Jihočeské univerzita v Českých Budějovicích

Přírodovědecká fakulta

Využití platformy Apache Pig pro zpracování vybraných dat z webového serveru

Bakalářská práce

Autor: Tomáš Čaněk

Školitel: doc. Ing. Ladislav Beránek, CSc. MBA.

České Budějovice 2015

Jihočeská univerzita v Českých Budějovicích
Přírodovědecká fakulta

ZADÁVACÍ PROTOKOL BAKALÁŘSKÉ PRÁCE

Student: ..... Tomáš Čaněk .....
(jméno, příjmení, tituly)

Obor – zaměření studia: Aplikovaná informatika.....

Katedra/ústav, kde bude práce vypracovávána: Ústav aplikované informatiky.....

Školitel: Doc. Ing. Ladislav Beránek, CSc. .....
(jméno, příjmení, tituly, u externího š. název a adresa pracoviště, telefon, fax, e-mail)

Garant z PřF:
.....
(jméno, příjmení, tituly, katedra – jen v případě externího školitele)

Školitel – specialista, konzultant: .....
(jméno, příjmení, tituly, u externího š. název a adresa pracoviště, telefon, fax, e-mail)

Téma bakalářské práce: .....

Využití platformy Apache Pig pro zpracování vybraných dat z webového serveru
Web server data processing with the Apache Pig platform
.....

Cíle práce:

Apache Pig je skriptovací jazyk pro vytváření programů nad platformou Hadoop. Umožňuje zvládnout problémy při analýze velkých objemů dat. Skripty napsané v jazyce Pig mohou být přeloženy do řady úloh MapReduce, které mohou běžet na Apache Hadoop clusteru. Tím se zvyšuje efektivita při zpracování dat. Skripty Pig lze spouštět i v jiných jazycích a pomocí něj lze vytvářet velké aplikace pro zpracování velkých objemů dat.

Cílem bakalářské práce je provést analýzu vlastností dotazovacího jazyka Pig, ukázat, jak Pig funguje a na příkladech ukázat využití některých příkazů. Součástí práce bude popis nastavení prostředí, které bude sloužit pro řešení praktické úlohy. Praktická úloha bude spočívat ve zpracování vybraných dat z webových serverů, např. logů nebo RSS zpráv. Autor bude definovat datový soubor, cíle analýzy a provede analýzu vstupních dat. Dále provede zpracování dat pomocí jazyka Pig, bude analyzovat a popisovat jednotlivé kroky svého postupu. Bude demonstrovat výhody i případné nevýhody navrženého řešení. Popíše závěry analýzy dat a podá doporučení. Nakonec popíše stručně některé další produkty, které jsou určeny podobně jako Pig pro analýzu velkých objemů dat včetně jejich stručného hodnocení.

Základní doporučená literatura:

Welcome to Apache Pig! [online]. [cit. 2015-09-19]. Dostupné z: https://pig.apache.org/ .

ALAN GATES. Programming Pig. 1. ed. Sebastopol, CA: Oreilly, 2011. ISBN 978-144-9302-641.

HOLMES, Alex. Hadoop in practice. Shelter Island, NY: Manning, c2012, xxiii, 511 p. ISBN 16-172-9023-8.

Financování práce: .....

Vedoucí práce: .....podpis: .....

U externích vedoucích fakultní garant práce: .....podpis: .....

Garant oboru bak. studia, pokud je obor zajišťován jinou katedrou/ústavem, než ze které je školitel (nepožaduje se u oboru biologie): .....podpis: .....

Vedoucí katedry/ústavu, kde bude práce vypracovávána: .....podpis: .....

Případný souhlas vedoucího ústavu AV: .....podpis: .....

V Českých Budějovicích dne 19. 12. 2015 .....Podpis studenta: .....

Bibliografické údaje

Čaněk T., 2015: Využití platformy Apache Pig pro zpracování vybraných dat z webového serveru.
Using of Apache Pig platform for the processing of selected data from the web server. Bc. Thesis, in Czech. – [45] p., Faculty o Science,
The University of South Bohemia, České Budějovice Czech Republic.

Anotace:

Tato bakalářská práce se zabývá možností využití Apache Pig pro zpracování a analýzu vybraných dat z webového serveru zpracovávané na Jihočeské univerzitě v Českých Budějovicích a uvedením použitých zdrojů. V teoretické části popisuje přípravy prostředí a nastavení nástrojů a vytváření skriptů pro Pig, v praktické části je otestována možnost použití vybraných vstupních dat a potvrzení možnosti použití Apache Pig pro zpracování vybraných dat z webového serveru. Citace jsou zpracovány podle vzoru ČSN ISO 690.

In English

This thesis deals with the possibility of using Apache Pig processing and analysis of selected data from the web server processed at the University of South Bohemia in the Czech Budejovice and putting the resources used. The theoretical part describes the preparation of an environment and tool settings and creating scripts for Pig in the practical part is tested the possibility of using the selected input data and confirm the possibility of using Apache Pig for the processing of selected data from a Web site. Quotations are processed according to the ISO 690.

Prohlašuji, že svoji bakalářskou práci jsem vypracoval samostatně pouze s použitím pramenů a literatury uvedených v seznamu citované literatury.

Prohlašuji, že v souladu s § 47b zákona č. 111/1998 Sb. v platném znění souhlasím se zveřejněním své bakalářské práce, a to v nezkrácené podobě elektronickou cestou ve veřejně přístupné části databáze STAG provozované Jihočeskou univerzitou v Českých Budějovicích na jejích internetových stránkách, a to se zachováním mého autorského práva k odevzdanému textu této kvalifikační práce. Souhlasím dále s tím, aby toutéž elektronickou cestou byly v souladu s uvedeným ustanovením zákona č. 111/1998 Sb. zveřejněny posudky školitele a oponentů práce i záznam o průběhu a výsledku obhajoby kvalifikační práce. Rovněž souhlasím s porovnáním textu mé kvalifikační práce s databází kvalifikačních prací Theses.cz provozovanou Národním registrem vysokoškolských kvalifikačních prací a systémem na odhalování plagiátů.

V Českých Budějovicích dne _____.

_____

Tomáš Čaněk

Obsah

1 Úvod

Paralelní zpracování dat v dnešní době najde mnoho různých využití, avšak při prohledávání velmi rozsáhlých textů najde obzvlášť uplatnění. Pro společnosti jako Google Inc. v globálním měřítku nebo Seznam a.s. v národním je prohledávání těchto velmi rozsáhlých textů denní chléb a pokud chtějí hledaných výsledků dosáhnout v co možná nejkratším čase s důrazem na jejich relevanci, potřebují ke své činnosti obrovský výpočetní výkon, který pokud by měl zastávat jediný server, bude extrémně drahý a zároveň by rychlost zpracování byla pravděpodobně nedostačující dnešní poptávce. V tomto případě je tedy paralelní zpracování dat jednoznačnou volbou jak z hlediska ekonomického tak časového.

Google Computing Engine (1) je jednoznačnou ukázkou hrubé výpočetní síly, která je rozložena na blíže nespecifikovaný počet hardwarových serverů, nad kterými běží virtuální prostředí, ve kterém je možné dynamicky alokovat potřebný výkon, který tak může být i nárazový a představuje vhodné řešení pro komerční využití v praxi, kdy není potřeba držet dostatečně výkonný server, ale stačí si tento výkon pronajmout na dobu nezbytně nutnou pro zpracování dat nebo jako kompenzaci špičkové zátěže serveru či webových stránek.

Apache Hadoop Framework (2)nabízí rozhraní pro paralelní zpracování dat velmi velkého rozsahu zároveň s dostatečně nízkou chybovostí. Toto rozhraní je možné libovolně škálovatelné a v tuto chvíli běží na více než desítkách tisíc strojů v komerčním prostředí, což dokazuje jeho připravenost pro využití při zpracování velmi rozsáhlých textů a dalších aplikacích. Například Facebook používá Apache Hadoop Framework pro ukládání vnitřních logů a jako rozsáhlý datový zdroj, který používá pro analýzy a strojové učení. Tyto dva klastry mají 1400 strojů s dohromady více jak 11000 procesorovými jádry a 15 Petabyty úložného prostoru (3).

Pig (4) slouží jako platforma pro analýzu velmi rozsáhlých datových souborů spojená s jazykem vyššího řádu pro zpracování a vlastní infrastrukturou pro jejich vyhodnocení. Tato platforma vytváří nadstavbu nad Hadoop rozhraním a MapReduce (5) algoritmem, pro kterou pomocí svého jazyka kompiluje sekvence instrukcí, pro jejichž paralelní zpracování je již MapReduce připraven.

V této práci se budu zabývat přípravou prostředí pro nasazení platformy Pig a její využití při zpracování RSS logů vybraných stránek, dále zpracování samotných RSS logů, které analyzuji a navrhnu skript pro Pig v jazyce Pig Latin (6) pro analýzu těchto logů a výsledky této analýzy od prezentuji v praktické části. Součástí této práce je také prozkoumání funkčnosti platformy Pig, popsání možností psaní příkazů a jejich vykonání a to vše je ukázáno na příkladech. Všechny kroky vedoucí k vyřešení problému na reálných datech jsou také detailně popsány v praktické části. Jeden z možných problémů může být sledování vývoje oblíbenosti určitého tématu podle výskytu slovních spojení v analyzovaném textu.

Tuto práci jsem rozdělil do devíti částí. První část je věnována představení technologií, které jsou použity při analýze a aplikování řešení. Druhá část se zabývá představení platformy Pig a jejím využitím. Třetí část dává podrobnější přehled nad problémem analýzy testovacích dat. Část čtvrtá popisuje přípravu a nastavení platformy Pig, bez které by tuto práci nebylo možné realizovat. V páté části se zabývám jejím popisem a podrobnějším prozkoumáním. Šestá část ukazuje na příkladech využití Pig krok za krokem a rozebrání dalších možností použití, než na které je zaměřena tato práce. Sedmá část se věnuje obtížím, které se vyskytly při nasazování a testování pro potřeby této práce. Osmá část se zmiňuje o dalších variantách dalších platformách kromě Pig, které lze na rozhraní Hadoop použít. V deváté části je uvedeno veškeré shrnutí této práce, závěry a diskuze, která se při zpracovávání vyskytla.

1.1 Formulace problému

Představme si, že chceme vytvořit systém pro analýzu a vyhodnocení dat založených na databázi, která nám podle specifikace vrátí požadované informace případně soubory v binárních podobě, které bude potřebovat konkrétní aplikace.

Při velkém množství dat by pro takový systém bylo jistě vhodné distribuované řešení, především z důvodu rozložení dat, konektivity, a v neposledním případě při použití složitějších dotazů i rozložení výpočetní zátěže.

1.2 Výzkumné otázky

Je možné realizovat claudový systém pro analýzu vstupních dat založený na čerpání vstupních informací z kanálových souborů webových serverů?

Dílčí otázky:

1. Jakým způsobem pracuje program Pig?
2. Jaké jsou výhody a nevýhody proti centralizovanému řešení?
3. Jakým způsobem bychom mohli realizovaný systém použít?

1.3 Cíl práce

Cílem této bakalářské práce je provést analýzu vlastností dotazovacího jazyka Pig, ukázat, jak Pig pracuje a na příkladech vysvětlit použití některých příkazů. Součástí práce bude také popis nastavení prostředí, které bude sloužit pro řešení praktické úlohy. Praktické úloha bude spočívat ve zpracování vybraných dat z webových serverů, například logů nebo RSS zpráv. Dále budu definovat datový soubor, cíle analýzy a provedu analýzu vstupních dat. Dále provedu zpracování dat pomocí jazyka Pig, budu analyzovat jednotlivé kroky, svého postupu a demonstrovat výhody i případně nevýhody navrženého řešení. Popíši závěry analýzy dat a nevrhnu doporučení. Nakonec stručně popíši některé další produkty, které jsou určeny podobně jako Pig pro analýzu velkých objemů dat včetně jejich stručného hodnocení.

