Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií

Evidenčné číslo

Matúš Cimerman

Analýza prúdu údajov

Bakalárska práca

Vedúci práce: Ing. Jakub Ševcech

december, 2014

Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií

Evidenčné číslo

Matúš Cimerman

Analýza prúdu údajov

Bakalárska práca

Študijný program: Informatika
Študijný odbor: 9.2.1 Informatika
Miesto vypracovania: Ústav informatiky a softvérového inžinierstva, FIIT STU Bratislava
Vedúci práce: Ing. Jakub Ševcech

december, 2014

Anotácia

Fakulta Informatiky a Informačných Technológií
Slovenská Technická Univerzita

Meno:	Matúš Cimerman
Vedúci bakalárskej práce:	Ing. Jakub Ševcech
Bakalárska práca:	Analýza prúdu údajov
Študijný program:	Informatika
December 2014	

Dnes sa stretávame so spracovaním a analýzou veľkého objemu dát (angl. Big Data) v mnohých oblastiach. S narastajúcim objemom dát rastie aj záujem o túto problematiku. Toto sa najviac dotýka oblastí kde je potrebné spracovávať dáta zo senzorov, sietí telekomunikačných operátorov, ale tiež sociálnych medií ako napríklad Twitter. Dáta z týchto zdrojov prúdia v obrovských množstvách, pričom sa v čase rýchlo menia, takéto prúdy nazývame jednoducho prúd dát. Tieto prúdy sú potenciálne nekonečné. Prúdy dát chceme spracovať a analyzovať v reálnom čase, aby sme mohli ponúknuť používateľom hodnotný výstup s čo najmenšou odozvou. Najčastejší spôsob spracovania dát je dávkové spracovanie s použitím MapReduce modelu, čo však nie je aplikovateľné pri viacerých úlohách spracovania dát. Táto metóda, ale nespĺňa naše požiadavky na spracovanie dát v reálnom čase. Na dosiahnutie našej požiadavky potrebujeme iný prístup, ktorý nazývame spracovanie prúdu dát. Jedna z paradigiem na spracovanie prúdu dát je napríklad, dátovody a filtre. Spracovanie a analýza veľkých prúdov dát je komplexný problém, pretože prúd musí byť spracovaný s nízkou odozvou, riešenie musí byť odolné proti chybám a horizontálne škálovateľné. V našej práci budeme analyzovať existujúce riešenia a rámce na analýzu prúdu dát. Poskytujeme overenie charakteristík týchto riešení v roznych problémoch, ktoré si vyžadujú spracovanie prúdu dát v reálnom čase. Na základe tohto navrhujeme a implementujeme aplikáciu, ktorá spracuje a analyzuje veľký prúd dát (napríklad prúd dát Twittru), ktorá umožní používateľom získať hodnotný výstup, ktorý sa mení v reálnom čase.

Annotation

Faculty of Informatics and Information Technology
Slovak University of Technology

Name:	Matúš Cimerman
Supervisor:	Ing. Jakub Ševcech
Bachelor thesis:	Data stream analysis
Course:	Informatics
2014, December	

Nowadays we can see Big Data processing and analysis in many domains. With increasing volume of data also growing up interest in this issue. The most affected domains where it is necessary to process data from sensors, networks, telecommunications operators, but also social media such as Twitter. Data from these sources flow in large amounts, while they are rapidly changing, these streams are simply called data streams. These streams are potentially infinite. We want to process and analyze data streams in real-time, to provide users valuable outputs with minimal latency. The most common method of data processing is batch processing using MapReduce model, which is not applicable in variety of data processing tasks. This method is not meeting our requirements to process data streams in real-time. To achieve these requirements, we need use different approach called data stream processing. One of paradigms to process data streams is for example, pipes and filters. Processing and analysis of big data streams is a complex issue, because stream needs to be processed with low-latency, the solution must be fault-tolerant and horizontally scalable. In our project, we will analyze existing solutions and frameworks for analyzing data stream. We provide verification of the characteristics of these solutions in various problems, that require data stream processing in real time. Based on this, we propose and implement a application, which processing and analyzing big data streams (e.g. Twitter data stream) and allows users to get valuable outputs in real-time.

