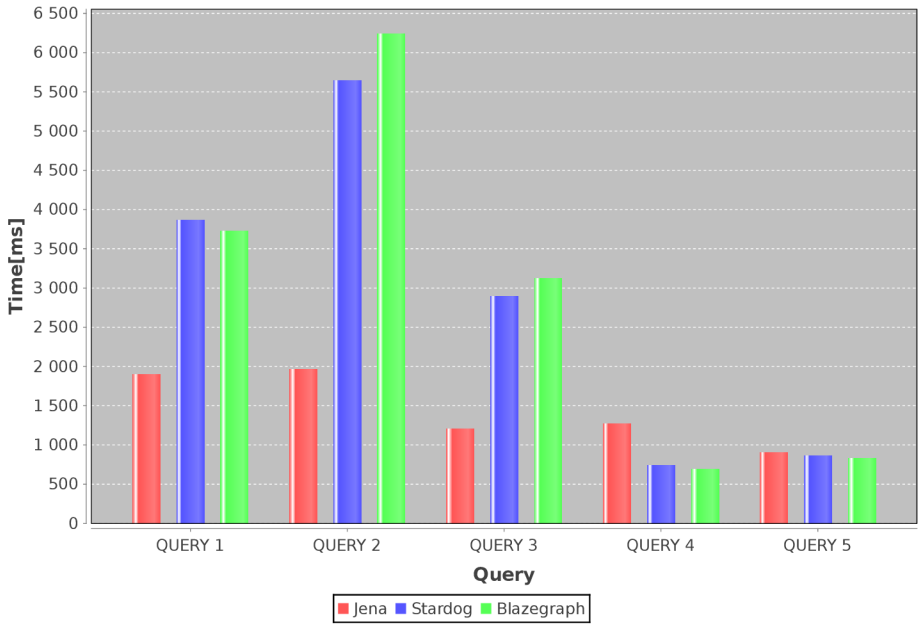
- Jena TDB - 60 ms
- Stardog - 30,8 ms
- Blazegraph - 72,4 ms

Provedení dotazů u Stardogu trvalo, v této testovací sadě, výrazně kratší dobu než u ostatních frameworků. Průměrné výsledky zbývajících frameworků, tedy Jeny a Blazegraphu jsou srovnatelné. U Blazegraphu ovšem docházelo k znatelně vyšším výkyvům.

5.4.3 Výsledky – Testovací sada 3

Pořadí provedených dotazů odpovídá pořadí proměnných uvedených v návrhu testů. (QUERY 1 - Stanley_Kubrick, QUERY 2 - Martin_Scorsese, QUERY 3 - Tim_Burton, QUERY 4 - Jim_Jarmusch, QUERY 5 - Danny_Boyle).

Dotazy v této testovací sadě vrátili 1007, 2775, 1996, 558, 561 výsledků.



Obrázek 5.5: Časy provedení dotazu pro testovací sadu 3

Průměr časů všech dotazů testovací sady:

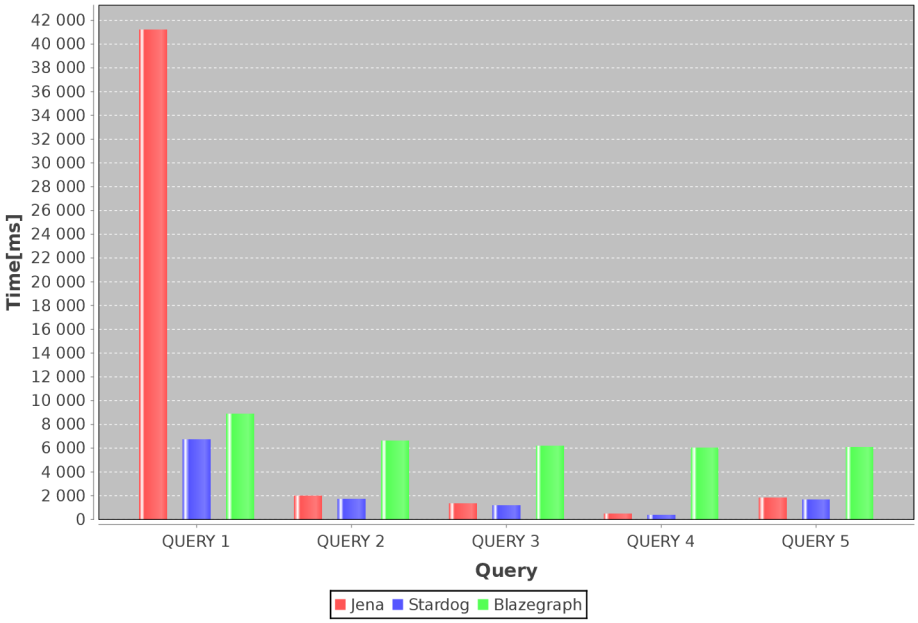
- Jena TDB - 1447,6 ms
- Stardog - 2801,8 ms
- Blazegraph - 2921,2 ms