1.4 Použité nástroje a metody

Při plnění úkolů jsem nejdříve využil teoretických metod. Jedná se především o studium odborné literatury, odborných diskuzí a fór a analýzy dostupných distribučních systémů.

Teoretické poznatky byly poté využity k provedení experimentu. Vytvořený modelový systém pro distribuci výpočetního výkonu a zpracování velkých dat bude otestován s ohledem na data z různých zdrojů.

2 Potřebné technologie

Při vypracovávání této práce bylo využito mnoho technologií a všechny se do této práce nevejdou, zmíním tedy jen ty podle mého názoru nejdůležitější.

2.1 Google Computing Engine (GCE)

Společnost Google a její inovativní technologie propojují denně mnoho lidí po celém světě, díky velmi vysokému využití jejích služeb si může účtovat reklamu od firemních partnerů a tu posléze distribuovat veřejnosti. Díky těmto možnostem a dostatečně velkému příjmu může investovat a vyvíjet další nové technologie a také je realizovat.

Google Computing Engine vychází z technologie claudového řešení distribuce výpočetního výkonu. Na pozadí tohoto claudu by mělo stát více než 1600 procesorů. Počátek tomuto procesorovému poli dala vniknout dohoda (7) mezi společností Google a společností IBM 8.10.2007, které se dohodly, že v rámci zlepšení dostupnosti a rozšíření nových vývojových metod pro paralelní programování a aplikace se škálovatelným zatížením, vytvoří hardwarové zázemí složené z několika stovek počítačů, které byly zkombinovány ze strojů Google, IBM BladeCenter a System x servers. Tyto servery běží na open source programech jako třeba Linuxové operační systémy, XEN systémové virtualizaci a Apache Hadoop projektu, což je open source implementace Googlem publikované výpočetní infrastruktury obzvláště MapReduce a Google File System (GFS).

2.2 Apache Hadoop na GCE

Společnost Google nabízí již předpřipravenou možnost spuštění Hadoop rozhraní jako „One click deployment“ v jejich webové aplikaci Cloud Launcher (8), která nabízí možnost spuštění i dalších rozhraní implementovatelné na GCE jako například Red Hat Enterprise Linux 7.1 nebo Windows Server 2013 R2. Tato aplikace nabízí již předpřipravená řešení, která usnadní první nasazení tohoto software do claudového prostředí a posléze jejich úpravu pro potřeby, jednotlivých aplikací a jejich požadavků.

Apache Hadoop je možné nasadit dvěma způsoby. Jedním způsobem je možné Hadoop nasadit přes Cloud Dataproc (9), což je Googlem vytvořená a spravovaná webová platforma pro management Spark a Hadoop služeb. Druhým způsobem je možné nastavit

a spustit Hadoop přes příkazovou řádku s pomocí bdutil (10), což je skript pro spouštění, správu a vypínaní Hadoop instancí.

2.3 Hadoop rozhraní

Hadoop rozhraní je připraveno pro zpracovávání velkých objemů dat, která přenáší přes všechny stanice. Tento software je rozšiřován a udržován společností Apache Software Foundation. Rozhraní běží na platformě JAVA a jejím programovacím jazyku a největším přispěvatelem do tohoto projektu je Yahoo! Inc.. Prvopočátkem tohoto projektu myšlenky, které vychází z nápadů prezentovaných společností Google Inc. v projektech Google MapReduce a Google File System. Tento nápad byl posléze tak zajímavý, že se z něj stal samostatný projekt pod názvem Hadoop Framework. V této sekci popíši základní principy fungování rozhraní Hadoop a MapReduce programového modelu, který vytváří návrh programovacího modelu pro Hadoop.

Hadoop není hlavním nástrojem pro uskutečnění této práce, ale slouží jako část infrastruktury, protože Pig je nadstavbou tohoto rozhraní a používá pod Hadoopem běžící MapReduce komponentu pro zpracovávání vytvořených skriptů. Přesto že bude MapReduce využit pro zpracovávání skriptů, tak žádná část není napsána jako program pro MapReduce a tedy Hadoop musí být na systému nainstalován ještě před tím, než se může vůbec nějaké hromadné zpracování dat odehrát. Pig slouží pouze jako abstraktní vrstva nad Hadoop rozhraním, pro kterou vytváří skripty, které MapReduce zpracuje.

2.3.1 MapReduce Programing Model

Pro porozumění funkčnosti Hadoop Frameworku se podíváme do historie jeho vzniku. Hadoop jako externí projekt pod Apache Software Foundation vzniknul po prezentaci nadějných nápadu na ze společnosti Google na konferenci ve městě San Francisco v roce 2004 a konkrétně téma „MapReduce: Simplified Data Processing on Large Clusters“ (5), který realizuje touto prací popsaný model pro zpracovávání velkých vstupních objemů dat na distribuovaném souborovém systému. Tento programový model využívá skupin počítačů pro rozdělení vstupních úkolů na jednotlivé počítače (nebo virtuální jádra), vypočítá mezivýsledky paralelně a ty se poté složí do konečného výsledku, který skupina vypočítá mnohem dříve než by se k těmto výsledku dopočítal serializací těchto vstupních úkolů jediný výkonný stroj.

Paralelizace zpracování vstupních úkolů přináší mnohé výhody, jako například možnost rozšíření výpočtů na teoreticky neomezené množství jednotlivých počítačů, drasticky kratší výsledný čas zpracovávání velkého objemu dat na jednou (podle velikosti skupiny) a zároveň možnost vyřešení nečekaného odstavení počítače z jakéhokoliv důvodu zatímco programové rozhraní odfiltruje od programátora starosti s dělením dat na menší části, plánováním procesů nebo vypořádání se s chybami hardwaru či softwaru.

Budu se zde věnovat shrnutí dokumentu „MapReduce: Simplified Data Processing on Large Clusters“ (5). Měl bych tím vytvořit jednoduchý přehled o MapReduce, popisující podrobnosti o programovém modelu stejně tak jako o realizaci společností Google. Společnost Google řešila problém zpracovávání velkého objemů dat v síti propojených počítačů. Aby řešení tohoto problému bylo dostatečně účinné, jako kritérium pro úspěšné řešení se stalo zvládnutí plánovaných úkolů jako dělení dat, distribuce nebo řešení problémů. Úspěšné řešení přineslo programátorovi možnost zaměřit se na psaní kódu aplikace pro daný úkol a ušetřením času věnovanému řešení plánovaných úkolů.

MapReduce vzniknul z nápadu použít mapovací a redukční funkce využívané v procedurálních programovacích jazycích. V těchto jazycích jsou tyto funkce aplikované na skupiny dat, takzvané Listy, což je v podstatě seznam objektů uložených v paměti. Výstupem funkce je nový List, na který je aplikovaná uživatelem definovaná funkce, jako například: “přičti ke každému číslu dané číslo“. Redukční funkce zde zase pomocí uživatelem definované funkce vybere elementy z Listu a vytvoří z nich nový datový objekt, který je dále zpracováván.

MapReduce implementace od Google je programována v jazyce C++. Rozhraní pomocí mapovací funkce definované uživatelem zpracuje vstupní data a mezivýsledky předá redukční funkci. Ta tyto výsledky zpracuje podle znovu podle zadaného způsobu a tímto se program dostane ke konečným výsledkům.

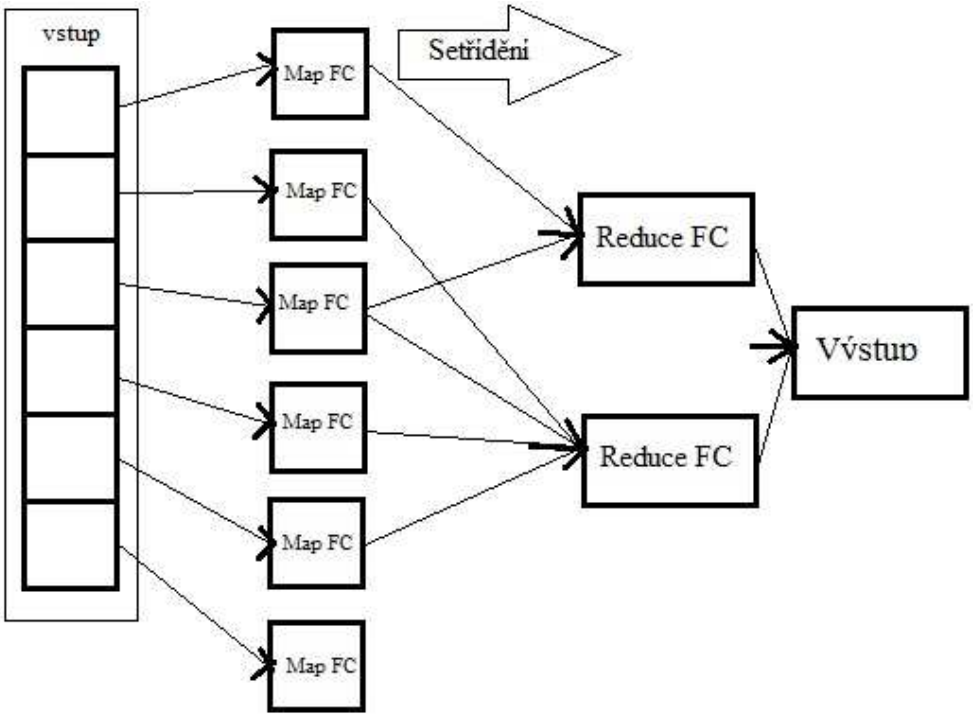
Celkové zpracování probíhá ve skupině počítačů, kde jsou rozdílné počítače přiřazeny k různým úkolům, některé zpracovávají mezivýsledky a jiné se dostávají ke konečným výsledkům, každý pracovává jinou část. Úkoly mezi ostatní rozděluje jedna hlavní instance programu zvaná „Master“ která se stará o veškeré pozadí plánovaných

úkolů jako distribuce dat po síti, rozdělování potřebných úkolů či řešení krizových situací jako například výpadek počítače zajištěním náhradního zpracování dat, který měl dotyčný počítač zpracovat. Ostatní instance programu jsou „Worker“ a jejich činnost je zpracovávat zadané úkoly od Mastera.

Master se stará o plynulý tok dat ze vstupních přes zpracování až na výsledky, shromažďuje mezivýsledky a ty rozděluje dále mezi Workery k redukci. Programátorovy používající MapReduce tedy stačí, aby nadefinoval dvě funkce, které budou požity na pro každý záznam v mapě a o zbytek běhu programu se již postará knihovna MapReduce zajišťující podrobnosti paralelizace, rozdělování úkolů a dalších.

Pro MapReduce se požívají vstupní data ve tvaru dvourozměrného pole definovaného jako (klíč;hodnota). V Mapovacím kroku jsou pomocí těchto hodnot vytaženy počáteční klíče a hodnoty ze vstupních dat a vytvoří se mezivýsledkové páry. Redukční krok je aplikován vždy na všechny páry se stejným mezivýsledkovým klíčem a vytvoří se konečné výsledky. Je možné napsat i další kombinační funkci pro mezivýsledkové páry.

Vstupní data jsou rozdělena typicky do X šestnácti až šedesáti čtyř Megabajtových částí. Program běží na skupině počítačů současně, kdy jeden ze skupiny pracuje jako Master a zbytek jako Worker. Master přiřazuje Mapovací a Redukční úkoly jednotlivým Workerům. Worker, který dostane úkol Mapovat, projde páry ze vstupních dat a vyhovující pošle dále na zpracování uživatelovo Mapovací funkcí. Mezivýsledkové páry vniknuvší Mapovacím krokem, jsou rozděleny funkcí definovanou uživatelem, rozděleny dál na Y částí. Mezivýsledky z tohoto kroku jsou shromažďovány v paměti a periodicky zapisovány na disk. Master poté pokyn pro Redukci zadá tak, že dotyčnému Workeru pošle umístění mezivýsledkových párů, na kterých má pracovat. Worker poté seřadí jednotlivé páry podle jejich klíče, a tedy páry se stejným klíčem jsou shromážděny u sebe. Postup MapReduce je znázorněn na obrázku č. 1.