Poďakovanie

Na prvom mieste sa chcem poďakovať vedúcemu mojej bakalárskej práce, inžinierovi Jakubovi Ševcechovi za všetky jeho rady, odovzdané skúsenosti a výborne vedenie pri tvorení práce.
Ďalej sa chcem poďakovať všetkým výskumníkom zo skupiny PeWe za hodnotné diskusie počas celého semestra a tiež spätnú väzbu k mojej práci.
V neposlednom rade sa chcem poďakovať celej mojej rodine a priateľom.

Matúš Cimerman

Obsah

1. Úvod

V ostatnom čase si téma spracovania prúdu údajov v kontexte veľkého objemu dát (angl. Big Data) vyžaduje stále väčšiu pozornosť. Čím ďalej, sa vo viacerých doménach stretávame s problémom spracovania narastajúceho objemu údajov, ktoré sú rozmanité. Tradičné prístupy Obchodných informácií (angl. Business intelligence) nie sú postačujúce pri riešení týchto problémov. (Liu et al., 2014).

Spracovanie Big Data sa stáva dôležitou časťou stále väčšieho množstva odvetví. Či už ide o veľkých telekomunikačných operátorov, komerčné podniky, vyhľadávače alebo sociálne média, všade sa stretávame s big data. Značným zdrojom dát sú v dnešnej dobe senzory, ktoré nachádzame stále častejšie v zariadeniach, ktoré sú súčasťou bežného života dnes. Chytré telefóny a hodinky, športové náramky, či vo všeobecnosti Internet vecí (angl. Internet of Things[1]) zmenšuje priepasť medzi svetom fyzických zariadení a prepojení s internetom. Napríklad šesť hodinový medzištátny let Boeingu 737 z New Yorku do Los Angeles vygeneruje počas letu celkovo 240 terabajtov dát(Higginbotham, 2010). Takéto rýchlo vznikajúce dáta, ktoré prúdia vo veľkých objemoch nazývame prúd dát alebo údajov (angl. data stream).

Preto stúpa motivácia a význam vybudovať infraštruktúru, ktorá bude schopná spracovať takýto masívny objem prúdiadich dáť v reálnom čase. Existuje veľa aplikácií, ktorých správne fungovanie môže byť kritické vzhľadom na správnosť výsledku z súvislého a nekonečného prúdu dát. Ako príklad môžme uviesť letové údaje zo senzorov lietadla Boeing 737 alebo analýza trendov na sociálnej sieti Twitter[2]. Keby sme takýto prúd spracovali tradičným prístupom spracovania Big Data, dávkovým spracovaním, mohlo by to mať kritický dopad na výsledký aplikácie. V prípade Boeingu 737 je jasné, že aplikácia musí poskytnúť výsledky v takmer reálnom čase, inak to môže mať katastrofické dopady. Pri analýze trendov na sociálnej sieti Twitter je veľmi pravdepodobné(Mathioudakis and Koudas, 2010), že sa v spracovanej dávke dát stratí trend, ktorý je aktuálny iba pre krátky časový úsek. Preto je dáta potrebné spracovávať okamžite po vstupe do aplikácie.

[1] TODO: Odkaz, na wiki?
[2] http://www.twitter.com/

TODO: Vysvetlit pojem real-time ako ho v tejto praci budeme chapat uz v uvode?

V našej práci sa venujeme spracovaniu prúdu dát. Cieľom práce je návrh a implementácia metódy, ktorá takéto prúdy dát dokáže efektívne spracovať. Práca je štrukturovaná do troch logických celkov.

TODO: Doplnit podla finalnej presnej struktury. Opisat klasicke rozdelenie zaverecnej prace - namapovat na moje a zdovodnit preco som moju pracu inak strukturoval.

Analyzujeme a sledujeme vlasnosti tejto navrhnutej metódy v porovnaní s klasickými prústupmi k spracovaniu big data.

2. Modely spracovania údajov

V tejto kapitole sa venujeme metódam, či prístupom k spracovaniu veľkého objemu údajov (angl. Big Data). Dávkové spracovanie je jednou z najčastejšie používaných metód spracovania big data, ale jej použitie zlyhýva v aplikáciach, pri ktorých môže byť čas odozvy kritický. Môžu to byť napríklad aplikácia, ktorá spracuje údaje zo senzorov lietadla alebo aplikácia na analýzu trendov na sociálnej sieti. Z tohto dôvodu v tejto kapitole porovnávame dva prístupy spracovania big data, *dávkové spracovanie* (angl. batch processing) a *prúdové spracovanie* (angl. stream processing), pričom druhá metóda hovorí, ako to vyplýva z názvu, o spracovaní prúdu dát.