Tato testovací sada vycházela z dotazu, který vrací větší množství výsledků. V této sadě dopadla nejlépe s výrazným náskokem Jena. Výsledky Stardogu a Blazegraphu jsou téměř totožné.

5.4.4 Výsledky – Testovací sada 4

Pořadí provedených dotazů odpovídá pořadí proměnných uvedených v návrhu testů. (QUERY 1 - poet, QUERY 2 - actor, QUERY 3 - musician, QUERY 4 - director, QUERY 5 - pointer).

Dotazy v této testovací sadě vrátili 61, 40, 6, 3, 7 výsledků.



Obrázek 5.6: Graf časů provedení dotazů pro testovací sadu 4

Průměrné časy vrácení výsledků:

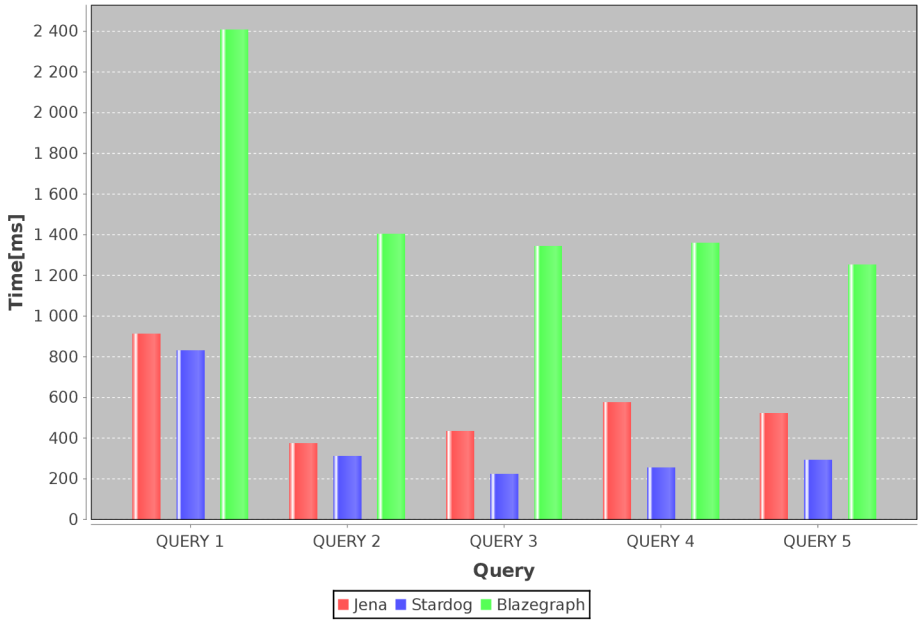
- Jena TDB - 9358,4 ms
- Stardog - 2322,8 ms
- Blazegraph - 6746,4 ms

V testovací sadě zaměřené na použití filtrů dosáhl nejlepších výsledků Stardog. V případě Jeny provedení prvního dotazu trvalo opakovaně několikanásobně déle, než u ostatních úložišť. Ostatní dotazy Jeny vracely časy srovnatelné se Stardogem. Blazegraph zaostával u všech dotazů.

5.4.5 Výsledky – Testovací sada 5

Pořadí provedených dotazů odpovídá pořadí proměnných uvedených v návrhu testů. (QUERY 1 - Moscow, QUERY 2 - New_Delhi, QUERY 3 - Tokyo, QUERY 4 - Helsinki, QUERY 5 - Stockholm).