Obrázek 1Paralelizace pomocí MapReduce (25)

2.3.2 Hadoop Implementace

Základní báze Hadoop projektu je tvořena ze tří menších projektů, které jsou velice těsně propojené. Jedná se o MapReduce, Hadoop Distributed File System (HDFS), Hadoop YARN a Hadoop Common. Dalšími součástmi poté určujeme zaměření použití, v době psaní této práce jsou k dispozici tyto: Ambari, HBase, Avro, Hive, Cassandra, Mahout , Chukwa, Pig, Spark, Tez a ZooKeeper. Některé ze jmenovaných projektů budou popsány později v osmé části (Příbuzné práce). Zde se podíváme, jak součásti HDFS a Hadoop YARN pracují.

Hadoop Distributed File System je souborový systém pro použití velkém počtu počítačů ve skupině. Je navržen tak, že jeho data jsou rozdělena na bloky a tyto bloky jsou rozdistribuovány mezi jednotlivé počítače a každý blok zkopírován a uložena na několika různých počítačích tak, aby se zajistila jeho dostupnost i v případě nedostupnosti počítače ze skupiny. Díky tomuto systému ukládání dat je HDFS schopný samostatně detekovat a automaticky opravovat chyby v datech, pokud by se vyskytly. Další nespornou výhodou a snížením nároků na síťový provoz a tím i na dobu zpracovávání je

vlastnost zadávat výpočty počítačům, které mají „blízko“ k datům, která mají zpracovávat (například jsou v daném počítači uložena).

Hadoop YARN by se dal označit jako takový motor celého soukolí. Stará se o plánování jednotlivých úkolů, rozděluje úkoly mezi jednotlivé stanice. Pokud vykonávání nějakého úkolu selže, je znovu naplánován. Jeho selhání také způsobí zastavení aktuálního průběhu (aby se zbytečně nevytěžoval systém, protože by se data stejně nepoužily. Přestože nápad Google je realizován v C++, je Hadoop realizován v Java.

2.4 Pig

Pig začínal jako projekt společnosti Yahoo! Inc., který převzala společnost Apache Software Foundation pod křídla Hadoop projektu. Později se ukázalo, že Pig má příliš velký potenciál na to, aby byl pouhý podprojekt a proto byl transformován jako jeden z hlavních projektů společnosti Apache v září 2010. Pig sehrál v této práci nejdůležitější roli a to hlavního tvůrčího prvku, se kterým jsem pracoval.

Pig slouží jako platforma pro rozsáhlé analyzování vstupních dat včetně jazyka pro jejich podrobný popis zpracování. Tento program byl vytvořen pro použití s MapReduce programovým modelem, tak jak je implementovaný v projektu Hadoop (4). Pig se již nezabývá konkrétními daty, slouží spíše jako unifikované rozhraní pro práci s informacemi agregovanými z dat, které pro Pig získá MapReduce. Tyto informace z agregačních funkcí (filtrování, počítání, vyhledávání v datech podle vzoru atd.) Pig zpracovává pomocí vlastního jazyka vyššího řádu, se kterým může stejné operace provádět nad různými daty právě díky „předzpracování“ kterým MapReduce zjistí z dlouhých textových souborů, jako jsou webové logy, RSS Feeds a podobně.

Díky tomu je programátor osvobozen od psaní Java MapReduce programů a může se soustředit na mnohem jednodušší psaní skriptů nebo dávkových souborů v jazyce Pig Latin pro splnění potřebných manipulací s daty. Pig přeloží následně skript nebo dávkový soubor, optimalizuje a vytvoří skupiny příkazů pro Hadoop, který poté vykoná požadované operace. Skripty a jejich překlad mohou být pouštěny i v lokálním prostředí bez paralelizace (2).

Podrobnější informace o funkčnosti budou rozebrány ve druhé části této práce.

2.5 RSS Standardy

V této práci jsou jako vstup použity RSS aktualizace, které obsahují shrnutí, co nového na dané webové stránce přibylo od poslední aktualizace. Formát RSS je standardizovaný podle formátu XML. V této části popisuji, jak by validní RSS mělo vypadat.

Zkratka RSS znamená Really Simple Syndication což by se dalo volně přeložit jako „velmi jednoduché shrnutí“. Web syndication (11) je termín požívaný v souvislosti s vytvořením nebo aktualizací webové stránky, kdy obsahuje souhrn všeho, co na stránce přibylo od poslední aktualizace. RSS je způsob, jakým zajistit, aby nebylo potřebné při návštěvě dané stránky procházet veškerý obsah, kterého na stránce může být velké množství. Díky RSS víme, který obsah na stránce přibyl a stačí tedy projít samotný nový obsah. Možnosti RSS využívá mnoho zpravodajských webů, webových blogů a spousta dalších stránek zabývající se sdělováním nových informací do světa. Tato služba slouží pravidelným uživatelům těchto webů jako navigační menu, které mohou využít pro rychlé vyhledání chtěného obsahu a ušetření času přeskočením proházení příspěvků, které například již četli.

Mnoho webů publikace RSS feed souborů používá a některé z nich budou vybrány jako vstup do této práce. Soubory obsahují stažená shrnutí z několika mnou vybraných stránek pro použití v praktické částí této práce.

XML jako Extensible Markup Language je jazyk, ze kterého standart RSS vychází. RSS standard popisuje, jaké XML tagy musí být v shrnujícím dokumentu přítomny a jak mají být čtenářem interpretovány. V době psaní této práce jako poslední standard pro RSS je verze 2.0.11, která byla vydána 30. března 2009 (12) a od té doby zatím nebyla aktualizována.

Základem RSS standardu je několik hlavních tagů, které jsou doplněny o další povinné či nepovinné segmenty. O některých těchto segmentech se rozepíši podrobněji v této části.

Hlavním tagem jakéhokoliv RSS dokumentu je umístění segmentu „rss“, pro který je povinný atribut „version“ který je v současné době vyplněn hodnotou „2.0“. Zároveň je pro segment rss povinná i přesně jeden segment „channel“ jehož povinnou součástí

jsou tři atributy, jsou to „title“, „description“ a „link“. Další segmenty a atributy jsou volitelné jako například „language“, „pubDate“ nebo „category“ které podrobněji specifikují nový příbytek v obsahu daného webu ze kterého je RSS feed stažený. Název nového souhrnu je obsažený právě v atributu „title“, dále v „description“ najdete stručný popis daného souhrnu a v atributu „link“ je uložen odkaz na stránky, ze kterých byl souhrn stažen.

K těmto třem povinným segmentům může být přidáno libovolné množství segmentů „item“, které tvoří vlastní jádro celého souhrnu. V tomto segmentu jsou popsány jednotlivé nové příbytky dané stránky. Vlastní atributy segmentu „item“ jsou stejně jako u nadřazeného kanálu „title“, „description“ a „link“ s jedním malým rozdílem a to že jsou nepovinné, stačí, aby byl přítomen jeden ze dvojce „description“ nebo „title“. Mohou zde být i další jako „author“, „category“ a „comments“. Atributu „author“ je posílán email na autora daného příspěvku, atribut „category“ v sobě drží zařazení do jedné nebo více kategorií, pod které článek spadá, v „comments“ je uložen odkaz na diskuzi k tomuto článku, popis nutného atributu je významově stejný jako u nadřazeného kanálu, tzn. „description“ obsahuje stručný popis publikovaného článku, avšak najdou se i výjimky, kdy je v tomto atributu článek celý. Toto rozhodnutí je ponecháno na autorovi RSS kanálu.

Zde následuje příklad validní RSS 2.0 feed.

```
<?xml version="1.0"?>
    <rss version="2.0">
        <channel>
        <title>Denni souhrn</title>
        <link>http://ukazkovyweb.cz</link>
        <description>Posledni update</description>
        <lastBuildDate>Tue, 01 Dec 2015 14:47:12
        GMT</lastBuildDate>
        <webMaster>admin@ukazkovyweb.cz</webMaster>
        <item>
            <title>Zavislost na PC hrach stoupa</title>
            <link>http://ukazkovyweb.cz/novinky/zavislost-
            na-pc-stoupa</link>
            <description>Vedci obevily souvislosti mezi
            hranim PC her a odtrzenim od reality
            </description>
            <pubDate>Fri, 01 Apr 2011 01:00:00
            GMT</pubDate>
        </item>
        <item>
            <title>Byl zabit vudce Islamskeho statu</title>
            <link>http://ukazkovyweb.cz/novinky/vudce-
            Islamskeho-statu-zabit</link>
            <description>Americka armada potvrdila uspesny
            utok na vudce Islamskeho statu</description>
            <pubDate>Sat, 02 Apr 2011 14:12:00
            GMT</pubDate>
        </item>
```

```
            <item>
                <description>Zatim neni co noveho
                publikovat</description>
                <pubDate>Wed, 02 Dec 2015 11:12:13
                GMT</pubDate>
            </item>
        </channel>
</rss>
```

3 Použití programu Pig

Tato část se zabývá návodem na používání programu Pig, spouštěním skriptů a dalšími věcmi potřebnými k úspěšnému zvládnutí úkolu, který je potřeba v programu vykonat ať už se jedná o procházení RSS souhrnů nebo jiných zadání, pro které je tento program vhodný. Dále budou též diskutovány různé způsoby k psaní a spouštění skriptů psaných v Pig Latin.

3.1 Pig Latin aneb jazyk pro Pig

Pro použití s programem Pig tvůrci vytvořily nový dotazovací jazyk, který se jmenuje Pig Latin. Je to dotazovací jazyk vyšší třídy podobný jazyku SQL (Structured Query Language), který umožňuje zadávání dotazů na relacích. Pig Latin skript je tvořen sérií příkazů kde u každý z nich má jednu ze tří funkcí. Tyto funkce pracují s daty a dji se shrnout jako načítání dat z jejich umístění, transformace dat do potřebné podoby a jejich uložení.

Pig Latin spoléhá stejně jako ostatní programovací jazyky na klíčová slova, která dají kompilátoru vědět, že toto je příkaz, který má něco vykonat. Klíčová slova, které v podstatě není možné vynechat pro jejich důležitou roli při zpracovávání dat, jsou mimo jiných, „UNION“, „GROUP“ a „FILTER“. Všechny ukázky kódu a popisy funkcí byly čerpány z tohoto zdroje (13).

LOAD 'data' [USING function] [AS schema];

Toto je základní syntaxe pro načítání vstupních dat. Sémantika napsané řádky nám říká: „Načti vstup ze souboru data v aktuální složce, použij k načtení tuto funkci a interpretuj data do tohoto schématu. “. Slova „LOAD“, „USING“ a „AS“ jsou klíčová slova a mají v této ukázce specifický význam. Klíčová slova „USING“ a „AS” je možné v této syntaxi vynechat a poté bude použita výchozí funkce „PigStorage“ a data nebudou dosazena do schématu.

PigStorage slouží jako výchozí uchovávající funkce. Popisuje, jako procházet vstupní data tak aby byla rozpoznána a stará se i o styl jejich ukládání. Tato funkce prochází vstupní soubor a s každou jeho řádkou zachází jako s jednorozměrným polem, které uloží do samostatné relace v databázovém pojetí oddělující jednotlivá pole

```
A = LOAD 'myfile.txt' AS (f1:int, f2:int, f3:int);

A = LOAD 'myfile.txt' USING PigStorage('\t') AS (f1:int, f2:int, f3:int);

DESCRIBE A;
a: {f1: int,f2: int,f3: int}

ILLUSTRATE A;
-------------------------------------------------------------
| a     | f1: bytearray | f2: bytearray | f3: bytearray |
-------------------------------------------------------------
|       | 4             | 2             | 1             |
-------------------------------------------------------------

---------------------------------------
| a     | f1: int | f2: int | f3: int |
---------------------------------------
|       | 4       | 2       | 1       |
---------------------------------------
```

Obrázek 2 Příklad použití finkce LOAD (23)

textovým řetězcem, který dostane jako vstupní parametr. Pokud je tento „string“ vynechán, je dosazena výchozí hodnota, což je „\t“, běžně známý jako tabulátor.