Najskôr je potrebné čo najpresnejšie definovať, čo je to prúd údajov. Prúd údajov môže naberať iný význam v závilosti na zdroji týchto údajov. Vo všeobecnosti, je ale prúd dát, potenciálne nekonečný a nijak ohraničený tok dát v pohybe. Takýto prúd má tri význačné vlastnosti, veľký *objem* (angl. volume), vysokú *premenlivosť* (angl. velocity) a *pestrosť* (angl. variety) prúdiacich správ za jednotku času(Kaisler et al., 2013). Pre priblíženie, objem sa pohybuje rádovo v stovkách tisícov správ za sekundu. Ako príklad môžme uviesť aktivitu na sociálnej sieti Twitter, ktorá presahuje 50 milionóv nových správ (tweetov) za jeden deň(Mathioudakis and Koudas, 2010).

Najčastejšie zdroje takýchto dát, ktoré vyžadujú prúdové spracovanie dát v reálnom čase pre dosiahnutie hodnotných výstupov:

- *sociálne média* ako napríklad Twitter, či Facebook produkujú masívny objem dát v reálnom čase, ktoré strácajú svoju informačnú hodnotu veľmi rýchlo.
- *údaje z logovacích súborov*, webové servery, databázy a rozne iné zariadenia produkujú súbory, do ktorých sú ukladané logy.
- *sieťové zariadenia alebo prvky* kontinuálne generujú hodnotné dáta v obrovských objemoch, tiež v závislosti na veľkosti siete.
- *senzory a snímače*, ktoré môžu merať fyzikálne veličiny.

Takéto prúdy má zmysel spracovať v reálnom čase, aby sme dosiahli hodnotné výstupy. Spracovanie v reálnom čase sa môže opäť meniť v kontexte problému. V

našom prípade to bude najčastejšie znamenať, že správnosť výsledku nebude závislá iba na jeho správnosti, ale aj na čase za aký bude poskytnutý (Stankovic et al., 1999).

Pre získanie hodnotného výstupu na základe dopytu používateľa je nutné vykonať operáciu nad všetkými dostupnými dátami(Marz and Warren, 2013, s. 8-10).

$$dopyt = fn(všetky\ dáta,\ ktoré\ sú\ k\ dispozícii)$$

Pri navrhovaní architektúry softvérového systému, ktorá ma za úlohu spracovať veľký objem dát je nevyhnutné aby spĺňala aspoň tieto nasledujúce vlastnosti:

1. *robustná a odolná voči chybám*, architektúra je robustná, ak bez chyby odolá nečakaným stavom, ako napríklad výpadok elektrickej energie. Je odolná voči chybám, ak chybné dáta alebo strata dát nemá za následok poškodený výstup.
2. *nízka odozva*, aplikácie postavené na tejto architektúre sú schopné poskytnúť odozvu v požadovanom čase. Typicky rádovo desiatky, maximálne stovky milisekúnd.
3. *horizontálne škálovateľná*, ak je možné zvýšiť výkon celej architektúry pridaním fyzického uzla bez dopadu na funkcionalitu.
4. *všeobecná*, architektúru je možné použiť na rôzne aplikácie.
5. *rozšíriteľná*, do fungujúcej architektúry je možné pridávať novú funkcionalitu za prijateľnej námahy.

2.1 Dávkové spracovanie veľkého objemu dát

Dávkové spracovanie veľkého objemu dát funguje na príncipe, ako napovedá názov, spracovania dát v dávkach. Nespracujú sa všetky dáta naraz, alebo postupne ako vznikajú, ale v istých dávkach. Tieto dávky môžu byť rozne veľké. Môžu byť ohraničené časovo alebo veľkosťou. Najčastejší prístup je vytvárať dávky na spracovanie po nejakých časových intervaloch, napríklad každých sedem hodín. Znamená to, že po dobu sedem hodín niekam akumulujeme nové dáta. Po uplynutí tohto času je nad týmito dátami a všetkými, ktoré boli doteraz zaznamenané, vykonaná nejaká operácia (napríklad priemer čísel) a jej výsledok je uložený vo forme *pohľadu*.

