Univerzita Pardubice

Fakulta ekonomicko-správní

Ukládání velkých objemů prostorových dat

Tomáš Jelínek

Diplomová práce

2014

Univerzita Pardubice
Fakulta ekonomicko-správní
Akademický rok: **2013/2014**

ZADÁNÍ DIPLOMOVÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Bc. Tomáš Jelínek**
Osobní číslo:	**E110081**
Studijní program:	**N6209 Systémové inženýrství a informatika**
Studijní obor:	**Informatika ve veřejné správě**
Název tématu:	**Ukládání velkých objemů prostorových dat**
Zadávající katedra:	**Ústav systémového inženýrství a informatiky**

Zásady pro vypracování:

Cílem práce je navrhnout vhodný způsob pro ukládání velkoobjemových prostorových dat a návrh realizovat.

Zásady:
- Charakteristika prostorových dat.
- Způsoby ukládání prostorových dat.
- Návrh vhodného způsobu pro ukládání velkých objemů prostorových dat, volba vhodného programového řešení.
- Realizace navrženého řešení a jeho zhodnocení.

Rozsah grafických prací:

Rozsah pracovní zprávy: **cca 55 stran**

Forma zpracování diplomové práce: **tištěná/elektronická**

Seznam odborné literatury:

CALUWE, R., TRÉ, G., BORDOGNA, G. Spatio-temporal databases: flexible querying and reasoning. Berlin, Heidelberg: Springer, 2004. 392 s. ISBN 9783540222149.
HAINING, R. Spatial data analysis: theory and practice. Cambridge: Cambridge University Press, 2003. 432 s. ISBN 9780521774376.
MANOLOPOULOS, Y., PAPADOPOULOS, A., VASSILAKOPOULOS, M. Spatial databases: technologies, techniques and trends. [s.l.]: Idea Group Inc (IGI), 2004. 340 s. ISBN 9781591403883.
PANG LO, C., YEUNG, A. Concepts and techniques of geographic information systems. USA: Prentice Hall, 2006. 2nd revised ed. 544 s. ISBN 9780131495029.

Vedoucí diplomové práce: **doc. Ing. Jitka Komárková, Ph.D.**
Ústav systémového inženýrství a informatiky

Datum zadání diplomové práce: **1. října 2013**
Termín odevzdání diplomové práce: **30. dubna 2014**

doc. Ing. Renáta Myšková, Ph.D.
děkanka

L.S.

prof. Ing. Jan Čapek, CSc.
vedoucí ústavu

V Pardubicích dne 1. října 2013

PROHLÁŠENÍ

Prohlašuji, že jsem tuto práci vypracoval samostatně. Veškeré literární prameny a informace, které jsem v práci využil, jsou uvedeny v seznamu použité literatury.

Byl jsem seznámen s tím, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorský zákon, zejména se skutečností, že Univerzita Pardubice má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona, a s tím, že pokud dojde k užití této práce mnou nebo bude poskytnuta licence o užití jinému subjektu, je Univerzita Pardubice oprávněna ode mne požadovat přiměřený příspěvek na úhradu nákladů, které na vytvoření díla vynaložila, a to podle okolností až do jejich skutečné výše.

Souhlasím s prezenčním zpřístupněním své práce v Univerzitní knihovně.

V Pardubicích dne 20. 4. 2014 Tomáš Jelínek

PODĚKOVÁNÍ:

Tímto bych rád poděkoval vedoucí mé diplomové práce doc. Ing. Jitce Komárkové, Ph.D. za její odbornou pomoc, cenné rady a poskytnuté materiály, které mi pomohly při zpracování diplomové práce. Dále bych chtěl poděkovat Ing. Martinovi Novákovi a Ing. Oldřichovi Horákovi za poskytnutý přístup do laboratoří a za technickou podporu při experimentech.

ANOTACE

Tato práce popisuje problematiku velkých objemů prostorových dat ve spojení s jejich ukládáním. Po teoretické stránce obsahuje jednotlivé možnosti při ukládání prostorových dat. Zachycuje kritéria, která jsou nutná zohlednit při zavádění řešení velkoobjemového ukládání prostorových dat. Na základě zvoleného kritéria se práce zaměřuje na jedno vybrané řešení, které je na univerzitním počítačovém clusteru testováno a konfigurováno pro dosažení nejlepších možných výsledků při ukládání.

KLÍČOVÁ SLOVA

Big data, Cloud computing, Hadoop, HDFS, Map Reduce, velkoobjemová data

TITLE

Storing of large volumes of spatial data

ANNOTATION

The following theses describes the issues of large volumes of spatial data in relation to their storage. Theoretical part includes the individual options of storing the spatial data. It also captures the criteria that are necessary to take into account when implementing the solution of bulk storage of spatial data. On the basis of the selected criteria, the work focuses on one selected solution, which is tested and configured at the university computer cluster to achieve the best possible results while in the process of storing.

KEYWORDS

Big data, Cloud computing, Hadoop, HDFS, MapReduce, large volume of data

OBSAH

SEZNAM TABULEK

SEZNAM ILUSTRACÍ

SEZNAM ZKRATEK A ZNAČEK

DFS	distributed file system
FS	file system
GIS	geographical information system
HDFS	Hadoop distributed file system
SQL	structured query language

ÚVOD

Velký počet společností, ať už malé, střední či velké, bojují s nekončícím nárůstem objemu dat, který tak vyvolává tlak na efektivní správu a zabezpečení společnosti. Přitom řešení nespočívá v nákupu novějších, efektivnějších, výkonnějších či větších serverů a úložišť dat, které se o to postarají. Výchozím bodem z této situace je především bezpečné, škálovatelné a hlavně efektivní ukládání dat. Tento problém nezasahuje pouze do určitých odvětví. Velkými objemy rychle rostoucích dat jsou zasažena všechna odvětví po celém světě. To jak se k tomu společnosti postaví, je už na každé zvlášť. Kvůli těmto důvodům bylo zvoleno téma diplomové práce.

Převážná většina všech dat, obsahuje prostorovou komponentu. Existuje velké množství nástrojů, které tato data ukládají a následně zpracovávají a vizualizují. Důležitým hlediskem před ukládáním prostorových dat je stanovení účelu, pro který budou prostorová data využívána. Na základě toho se rozhoduje, jakým způsobem se prostorová data budou ukládat.

Cílem diplomové práce je návrh vhodného způsobu pro ukládání velkých objemů prostorových dat a následná volba vhodného programového řešení, které bude vhodné pro malý univerzitní cluster skládající se z běžných počítačů. Pro výběr programového řešení bude zapotřebí stanovit si sadu kritérií, za pomocí kterých se vybere optimální řešení a toto řešení bude následně na univerzitním clusteru nainstalováno a testováno.

1 PROSTOROVÁ DATA

V této kapitole bude čtenář seznámen se základními pojmy zabývajícími se problematikou prostorových dat (někdy označována jako geodata).

Prostorová data představují informace, které se vztahují k určitým místům v prostoru a pro která jsou na potřebné úrovni rozlišení známé lokalizace těchto míst. Uživatel pracující s prostorovými daty tedy ví, kde se data vyskytují a je schopen určit jejich polohu. Údaje, které zajišťují vazbu dat na konkrétní místo v prostoru, se nazývají georeference. Většinou se jedná o údaj, který zprostředkovává prostorovou lokalizaci nepřímo, jako je adresa, název státu, města apod.[48]

Nejčastěji jsou prostorová data prezentována v podobě map. Bývají to mapy analogové (tištěné na papíře), nebo digitální, které se zobrazují například na monitoru počítače.[13]

1.1 Reprezentace prostorových dat

Jak je již zmíněno v předchozím odstavci, prostorová data mohou být prezentována analogově, či digitálně. Tato práce se zaměřuje pouze na digitální zpracování, tudíž se dále analogovou reprezentací nebude zaobírat. Digitálně prezentovaná prostorová data bývají označovány jako datové modely.