AS umožňuje využití pojmenování jednotlivých sloupců v polích a také určení datových typů, které budou pro tyto sloupce použity. V následujícím příkladu budou využity tři sloupce s názvy f1 až f3 a každý z těchto sloupců bude používat datový typ Integer.

LOAD 'myfile.txt' AS (f1:int, f2:int, f3:int);

USING jako volitelná položka syntaxe umožňuje programátorovi nadefinovat i vlastní funkce pro načtení vstupního souboru a leze tedy přizpůsobit pro načítání jakéhokoliv vstupního souboru, který je vhodný pro strojové zpracování (má jednoznačně definovaná pravidla pro jeho strukturu jako například XML). Na Obrázku č. 2 je názorně zobrazeno použití těchto dvou klíčových slov a zobrazení jejich výsledků.

Použití klíčového slova AS a pojmenování sloupců nám umožňuje později tyto sloupce adresovat právě pomocí těchto jmen, která jsou, pro jednotlivé sloupce zadána. Pokud však toto pojmenování vynecháme, musí zde být možnost jednotlivé sloupce adresovat i nadále. Tato možnost existuje v podobě programátory běžně používaného adresování pole, ve formě počítaní vzdálenosti od prvního sloupce daného pole, tzn. $0 pro první sloupec a o jedničku vyšší pro každý další.

Pig používá základní jednoduché datové typy a pár komplexních. Mezi jednoduché datové typy, které Pig používá, patří „int“ a „long“ pro celé čísla včetně znaménka, „float“ a „double“ pro čísla s plovoucí desetinnou čárkou. Dále je možné využít datové typy jako „bolean“ pro hodnoty pravda/nepravda nebo „datetime“ pro zaznamenání data včetně času. Mezi jeho jednoduché datové typy patří dva druhy polí, jeden se nazývá „chararray“ a slouží pro uchovávání textových řetězců ve formátu UTF-8, druhý je znám pod jménem „bytearray“. Podle definice v dokumentaci jde o blob, což si nejsem jistý, jak přesně by měly data uložené v tomto poli vypadat.

Komplexní datové typy používané programem Pig jsou označeny jako „tuple“, „bag“ a „map“. Tuple neboli pár označuje seřazenou sadu polí (např.: „(19,2)“). Bag označuje kolekci těchto párů (např.: „{(19,2), (18,1)}“) a nakonec do Map ukládáme páry klíčů s hodnotami (např.: „[open#apache]“). Tímto systémem by se daly tyto komplexní datové typy přirovnat k databázovému systému, kdy pár funguje jako řádek databáze a Bag jako tabulka. Je tu však jeden veliký a zásadní rozdíl oproti databázím a to v tom, že jednotlivé páry (Slovo pár je vyvozeno z překladu a nesouvisí se skutečnými páry ve smyslu počtu sloupců.) v Bag mohou mít různé počty sloupců, čímž není dodržen princip ACID používaný v relačních databázích.

Syntaxe pro Tuple:

(field [, field ...])

Syntaxe pro Bag:

{tuple [, tuple ...] }

Syntaxe pro Map:

[key#value <, key#value ...>]

Jak je vidět na příkladu syntaxí Tuple a Bag, jsou tyto komplexní datové typy připraveny na obsahování sebe sama, neboli uvnitř Tuple může být další vnitřní Tuple a Bag může obsahovat též vnitřní Bag, ty se poté rozlišují pomocí hranatých závorek, jak je uvedeno v syntaxi. V této práci pokud se budu zmiňovat o relaci, či tabulce, bude se mluvit o vnějších, tedy celých pytlích (Bag).

Dále si představíme několik operátorů, se kterými manipulujeme s daty, řadíme, případně říkáme, s čím mají funkce pracovat.

alias = FOREACH { block | nested_block };

Alias zde zastupuje název proměnné, podobně jako tomu bylo u schématu při načítání vstupního datového souboru a pojmenovávání jednotlivých sloupců podle schématu. Klíčové slovo FOREACH říká, pro každý prvek z relace vykonej danou

```
X = FOREACH A GENERATE *;

DUMP X;
(1,2,3)
(4,2,1)
(8,3,4)
(4,3,3)
(7,2,5)
(8,4,3)
```

Obrázek 3 Příklad použití funkce FOREACH (13)

funkci. Příklad použití je na následujícím obrázku.

X bude stejné s relací, která vznikne aplikováním funkce klíčového slova GENRARTE na každý prvek v relaci A, jednoduše by se dalo říci, že zkopírujeme A do X. GENERATE s parametrem hvězdičky totiž díky tomu, že je hvězdička zástupný znak vygeneruje to samé, co této funkci přijde na vstup.

Díky funkci FOREACH můžeme počítat, kolik prvků se v pytli nachází a tedy zjistit, kolik vnitřních párů, případně dalších vnitřních pytlů relace obsahuje.

Velice užitečnou funkcí je funkce FILTER, jíž použijeme v případě, že potřebujeme odfiltrovat nepotřebná data. Následuje syntaxe:

alias = FILTER alias BY expression;

Znovu se zde opakuje alias, který nám udává, jak se bude jmenovat nově vzniklá relace, dále klíčové slovo FILTER, poté jméno relace, ze které budeme filtrovat, další je klíčové slovo AS, jehož definice již byla zmíněna u funkce LOAD a jeho význam je zde stejný, včetně výrazu expression.

Občas je nutné vyfiltrovat data od opakujících se prvků, které by se nám mohli plést do dalších výpočtů, jako je například spočítání jedinečných prvků v dané relaci, při

němž potřebujeme opravdu pouze unikátní záznamy. Toto odfiltrování provedeme klíčovým slovíčkem DISTINCT v následující syntaxi:

```
alias = DISTINCT alias [PARTITION BY partitioner] [PARALLEL n];
```

Zde již po několikáté alias výsledného pytle, dále klíčové slovo DISTINCT a jeho parametrem je vstupní relace. Klíčové slovo „PARTITION BY“ nám udává, který rozdělovač má být použit, podobně jako klíčové slovo PARALLEL, jehož vstupem je počet redukčních kroků, kterým se může zvýšit paralelizmus, jsou tyto položky nepovinné a více je o nich napsáno ve zdroji (13).

Spolu s databázemi má Pig nejenom společné některé vlastnosti ale i klíčová slova, jedním z nich je klíčové slovo JOIN, které spojuje jednotlivé relace dohromady tak, jak se tomu děje u relačních databází včetně jeho variant LEFT, RIGHT, FULL atd. pro outer JOIN, ale není zde omezení použití jednoho typu slučování jako u MS-SQL databází.

Syntaxe pro vnitřní JOIN(inner):

```
alias = JOIN alias BY {expression|'('expression [, expression ...]')'} (, alias BY {expression|'('expression [, expression ...]')'} ...) [USING 'replicated' | 'skewed' | 'merge' | 'merge-sparse'] [PARTITION BY partitioner] [PARALLEL n];
```

Syntaxe pro vnější JOIN(outer):

```
alias = JOIN left-alias BY left-alias-column [LEFT|RIGHT|FULL] [OUTER], right-alias BY right-alias-column [USING 'replicated' | 'skewed' | 'merge'] [PARTITION BY partitioner] [PARALLEL n];
```

V každém sloučení (JOIN), máme na výběr z několika možností, které použít, všechny možnosti jsou znázorněny v syntaxi nahoře. Klíčová slovy „BY“ určují, co s čím se má spojit, tedy, co spolu souvisí a má být umístěno do jedné řádky. Další nové klíčové slovo USING má několik vstupních parametrů. Vstupní parametr „replicated“ nám umožňuje spojit relaci se sebou samou, tím, že se znovu zkopíruje. Pro tento parametr je jde použít pouze vnější levé sloučení. Vstupní parametr „skewed“ je podrobněji rozebrán v dokumentaci. Další parametr „merge“ slouží pro použití v situaci, kdy byly jednotlivé

tabulky již seřazeny jinou funkcí JOIN a bylo by tedy zbytečné znovu procházet fázemi řazení a třídění, což umožňuje značné ušetření výpočetního výkonu vynecháním několika průchodů skrz veškerá data, které spojované tabulky obsahují.

3.2 Běh programu Pig

Ve třetí části této práce bude řešen hlavní úkol této práce pomocí prostředků programu Pig. Aby ale toto mohlo být uskutečněno, je potřeba vytvořit jakýsi úvod do používání tohoto programu a jeho specifických technických detailů, na jejichž základech program pracuje. Tyto informace jsou čerpány ze zdrojů (13), (14) a (15).

Příkazy vytvořené v jazyku Pig Latin mohou být v programu Pig zpracovány pomocí skriptů nebo i spuštěny v případě potřeby pomocí podprogramu Grunt. Pig umožňuje několikero verzí módů, ve kterých poté pracuje. Uživatel si může vybrat mezi použitím lokálního módu, pouze s použitím lokálního výpočetního výkonu bez rozšíření o paralelizaci nebo použije mód s použitím MapReduce rozhraní a paralelizací zpracování úkolů.

Grunt funguje jako interaktivní příkazová řádka která je součástí programového vybavení prostředí Pig, kterou je možné spustit přímo z příkazové řádky pomocí následujícího příkazu.

$ pig

Pomocí tohoto příkazu dojde ke spuštění Grunt příkazové řádky, ze které je možné pomocí dalších příkazů provádět další operace.

Pokud chceme specifikovat spouštěný mód, ve kterém má Pig běžet uděláme to následujícími příkazy:

$ pig –x local

Pro spuštění v Lokálním módu bez paralelizace.

$ pig –x mapreduce

Tento mód je nastaven jako výchozí a tedy je jedno, jestli spustíte Pig s tímto vstupním parametrem nebo bez parametru jako je uvedeno několik řádek výše. Po spuštění je tedy spuštěn Grunt, který vypadá takto:

```
grunt>
```

V tuto chvíli je možné začít vkládat jednotlivé příkazy v jazyce Pig Latin a Grunt je začne jeden po druhém vykonávat. Pokud bychom chtěli tyto příkazy spustit najednou v dávkovém skriptu, existují pro nás dvě možnosti jak to provést.

1. V prostředí Grunt po spuštění zavolat příkaz „exec“ nebo „run“ a za něj zadat název skriptu např. „mujSkript.pig“ ve tvaru:

```
$ pig
grunt> run mujSkript.pig
```

2. Nebo rovnou při spouštění programu zadat jako vstupní nebo další vstupní parametr název skriptu, který se má po spuštění vykonat. Touto možností je ošetřena možnost ekvivalencí základního spuštění skriptu v obou případech.

```
$ pig mujSkript.pig
$ pig –x local mujSkript.pig
```

Dalším využitím vstupních parametrů může být zadání vstupních proměnných, které de dají použít ve skriptu jejich substitucí. Takto zadávané vstupní parametry by se zadávaly tímto způsobem:

```
-param promena1=hodnota1 -param promena2=hodnota2
```

Pokud bychom chtěli k těmto proměnným přistupovat ve skriptu, stačí použít jednoduchý řádek kódu v jazyce Pig Latin, který by vypadal podobně jako tento:

```
alias= FILTER data BY field = ‘$promena1‘;
```

Seznam těchto proměnných lze naimportovat i ze souboru, stačí tento soubor vytvořit a uložit do něj proměnné v tomto tvaru:

```
promena1=hodnota1
promena2=hodnota2
promena3=hodnota3
```

A poté už jenom specifikovat umístění a název tohoto souboru jako vstupní parametr při spuštění Pig

-param file mujSouborSParametry

Pro mě velice zajímavá možnost, co se týče zaujetí a využití v budoucnu, ne však v této práci, se ukázala možnost využití Hadoop projektu s nadstavbou Pig v externích Java programech, kdy pro jejich použití je potřeba pouze importovat jejich jar knihovny do projektu a poté už lze používat jejich třídy podle potřeby.