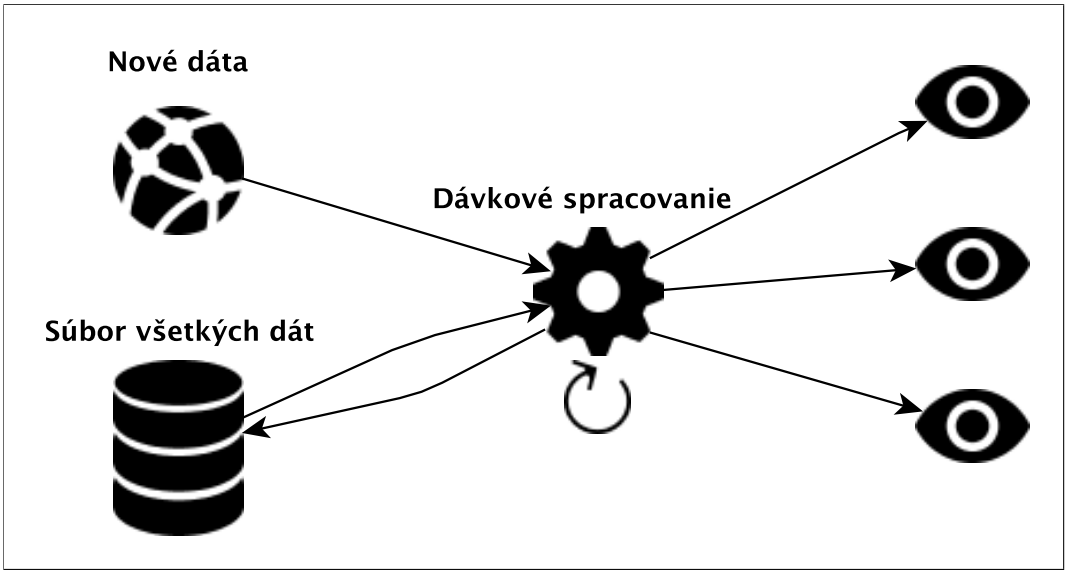

Figure 2.1: Obrázok zobrazuje dávkové spracovanie dát, pričom na pravej strane sú vytvárané rôzne pohľady.

S takýmto prístupom bude len veľmi ťažké dosiahnut nízku odozvu na dopyt používateľa. Nakoľko minimálne v jednom momente bude *pohľad* sedem hodín neaktuálny. Tomuto by sa teoreticky dalo zabrániť rapídnym znížením časového intervalu.

Pri tomto prístupe sú všetky dáta ukladané do databázy, tak ako prišli. Nie sú vôbec modifikované, takže sa pracuje s čistými dátami. Súbor dát, ktorý uchovávame môže byť preto veľmi objemný, rádovo terabajty. To znamená, že každé spracovanie dávky je časovo a výpočtovo náročná operácia a môže niekedy trvať aj celé hodiny.

Nami poskytnuté teoretické riešenie ako sa vyhnúť vysokej odozve znížením časového intervalu, po ktorom je vykonané dávkové spracovanie, by nebolo riešením. To čo sa môže zdať ako nevýhoda, je zároveň aj výhoda, pretože ukladaním všetkých dát dosahujeme odolnosť voči chybám (aj spôsobným človekom) a pri vytváraní pohľadu zohľadňujeme údaje z minulosti.

Dávkové spracovanie dát nám poskytuje distribuované a spoľahlivé (tj. odolné voči chybám) riešenie. Tento prístup tiež poskytuje škálovateľnosť, je dostatočne všeobecný a rozšíriteľný. Avšak neposkytuje nízku odovzu. Nasledujúci zoznam hovorí o najdôležitejších vlastnostiach dávkového spracovania:

1. *neohraničené počítanie*, znamená že máme teoreticky neobmedzenú výpočtovú silu a neobmedzený čas výpočtu. Teoreticky môžeme vykonať dávkové spracovanie nad akýmkoľvek dátovým súborom.

2. *ukladanie normalizovaných dát*, neexistuje potreba dáta denormalizovať, pretože pohľady sú generované často.