Datové modely představují prostorová data, která jsou uložená v souborech nebo databázích (více informací o způsobech ukládání bude popsáno v následující kapitole) a mají pevně danou strukturu dat. S datovými modely úzce souvisí pojem datové modelování, což je proces abstrakce, při kterém jsou podstatné elementy reálného světa vyčleňovány a nepodstatné eliminovány. Zohledňuje se cíl, který má toto modelování splnit.[49]

Datové modely využívají geografické informační systémy (dále jen GIS). Tyto systémy umožňují provádět rozsáhlé analýzy nad daty, zobrazují různé přehledové mapy, provádějí různé vizualizace dat atd. Aby GIS mohly s datovými modely pracovat, určují si požadavky na datovou strukturu. Datová struktura představuje uspořádání datového modelu. Datový model musí uchovávat geometrickou, topologickou a tematickou informaci.[22]

Datové modely ve vztahu ke GIS se dle zdroje [49] člení na klasické a objektově orientované. Mezi klasické datové modely náleží rastr a vektor.

Rastrový datový model

Rastrový datový model vychází z rozdělení rovinného prostoru pravidelnou mřížkou na jednotlivé dílky, které se označují jako buňky (angl. cell). Tyto buňky představují nejmenší, a zpravidla nedělenou, prostorovou jednotku. Prostorové vztahy mezi jednotlivými prvky jsou jednoznačně (implicitně) dány přímo v rastru. Každá buňka obsahuje určité číslo, které reprezentuje hodnotu mapovaného atributu. Prostorové vztahy geoprvků a lokalizace geoprvků nejsou přímo dostupné. Pokud uživatel tyto informace potřebuje, musí si sestavit agregaci buněk náležících jednotlivým geoprvkům.

V rastrovém modelu se nevyskytuje popis jedinečných geoprvků, které leží v zájmové oblasti. Vyskytuje se zde jen popis rozložení jedinečných atributů v této oblasti. Dále zde neexistuje ani explicitní popis geometrie geoprvků.[36]

Dle zdroje [50] dosahuje velikost obrázku rastru, při vysokém rozlišení jednotek megabytů a v profesionální grafice se operuje s daty o desítkách megabytů.

Vektorový datový model

Vektorový datový model, na rozdíl od rastrového modelu, zavádí schématické členění dat podle geoprvků. Každému geoprvku je přidělen jedinečný identifikátor. Geometrická složka popisu se ukládá odděleně do tematické složky popisu. Tyto dvě složky jsou propojeny jedinečným identifikátorem geoprvku. Vektorová data dosahují mnohonásobně menších datových objemů. Mezi základní geometrické prvky vektorového datového modelu patří bod, linie (tvořena pomocí počátečního a koncového bodu) a polygon (plocha).[36]

Objektově orientované datové modely

Tyto modely vycházejí z objektově orientovaného programování. Data jsou spravována jako objekty, což více přibližuje model reálnému světu. Každý objekt je popisován nejen jeho vlastnostmi (atributy), ale i způsobem jeho chování. Objekty spolu komunikují pomocí zpráv. Tvorba objektové databáze je náročnější než relační, je totiž nutné sestavit složitou hierarchii objektů a naprogramovat metody.[32]

1.2 Způsoby ukládání prostorových dat

Pokud se prostorová data mohou prezentovat analogově a digitálně, dají se také těmito způsoby ukládat. Ovšem vzhledem k tématu diplomové práce následující text bude zaměřen

pouze na digitální způsob ukládání. Existují dva základní způsoby digitálního ukládání prostorových dat. Jedná se o souborový a databázový.

Databázové ukládání

Prvním způsobem je ukládání do běžné databáze, případně do geodatabáze. Pro obě databáze platí, že data jsou uložena v databázi, kterou řídí SŘBD (systém řízení báze dat). SŘBD anglicky označováno jako DBMS (database management system) je software, který prostřednictvím uživatelského rozhraní pracuje s databází.[57]

Geodatabáze, neboli prostorová databáze je speciálně vyvinutá databáze firmou Esri a vyznačuje se tím, že umí pracovat s geoprvky (bod, linie, polygon). Více o databázích se čtenář dočte v kapitole číslo 2.[21]

Souborové ukládání

Druhým způsobem jak ukládat prostorová data do počítače či na server, je ukládání souborů na lokální disk. Data se ukládají do souborů v adresářové struktuře počítače, což přináší jisté výhody i nevýhody. Při editaci souboru přistupují uživatelé přímo k obsahu souboru. Tudíž více uživatelů nesmí pracovat najednou se stejnými daty.[16]

Příkladem souborového uložení je datový formát shapefile. Shapefile je tvořen minimálně třemi soubory (DBF – atributy v tabulce databáze, SHP – kresba, SHX – indexový soubor). Dále může obsahovat ještě soubor označovaný jako PRJ, který obsahuje údaje o souřadnicovém systému. Výhodou shapefilu je snadná přenositelnost mezi softwary a jednoduchost. Ovšem není vhodný pro ukládání většího počtu dat. Shapefile používá společnost Esri ve svém geografickém informačním systému ArcGIS for desktop.[19]

1.3 Datové formáty

Datové formáty jsou standardy grafických dat, které slouží k jejich přenosu, zpracování, ukládání a prohlížení. Těchto formátů je velké množství. Spoustu firem, které vyvíjejí software pro práci s grafikou, používají svůj vlastní firemní formát. Z tohoto důvodu budou v této části diplomové práce popsány pouze ty, které budou dále v práci využity.

1.3.1 Shapefile

Shapefile je vektorový formát firmy Esri, který byl poprvé představen v programu ArcView 2 začátkem devadesátých let. Objekty, kterými jsou bod, linie, plocha, se ukládají

ve formě vektorových souřadnic klíčových bodů. Geometrie prvku se ukládá jako sekvence bodů (např. GPS souřadnice). Jeden soubor většinou reprezentuje jeden typ mapového prvku. Tím může být například silnice, jezera, obce, železnice atd. Shape file neukládá topologii dat.[55]

ShapeFile se skládá z hlavního souboru, který má příponu .shp, indexového souboru .shx a databázové tabulky .dbf. Dále pak může obsahovat nepovinné soubory jako jsou .prj (zdrojový souřadnicový systém), .cpg (kódování v .dbf souboru) a další. Hlavní soubor je záznamový, jelikož každému záznamu se připisuje typ objektu se seznamem klíčových bodů. Indexový soubor propojuje prvek v hlavním souboru se záznamem v atributové tabulce. Databázová tabulka obsahuje atributy jednotlivých prvků ve stejném pořadí, jak jsou zaznamenány v hlavním souboru. U nás poskytuje data ve formátu Shapefile zeměměřičský úřad.[19]

Výhody ShapeFile [55] **:**

- snadná editace bodů, otevřený formát, rychlá vizualizace geodat,
- jednoduše pochopitelná struktura, lehce dostupná struktura,
- podpora v GIS softwarech, snadná projekce do jiných souřadnicových systémů.

Nevýhody ShapeFile [55] **:**

- neukládá topologii dat, redundance dat,
- datově náročná manipulace s detailními shapefile,
- špatná podpora Unicode.