Dotazy v této testovací sadě vrátili 171, 132, 382, 310, 191 výsledků.



Obrázek 5.7: Graf časů provedení dotazů pro testovací sadu 5

Průměr časů všech dotazů testovací sady:

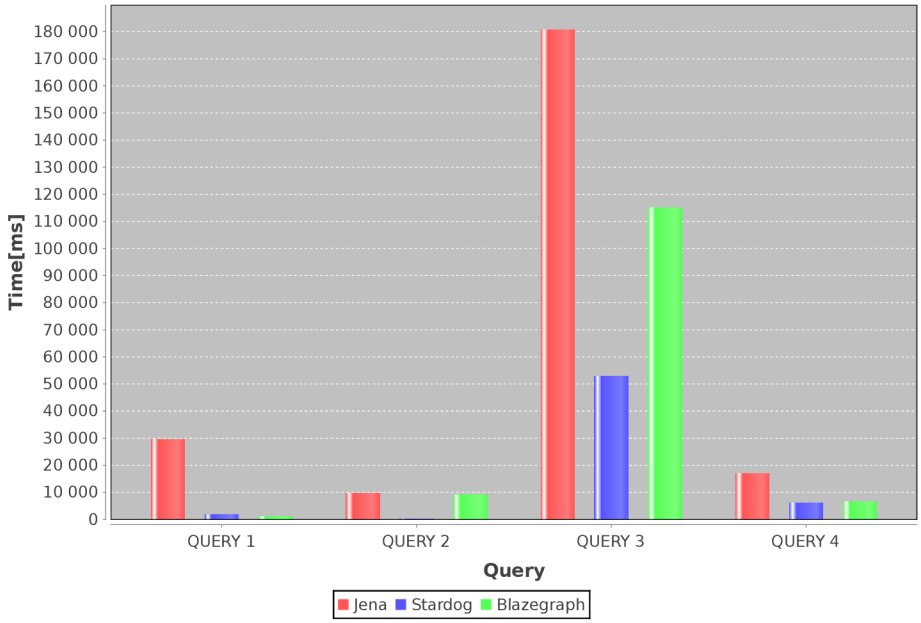
- Jena TDB - 563,4 ms
- Stardog - 375,8 ms
- Blazegraph - 1553,6 ms

V této testovací se opět jako nejefektivnější úložiště ukázal Strardog těstně následovaný Jenou. Blazegraph je zde výrazně pomalejší.

5.4.6 Výsledky – Testovací sada 6

V této testovací sadě jsou jednotlivé dotazy odvozeny od dotazů z testovacích sad 1-4. Bližší informace se nachází v kapitole návrhu 4.3.6.

Dotazy v této testovací sadě vrátili 5809, 277, 93513 a 61629 výsledků.



Obrázek 5.8: Graf časů provedení dotazů pro testovací sadu 6

Průměr časů všech dotazů testovací sady:

- Jena TDB - 88623,75 ms
- Stardog - 15307 ms
- Blazegraph - 32910 ms

Suverénně nejrychlejší se v této testovací sadě ukázal Stardog. Blazegraph má srovnatelné časy se Stardogem v prvním a čtvrtém dotazu ve zbylých dotazech je čas provedení znatelně vyšší. Jena se ukázala nejpomalejší ve všech provedených dotazech.