Pro používání Pig dotazů z Java aplikací musí být vytvořena instance „PigServer()“ třídy včetně nastavení pracovního módu, jestli má server běžet jako lokálně nebo využít MapReduce. Jednotlivé dotazy poté provádíme použitím funkce „RegisterQuery()“ na vytvořené instanci serveru a zadáním dotazu v Pig Latin jako argument použité funkce. Ostatní funkce používáme stejně jako přímo klíčová slova v Grunt, takže pokud chceme výstup dotazu uložit do souboru, zavoláme funkci „Store()“ z instance serveru a jako její parametry zadáme jméno a cestu, kam má být soubor uložen.

Zajímavá je i možnost použít vytvořené skripty i v externím Java programu zavoláním funkce „RegisterScript()“. Nakonec je tu ještě jedna, celkem důležitá informace, není to nic závažného, jen je potřeba si na to dá pozor. Pokud je Hadoop/Pig používán externě, tak dokud při procházení skriptu nenarazí na klíčové slovo „DUMP“ nebo „STORE“ v Pig Latin jazyce, tak neprovede žádnou operaci s daty, nic nenačte, nic nezpracuje.

Díky tomuto Application Programming Interface (API), je možné využití přímo ve webových aplikacích a dalších programech, které mají přístup na internet a mohou tedy komunikovat například právě s GCC a náročné úkoly zpracovávat tam.

3.3 Příklady použití

Abychom si udělali lepší představu, jak Pig pomocí Pig Latin pracuje. Představím zde kompletní skript na zpracování dat jako ukázku, na které si ukážeme, jak tento program pracuje. Ukážeme se zadání zpracování dat a vyřešení tohoto úkolu pomocí skriptu, včetně komentáře vysvětlujícího funkce programu, jeho vstupy a výstupy.

Vstupem pro tento vzorový příklad bude textový soubor, ve kterém budou uloženy ve formátu, kdy jednotlivé sloupce budou odděleny tabulátory, řádky budou jednotlivé řádky. V této databázi budou uloženy jména hráčů, her, které hrají a jejich

skóre v dané hře. Jméno souboru je „hernivysledky.in“ a příkladem by mohla být následující řádka.

```
Tom	Mario	1234
```

Zde máme jednoduchý příklad, jak by mohla být data uložena. Máme zde jméno hráče „Tom“, dalším záznamem je jméno hry, kterou hrál „Mario“ a dosažené skóre „1234“ v bodech.

Dalším vstupním souborem budou informace o hráčích, jméno souboru je „hraci.in“. Následuje příklad, jak by mohly být uloženy informace o věku.

```
Tom	19
```

V tomto případě je zase první jméno hráče „Tom“ a za ní následuje informace o věku oddělená tabulátorem „19“. Tato data jsou zpracována za účelem zjištění průměrného věku pro každou hru. Tato data jsou smyšlená a slouží pouze k demonstračním účelům.

Z těchto vstupních dat zjistíme, kdo nahrál nejvyšší skóre pro každou hru a jaký je průměrný věk hráče dané hry. Pro účely v reálném světě by tato studie mohla sloužit například pro potřeby vývojářů nebo developerů jako informace, na kterou věkovou skupinu mají danou hru zaměřit.

Následuje příklad, jak by takovýto skript mohl vypadat:

```
vysledky = LOAD '$vysledky_vstup' USING PigStorage('\t') AS
(jmeno:chararray, hra: chararray, vysledek: int);
herni_vysledky = GROUP vysledky BY hra;
top_herni_vysledky = FOREACH herni_vysledky
{	top_vysledky = ORDER vysledky BY vysledek DESC;
	top_vysledky_bag = LIMIT top_vysledky 1;
	GENERATE FLATTEN(top_vysledky_bag);	}
DUMP top_herni_vysledky;
hraci = LOAD '$hraci_vstup' USING PigStorage('\t') AS
(jmeno:chararray, vek:int);
vysledky_hrace= JOIN hraci BY jmeno, vysledky BY jmeno;
```

```
sdruzene_vysledky_hrace= GROUP vysledky_hrace BY hra;
prumerne_veky = FOREACH sdruzene_vysledky_hrace GENERATE
skupina,
AVG(vysledky_hrace.vek) AS vek:float;
DUMP prumerne_veky;
```

Ve skriptu „nejlepsi_vysledky.pig“ jsou použité dvě proměnné jako parametry z příkazové řádky, jedná se o „vysledky_vstup“ a „hraci vstup“. Tyto dvě proměnné jsou využity pro vložení dvou vstupních souborů. Pro spuštění s příkladovými vstupními soubory, které jsou umístěny v aktuální složce, bychom zadali takovýto vstup do příkazové řádky:

```
$ pig –param výsledky_vstup=hernivysledky.in –param
hraci_vstup=hraci.in nejlepsi_vysledky.pig
```

Zpracování proběhne následujícím způsobem. Program načte obsah souboru „hernivysledky.in“ a uloží si je do paměti jako Bag se jménem „vysledky“ a názvy sloupců od prvního: “jmeno“, „hra“ a „vysledek", první dvě budou typu „chararray“ a poslední typu „int“. Po načtení bude provedeno sdružení příbuzných záznamů nad sloupečkem výsledky podle jednotlivých her, což nám vytvoří relaci „hra_vysledeks“, která má v jednom sloupečku názvy her a v druhém vnitřní Bag, který obsahuje všechny záznamy, kde se vyskytuje název dané hry.

Dalším příkazem se výsledky jednotlivých her seřadí od nevětšího po nejmenší a pouze první pár od každé hry je vygenerován jako výstup. Příkaz „DUMP“ vypíše výsledky na obrazovku, pokud bychom chtěli výstup uložit do souboru, co by pro další využití bylo určitě vhodnější, použili bychom příkaz „STORE“ a výstup bychom tak uložili. Následuje příklad výstupu:

```
(Tom,Mario,1234)
(Pepa,Tetris,897)
```

Poté je načten druhý vstupní soubor se jménem „hraci.in“ pomocí zástupné proměnné ve skriptu a nyní se spojí „hraci“ a „výsledky“ dohromady v jeden celek, v jednu tabulku, kde jsou relace propojeny s použitím jména hráče. S tímto spojením je

možné určit, u které hry je jaký průměrný věk jejích hráčů. Stačí jednoduše použít funkci „AVG“, následuje příklad výstupu:

```
(Minecraft,22,5)
(Tetris,20)
```

Tento skript byl spuštěn na vytvořeném testovacím prostředí, popsaném v první části (GCC), na které byl nainstalován Hadoop a Pig ve verzi 0.15.0, poslední dostupná dokumentace je pro verzi 0.14.0 a na tuto dokumentaci jsou navázány zdroje v závěru této práce.

4 Překážky

Stránek, které poskytují RSS feeds pro své návštěvníky je opravdu mnoho a protože je toto naprosto běžným způsobem informování o novinkách, byly vytvořeny standardy pro toto informování, určující přesný formát, které tyto novinkové kanály musí splňovat, aby jejich novinky mohly být správně čteny programy na čtení a agregaci těchto kanálů. Tyto formáty zahrnují RSS a Atom standardy. Pro účely této práce, tedy analýzy uplatnění programu Pig a analýzy dat velkých rozsahů jsem použil RSS soubory jako vstup a metody a nástroje byly použity na těchto souborech.

4.1 Předpokládané výsledky

V této části se budu věnovat zkoumání otázek, které při analýze dat aplikací je potřeba zodpovědět. Tři různé druhy výsledku mohou být výstupem, ke kterému bychom se mohli dostat:

1. Seznam populárních slov v denních novinkách podle data
2. Počet u vícenásobného výskytu určitých slov v těchto článcích
3. Seznam novinek, které splňují hledané slovo nebo sousloví

První, co budeme v datech hledat, jsou výskyty slov v novinkách podle data, abychom si udělali představu, jaká slova se běžně vyskytují při publikaci zpráv. Dalším krokem bude sledovat tyto trendy v průběhu času, jak se tyto počty mění. Pokud nastane například více medializovaná událost, jako například nešťastné události, přírodní katastrofy či politické přešlapy, měly by se postupně slova, spojená s touto událostí, stát častější ve zveřejňovaných zprávách. Postupem času, jak informování o těchto událostech odborně „vyšumí“, měly by se počty slov spojované s těmito událostmi zase snížit. Aplikace, analyzující tyto datové proudy, by měla tento jev zachytit a po použití dotazů odpovědět seznamem nejfrekventovanějších a zároveň správných slov ve zprávách pro daný den za určité období.

Abychom se mohli detailněji dotazovat a provést hlubší analýzu chování těchto trendů, potřebujeme zjistit jejich funkcionalitu, jak často se některá slova vyskytují dohromady v různých novinkách podle data. Program přijímá jako vstup ve formě parametrů libovolný počet slov a spočítá všechny příspěvky podle data, kde se tyto slova vyskytují dohromady. Dále jsou tyto výsledky sloučeny do skupin podle zdroje, ze

kterého pocházejí, podle čehož zjistíme, jak který zdroj toto téma pokrývá v celkovém spektru zdrojů a ukáže rozdíly mezi jednotlivými zdroji.

Můžeme též vyhledávat podle regulárního výrazu, kdy již nezískáme pouze seznam slov, která se nejčastěji opakují, ale přímo daný příspěvek. Uživatel pouze musí vytvořit regulární výraz, podle kterého se má článek vyhledat. Toto umožní uživateli full-textové vyhledávání.

4.2 Omezení

Řešení výsledků této práce není mířeno na vyřešení všech problémů s automatickým zpracováním webových kanálů, ale pouze slouží jako příklad použití programu Pig. Proto jsem určil určité mantinely, abych se mohl soustředit na jádro problému a neřešil méně důležité detaily.

Pro zjednodušení analýzy vstupů bylo použito pouze RSS kanálů, aby nebylo nutné upravovat algoritmy pro zpracování různých druhů dat (např. Atom). Dále některé stránky nedodržují standardy RSS úplně do detailu, takže pouze stránky, které se jich drží, byly vybrány k analýze.

RSS specifikace neurčuje, jestli se v „description“ bude posílat pouze stručný popis nebo článek celý, toto rozhodnutí je ponecháno na vydavateli onoho kanálu. Přesto, že by zpracování celého článku přineslo více světla na danou událost, přineslo by to více problémů než užitku. Několik problémů zde zmíním pro představu. Každá webová stránka má vlastní design, který může být naprosto odlišný od jiné, tudíž by bylo potřeba nejdříve každou stránku manuálně analyzovat a připravit pro ni vstupní rozhraní, abychom mohli extrahovat pouze chtěný článek s vynecháním ostatních článků a třeba i reklamy, která se na zpravodajských webech vyskytuje velice často. Navíc toto rozhraní by bylo závislé na konkrétním designu, a pokud by se tento design změnil, bylo by potřeba upravit i toto vstupní rozhraní. Možná by v tomto případě mohlo být využito technik strojového učení, ale toto není v zaměření této práce. Z těchto důvodů bude analyzován pouze výstup z RSS kanálů.

Dalším neméně vážným problémem k zamyšlení je použití různých jazyků. Všechny použité RSS kanály webových stránek jsou pouze v jednom jazyce, protože při použití více jazyků by bylo potřeba překládat jednotlivá klíčová slova nemluvě o jejich rozdílném

významu, tedy rozdílech v sémantice, kdy každé slovo nemusí mít smysluplný překlad do všech použitých jazyků, aby mohli být výsledky relevantní, bylo vybráno pouze několik zdrojů. I když by teoreticky mohlo být použito překládacího modulu, není to v zaměření této práce.

Přesto, že je možné agregovat zprávy z různých časových zón, byla použita pouze jedna časová zóna pro minimalizování indiferencí rozdílného času vydání novinek v kanálu a všechny časy byly akceptovány jako jedna veliká společná časová zóna. Více informací o tomto problému bude rozebráno v kapitole 8. Pokud by nebylo prozatím použito zdrojů z jedné časové zóny, chyba by přesto nebyla příliš velká, protože okno pro tuto chybu je pouze 24 hodin, tedy od -12 do +12 hodin od UTC.

5 Nasazení platformy Pig

Data zpracovávaná v této práci a celé prostředí běží na GCC od společnosti Google Inc. Tato část obsahuje podrobný návod na přípravu prostředí a nastavení všech součástí pro potřeby této práce.