1.3.2 Data získána z výškového skenování

Technologií leteckého laserového skenování, která umožňuje rychlé a přesné zachycení výškových dat se nazývá LIDAR (Light Detection And Ranging). Základním principem je dálkoměrné měření pomocí laserového svazku paprsků, přičemž musí být známa přesně poloha skeneru a přesný směr vysílání paprsku. Svazek paprsků je odrážen zpět a je zaznamenávána doba mezi vysláním paprsku a přijetím jeho odrazu.

Data z těchto měření se uchovávají ve formátu GRID, který je určen především pro program ArcGIS for desktop. Soubor obsahuje x, y a z souřadnice, které se dají exportovat do databázových či textových formátů (ASCII soubor).[61]

Shrnutí první kapitoly

V této kapitole byla popsána problematika prostorových dat, čím jsou reprezentována, jak se dají ukládat a v poslední řadě zde byly zmíněny datové formáty, které budou v práci dále vyžity. Na základě reprezentace prostorových dat bylo rozhodnuto, že se práce dále bude zabývat pouze digitální prezentací. Dále bylo také zjištěno, že prostorová data v podobě datových modelů se využívají v geografických informačních systémech (GIS). Ve druhé části této kapitoly bylo zjištěno, že prostorová data se dají ukládat souborově, nebo databázově, čímž se autor bude detailněji zabývat v následující kapitole. Třetí část se zaměřuje na datové formáty. Jelikož datových formátů prostorových dat je velké množství a spousta firem používá své vlastní, proto nebylo podstatné popisovat všechny formáty, ale pouze ty, které budou v práci dále využity.

2 UKLÁDÁNÍ DO DATABÁZÍ A SOUBOROVÝCH SYSTÉMŮ

Tato kapitola se bude zabývat nejrozšířenějšími typy pro ukládání velkých objemů prostorových dat. Budou zde vystiženy jejich výhody a nevýhody a vhodnost jejich použití.

2.1 Základní rozdíly mezi databází a souborovým systémem

Souborový systém představuje snadný způsob ukládání a organizování souborů. Databáze obsahuje počítačový software, který spravuje databázi. Nazývá se jako systém řízení báze dat (SŘBD) a v angličtině jako DBMS (Database management system). Databáze je velice rychlá pro zprostředkování souvisejících dat. Samozřejmě zde záleží na velikosti uložených dat v databázi. Data v databázi mohou být načítána několika uživateli, zatímco v souborovém systému uživatel přistupuje přímo k souboru a bývá tak problém při nutnosti práce více uživatelů na jednom souboru. Všechny databázové architektury poskytují metody pro organizaci dat do tabulek, které mohou být aktualizovány.[59]

Databáze se obvykle používají pro ukládání **strukturovaných dat** pro definované formáty dat s efektivním způsobem pro ukládání, aktualizaci a zpětné načítání z databáze (závisí na aplikaci). Souborový systém je spíše určen pro **nestrukturovaná data**, tedy pro uložení libovolných, nesouvisejících dat. Existují zde také rozdíly v očekávání od těchto služeb. Zatímco databáze musejí být konzistentní v každém okamžiku (například když banky sledují pohyby peněz), poskytují izolované transakce a trvalé zápisy. Souborový systém poskytuje mnohem menší záruky týkající se konzistence, izolace a trvanlivosti dat. Databáze používá sofistikované algoritmy a protokoly pro ukládání a právě díky těmto algoritmům se databáze stávají robustnější a také dražší pokud jde o náklady na zpracování a ukládání dat. Z tohoto důvodu se stávají souborové systémy jako ideální volba pro data, která nevyžadují žádné větší záruky.[59]

Základní fakta

- U souborového systému se soubory ukládají lokálně, kdežto u databáze se data ukládají přímo do ní.
- Souborový systém ukládá soubory na dočasná místa, zatímco databáze ukládá data do přehledného a trvalého úložiště v databázi.

- Souborový systém umožňuje pouze základní operace s daty. Databáze poskytuje mimo vkládání a mazání také zobrazení dat, úpravy dat a další.
- Data v souborovém systému jsou přístupná z jednoho, nebo více různých souborů. Přístup k datům v databázi je řešen přes tabulky.
- Pro ukládání všech vztahů mezi adresáři v souborovém systému se využívá správce souborů. Databáze ukládá vztahy mezi jednotlivými daty do strukturovaných tabulek.
- Data v databázích jsou bezpečnější ve srovnání s daty v souborech.[59]

Současné trendy

Vzhledem k tomu, jak technologie postupují kupředu a hlavně data čím dál více a rychleji přibývají na svém objemu, souborové systémy začínají obsahovat funkce, které dříve byly pouze doménou databází (různé transakce s daty, pokročilé dotazování). Databáze zase na druhou stranu kvůli jejich robustnosti pomalu upouští od rozsáhlé konzistence, izolovanosti a trvanlivosti zápisu.[58]

2.2 Distribuovaná databáze

Distribuovanou databázi představuje množina databází, která je uložena na několika počítačích. Ovšem uživateli se jeví jako jedna databáze. Dle zdroje [5] má takováto databáze tři hlavní vlastnosti:

- **Transparentnost** – z pohledu uživatele se databáze chová, jakoby všechna data byla zpracovávána na jednom serveru v lokální databázi. Uživatel používá shodné příkazy pro lokální i vzdálená data, nespecifikuje místo uložení dat. O to se stará distribuovaný SŘBD.
- **Autonomnost** – s lokální bází dat, která patří do distribuované databáze, uživatel pracuje nezávisle na ostatních databázích. Lokální databáze pracuje funkčně samostatně. Propojení do jiné distribuované databáze se zřizuje v případě potřeby dynamicky. Distribuovaná databáze nemá žádný centrální uzel nebo proces odpovědný za řízení funkcí celého systému. Díky tomu se výrazně zvyšuje odolnost systému proti výpadkům jeho částí.
- **Nezávislost na architektuře počítačové sítě** – distribuovaná databáze podporuje různé typy lokálních i globálních architektur počítačových sítí. Pro komunikaci se používá jazyk SQL.

2.3 Distribuovaný souborový systém

Jedná se o sdílený (distribuovaný) systém souborů, často označován jako DFS (Distributed File System), který se používá pro clustery. Umožňuje uživateli pracovat se vzdálenými daty pohodlně a jednoduše, jako by to byla data uložená na lokálním disku. Tento systém nemá odlišenou serverovou a klientskou část. Pro zamykání souborů využívají většinou speciálního správce. DFS poskytuje transparentnost umístění a redundanci dat, čímž zajišťuje lepší dostupnost sdílených dat. Dále také zvyšuje odolnost vůči chybám a přetížení sítě.[17]

Pokud uživatel přistupuje k položce v adresáři DFS, pak ve skutečnosti přistupuje pouze k odkazu. DFS jej poté transparentně přesměruje na správný zdroj. Proti výpadkům sítě a poruchám na jednotlivých uzlech DFS využívá replikaci dat. Tudíž pokazí-li se jeden uzel clusteru, bude vyjmut a potřebná data pro výpočet se načtou z repliky, která je umístěna na jiném, nejblíže položeném počítači clusteru. Jelikož si vybírá nejdosažitelnější kopii, zvyšuje tak rychlost a spolehlivost systému. Data tak mohou být za běhu jednoduše přesouvána mezi jednotlivými uzly, aniž by uživatel cokoliv zpozoroval. Dále se také může stát, že uživatel pracuje se souborem, který je uložen na uzlu, kde dochází kapacita. Systém soubor během práce uživatele přesune na jiný a uživatel vůbec nic nezpozoruje. Je to také ideální řešení pro rozkládání zátěže mezi jednotlivými uzly. Ekonomicky výhodné u DFS je zapojení vícera levnějších počítačů do clusteru, než menší počet drahých superpočítačů.[54]

Shrnutí druhé kapitoly

Jisté je, že pro velké objemy ať už prostorových, či neprostorových dat, dnes běžné relační databáze nestačí. Jsou velmi robustní a ukládání takových objemů je velice zdlouhavé. Řešení se ubírá k jednoduchosti těchto systémů, jako jsou distribuované souborové systémy. Nemají sice takové možnosti jako relační databáze, ovšem jsou velice rychlé a umožňují nasazení na více počítačovém výpočetním clusteru.