Kapitola 6

Vyhodnocení provedených testů

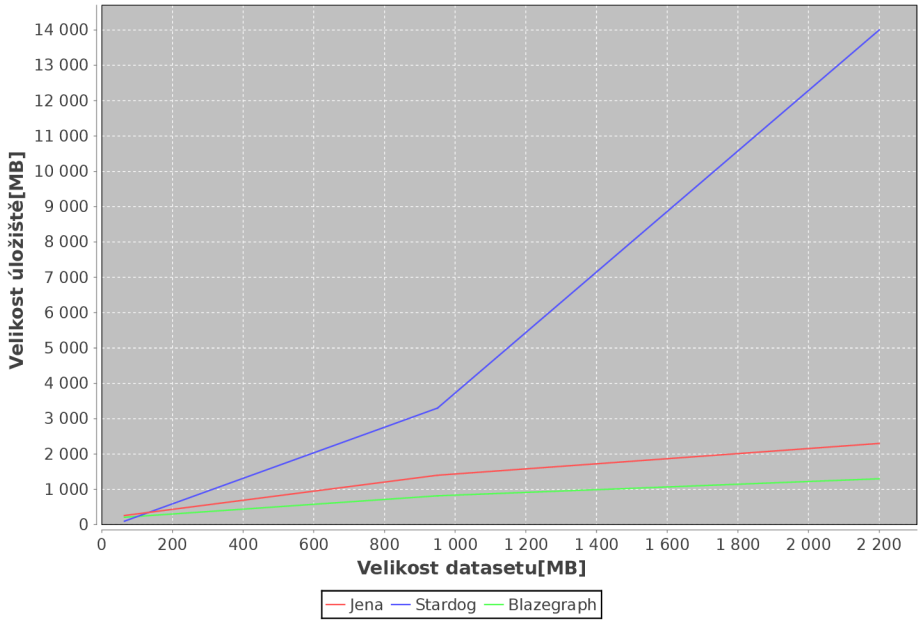
Při vyhodnocování testů byla brány v potaz následující kriteria – velikost persistentního úložiště, čas vytvoření persistentního úložiště a průměrné časy provedení všech dotazů dané testovací sady.

6.1 Vyhodnocení velikosti persistentního úložiště

V případě větších datasetů zabíral nejméně místa na disku Blazegraph. Blazegraph vytváří pouze žurnálový soubor, který neobsahuje samotný dataset. Ačkoli dokumentace úložiště neposkytuje detaily o implementaci, které by pomohly doložit toto tvrzení, je pravděpodobné, že tento fakt se negativně podepsal na rychlosti provádění dotazů.

Velikost úložiště vytvořeného Jenou se s rostoucí velikostí vstupních dat přibližuje velikosti datasetu.

Úložiště vytvořené Stardogem několikanásobně převyšuje velikost datasetu. Výsledky naznačují, že velikost úložiště vůči datasetu by se mohla blížit kvadratické složitosti. Licence Stardogu bohužel nedovolila použití datasetů, které by obsahovaly stovky milionů trojic.



Obrázek 6.1: Graf závislosti velikosti úložiště na velikosti datasetu

Co do úspornosti úložiště vychází nejlépe použití Blazegraphu nebo Jeny pro větší a střední datasety.

6.2 Vyhodnocení rychlosti načtení nebo vytvoření persistentního úložiště

Pro první dva datasety proběhlo nejrychleji vytvoření úložiště u Jeny a Blazegraphu. Stardog je zde jednou pomalejší než ostatní úložiště. V případě většího datasetu jsou časy načtení u všech úložišť srovnatelné.

6.3 Vyhodnocení výsledků testovaných úložišť

Ze zvolených testovacích dat vyplývá, že největší vliv na provedení dotazu má použití filtrů a počet výsledků. Velikost samotného úložiště ve zvolených případech má malou roli.

6.3.1 Stardog

Ve zvolených testovacích sadách si vedl nejlépe Stardog, který měl nejkratší dobu provedení dotazu v pěti z šesti testovacích sad. Ve třech případech byly průměrné časy jednou tak kratší než u ostatních úložišť. Mezi další výhody toho úložiště patří poměrně detailní dokumentace, rychlá odezva na diskusních fórech a možnost použití různých aplikačních rozhraní pro vytvoření úložiště (Sesame, Jena a vlastní rozhraní SNARL). Nevýhodou jsou omezení *Community* licence [13], stejně tak vyšší velikost persistentního úložiště pro větší (desítky miliónů trojic) datasety.

Stardog je vhodné použít pro středně velká úložiště, která nepřevyšují 25 miliónů trojic na databázi a vyhovují dalším požadavkům *Community* licence. Případně využití *Enteprise* licence.

6.3.2 Jena

Jena si vedla dobře ve třetí testovací sadě, která obsahovala zanoření trojic s použitím filtrů. V ostatních tetovacích sadách zaostává za Stardogem, ale často je to způsobeno výkyvy v časech některých dotazů. Jena je poměrně dobře dokumentována a podléhá svobodné *Apache licenci*. Persistentní úložiště jen mírně překročili velikost datasetu, takže použití Jeny není příliš náročné na diskový prostor.