3. *architektúra je horizontálne škálovateľná.*

4. *dáta sú konzistentné*, za predpokladu, že vstupné dáta sú korektné.

2.1.1 Metódy dávkového spracovania

MapReduce je programovací model a príslušná implementácia na spracovanie a generovanie masívnych údajových korpusov (Dean and Ghemawat, 2008). Tento model bol vymyslený kvôli výpočtovej zložitosti a problému distribuovať a paralelizovať výpočet. Je založený na primitívach prezentovaných v programovacom jazyku Lisp, *mapovanie* a *filtrovanie*. Výpočet je vykonávaný za použitia len týchto dvoch funkcií. Funkcia *mapovanie* má ako vstupný argument dvojicu *kľúč/hodnota* s kľúčom K, nad ktorou spraví používateľom definovanú operáciu, a vytvorí dočasnú dvojicu s kľúčom K. Túto hodnotu s kľúčom K ďalej konzumuje funkcia *filtrovanie*, ktorá najčastejšie spojí hodnoty do jednej (napríklad operácia sčítania). Ako jednoduchý príklad uvádzame problém spočítania výskytov slova vo veľkej kolekcii dokumentov (Dean and Ghemawat, 2008):

Dátové kocky (angl. Data cubes) Dátová kocka, tiež známa pod pojmom OLAP[1] kocka, je dátová štruktúra, ktorá umožňuje rýchlu analýzu dát. Hlavná motivácia prečo vytvárať dátove kocky je materializácia všeobecných údajov. Takáto kocka má dve hlavné atribúty, *dimenzia* a *hodnota*. Pričom dimenzie predstavujú nezávislé hodnoty, napríklad vek a hodnota je závislá, pretože bude viazaná k veku v konkrétnom zázname. Viac rozmerný priestor je vymedzený dimenziami kocky. Nad dátovými kockami poznáme niekoľko základných operácií, *krájanie* (angl. slicing), *delenie* (angl. dicing), *pivotovanie* (angl. pivoting), *zrolovanie* (angl. roll up), *drill-down*(Franek, 2013).

Hviezdicová schéma (angl. Star schema) Hviezdicová schéma sa používa na modelovanie viac dimenzionálnych dát a vychádza priamo z princípov relačných databáz. Pričom priamo v tabuľke je uložených len niekoľko základných faktov a kľúčov. Tieto kĺúče ďalej odkazujú na ďalšie tabuľky, ktoré môžu odkazovať na ďalšie tabuľky(Chaudhuri and Dayal, 1997). Dopytovanie sa do takéjto schémy

[1] Online Analytical Processing

je pomerne efektívne, nakoľko nie je vždy potrebné dotazovanie vo všetkých dimenziách.

Dátové sklady (Datawarehouse) Pojem dátový sklad definoval v roku 1990 Bill Inmon. Dátovy sklad je subjektovo orientovaná, integrovaná, stála a časovo rozdielna kolekcia dát podporujúca rozhodujúce procesy(Chaudhuri and Dayal, 1997). Dátové sklady nám poskytujú zovšeobecnené a ustálené dáta vo viacerých dimenziách. Dátové sklady nám tiež poskytujú prostriedky ako napríklad OLAP, ktorý bol vyšie spomenutý v súvislosti s dátovými kockami. Dátove sklady sú typicky udržiavané mimo hlavnej databázy, resp. úložiska dát.

2.2 Prúdové spracovanie dát

Správne fungovanie niektorých aplikácií je priamo závislé na takmer okamžitom výsledku z nekonečného spojitého prúdu dát. Keďže prúd je nekonečný, nové dáta prichádzajú kontinuálne a teda aj výsledky a zmeny z nich vyplývajúce. Dáta musia byť spracované tak ako prichádzajú. Spracovanie takýchto nekonečných prúdov dát aplikovaním tradičných metód môže byť problém, keďže výpočtové prostriedky sú limitované (Babcock et al., 2002). Množstvo predtým (rádovo sekundy) vygenerovaných dát je zvyčajne veľké a preto sú často po spracovaní zahadzované, ale môžu byť uložené na disku. V prípade keby sme aj chceli pristupovať k dátam z minulosti (môže byť užitočné pri detekcii trendov) bolo by to výpočtovo náročné, pretože dopyt sa musí vykonať nad všetkými doteraz uloženými dátami pre získanie jedného výstupu(Silvestri, 2006). Na tkéto uloženie sa najčastejšie používajú NoSQL databázy, pretože tradičné extrahuj-transformuj-načítaj (ETL)[2] relačné databázy sú priveľmi štrukturované a pomalé (Liu et al., 2014). Preto sa budeme zaoberať v tejto kapitole iba nástrojmi, ktoré adresujú spracovanie v takmer-reálnom čase.