3 PROBLEMATIKA VELKÝCH OBJEMŮ DAT

Problematiku velkých objemů dat řeší nejenom obrovské IT firmy jako je Google, Microsoft, IBM ale i další. V dnešní době se musí s touto problematikou vypořádat i mnohem menší firmy a organizace. Zdroj [11] udává, že data, která vznikají každý den, přesahují 2,5 exabytů (což odpovídá zhruba milionu terabytů). Dle zdroje [10] pouhý Twitter během roku 2011 zaznamenal v průměru více jak 200 milionů uživatelských příspěvků za den. Ve stejném časovém období generoval Facebook denně přes 130 terabytů. Podle zprávy IBM z roku 2013 bylo dokonce 90 % objemu veškerých dat vygenerováno během posledních dvou let. Například během letu přes Atlantský oceán čtyřmotorovým letadlem je vygenerováno asi 640 terabytů. Podobné to je i v jiných odvětví jako je energetika či telekomunikace.

Pro problematiku velkých objemů dat vznikl název „**big data**“, přeloženo jako obrovská data. V některých českých textech se pojem big data ani nepřekládá. Pro sjednocení názvosloví se tato práce bude držet termínu „**velké objemy dat**“. Ovšem definic tohoto pojmu existuje více. Dle zdroje [20] poradenská firma Gartner považuje velké objemy dat za objemy, které se již nedají ukládat a zpracovávat na běžně používaném softwaru a hardwaru v rozumném čase. Velké objemy dat se staly nejvíce diskutovaným problémem během posledních dvou let. Při pohledu na akademické publikace bylo zjištěno, že více než 70 % všech hodnocených prací, které se zabývaly big daty, byly publikovány během posledních dvou let. Velkými daty jsou zasaženy téměř všechny obory. Především se jedná o vědní obory, vládní organizace, odvětví jako jsou média, telekomunikace, zdravotnictví, nebo strojírenství. Všechny tyto obory čelí obrovskému množství dat a novým technologiím na ukládání, zpracování a analýzu těchto dat.[11]

Velké objemy dat nepředstavují pouze objemná data, jako jsou gigabyty, terabyty, nebo petabyty, ale také rychlost jejich tvorby a přenosu z hlediska různorodosti typů. Takováto data se obvykle ukládají do datových skladů. Mění se také struktura ukládaných souborů. Nyní už není kladen takový důraz na centralizaci a vysokou strukturovanost jako dříve. V současné době se data ukládají volně, bez velkých struktur, jsou vysoce distribuovaná a mají vzrůstající objem.[8]

Velké objemy dat jsou založeny na čtyřech pojmech začínajících písmenem „V“. Jedná se o volume (objem), velocity (rychlost), variety (různorodost) a veracity (věrohodnost).

Volume znamená objem v tom smyslu, že objem dat narůstá exponenciálně. Zvýšení tohoto objemu je způsobeno díky velkému množství informačních transakcí a prostřednictvím vznikajících nových datových typů.

Velocity (rychlost) – představuje myšlenku, jak rychle data vznikají a jak rychle by měla být zpracována. Objevují se úlohy, které vyžadují okamžité zpracování velkého objemu průběžně vznikajících dat. Například data produkované kamerou.

Variety – různorodost, neboli variabilita znamená, že velké objemy dat bývají nestrukturovaná a plná různých datových typů jako jsou e-maily, hierarchické údaje, naměřené údaje, videa, obrázky, zvukové obrazy a spoustu dalších. Podle zdroje [8] obecně platí, že více než 80 % všech dat v podniku má nestrukturovanou formu.

Veracity popisují věrohodnost dat. Existuje totiž nejistá věrohodnost dat v důsledku jejich nekonzistence, nejasnosti, neúplnosti a podobně. Příkladem mohou být data čerpaná ze sociálních sítí.[34] [9]

Vedoucí pracovníci a ředitelé firem si uvědomují důležitost IT řešení, které si poradí s obrovským objemem informací, rychlostí a různorodostí ukládaných dat. Ovšem mnoho z nich považuje tuto problematiku v rámci jejich organizace za neřešitelné. Podle celosvětového průzkumu, provedeného firmou HP, uvedla více než polovina dotazovaných manažerů, že společnosti, ve kterých pracují, nedisponuje vhodným řešením pro vytěžení informací a analýzu velkých objemů dat. Dále prozradili, že jim chybí také znalosti a strategie, která by sjednotila všechny komponenty a propojila původní a nové struktury.[27]

Velké objemy prostorových dat

Doposud zde byly zmíněny informace o obecných velkých objemech dat. Tato část bude zaměřena pouze na prostorová data v kontextu s big daty.

Jak již bylo v 1. kapitole napsáno, prostorová data představují informace, které se vztahují k určitým místům v prostoru a pro která jsou na potřebné úrovni rozlišení známé lokalizace těchto míst. Uživatel pracující s prostorovými daty tedy ví, kde se data vyskytují a je schopen určit jejich polohu.

Dle zdroje [60] zhruba 80 % veškerých informací obsahuje prostorovou komponentu. Mnoho funkcí je tak závislých na prostorových datech. Z tohoto důvodu také vznikají nové integrační služby nebo služby spojené se sdílením prostorových informací. Nutno také

podotknout, že v současné době existuje mnoho faktorů, jako jsou: nízká systémová bezpečnost, slabý hardwarový výkon, nákladná údržba, náklady na software a spoustu dalších faktorů, které brzdí vývoj služeb pracujících s prostorovou informací.

Mezi nejrozsáhlejší prostorová data patří především snímky získávané z dálkového průzkumu Země. Dle zdroje [41] organizace EUMETSAT, která buduje a provozuje operativní meteorologické družicové systémy pro potřeby dvaceti evropských států, uvádí, že velikost přijímaných dat za hodinu činí 540 MB v téměř 7 000 souborech. Denní objem všech původních dat představuje kolem 14 GB. Organizace doporučuje ukládat snímky ve formátu JPEG. Tím se zredukuje denní objem na 1 GB.

Shrnutí třetí kapitoly

Klíčové označení pro problematiku velkých objemů dat se nazývá big data. V některých českých textech se tento pojem ani nepřekládá. Ovšem pro zachování sjednoceného názvosloví je pojem „big data“ brán jako synonymum pojmu „velké objemy dat“. Dále v práci se bude vyskytovat pouze jako „velké objemy dat“. Denně vzniká 2,5 exabytů dat. Velké objemy dat bývají nestrukturovaná a různorodá. Podle celosvětového průzkumu zhruba polovina firem si je vědoma, že nedisponují vhodným řešením pro velké objemy dat. Dalším zajímavým zjištěním bylo, že zhruba 80 % veškerých dat obsahuje prostorovou informaci. Z této kapitole je patrné, že v blízké době se bude muset zabývat čím dál více společností a institucí problematikou velkých objemů dat.