Jena je díky své licenci a výkonu vhodná jako úložiště středního nebo většího rozsahu v komerčním i nekomerčním prostředí.

6.3.3 Blazegraph

V testech dosahoval nejhorších výsledků Blazegraph. Lepších výsledků lze pravděpodobně dosáhnout konfigurací úložiště dle kriterií daného datasetu. Blazegraph bohužel neobsahuje příliš obsáhlou dokumentaci.

Blazegraph je vhodný pro zkušené uživatele kladoucí na úložiště specifické požadavky.

Kapitola 7

Implementace testovacího nástroje

7.1 Použité nástroje

Volba implementačního jazyka se odvíjela od zvolených RDF úložišť, které jsou implementovány v *Javě*. Java je objektově orientovaný jazyk mezi jehož největší výhody, oproti jiným objektově orientovaným jazykům, patří přenositelnost aplikací mezi různými platformami.

Pro vytvoření buildu a získání potřebných závislostí byl užit nástroj *Apache Maven*. Maven popisuje projekt pomocí Project Object Model, který standardně bývá zapsán v souboru `pom.xml`. K vývoji Java aplikace existuje řada vývojových prostředí (IDE). Mezi nejvýznamnější patří Netbeans, Eclipse a IntelliJ. Pro tvorbu toho nástroje bylo použito Eclipse[1], které podporuje celou řadu doplňků včetně pluginu pro práci s Maven.

Pro vývoj středních a větších projektů je vhodné použít některý z systémů správy verzí jako je například *GIT*.

Při spojování použitých frameworků do výsledné aplikace se vyskytl problém závislosti na různých verzích stejné knihovny. Testované frameworky Stardog a Blazegraph jsou závislé na knihovně *Lucene*, ale každý používá jinou verzi. Z tohoto důvodu byla aplikace rozdělana na dvě části. První umožňuje testovat Jenu a Stardog. Druhá pak Jenu a Blazegraph.

7.2 Užití funkcí pro měření času

V Javě existují dvě funkce pro měření času `System.currentTimeMillis()` a `System.nanoTime()`. Pro testy byla použita funkce `System.currentTimeMillis()`. Pomocí které byl počítán reálný čas provedení dotazu.

Použití funkce `System.nanoTime()` vrací procesorový čas, který zabere provedení dotazu. Tento způsob se ukázal neefektivní. Všechny testované úložiště používají vlastní vlákna. Při volaní příslušné metody provedení dotazu dochází, také pouze k naplánování provedení. K samotnému provedení dotazu dochází pravděpodobně až v době žádosti o navrácení výsledků.

7.3 Struktura testovacího nástroje

Java umožňuje členit projekt pomocí *balíků (package)*, tato kapitola popisuje nejvýznamnější rysy jednotlivých balíků. Cílem této kapitoly není popsat implementační detaily.

[1] Eclipse je dostupné z: https://www.eclipse.org/downloads/.

7.3.1 Balík `rdf.strore.api`

Tento balík obsahuje implementace jednotlivých RDF úložišť, které se nachází v balících `rdf.strore.api.jena`, `rdf.strore.api.blazegraph` a rdf.strore.api.stardog. Třídy nacházející se v těchto balících rozšiřují třídu `StoreAPI`, která obsahuje veřejné rozhraní pro měření času inicializace a provedení dotazu.

Inicializaci lze provést provést buď do operační paměti nebo na pevný disk (do persistentního úložiště). Nástroj podporuje pouze provedení dotazu `SELECT`. Měření času provedení dotazu je umožněno skrze dva režimy (s/bez započítané iterace výsledků).

Měření času provedení inicializace a dotazu je možné dvěma způsoby. Pomocí funkce `System.NanoTime()`, která měří procesorový čas. Nebo pomocí funkce `System.currentTimeMillis()`, která měří reálný čas.