5.1 Nastavení prostředí

Abychom mohli začít s přípravou a realizací claudového prostředí, ve kterém budeme zpracovávat data, musíme nejdříve provést registraci u společnosti Google. Na konci této sekce najdete seznam potřebných informací, bez kterých není možné začít tyto služby využívat. Abychom mohli začít využívat služeb GCC, budeme potřebovat účet u společnosti Google.

Začneme založením emailové schránky. Do internetového prohlížeče zadáme adresu www.google.cz, v pravém horním rohu klikneme na odkaz „Gmail“. Otevře se nám stránka accounts.google.com, kde klikneme na odkaz „Vytvořit účet“. V tuto chvíli jsme přesměrováni na stránku „Vytvoření účtu Google“. Po levé straně můžeme zjistit, k čemu všemu nám může být účet Google prospěšný, jaké všechny služby můžeme využívat. Na pravé straně sídlí tabulka, kde doplníme kolonky Jméno, Příjmení, do další kolonky vyplníme zvolené uživatelské jméno. Následující kolonka bude obsahovat heslo, které musí splňovat délku alespoň 8 znaků. Při kliknutí do této kolonky obdržíme doporučení, abychom nepoužili heslo z jiného webu nebo snadno předvídatelné slovo z bezpečnostních důvodů, je dobré na tuto bezpečnost myslet již dopředu, neboť díky tomuto účtu můžeme využívat velké množství služeb a některé jsou placené, nemluvě o další službách či dalších věcech navázaných na emailovou adresu. Kolonka pod touto je vyžadována pro ověření hesla, aby bylo zajištěno, že víme, jaké heslo jsme zadali a pamatujeme si ho. Následuje řádek pro datum narození složený ze tří polí, jsou pro den, měsíc a rok. Další řádek je pro telefonní číslo, sice v tuto chvíli není vyžadováno, ale souží k dodatečnému zabezpečení Vašeho účtu, doporučuji vyplnit. Zbývá už jen Capcha pro ověření, že toto není účet vytvářený automatizovaným nástrojem a místo, tedy stát, ze kterého je účet zakládán. Dalším krokem je potvrzení souhlasu s ochranou soukromí a smluvními podmínkami. Po odsouhlasení jsme přesměrováni na přivítání, ze které ho pokračujeme tlačítkem „Pokračovat do služby Gmail“

V době psaní této práce společnost Google nabízí dotování jejích služeb na Google Cloud Platform (GCP) (16) v hodnotě 300$ (17), pro využití této dotace je potřeba přihlášení ke svému účtu u společnosti Google a doplnění několika dalších údajů do registračního formuláře.

Na stránce Google Cloud Platform (16) klikneme vpravo nahoře na modré tlačítko „Free-Trial“, které nás přesměruje na stránku registračního formuláře. Po pravé straně jsou zobrazeny dodatečné informace o službě a nejčastěji kladené dotazy. Po levé straně začneme vyplňovat registrační formulář. Jak jsem již zmínil dříve, tato platforma slouží pro komerční účely a je orientována na podnikatelské subjekty, to sice nevylučuje použití pro osobní nebo studijní účely, ale nenabízí služby zdarma, tedy všechny služby jsou placené, a i když využíváte dotovaný kredit, přijde Vám faktura.

Prvním polem formuláře je země, tato země je předvyplněná podle volby země z registrace účtu Google, ale je možné zvolit jinou. Následující pole obsahuje měnu, tato měna je v tuto chvíli pevně stanovená na Americký dolar a Google nenabízí její změnu. Další pole se jmenuje „Tax information“ a slouží k vyplnění daňových údajů. Pokračujeme vyplněním obchodního jména, které se objeví na vystavené faktuře (obchodní jméno, jméno, adresa, PSČ, město) a hlavním kontaktem pro společnost.

Poslední dvě části se věnují způsobu platby za služby a komunikačnímu jazyku v platebním styku. Pro aktivaci účtu na Google Cloud Platform je potřeba vlastnit bankovní účet s debetní nebo kreditní kartou a povolenými platbami na internetu. Příprava takového účtu je mimo zaměření této práce. Zadáme číslo karty z její přední strany do prvního pole, další dvě pole zaplníme měsícem a rokem konce platnosti karty ve tvaru MM/RR a následující posle slouží k zadání bezpečnostního kódu uvedeného z druhé strany karty, který je na druhé straně, aby nebylo možné jednoduše zaznamenat pouze jednu stranu karty a již kartu zneužít. Poslední pole platebních údajů, pokud je zaškrtnuté pole, které uvádí, že adresa u karty je shodná s adresou uvedenou výše je připraveno pro zadání jména držitele karty. Následuje výběr komunikačního jazyku v platebním styku a potvrzení souhlasu a přečtení „Google Cloud Platform Free Trial Terms of Service“ (18). Celý formulář odešleme kliknutím na modré tlačítko „Accept and start free trial“ hned pod formulářem.

Pokud je vše v pořádku, jsme přesměrováni do úvodní obrazovky GCP a dotázáni na název našeho projektu (pokud je toto náš první projekt). Pokračování najdete v části nastavení nástrojů.

5.1.1 Nezbytné informace potřebné pro přípravu prostředí

Jedná se o informace, které jsou nezbytné, a bez těchto informací nelze v registraci pokračovat, ostatní informace zmíněné v textu výše jsou jen doplňkové a mají pouze vliv na komfort, bezpečnost či jiné aspekty používaní zmíněných účtů.

5.1.1.1 Informace pro založení účtu Google

- Jméno a Příjmení
- Uživatelské jméno
- Heslo
- Stát

5.1.1.2 Informace pro registraci v Google Cloud Platform

- Stát
- Obchodní jméno a/nebo Jméno
- Fakturační adresu
- Hlavní kontakt
- Číslo platební karty, její platnost a bezpečnostní kód a jméno držitele

5.2 Nastavení nástrojů

Nastavení nástrojů lze upravovat po úvodním přihlášení do GDC. V levém horním rohu, vedle názvu se nachází tři vodorovné čárky, které jsou vstupem do menu konzole, kde se provádí potřebné nastavení.

5.2.1 Google Compute Engine

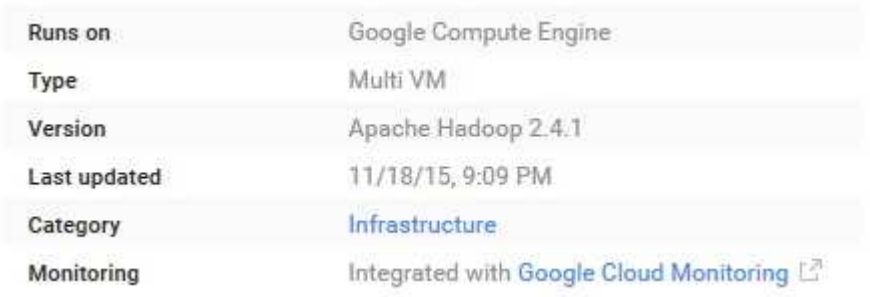
Pro nastavení virtuálních strojů, na kterých poběží budoucí analýza, nastavujeme kliknutím na menu konzole a v sekci „COMPUTE“ vybereme možnost „Compute Engine“ kdy se nám otevře ovládací panel pro vytváření a spravování výpočetního systému. Na levé straně jsou uvedeny jednotlivé sekce jako „VM instances“, „Disks“, „Metadata“ a další. Zvolíme možnost VM instances, pokud by nebyla načtena jako výchozí. První, na co nám padne pohled, je graf využití CPU v procentech. Nad tímto

grafem se nachází menu pro ovládání virtuálních instancí (vytvoření, spuštění, restart atd.) klikneme na vytvoření (New instance), následujícím formuláři zvolíme jméno, geografickou zónu, v jakém data centru mají servery běžet fyzicky, druh virtuálního stroje. Pro zkušební účet existují omezení na maximální možnost 2 virtuálních jádry s několika různými možnostmi nastavení od 3,5 GB paměti pro jádro případně více paměti nebo rychlejší procesor, po vylepšení druhu našeho účtu je možné využívat až 32 jádrové virtuální procesory. Formulář pokračuje nastavením boot-disku, kde máme na výběr z širokého výběru od linuxových systémů až po Windows server 2012 R2 DcE. U nastavení boot-disku najdeme i možnost výběru, jestli bude disk SSD nebo plotnový. Už zbývá jen nastavení Firewallu a nastavení oprávnění pro jednoduší verzi vytvoření virtuálního stroje, pro složitější následuje rozklikávací odkaz na další nastavení managementu, sítí a přístupu. Tuto nastavenou instanci můžeme buď vytvořit tlačítkem „Create“ nebo zrušit tlačítkem „Cancel“. Pro použití Hadoop na Google Computing Engine není potřeba vytvářet samostatnou instanci, tento popis slouží pouze pro představu vlastnoruční přípravy, viz níže.

5.2.2 Apache Hadoop

Apache Hadoop na virtuální prostředí GDC nasadíme snadno díky předpřipravenému řešení od společnosti Google, které sice nepoužívá nejnovější verzi Apache Hadoop hned po vydání, ale zkušenosti z komerčních systémů ukazují, že komerčně nabízené aplikace jsou pozadu s nasazováním nových verzí z důvodů testování stability nové verze a ta je nabízena ověření stability aby se zajistila požadovaná kvalita nabízených služeb, přesto tedy starší verze není na škodu.

Začneme kliknutím na menu konzole a vybereme možnost „Cloud Launcher“. Objeví se nám rozhraní Cloud Launcher, kde máme možnost spustit v tuto chvíli až na 161 různých projektů a tyto projekty můžeme prohledávat zadáním části názvu do vyhledávací řádky. Pokud nevidíme projekt Apache Hadoop na první stránce, zadáme do vyhledávání „Hadoop“ a klikneme na nalezenou položku. Na stránce, která se načte po kliknutí a zobrazí nám stručné informace o projektu a odkaz na stránky vývojáře (Apache Software Foundation) a dále důležité modré tlačítko „Launch on Compute Engine“ jehož zmáčknutím začíná celá procedura.



Obrázek 4 Informace o Apache Hadoop při realizaci praktické části (24)

Jsme přesměrováni na formulář nasazení Apache Hadoop na GCP v DGC, prvním polem formuláře určíme jméno nasazovaného projektu, které musí být unikátní mezi všemi ostatními projekty. Druhé pole obsahuje geografickou lokaci serveru, k dispozici jsou volby střední a východní Amerika, západní Evropa a východní Asie. Abychom mohli používat distribuované úložiště, které nemá uniformní jednotku (disk či pole), kromě velikosti, byl zaveden pojem „bucket“ který označuje datový prostor, který obsahuje nesetříděné objekty a má určitou velikost definovanou při vzniku (19). Vybereme tedy požadovaný „kbelík“, tedy pokud existuje, pokud ne, vytvoříme ho přes „Create new bucket/CREATE BUCKET“ pomocí jména, druhu dostupnosti (SLA) a umístění. Další dvě části se věnují nastavení virtuálních strojů, nejdříve určíme, kolik pracovních strojů chceme využívat, poté zvolíme jejich typ podobně jako u GCE, jen s tím rozdílem, že zde je možné využít až 16 jádrové virtuální procesory. V druhé části nastavujeme Hadoop Master stroj, který se bude starat o správu a přidělování úkolů. Posledním krokem je kliknutím na modré tlačítko „Deploy Apache Hadoop“. Operační systém, který mý Google připravený pro toto řešení je Debian 7 backports.