4 UKLÁDÁNÍ DAT DO DATACENTER

Spolu s ukládáním velkých objemů dat úzce souvisí také zpracování a práce s těmito uloženými daty. Proto v této kapitole budou popsány nejen současné trendy ukládání velkých ojbemů dat, ale bude zde také nastíněná technologie pro zpracování.

V současné době se do popředí dostává technologie zvaná **cloud computing**. Je to technologie, která využívá techniky k vytváření nových řešení. Stříhání HD videa na starém počítači, mobilní telefon vybaven 500 GB datového prostoru, spouštění jakéhokoli softwaru a vyvolání dat na jakémkoli PC, to vše tato technologie, nazývaná cloud computing, umožňuje. Cloud, přeložený jako oblak, představuje virtuální místo na internetu, kde se odehrávají veškeré činnosti (výpočty, ukládání, zpracování, odesílání výsledků). Samozřejmě je důležité mít kvalitní připojení k internetu, podle toho jakou cloudovou službu zrovna uživatel potřebuje využívat.

Základní myšlenkou cloud computingu je, že všechny aplikace pracují přímo na webu a to od jednoduchého softwaru až po operační systémy. Uživatel si nemusí obstarávat drahý hardware, nemusí řešit aktualizace a jiné nepříjemnosti s tím spojené.[14]

4.1 Datová centra

V této kapitole bude čtenář seznámen s prostory a zařízeními, kam se velké objemy prostorových i neprostorových dat ukládají a kde se dále zpracovávají.

Dle zdroje [15] datová centra představují zdroje, kde jsou uložené veškeré weby, e-maily a různá digitální (ať už prostorová či neprostorová) data. I v České republice v současné době existuje několik kvalitních datacenter. A jak se pozná kvalitní datacentrum? Dle následujících vlastností:

Konektivita

Konektivita představuje rychlost (propustnost) k připojení do páteřní sítě internetu. Většina velkých datových center nabízí srovnatelné parametry. Proto je zapotřebí podívat se na ostatní vlastnosti a služby, které datacentra nabízejí.

Prostředí

Pro dlouhé a spolehlivé fungování běžících procesů je nejideálnější konstantní teplota 19 - 23 °C a vlhkost kolem 40 – 60 %. To vše zajišťují speciální klimatizační systémy.

Důležité také je udržování bezprašného prostředí. To zabezpečují filtry. Další důležitou vlastností pro ideální prostředí je, aby datacentrum bylo umístěno v lokalitě bez otřesů a hrozby vyplavení vodou.

Protipožární ochrana

Všechna datová centra by měly mít nainstalovaný elektronický požární systém, jehož součástí jsou čidla a multisenzory. V případě zpozorování kouře nebo vysoké teploty spouští automatický hasící systém, zpravidla se zhášecím plynem FM200, aby nedošlo k poškození okolních serverů. Tato substance je velice nákladná. Náklady na pokrytí jedné místnosti přesahují milion korun. Je ovšem nejšetrnější k hašení serverů, protože způsobuje tepelný rozklad molekul odebráním tepla a vytlačením kyslíku z bezprostřední oblasti hoření.

Elektřina

Datacentrum musí být připraveno na výpadky elektrické energie. Standardem je tedy napájení datacentra ze dvou nezávislých elektrických sítí. Náhlé výpadky jsou jištěny záložními zdroji napětí, které zároveň pomáhají při odstraňování některých anomálií v elektrické síti. Ocitne-li se datacentrum úplně bez proudu, jsou poslední možností naftové dieselagregáty.

Zabezpečení

Jedná se o jednu z nejdůležitějších vlastností datacentra. Nikdo si totiž nepřeje, aby ke svým datům v datacentrům přistupovala nějaká neoprávněná osoba. Jednotlivé racky jsou uzamčeny specializovaným klíčovým systémem a zabezpečeny proti vylomení. Neméně nebezpečné je také zabránění vniknutí do celého datacentra. Z tohoto důvodu se datacentra stavějí bez oken, každá návštěva musí být hlášena a prokázána u vstupu svým občanským průkazem, nebo ID. Samozřejmostí je i přísně střežený přístup, monitorování a systém čipových karet. V datacentrech se také pohybují příslušníci bezpečnostní služby a také jsou datacentra pokryta sítí průmyslových kamer.[15]

4.2 Spotřeba energie v datacentrech

Datacentra vyžadují opravdu hodně energie. Každé hlavní výpočetní středisko spotřebovává stejné množství elektřiny jako česká metropole. Datacentra jsou proto zřizována v místech, kde je možné zajistit zásobování energií na finančně přijatelné úrovni. Společnosti přitom příliš nespoléhají na obnovitelné zdroje energie.[33]

Dominantní část spotřeby je spojena s přímou spotřebou pro napájení serveru. Tato spotřeba roste se zvyšujícím se taktem procesoru. Možností úspor při zachování či spíše neustálém růstu výkonu zdánlivě mnoho není. Výrobci procesorů ale už pociťují potřebu reagovat a nechávají své produkty „zelenat“.[31]

Celková spotřeba energie

Hlavními odběrateli elektřiny jsou počítačová střediska – na celém světě spotřebuje cloud computing dvojnásobné množství elektřiny než celá Velká Británie, což vystihuje Příloha C, kde je také vložen přehled největších datacenter IT firem.[12]

4.3 „Zelená“ datacentra

Část energie na provoz datového centra je možné získávat z obnovitelných zdrojů. Možností ovšem není příliš mnoho. Využití větru například v husté městské zástavbě není zrovna ideální řešení. Z tohoto důvodu se v současné době staví datacentra v odlehlých místech od měst. Další možností jsou solární panely a jejich umístění na střechu datacentra. I z tohoto důvodu se datacentrum nebudují ve městě, kde se vyskytuji smogové znečištění a účinnost solárních článku by byla poté menší. Solární článek na venkově je o 30 % výkonnější nežli ve městě, kde žije jeden milion obyvatel. Ekologicky výhodné může být i využití odpadního tepla z datového centra. Ovšem vzhledem k tomu, že odpadní teplo je nízkoenergetické, není možností mnoho. Prakticky se aplikují pouze na částečné vyhřívání objektů nebo zařízení. Příkladem jsou obytné domy, venkovní bazény, zahradnické skleníky atd. Jelikož se datacentrum umisťuje mimo záplavové oblasti, ani vodní elektrárna není na pravém místě. Hlavním důvodem je to, že datová centra musí být v bezprostřední blízkosti datových uzlů, nebo na místech dobrého připojení.

Dalším způsobem jak „zazelenat“ datová centra je nákup tzv. zelené elektřiny. Datacentra v zemích s vysokou mírou citlivosti veřejnosti vůči ekologickým otázkám, jako je Německo a Holandsko, se skutečně snaží získat PR body deklarovaným nákupem části spotřeby elektrické energie z obnovitelných zdrojů. Nicméně její nákup je víceméně pouze účetní operace, jelikož nemá s realitou mnoho společného. Všechna energie se totiž v přenosové soustavě míchá, a tak odběratel spotřebovává v drtivé míře energie z jádra a hnědouhelných elektráren.[31]

Shrnutí čtvrté kapitoly

Ve čtvrté kapitole se čtenář dočetl, že ukládání dat je silně spojeno s jeho zpracováním. Hlavním trendem současné doby pro zpracovávání dat je cloud computing. Dále zde bylo vysvětleno ukládání velkých objemů dat do datacenter, které musí mít kvalitní propustnost, ideální teplotní a vlhkostní hodnoty zajišťované klimatizací a v neposlední řadě protipožární ochranu a zabezpečení. Jelikož je datacentrum obrovský spotřebitel elektrické energie, začínají se řešit otázky spojené s alternativním získáváním elektrické energie. Sympatickou cestou jde společnost Google Inc., která si vyrábí elektřinu vlastní cestou pomocí obnovitelných zdrojů. Z hlediska zjištění energetické náročnosti provozu datacentra se autor pokusí, po zvolení softwarového řešení pro ukládání velkých objemů dat, zaměřit i na energetickou spotřebu při ukládání prostřednictvím vybraného řešení.