7.3.2 Balík `rdf.strore.compare`

`rdf.strore.compare` je stěžejním balíkem aplikace obsahuje třídu `RdfStoreCompare`, která provádí operace nad vybraným úložišti. Balík `rdf.strore.compare.benchmark` poskytuje třídy odvozené od StoreBenchmark. Tyto třídy přizpůsobují implementace RDF úložiště pro použití nástroje. Třída `BenchmarkQueryResults` reprezentuje jednotné rozhraní přístupu k výsledkům testů. Příklady použití implementací jednotlivých úložišť a příklad generovaní grafu z výsledků obsahuje balík `rdf.strore.compare.example`.

7.3.3 Balík rdf.strore.compare.gui

Obsahuje implementaci grafického uživatelského rozhraní, které se skládá z hlavního okna a okna ve kterém se vykreslují grafy. Pro specifické potřeby uživatelského rozhraní byli vytvořeno několik komponent, které se nachází v `rdf.store.compare.gui.component`. Komponenta přidávající čísla řádků k existující komponentě `TextArea` byla vytvořena Robem Camickem [2].

[2] Rob Camick zveřejnil svou komponentu `TextLineNumber` na: https://tips4java.wordpress.com/2009/05/23/text-component-line-number/.

Kapitola 8

Závěr

Cílem této práce bylo porovnat několika vybraných úložišť pro RDF data. K porovnání byli vybrány úložiště Blazegraph, Jena TDB a Stardog.

Pro dosažení cíle bylo nejprve nutné nastudovat problematiku sémantického webu. Na rozdíl od původní verze webu, která je založena na dokumentech, je sémantický web založen na samotných datech. Myšlenka sémantického webu byla představena před téměř patnácti lety a od této doby odvětví pozvolna získává na oblibě. Jeho potenciál je nesporně obrovský.

Pro testování jednotlivých úložišť bylo vybráno několik vzorků tetovacích dat, které jsou volně dostupné na stránkách DBPedie. Tvorba jednotlivých dotazů byla inspirována existujícími benchmarky pro RDF data jako SBSPB, BSBM a RDF Store Benchmarks with DBPedia. Šest vytvořených testovacích sad bylo použito k měření rychlosti provedení dotazu. Součástí testů bylo rovněž měření času vytvoření persistentního úložiště a jeho velikosti.

Nejrychlejší časy provedení dotazu prokázalo úložiště Stardog. Úložiště poskytuje dobře zpracovanou dokumentaci a širokou komunitu. Mezi jeho nevýhody patří omezená licence a vyšší požadavky na diskový prostor. Toto úložiště je vhodné pro komerční sféru, kde je třeba rychlé spolehlivé úložiště s placenou podporou. Jena si vedla v testech také dobře, ačkoliv se v jejím výkonu objevilo několik výkyvů. Úložiště poskytuje rovněž dobře zpracovanou dokumentaci a aktivní komunitu. Na rozdíl od Stardogu je Jena dostupná pod svobodnou licencí, její použití je tedy vhodné pro nekomerční použití. Blazegraph byl ve všech testech nejpomalejší. Blazegraph poskytuje značné množství nastavení, které bohužel není příliš dobře dokumentováno. Toto úložiště je vhodné pro zkušené uživatele kladoucí na úložiště specifické požadavky.

Jako rozšíření této práce bylo možné pro vyhodnocování testů použít netriviální metody query-log mining nebo query clustering. Testovací sady by mohli být navrženy v návaznosti na často prováděné dotazy na realně použitém RDF úložišti. Z provedených testů vyplývá, že vliv na rychlost provedení testu měli vestavěné funkce. Proto by rozšíření navržených testů mohlo být zaměřené na vestavěné funkce jazyka SPARQL (FILTER, MINUS, EXISTS, UNION a další). Další možné rozšíření je tvorba benchmark nástroje, která by pravděpodobně vyžadovala alespoň dva členy vývojového týmu.

Literatura

[1] ALLEMANG, D.; HENDLER, J.: *Semantic web for working ontologist*. Amsterdam ; Boston : Elsevier ; Morgan Kaufmann, 2008, iSBN 978-0-12-373556-0.

[2] BECKER, C.: RDF Store Benchmarks with DBPedia [online]. URL `http://wifo5-03.informatik.uni-mannheim.de/benchmarks-200801/#queries`, [2015-02-05].