5.2.3 Java Runtime

Aplikace Pig je napsána v jazyce Java a je tedy pro její běh nezbytné mít nainstalovanou podporu pro tento jazyk. Pig vyžaduje instalaci verze Java6, kterou získáme ze software repository, což je takový softwarový sklad pro Linux. Pokud ještě nemáme Javu vůbec, tak nejsnadnější cesta vede přes příkaz zadaný do příkazové řádky.

```
$ sudo apt-get install openjdk-6-jdk
```

5.3 Instalace aplikace Pig

Ve chvíli, kdy máme připravenu podpůrnou infrastrukturu, můžeme začít s instalací aplikace Pig. Aplikaci nainstalujeme na jednotlivé stroje pomocí několika příkazů do příkazové řádky.

```
$ wget ftp://mirror.hosting90.cz/apache/pig/pig-0.15.0/pig-0.15.0.tar.gz
```

Jakmile je soubor stažen, můžeme ho rozbalit dalším příkazem.

```
$ tar xvf pig-0.15.0.tar.gz
```

Spouštěcí soubory jsou nyní umístěny ve složce /home/canek_t/Pig/pig-0.15.0/bin, abychom mohli Pig začít využívat, potřebujeme spouštěcí soubor přidat do systémové proměnné PATH, to učiníme příkazem, který musí mít jako vstupní parametr úplnou cestu ke spouštěcímu souboru.

```
$ export PATH=/home/canek_t/Pig/pig-0.15.0/bin:$PATH
```

6 Popis použití platformy Pig

6.1 Sběr dat

Data pro otestování využití platformy Pig byla stažena z několika webů pro možnost porovnání pokrytí mezi jednotlivými tématy. Sběr jednotlivých RSS souborů probíhal vyhledáním zdrojové adresy pro daný soubor webové stránky na dané stránce. Tento soubor byl poté stažen a uložen pod názvem webové stránky, ze které byl stažen. Stránky byly vybrány s ohledem na podobnost dodržování standardu, aby nebylo nutné provádět větší úpravy kódu. Některé druhy stránek mohou využívat jiného značení data vydání nebo ho nepoužívat vůbec.

6.2 Zpracování dat

Zpracování dat je provedeno manuálním zadáváním pokynů do příkazové řádky ve virtuálním stroji na GCP.

Hlavní úkoly pro zpracování dat

1. Separace jednotlivých novinek z XML zprávy
2. Rozebrání jednotlivých zpráv na jednotlivé segment
3. Naformátování a vyčištění vstupních dat
4. Vytvoření relací jednotlivých slov ke správám, v nichž se nacházejí

Abychom mohli začít zpracovávat data, je třeba je nejdříve připravit, prvním úkolem je analýza struktury dat a adaptace skriptu, který tento vstupní soubor načte z jeho původní podoby a upraví tento vstup na jednotlivé segmenty, po rozložení vstupu na jednotlivé novinky pokračujeme rozkladem na jednotlivá slova a jejich přiřazení k jednotlivým zprávám, tyto slova jsou ještě mezitím odfiltrováno od běžně používaných slov, jako jsou spojky a další doplňující slova (jedná se o slova, která určují co, kde, jak atd.).

Pokud bychom však použili výchozí načítací funkci, byl by každý řádek souboru uložen jako jedna relace což se nám, ale při čtení XML souboru moc nehodí, protože pokud bude například pole „description“ rozmístěno na více řádcích, pak nám je tato načítací funkce rozdělí do více relací a pro účely této práce nepoužitelné. Přesto, že Pig

ve verzi 0.15.0 nepodporuje přímé načítání dat z XML souboru, přesto v knihovně uživateli definovaných funkcí jedna existuje. Vstupní soubor bude načten s použitím funkce „XMLLoader“

```
zpravy =  LOAD $soubor1' using
org.apache.pig.piggybank.storage.XMLLoader('item') as
(zprava:chararray);
```

Dalším postupem jsou data z jednotlivých zpráv rozebrána na menší segmenty, jako jsou title, link a description.

```
zdrojA = FOREACH zpravy GENERATE   REGEX_EXTRACT(item,
'<link>(.*)</link>', 1) AS link:chararray,
REGEX_EXTRACT(item, '<title>(.*)</title>', 1) AS title:chararray,
REGEX_EXTRACT(item, '<description>(.*)</description>', 1) AS
description:chararray,
'$zdrojA' AS source:chararray;
```

Jednotlivé sloupečky budou uloženy, jak je naznačeno výše ve skriptu, tedy ve tvaru:

```
zpravy: {link:chararray, title:chararray,description:chararray,
source:chararray}
```

V tuto chvíli máme načtená všechno potřebná data pro analýzu, jen je potřeba je trochu upravit. V tuto chvíli neexistují duplikátní data, takže je není třeba odstraňovat, avšak v reálné použití by to bylo nezbytné. Zbývá tedy rozložení textů na jednotlivá slova. Toho je docíleno Funkcí TOKENIZE a jejím použitím na každý description.

```
slovnik = FOREACH zpravy GENERATE
flatten(TOKENIZE(LOWER(zprava))) AS slovo:chararray, link,
source;
```

6.3 Analýza dat

Jakmile máme data zpracována a připravena pro analýzu můžeme pokračovat s následujícími kroky.

Potřebné úkoly pro analýzu

1. Počítání výskytů všech slov ve zprávách
2. Počítání dalších výskytů slov zadaných uživatelem

V obou těchto případech využijeme slovníku, jež jsme si vytvořili v předchozím kroku. Abychom měli funkční počítaní slov a nevyskytovaly se zde nadbytečná slova, která se běžně vyskytují v projevu, jako jsou předložky či spojky, použijeme regulární výraz, abychom je odfiltrovali. Všechny slova ve slovníku jsou poté zkontrolovány a nadbytečná slova jsou odfiltrována. Poté se spočítají jednotlivé výskyty slov.

Pro účel počítání slov zadaných uživatelem vybereme záznamy, které odpovídají slovu zadané uživatelem. Tyto slova by mohla být sledována po vytvoření slovníku nebo rovnou při procházení XML souboru pomocí regulárního výrazu, čímž by se ušetřil výkon nutný k vytvoření slovníku.

6.4 Textové vyhledávání

Textové vyhledávání může být využito na vyhledávání konkrétních příspěvků, které obsahují určitý text, který je možné vyhledat pomocí regulárního výrazu. Toto vyhledávání je umožněno jednoduchým skriptem, který nám umožní vyhledat zprávy, v jejichž title nebo description výraz odpovídá zadanému regulárnímu výrazu.

```
vysledek = FILTER slovnik BY title MATCHES '$vyraz' OR
description MATCHES '$vyraz';
```

Datová analýza nám dává možnost zjistit, jak se vyvíjejí trendy v jednotlivých tématech, ale textové vyhledávání nám vyhledá něco ke čtení.

7 Příklad použití

V této část se budu věnovat postupu při analýze datových vstupů z pohledu uživatele použitím nástrojů již zmíněných nástrojů. Některé výsledky budou předvedeny včetně procesu jejich získávání z jejich zdrojových zpráv.

7.1 Datové sady

Datové sady byly staženy ze dvou českých webů pro eliminaci časových posunů, které tak není nutno řešit a pro podobnost interpretace RSS standardu, kdy v jejich výstupních kanálech bylo minimum změn, ostatní české zpravodajské kanály používají buď ATOM, nebo vykazují přílišné odlišnosti od zpracovávaných kanálů.

1. Novinky.cz – http://www.novinky.cz/rss/
2. Ihned.cz - http://www.ihned.cz/rss/

Kanály stažené z těchto stránek jsou použity jako datový základ pro testování.

7.2 Určení často se opakujících slov

Pro určení často se opakujících slov s vynecháním běžně používaných slov spustíme skript poprvé na daných datech s z celkového seznamu slov poté vybereme slova, která se běžně používají a upravíme regulární výraz pro jejich odfiltrování. Program Pig spustíme příkazem ze složky se skriptem a testovacími daty.

```
$ pig –x mapreduce –param soubor1=ihnedcz-
13.12.20152145.xml –param soubor2=novinkycz-
13.12.20151630.xml ReadXml.pig
```

Tím to se data zpracují a výstupem je tabulka s počtem slov v aktuálních souborech.

```
aut,        4
auto,       4
demonstrace,        1
president,          3
presidenta,         4
```

Jak je vidět, z příkladu vstupu není nejvhodnější jazyk pro neupravené použití, které by se dalo upravit pomocí dalšího regulárního výrazu kterým bychom ošetřili předpony a přípony se základem slova.

7.3 Určení aktuálních trendů

Abychom mohli určit aktuální trendy, bylo by potřeba provádět sběr vstupních dat na základě pravidelného intervalu, tak abychom neměli zbytečně duplikovaná data a zároveň nám některé zprávy neunikly a toto časové okno je velice obtížně určit. Použitím většího množství získaných informaci ze vstupů a jejich svázání s datem jejich vydání získáme přehled, jak jsou jednotlivá témata oblíbená v médiích při využití počtu výskytů jednotlivých slov. Pokud bychom určily, které slovo jednoznačně označuje téma, mohli bychom na tomto základě vytvořit i statistiku.

7.4 Vyhledávání zpráv

Vyhledávání v textu je užitečné, pokud chceme vyhledat konkrétní článek na základě již nalezených slov nebo zadáním slov, která spolu mají mít souvislost. Spuštěním vyhledávacího skriptu je prohledaná celá báze dat, jestli se v ní nachází výskyt hledaného výrazu. Od uživatele je specifikován vstup, který je vložen do regulárního výrazu, přes který je vyhledán odkaz na hledaný článek.

Příkladem použití by mohlo být, že po určení trendu by nás zajímalo, co konkrétně bylo o daném tématu napsáno, v tomto případě zadáme klíčová slova a po vyhledání nám budou odpovědí odkazy na jednotlivé články, v nichž se zadaná slova vyskytují. Například při vyhledávání slov „Demonstrace“ a „prezidenta“ bude nám odpovědí odkaz na článek o demonstraci na podporu prezidenta, který se ve vstupních souborech vyskytuje.

http://domaci.ihned.cz/c1-64971770-demonstrace-na-podporu-prezidenta-se-zucastnilo-asi-tisic-lidi-zahrala-na-ni-i-kapela-ortel?utm_source=mediafed&utm_medium=rss&utm_campaign=mediafed

8 Vyskytnuté problémy

Kapitola popisuje různé překážky, které bylo potřeba vyřešit při vývoji systému analýzy. Jejich řešení či náhradní řešení, které bylo použito, bude též diskutováno.

8.1 Frekvence sběru datových sad

Pro sběr dat je důležité vybrat správnou dobu, po které by mělo stažení být zopakováno. Tento časový interval by měl být vybrán tak aby dva soubory, po sobě stažené měly minimální pravděpodobnost, že budou stejné, ale zároveň nové zprávy by neměly být opomenuty.

Ve skutečnosti vyloučení stejných záznamů není až tak jednoduché, jak se zdá. Různé stránky vydávají novinky do svých kanálů s různou četností, a tedy by stahování z různých kanálů mělo být vyřešeno individuální pra každý zdroj. RSS standard uvádí nepovinný segment „ttl“ neboli „time to live“, který uvádí, kolik času, má uplynout v minutách, než by měl být daný kanál stažen znovu. Přestože i kdybychom zavedli využití segmentu ttl, není problém duplikace dat odstraněna, protože kanály jsou doplňovány o nové zprávy po malých krocích, které nenahradí všechny zprávy v daném kanálu, ale pouze přidají první a odeberou poslední příspěvek ve frontě „FIFO“. Navíc pouze některé kanály tento segment používají.

Pro účely této práce bylo staženo několik kanálů, ze kterých byly extrahovány zprávy pro účely otestování skriptů a spočítání četnosti jednotlivých slov. Ideální řešením, které by problém vyřešilo, by bylo stahování zpráv v krátkém časovém intervalu, čímž by se zamezilo ztrátě dat a vytvořením databáze, do které by se zaznamenávaly nové zprávy, čímž by mohla být duplicita odstraněna před přidáním do databáze a nebylo by tento výpočetně náročný úkon provádět dodatečně. Vybudování takovéto aplikace je však nad rámec této práce.

8.2 Procházení XML

Jeden z problémů, který se vyskytl při budovávání, jak již bylo řečeno dříve, RSS je formát založený na XML (12) a přestože je Pig vhodný pro zpracovávání vstupu ve tvaru řádek textu s poli rozdělenými určitými řetězci a také může extrahovat data ze

vstupních řádek s pomocí předpřipraveného formátu při použití regulárních výrazů, tak XML data nepatří ani do jednoho z těchto případů.