5 VELKÉ OBJEMY DAT VE SPOLEČNOSTI GOOGLE

Google Inc. ukládá svá data (dále se bude vyskytovat název společnosti zkráceně jako Google) do vlastních datacenter. Datacentra jsou podrobně popsána v kapitole 4. Pro připomenutí, jedná se o úložná zařízení, která obsahují až tisíce serverů. Každý server musí být poháněn elektrickou energií, proto se Google rozhodl jít cestou výroby energii vlastními obnovitelnými zdroji.[38]

Filozofie Googlu je založena na jednoduchém principu. Místo toho, aby nakupovali drahé superpočítače na zpracování a ukládání dat, sestavují si vlastní počítače z levných součástek do takzvaných clusterů (skupina propojených počítačů, většinou LAN sítí). Jelikož se každý hardware opotřebovává a kazí, i hardware Googlu má poruchy. V objemech technického vybavení, kterými Google disponuje, je tento problém o to složitější. Společnost se na tyto skutečnosti připravuje tak, že se snaží predikovat chyby. Dle zdroje [24] společnost Google disponuje 37 rozměrnými datovými centry. Tato datacentra jsou rozmístěna po celém světě, především tam kde je levnější elektřina. Jak již bylo zmíněno, datacentra mají obrovské energetické nároky a proto Google staví vlastní větrné a solární elektrárny. V jednom datovém centru může být až sto tisíc serverů. V roce 2007 analytická společnost Gartner odhadla počet všech serverů Googlu na jeden milion. Dnes už bude toto číslo několika násobně vyšší. Postačujícími prvky hardwaru potřebných pro fungování počítačů v datových centrech je procesor, operační paměť, samozřejmě disk a síťové propojení. Na základní desce nejsou potřeba žádné grafické či zvukové karty, pouze síťová.[38]

Také programové vybavení je odlišné od softwaru, který využívá běžný uživatel. Google používá vlastní BIOS a speciálně upravený operační systém Linux. V operačním systému jsou nainstalovány speciální aplikace pro správu serverů. Nepostradatelnou aplikací je například obsluha senzorů teploty. Program monitoruje na jakém počítači se co děje a pokud se někde stane porucha, snaží se chybu vyhodnotit a určit, kde mohlo k problému dojít. Další důležitou aplikací je program pro zajišťování úspěšného přenosu dat. Při blížícím se konci úlohy aplikace vyvolá spuštění více počítačů, aby byl proces úspěšně a bezpečně dokončen.[38]

Google file system

Google file system, označován jako GFS, je škálovatelný síťový souborový systém, který využívá Google pro práci s velkými objemy dat. Mezi hlavní výhody GFS patří především již zmíněná škálovatelnost, spolehlivost a dostupnost.[23]

Přehled systému

- Hardware, se kterým GFS pracuje je sestaven z mnoha levných komponent, které často selhávají. Jedná se především o paměti, disky napájecí zdroje. Proto je potřeba jednotlivé uzly neustále monitorovat a zotavovat je z jejich selhání.
- Systém je schopný ukládat miliony souborů o velikosti přesahující 100 MB. Často musí systém ukládat i několika-gigabytové soubory a proto musí být zpracovány efektivně. Samozřejmě GFS ukládá i malé soubory, pro které se nemusí systém optimalizovat.
- Běžné operace se soubory představuje sekvenční čtení, typicky stovek kB a náhodné čtení několika kB na určitém místě. Zápisy se provádí na konec souboru pomocí atomického zápisu. Díky atomickému zápisu, kde dochází k neoddělitelnosti zápisu, mohou být soubory využívány jako fronty mezi producentem a konzumentem nebo pro slučování dat z více zdrojů.
- Systém je implementován tak, aby několik uživatelů mohlo používat jeden soubor současně.[23]

Rozhraní

GFS nabízí klasické rozhraní souborového systému, ačkoli není zde implementováno nějaké standartní API. Soubory jsou hierarchicky organizovány v adresářích a přistupuje s k nim pomocí cest. GFS podporuje běžné operace, jako jsou: create (vytvoř), delete (smaž), open (otevři), close (zavři), read (čti) a write (zapiš). Dále pak další dvě: snapshot a record append. Snapshot vytváří rychlé kopie souboru nebo stromu podadresáře (řešeno pomocí Copy-on-write). Record append GFS využívá pro současný zápis více klientů na konec souboru.[23]

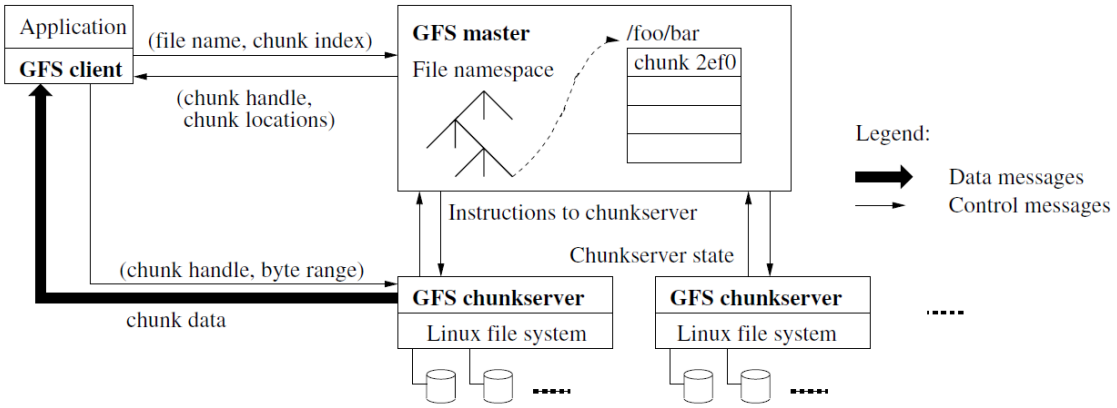
Architektura

Počítače (servery) Googlu jsou spojeny do několika clusterů. Jeden cluster může obsahovat až tisíce počítačů. Každý cluster se skládá z jednoho master serveru a zbytek serverů jsou označovány jako chunk servery, na které se ukládají data. Soubory, které Google ukládá, se rozdělují do tzv. chunků (kusů) pevné délky. Každý chunk je identifikován 64 bitovým číslem, které mu je přiděleno master serverem v okamžiku vytvoření. Chunk servery ukládají chunky na lokální disk jako běžné linuxové soubory. Jeden chunk zabírá velikost 64 MB, což je o dost více než v běžném souborovém systému. Data čtou a zapisují na základě požadavků specifikovaných číslem chunku a rozsahem. Z důvodu spolehlivosti je

každý chunk uložen na několik míst zároveň. Standardně se ukládají tři kopie. Jednotlivé kopie se poté fyzicky rozmísťují do různých datacenter podle intenzity čtení, podle poměru čtení/zápis a podle množství požadavků v určitém regionu. Pomocí zálohování si systém dokáže automaticky poradit s výpadky hardwaru a nepotřebuje k tomu speciální hardwarové řešení.