[3] BERNERS-LEE, T.; HENDLER, J.; LASSILA, O.: The Semantic Web. 2001.

[4] Christian BIZER, A. S.: Berlin SPARQL Benchmark [online]. URL `http://wifo5-03.informatik.uni-mannheim.de/bizer/berlinsparqlbenchmark/spec/20110607/`, [2015-01-28].

[5] DYTRYCH, J.: Sémantický web [online]. URL `http://www.fit.vutbr.cz/study/courses/VPD/public/0910VPD-Dytrych.pdf`, 2010.

[6] HRUŠKA, T.: Slidy k předmětu WAP. URL `http://www.fit.vutbr.cz/study/courses/VPD/public/0910VPD-Dytrych.pdf`, 2014.

[7] MATULÍK, P.; PITNER, T.: Sémantický web a jeho technologie [online]. URL `http://ics.muni.cz/bulletin/articles/296.html`, 2004, iSSN 1212-0901.

[8] MORSEY, M.; LEHMANN, J.; AUER, S.; aj.: DBpedia SPARQL Benchmark [online]. URL `http://iswc2011.semanticweb.org/fileadmin/iswc/Papers/Research_Paper/03/70310448.pdf`, [2015-01-28].

[9] W3C RDF Data Access Working Group: World Wide Web Consortium - RDF current status [online]. URL `http://www.w3.org/TR/rdf-sparql-query/`, [2014-12-16].

[10] WWW stránky: Apache Jena. URL `https://jena.apache.org/`, [2014-12-03].

[11] WWW stránky: Apache Jena - architecture overview [online]. URL `https://jena.apache.org/about_jena/architecture.html`, [2014-12-03].

[12] WWW stránky: Blazegraph: RDF Database. URL `http://www.blazegraph.com/develop`, [2014-12-06].

[13] WWW stránky: Stardog commercial licensing. URL `http://stardog.com/#plans`, [2014-12-06].

[14] WWW stránky: Stardog: The Enterprise Graph Database.
URL http://stardog.com/, [2014-12-06].

[15] WWW stránky: World Wide Web Consortium - RDF current status [online].
URL http://www.w3.org/standards/techs/rdf#w3c_all, [2014-12-14].

[16] WWW stránky: List of subject-predicate-object databases [online].
URL http://en.wikipedia.org/wiki/List_of_subject-predicate-object_databases,
[2015-04-16].