V kolekci uživatelem definovaných funkcí nazývající se „Piggybank“ je sice možné použít funkci „XMLLoader“ která je určena pro čtení dat z XML, tato funkce vezme jako vstupní parametr textový řetězec, který reprezentuje XML tag, který má být ze vstupního souboru načten. Háček je v tom, že tato funkce načte všechny výskyty tohoto tagu do jednoho sloupce jako jednotlivé řádky včetně počátečního a koncového tagu, ale již nerozlišuje jednotlivé další vnořené XML tagy, a tím pádem automaticky nezískáme stromovou strukturu pomocí vnořených relací.

Jednotlivé elementy jsou tedy vnořené v textovém řetězci a je třeba je z tohoto řetězce vyextrahovat do párů jednotlivými příkazy. Pig nemá žádnou vestavěnou funkci pro extrahování obsahu z jednotlivých XML tagů pro vytvoření stromové struktury a dokonce ani v Piggybank takováto funkce neexistuje, proto padlo rozhodnutí, že pro vyhledávání v jednotlivých segmentech XML použiji regulárních výrazů zadávaných do funkce „RegexExtract“. Procházení XML pomocí regulárního výrazu je nepopsatelně zpátečnický krok, protože formát XML je navržen pro rychlé a snadné orientování, avšak Pig neobsahuje žádnou vestavěnou funkci pro efektivnější procházení XML.

8.3 Relevance informací o časové zóně

Přestože RSS standard uvádí, jak mají být data z kanálu interpretována, přesto to nechává určitou volnost, jak mohou být vydavateli pochopena. Abych eliminoval tento problém, využívám kanály, které interpretují tyto standardy podobným způsobem, aby bylo potřeba minimálního počtu modifikací mezi jednotlivými zdroji.

Stále však je potřeba vyřešit problém, který jsem již nastínil dříve a to s překladem informací o časové zóně definované v RFC 822. Standard RFC 822 definuje možnost označení časové zóny jako třípísmennou zkratku stejně jako hodinový offset od UTC (GMT). Pig sice podporuje práci s časem pomocí „datetime“ datového typu, avšak překlad třípísmenných zkratek naráží na problém neunikátnosti těchto zkratek, například zkratka „CST“ by mohla být Americký „Central Standard Time“ nebo „China Standard Time“. (20) a proto tedy není informace o časové zóně v tuto chvíli využitá.

9 Příbuzné práce

Tato práce využívá analýzy dat pomocí nadstavby Apache Pig, tato nadstavba však není jediná nadstavba, která může být na Hadoop využita. Popíši zde další možnosti zpracování dat podobné Pigu.

Projekt Mahout je využíván jako tvůrce knihovny pro podporu strojového učení, disponujícího k tomuto účelu vekou škálou různých matematických a algoritmických funkcí, které využívají MapReduce jako základ pro rychlé zpracování. Mahout je vytvořen pro filtrování, clusterování a klasifikaci vstupních dat. Jeho hlavní funkce by se daly shrnout jako doporučování na základě uživatelského chování, řazení dokumentů do clusterů podle shody v jejich obsahu a ohodnocení dokumentu podle již uložených a ohodnocených dokumentů.

Hive je koncipován jako infrastruktura pro databázové skladiště, který vyvinula společnost Facebook. Hive nabízí dotazovací jazyk podobný SQL, který využívá deklarační programovací model (21), a tím je potřeba používat více procedurální přístup. Výhodou Hive oproti SQL databázím je vysoká a levná škálovatelnost díky jeho vlastnosti uchovávat data v původních formátech, což je na druhou strano vykoupeno vysokým dotazovacím čase, omezení možností operací pouze na čtení atd. Přestože Pig a Hive jsou navrženy pro podobné úkony, jejich rozdíl spočívá ve zpracování vstupů (22), kdy Hive je připraveno na využití v datových skladech na rozdíl od Pigu, který je lépe vybaven na přípravu vstupních dat.

Apache HBase je databázový systém nevržený po vzoru BigTable od společnosti Google. Jeho základem je přístup k datům v databázi v nulovém čase, takzvaně „real-time“ avšak navržený pro databáze o velikosti milionů sloupců a miliardách řádků. Dalšími funkcemi jsou podpora API přístupu pro Pig a Hive, například Pig disponuje funkcemi pro načítání dat z HBase, avšak zde toho není využito, protože objem dat je pro toto použití s trochou nadsázky směšný.

10 Závěr

V této práci jsem představil platformu Pig pro zpracování a analýzu vstupních dat. Vysvětluje způsoby fungování využité v programování v dotazovacím jazyce Pig Latin a vysvětluje, jakým způsobem je Pig spojený s MapReduce modelem.

Abych ukázal využitelnost platformy Pig pro zpracování vybraných dat z webového serveru, analyzoval jsem prostředí Hadoop a na něj navazující modul Pig. Analyzoval jsem možnosti dotazovacího jazyka Pig Latin a vybral vstupní data z webového serveru na základě zaměření této práce. Provedl jsem analýzu vstupních dat a vytvořil skripty pro zpracování v jazyce Pig Latin.

Do této práce jsem popsal využití claudové platformy Google Computing Cloud při použití s projektem Hadoop a jeho nadstavbou Pig s dotazovacím jazykem Pig Latin. Jsou zde uvedeny detaily implementace, nasazení a použití a některé výsledky ze zpracovaných dat.

Byla zde ukázána možnost využití Apache Pig pro zpracování vybraných dat, jako například RSS feeds, z webového serveru, byť tato možnost není ze strany Pig přímo podporována vestavěnými funkcemi. Pig představuje užitečné zjednodušení na základě MapReduce modelu a umožňuje vytváření příkazů, které mohou zpracovávat a dotazovat se na vstupní data v jazyce programovacím jazyce vyšší úrovně a díky základu na MapReduce modelu jsou tyto příkazy automaticky paralelizovány a tím je zajištěno škálovatelné zpracování i pro velmi velká vstupní data. Přestože Pig zatím není připraven na práci s XML formáty, je tento systém vhodný pro zpracování problémů, které se dají rozložit na podproblémy, které se dají strojově zpracovat.

V tuto chvíli byla funkčnost otestována na vzorku vstupních dat ve formátu RSS, ale do budoucnosti by bylo vhodné rozšířit sběr dat o více zdrojů a na delším časovém intervalu, protože některé závěry nemohly být uskutečněny z důvodu malého množství vstupních dat, pro které bylo vhodné vytvořit nástroj, který by tato data automaticky shromažďoval a ukládal, aby bylo možné využít báze dat k vytvoření smysluplných statistik, což už není obsahem této práce.

11 Citace

Citace byly zpracovány podle normy ČSN ISO 690

1. **Google Inc.** Compute Engine. *Google Cloud Platform.* [Online] Google Inc., 24. 7 2014. [Citace: 27. 11 2015.] cloud.google.com/compute/docs.

2. **The Apache Software Foundation.** Apache Hadoop. *Apache.* [Online] The Apache Software Foundation, 31. 10 2015. [Citace: 27. 11 2015.] hadoop.apache.org.

3. —. Powered By. *wiki.apache.org.* [Online] The Apache Software Foundation, 2. 11 2015. [Citace: 27. 11 2015.] wiki.apache.org/hadoop/PoweredBy#F.

4. —. Apache Pig. *Apache.* [Online] The Apache Software Foundation, 6. 6 2015. [Citace: 27. 11 2015.] pig.apache.org.

5. *MapReduce: Simplified Data Processing on Large Clusters.* **Jeffrey Dean and Sanjay Ghemawat.** San Francisco : Google Inc., 2004.

6. **The Apache Software Foundation.** Pig Latin Basics. *Apache.* [Online] The Apache Software Foundation, 21. 11 2014. [Citace: 27. 11 2015.] pig.apache.org/docs/r0.14.0/basics.html.

7. **Google Inc.** google-and-ibm-announce-university_08. *googlepress.blogspot.cz.* [Online] 10. 10 2007. [Citace: 27. 11 2015.] http://googlepress.blogspot.cz/2007/10/google-and-ibm-announce-university_08.html.

8. —. Google Cloud Platform. *Cloud Launcher.* [Online] Google Inc., 1. 9 2015. [Citace: 1. 12 2015.] cloud.google.com/launcher/.

9. —. Google Cloud Platform. *Cloud Dataproc.* [Online] Google Inc., 30. 11 2015. [Citace: 1. 12 2015.] cloud.google.com/dataproc/.

10. —. Google Cloud Platform. *bdutil.* [Online] Google Inc., 30. 11 2015. [Citace: 1. 12 2015.] https://cloud.google.com/hadoop/bdutil.

11. **Wikipedia contributors.** Web syndication . *Wikipedia.org.* [Online] Wikipedia, The Free Encyclopedia. , 30. 10 2015. [Citace: 12. 12 2015.] https://en.wikipedia.org/w/index.php?title=Web_syndication&oldid=688226649.

12. **Board Members.** RSS 2.0 Specification. *RSS Advisory Board.* [Online] RSS Advisory Board, 30. 3 2009. [Citace: 12. 12 2015.] http://www.rssboard.org/rss-specification.

13. **Apache Software Foundation.** Pig 0.14.0 Documentation. *Apache Hadoop.* [Online] Apache Software Foundation, 21. 11 2014. [Citace: 3. 12 2015.] http://pig.apache.org/docs/r0.14.0/start.html.

14. —. Shell abd Utility Commands. *Apache Hadoop.* [Online] Apache Software Foundation, 21. 11 2014. [Citace: 4. 12 2015.] http://pig.apache.org/docs/r0.14.0/cmds.html.

15. —. Pig Setup. *Apache Hadoop.* [Online] Apache Software Foundation, 5. 14 2010. [Citace: 4. 12 2015.] https://pig.apache.org/docs/r0.7.0/setup.html.

16. **Google Inc.** Google Cloud Platform. *Google.* [Online] Google Inc., 8. 12 2015. [Citace: 10. 12 2015.] https://cloud.google.com.

17. —. Google Cloud Platform/Free-trial. *Google.com.* [Online] Google Inc., 8. 12 2015. [Citace: 12. 12 2015.] https://cloud.google.com/free-trial.

18. —. Supplemental Terms and Conditions For Google Cloud Platform Free Trial. *Google.com.* [Online] Google Inc., 10. 12 2015. [Citace: 10. 12 2015.] https://console.developers.google.com/billing/freetrial/terms.

19. **contributors, Wikipedia.** Bucket (computing). *Wikipedia.org.* [Online] Wikipedia, The Free Encyclopedia., 26. 2 2015. [Citace: 11. 12 2015.]

20. **Oracle America, Inc.** Java 6 Class TimeZone. *docs.oracle.com.* [Online] Oracle America, Inc., 20. 11 2015. [Citace: 12. 12 2015.] http://docs.oracle.com/javase/6/docs/api/java/util/TimeZone.html.

21. **Apache Software Foundation.** Apache Hive. *apache.otg.* [Online] Apache Software Foundation, 14. 1 2014. [Citace: 12. 12 2015.] hive.apache.org.

22. **Gates, Alan.** Pig and Hive at Yahoo! *developer.yahoo.com.* [Online] Yahoo Inc., 18. 8 2010. [Citace: 12. 12 2015.] https://developer.yahoo.com/blogs/hadoop/pig-hive-yahoo-464.html.

23. **Apache Software Foundation.** Pig 0.14.0 Documentation. *Apache Hadoop.* [Online] Apache Software Foundation, 21. 11 2014. [Citace: 3. 12 2015.] http://pig.apache.org/docs/r0.14.0/basic.html#load.

24. **Google Inc.** Google Developers Console/Apache Hadoop Beta. *Google.com.* [Online] Google Inc, 2. 12 2015. [Citace: 11. 12 2015.] https://console.developers.google.com/launcher/details/click-to-deploy-images/hadoop.

25. **Donkin, R. B.** Hadoop And Friends. *people.apache.org.* [Online] Apache Software Foundation, 16. 6 2008. [Citace: 13. 12 2015.] http://people.apache.org/~rdonkin/hadoop-talk/hadoop.html.