Master server spravuje veškerá metadata, která zahrnují jmenný prostor, údaje o přístupových právech, mapování souborů na chunky a aktuální umístění chunků. Dále master server umožňuje uživatelům přistupovat k jednotlivým chunkům a pracovat s nimi. Zajišťuje také mazání osiřelých chunků a přemísťování z jednoho serveru na jiný. Poslední důležitou prací master serveru je pravidelná kontrola chunk serveru. V určitých intervalech zasílá zprávy a kontroluje, zda chunk server na ně odpovídá či nikoliv. Skrze master server nejsou prováděny žádné operace s daty. Nicméně i kdyby došlo k poruše na master serveru, je GFS vybaven záložním master serverem.

Po několika úspěšných zápisech GFS garantuje integritu dat. Pak tedy změněná oblast bude konzistentní a bude obsahovat data z posledního zápisu. To zajištuje aplikování změn ve stejném pořadí na všechny repliky. Při aktualizaci souborů master server přiřadí chunk jedné z replik, kterou poté označí jako primární. Na této primární replice se provedou změny a ostatní repliky se změní podle ní. Data mohou být také poškozena poruchou hardwaru. Master server odhalí nefungující chunk server, při pravidelných kontrolách chunk serverů. Poškození dat se ověřuje pomocí kontrolních součtů. Pokud se objeví problém, data jsou obnovena z nepoškozených replik. Chunk je na vždy ztracen, pokud jsou všechny jeho repliky poškozeny dříve, než GFS stihne zareagovat. Architekturu GFS vystihuje Obrázek 1.[23]



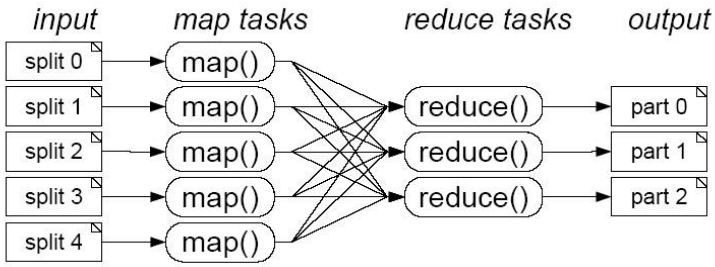
Obrázek 1: Architektura GFS

zdroj: [23]

MapReduce

MapReduce je další nepostradatelným systémem na zpracování velkých objemů dat, který Google využívá. GFS jak již bylo zmíněno je distribuovaný souborový systém a MapReduce na ním provádí výpočty. Podstatou tohoto systému je, že umožní spustit jeden rozsáhlý proces na velkém množství počítačů a tím dojde několikanásobnému urychlení. Funkce Map a Reduce vždy pracují s dvojicí hodonota – klíč. Každá z funkcí přijímá tuto dvojici jako vstup a poskytuje jako výstup. Výstup modelu MapReduce obvykle obsahuje více výstupních prvků podle definovaných klíčů. Agregují se všechny hodnoty se stejným klíčem nezávisle na ostatních. Výstupy funkce Map se nazývají Mappers. Příkladem metody Map může být převod textového řetězce na nový textový řetězec s velkými písmeny.

Metoda Reduce spojuje vstupní datové prvky v jednu výstupní hodnotu podle zadaného parametru. Běžným příkladem funkce je sumarizace dat. Výsledky těchto funkcí se nazývají Reducers. Podstatu MapReduce vystihuje Obrázek 2.[4]



Obrázek 2: MapReduce model

zdroj: [4]

Důležitou vlastností, kterou MapReduce disponuje, je přidělování úloh jednotlivým počítačům. Pokud systém zjistí, že nějaký počítač se začne přehřívat nebo má jakýkoli jiný problém a nebude moci danou úlohu zpracovat plným výkonem, přidělí úlohu jinému počítači a ten jí dokončí za něj.[38]

Sawzall

Je silně specializovaný procedurální programovací jazyk, který Google využívá pro obsluhu Map Reduce. Sawzall je předurčen pro zpracování logů. Log (žurnál) se v informatice označuje jako záznam, nebo soubor záznamů, který složí programům jako informace o průběhu činnosti a běhu programu. Nutno také podotknout, že Sawzall běží na neomezeném množství počítačů.[45]

6 VOLBA VHODNÉHO PROGRAMOVÉHO ŘEŠENÍ

Způsobů, jak ukládat velké objemy prostorových dat a následně je efektivně zpracovávat, existuje hned několik. Cílem této práce je volba vhodného programového řešení pro malý univerzitní cluster, který je vybaven obyčejnými počítači.

6.1 Stanovení kritérií

Jelikož univerzita nemá zájem o placené řešení, je možné vybírat pouze z open source řešení. Open source řešení existuje dle zdrojů [2] a [43] hned několik, proto bylo vhodné použít metody vícekriteriálního rozhodování. Nejdříve byla vytvořena sada deseti kritérií, které byly vytvořeny z potřeb univerzitního clusteru a z nabytých znalostí během rešerše.

Popis jednotlivých kritérií:

Distribuované ukládání (A) - Pro potřeby ukládání je potřeba, aby se data ukládala prostřednictvím distribuovaného ukládání.

Paralelní zpracování (B) - Univerzitní cluster nebude složit pouze pro ukládání ale hlavně také pro zpracování. Aby bylo využito všech počítačů, je potřeba paralelní zpracování dat.

Nenáročnost na hardware (C) - Univerzita si nemůže dovolit drahé výkonné superpočítače. Je potřeba řešení pro běžný hardware.

Podpora strukturovaných dat (D) - Jelikož se bude na univerzitě ukládat zpracovávat především strukturovaná data.

Zálohování dat (E) - Zálohováním se zabrání případné ztrátě dat.

Cloud computing (F) - Nejedná se o úplně nutné kritérium, ovšem pokud programové řešení tuto možnost nabízí určitě dobře pro univerzitu.

Grafické výstupy (G) - Grafický výstup po zpracování dat, především možnost reportů.

Aktualizace programu (H) - Každé kvalitní programové řešení by mělo být aktualizované.

Podpora nestrukturovaných dat (I) - V případě, že by univerzita potřebovala ukládat a zpracovávat nestrukturovaná data. Jelikož se jedná o velké objemy dat, programové řešení by na to mělo být připraveno.

Škálovatelnost (J) - Možnost rozšíření, či redukování clusteru dle aktuálních potřeb.

6.2 Stanovení vah kritérií

Váhy jednotlivých kritérií byly stanoveny metodou párového srovnání (Fullerova metoda), viz Tabulka 1. Tato metoda zjišťuje počet preferencí vzhledem ke všem ostatním kritériím. Vždy se zjišťuje preference kritéria uvedeného v řádku před kritériem ve sloupci. Poté se vyplní 1, nebo 0. Aby nedošlo k vyloučení kritéria s nulovou preferencí, byl použit vzorec dle zdroje [40] :

$$k_i = n + 1 - p_i ,$$

kde $\mathbf{k_i}$ určuje nenormovanou váhu, **n** je počet kritérií a $\mathbf{p_i}$ je pořadí i-tého kritéria v jeho preferenčním uspořádání.

Nenormované váhy poté normujeme dle vztahu [40] :

$$v_i = nv_i \Big/ \sum_{j=1}^{m} nv_j , \text{ pro } i \text{ a } j = 1, 2, \ldots, m,$$

kde nv_i zde představuje k_i.

Nutno podotknout, že všechna kritéria jsou maximalizační.