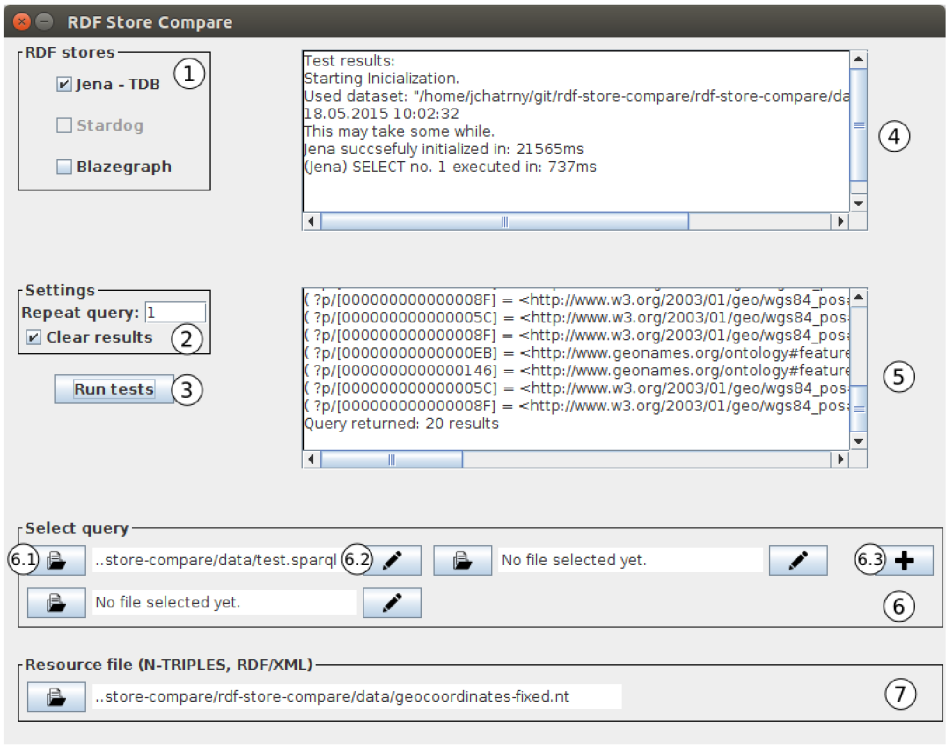
Příloha A

Konfigurace úložiště BlazeGraph

com.bigdata.journal.AbstractJournal.bufferMode=DiskRW
com.bigdata.rdf.sail.truthMaintenance=**false**
com.bigdata.rdf.sail.BigdataSail.bufferCapacity=100000
com.bigdata.rdf.store.AbstractTripleStore.axiomsClass=com.bigdata.rdf.axioms.NoAxioms
com.bigdata.rdf.store.AbstractTripleStore.bloomFilter=**false**
com.bigdata.rdf.store.AbstractTripleStore.quads=**false**
com.bigdata.rdf.store.AbstractTripleStore.statementIdentifiers=**false**
com.bigdata.rdf.store.AbstractTripleStore.justify=**false**
com.bigdata.rdf.store.AbstractTripleStore.inlineDateTimes=**true**
com.bigdata.rdf.store.AbstractTripleStore.textIndex=**false**
com.bigdata.rdf.store.AbstractTripleStore.textIndex.datatypeLiterals=**false**
com.bigdata.btree.writeRetentionQueue.capacity=4000
com.bigdata.journal.AbstractJournal.initialExtent=209715200
com.bigdata.journal.AbstractJournal.maximumExtent=209715200
com.bigdata.journal.AbstractJournal.writeCacheBufferCount=12
com.bigdata.btree.BTree.branchingFactor=128
com.bigdata.namespace.kb.lex.com.bigdata.btree.BTree.branchingFactor=400
com.bigdata.namespace.kb.spo.com.bigdata.btree.BTree.branchingFactor=1024

Příloha B

Manuál



Obrázek B.1: Hlavní okno aplikace.

Popis ovládání aplikace:

- RDF stores (1) – zde je možné vybrat, která úložiště budou testována.
- Settings (2) – jednoduché nastavení programu. *Repeat query* stanovuje množství opakování každého z dotazů vybraných v komponentě (6). Pokud je vybráno *Clear results*, při spuštění nového testu současné výsledky dotazu (komponenta (5)) jsou nahrazeny novými. V opačném případě jsou nové výsledky dotazu přidány nakonec komponenty.
- Run tests (3) – spuštění testů.
- Test Results (4) – obsahuje informační výpisy a výsledky testů.

- Query Results (5) – obsahuje výsledky dotazů.
- Select Query (6) – komponenta pro přidávání dotazů.
  - Ikona složky (6.1) – umožňuje vybrat dotaz ze souboru
  - Ikona tužky (6.2) – v případě, že je soubor vybrán umožňuje editovat dotaz (změny provedené v rámci editace nejsou uloženy do souboru). Pokud soubor není vybrán je možné vytvořit soubor pomocí editačního dialogu.
  - Ikona plus (6.3) – přidání další komponenty pro výběr souboru a editaci. Maximální počet vybraných dotazů je nastaven na 8 z důvodu limitace rozložení komponent aplikace.
- Resource file (7) – komponenta pro přidání souboru obsahujícího serializované RDF data. Testovány byli formáty N-TRIPLES a RDF/XML.

Příloha C

Obsah CD

```
+-- app – soubory aplikace.
| |
| |
| +-- bin – obsahuje spustitelné soubory typu jar.
| |
| |
| +-- src – zdrojové soubory programu.
| |
| |
| +-- data   – testovací data a testovací dotazy.
| |
| |
| +-- readme.txt – popis instalace a ovládání aplikace.
|
|
+-- doc – soubory dokumentace.
| |
| |
| +-- src – zdrojové soubory dokumentace.
| |
| |
| +-- dokumentace.pdf – projektová dokumentace.
|
|
+-- readme.txt – popis aplikace.
|
|
+-- licence.txt – licence použitých frameworků.
```