Pri prúdovom spracovaní sa stretávame s niekoľkými špecifikami:

Prúd dát

Problémy

[2] extract-transformation-load

3. Existujúce riešenia modelov spracovania údajov

V tejto kapitole bližšie rozoberieme existujúce riešenia, ktoré sme analyzovali. Zamerali sme sa najmä na riešenia, ktoré nie sú proprietárne a čo najviac spĺňajú naše požiadavky na prúdové spracovanie údajov.

3.1 Dávkové spracovanie

Apache Hadoop je open-source implementácia MapReduce programovacieho modelu. Hadoop pozostáva z dvoch základných komponentov: *Distribuovaný Súborový Systém Hadoop* (angl. Hadoop Distributed File System) a *MapReduce programovacieho rámca* (Liu et al., 2014).

3.2 Prúdové spracovanie

S4 Skratka S4 znamená *Jednoduchý Škálovateľný Prúdový Systém* (angl. Simple Scalable Streaming System), tento systém je založený čiastočne na modely MapReduce. S4 je všeobecná, distribuovaná a škálovateľná platforma, ktorá je čiastočne odolná voči chybám. Napríklad, ak spracujúci uzol v topológií vypadne kvôli chybe, spracovanie je automaticky presunuté na iný uzol, ale stav toho uzla je stretený a nemôže byť obnovený (Neumeyer et al., 2010). Tento systém nepoužíva zdieľanú pamäť a je založený na modeli Hráčov (angl. Actor model) (Agha, 1985). S4 má decentralizovanú symetrickú architektúru, v ktorej sú všetky uzly na rovnakej úrovni (rozdiel oproti master-slave architektúre). Tieto uzly sú nazývané *spracujúce elementy* (angl. processing elements, ďalej len PEs). S4 si vymieňa informácie medzi PEs vo forme dátových udalostí, čo je aj jediná možnosť interakcie a výmeny informácií medzi PEs. Strapec (angl. cluster) S4 pozostáva z PEs na spracovanie týchto udalostí. Vzhľadom na to, že dáta prúdia medzi PEs nie je potrebné ukladanie na disk. Z čoho tiež vyplýva, už spomenutá, čiastočná odolnosť voči chybám (Liu et al., 2014).

Spark Spark (Apache, 2015) clustrový výpočtový systém. Cieľom Spark-u je poskytnúť rýchlu výpočtovú platformu pre analýzu dát. Spark poskytuje všeobecný model výkonávania ľubovolných dopytov, ktoré sú výkonávané v pamäti. Takže opäť nie je potreba ukladať dáta počas spracovania na disk, čo by mohlo mať za následok nežiadúce spomalenie aplikácie.

3.3 IBM InfoSphere Streams

3.4 Storm

Strom je programovací rámec vytvorený na spracovanie prúdu dát. Je to open-source riešenie, ktoré poskytuje spracovanie prúdu dát s nízkou odozvou. Storm pozostáva z viacerých častí vrátane koordinátora (ZooKeeper), manažéra stavov (Nimbus) a spracovávajúcich uzlov (Supervisor). Implementuje model kde dáta neustále prúdia sieťou, tiež nazývanou topológiou uzlov a ústí. Topológia väčšinou záčína ústím a ďalej nasledujú len uzly, ktoré spracujú dáta a posielajú ich ďalej na spracovanie. Abstrakcia nad prúdiacimi údajmi sa v skratke nazýva *prúd* (Liu et al., 2014), čo je neohraničená sekvencia n-tíc.

TODO: Porovnanie existujucich rieseni a odovodnenie preco som sa rozhodol prave pre storm

4. Aplikácia pre filtrovanie v prúde dát

V predchádzajúcej kapitole sme sa venovali existujúcim riešeniam na spracovanie veľkého objemu dát. Niektoré sa zameriavajú na dávkové spracovanie veľkých dát, iné na spracovanie "rýchlych dát" v reálnom čase. Navrhujeme metódu na analýzu prúdu údajov. Dáta prúdia zo zdroja v reálnom čase. Naša metóda spracuje prúd údajov v reálnom čase a poskytytuje používateľovi hodnotný výstup na základe dopytu. Metódu navrhujeme pre doménu sociálnych sietí, konkrétne Twitter. Cieľom metódy je overiť vlastnosti spracovania a filtrovania prúdu správ pomocou rámca Storm. Vstupom do procesu spracovania prúdu údajov je potenciálne nekonečný prúd údajov.

4.1 Všeobecná topológia

Ako bude vyzerať moja topológia? Mala by byt prisposobena na paralelne spracovanie dopytov.

5. Implementácia navrhnutej metódy

6. Overenie a experimenty

7. Zhodnotenie a budúca práca

7.1 Zhodnotenie a záver

7.2 Budúca práca

A. Plán na letný semester

- *december*
  1. týždeň,
  2. týždeň, odovzdanie priebežnej verzie dokumentu.
  3. týždeň, príprava vzorového datasetu z Twittru.
  4. týždeň, dokončenie navrhovanej metódy.
- *január*
  1. týždeň, revízia metód spracovania, existujúcich riešení a navrhovanej metódy.
  2. týždeň, revízia navrhovanej metódy a implementácia navrhovanej metódy.
  3. týždeň, implementácia navrhovanej metódy + návrh experimentov.
  4. týždeň, implementácia navrhovanej metódy + realizácia experimentov.
- *február*
  1. týždeň, príprava abstraktu na IIT.SRC.
  2. týždeň, príprava príspevku na IIT.SRC.
  3. týždeň, odoslanie príspevku na IIT.SRC.
  4. týždeň, práca na ďalších experimentoch.
- *marec*
  1. týždeň, vyhodnotenie experimentov.
  2. týždeň, revízia implementácie a experimentov.
  3. týždeň, implementácia webovej aplikácie.
  4. týždeň, implementácia webovej aplikácie.
- *apríl*
  1. týždeň, technická dokumentácia a používateľská príručka.
  2. týždeň, technická dokumentácia a používateľská príručka.
  3. týždeň, revízia dokumentu.
  4. týždeň, azda IIT.SRC

Literatúra

Agha, G. A. (1985). Actors: a model of concurrent computation in distributed systems.

Apache (2015). Apache spark.

Babcock, B., Babu, S., Datar, M., Motwani, R., and Widom, J. (2002). Models and issues in data stream systems. In *Proceedings of the twenty-first ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 1–16. ACM.

Chaudhuri, S. and Dayal, U. (1997). An overview of data warehousing and olap technology. *ACM Sigmod record*, 26(1):65–74.

Dean, J. and Ghemawat, S. (2008). MapReduce : Simplified Data Processing on Large Clusters. 51(1):107–113.

Franek, B. L. (2013). Importy dat z relační databáze do olap datovỳch kostek.

Higginbotham, S. (2010). Sensor networks top social networks for big data. [Online; zverejnené 13-September-2010].

Kaisler, S., Armour, F., Espinosa, J. A., and Money, W. (2013). Big data: Issues and challenges moving forward. In *System Sciences (HICSS), 2013 46th Hawaii International Conference on*, pages 995–1004. IEEE.

Liu, X., Iftikhar, N., and Xie, X. (2014). Survey of real-time processing systems for big data. In *Proceedings of the 18th International Database Engineering & Applications Symposium*, pages 356–361. ACM.

Marz, N. and Warren, J. (2013). *Big Data: Principles and best practices of scalable realtime data systems*. O'Reilly Media.

Mathioudakis, M. and Koudas, N. (2010). Twittermonitor: trend detection over the twitter stream. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*, pages 1155–1158. ACM.

Neumeyer, L., Robbins, B., Nair, A., and Kesari, A. (2010). S4: Distributed stream computing platform. In *Data Mining Workshops (ICDMW), 2010 IEEE International Conference on*, pages 170–177. IEEE.

Silvestri, C. (2006). Distributed and stream data mining algorithms for frequent pattern discovery.

Stankovic, J. A., Son, S. H., and Hansson, J. (1999). Misconceptions about real-time databases. *Computer*, 32(6):29–36.