Tabulka 1: Metoda párového porovnání - stanovení vah kritérií

	A	B	C	D	E	F	G	H	I	J	Preference v řádku	Preference ve sloupci	Celkem (fi)	Pořadí kritéria	K_i	Váhy
A		0	1	0	1	1	1	1	1	1	7	0	7	3	8	0,145
B			1	0	1	1	1	1	1	1	7	1	8	2	9	0,164
C				0	1	1	1	1	1	1	6	0	6	4	7	0,127
D					1	1	1	1	1	1	6	3	9	1	10	0,182
E						1	1	1	1	1	5	0	5	5	6	0,109
F							1	0	1	1	3	0	3	7	4	0,073
G								0	0	0	0	0	0	10	1	0,018
H									1	1	2	2	4	6	5	0,091
I										0	0	1	1	9	2	0,036
J											0	2	2	8	3	0,055
															55	1

Zdroj: vlastní zpracování

6.3 Ohodnocení alternativ

Jak již bylo zmíněno, alternativ open source řešení existuje hned několik. Nejdříve zde budou alternativy představeny, poté budou ohodnoceny a na závěr bude vybrána optimální alternativa.

Open source alternativy:

Apache Hadoop – jedná se o Java open source framework pro distribuované ukládání a zpracování velkých objemů dat. Informace o tomto řešení byly čerpány z [4] .

Apache Spark – další produkt od společnosti Apache. Taktéž Java open source Framework využívající Hadoop jako základ. Využívá tzv. in-memory zpracování, tedy data ukládá do operačních pamětí. Informace o tomto řešení byly čerpány z [52] .

HPCC – velký konkurent Apache Hadoop. Open source Framework pro ukládání a zpracování velkých objemů dat od společnosti LexisNexis. Informace čerpány z [28] .

Grid Gain – další zástupce in-memory zpracování, který může posloužit jako akcelerátor Apache Hadoop nebo samostatné programové řešení. Informace čerpány z [46] .

Oracle – open source in-memory databáze s názvem Oracle TimesTen In-Memory Database a Oracle In-Memory Database Cache. Informace o tomto řešení byly čerpány z [44] .

Sector/Sphere – další programové řešení pro distribuované ukládání velkých objemů dat a následné zpracování, které je schopné operovat prostřednictvím sítě WAN.[51]

InfiniDB – Jedná se o databázi pro velké objemy dat s pokročilými analýzami. Informace o tomto řešení byly čerpány z [29] .

Tabulka 2: Ohodnocení variant

	A	B	C	D	E	F	G	H	I	J	Skóre
Apache Hadoop	1	1	1	1	1	1	1	1	1	1	1
Apache Spark	0	1	0	1	1	1	1	1	1	1	0,728
HPCC	1	1	1	1	1	1	1	1	1	1	1
Grid Gain	1	1	0	1	1	1	1	1	1	1	0,873
Oracle	1	0	0	1	1	1	1	1	1	1	0,709
Sector/Sphere	1	1	1	1	1	1	1	1	1	1	1
InfiniDB	0	1	1	1	1	1	1	1	1	1	0,855

Zdroj: vlastní zpracování

Z výsledku, který zachycuje Tabulka 2, je patrné, že vhodná open source řešení pro malý univerzitní cluster, jsou hned tři. Jedná se o Apache Hadoop, HPCC a Sector/Sphere. Z hlediska velikosti komunity a dostupnosti literatury o jednotlivých řešení, bylo rozhodnuto o výběru programu **Apache Hadoop**. Dalším ukazatelem pro výběr tohoto programu je, že bývá využíván jako základ velkých proprietárních řešení jako jsou IBM, DELL, Cisco a další.

7 APACHE HADOOP

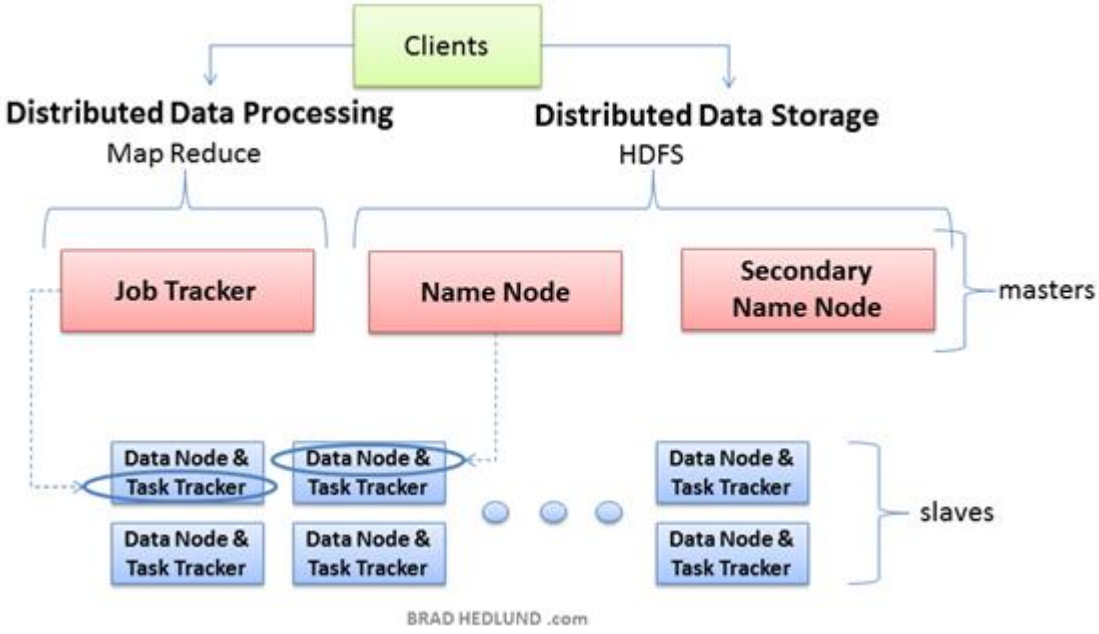
V předchozí kapitole byl zvolen za vhodné řešení pro ukládání velkých objemů prostorových dat program Apache Hadoop a z tohoto důvodu se následující kroky diplomové práce budou zabývat tímto programem. V dalších kapitolách bude dopodrobna popsána jeho architektura, princip a dále testování možností tohoto softwaru v univerzitní laboratoři, která se skládá z pěti běžných počítačů.

Apache Hadoop (dále jen Hadoop) je Java open source framework pro distribuované ukládání a zpracování velkých objemů dat. Hadoop je instalován na takzvané clustery. Pod pojmem cluster si může čtenář představit skupinu propojených počítačů, které spolu úzce spolupracují a na venek se chovají jako jeden počítač. Obvykle jsou propojeny počítačovou sítí. Podstata Hadoopu spočívá v ukládání dat na velké množství počítačů. Jedná se o počítače s běžným hardwarem, které tak nahradí výkony drahých superpočítačů a vyjdou mnohem levněji. Faktem je, že se zvyšujícím se počtem zapojených počítačů roste pravděpodobnost výskytu chyb. Může docházet k poruchám na jednotlivých součástkách v počítači. Dle zdroje [62] jsou velice poruchové pevné disky, zvláště v každodenním provozu. Především po dvou letech provozu, kdy končí záruka disků, značně roste selhávání. Může také docházet k výpadkům sítě. Možnosti paralelních výpočtů přiblíží následující příklad. Běžný pevný disk čte data rychlostí 100 MBps. Čtyř jádrový procesor poskytuje teoretickou rychlost čtení 400 MBps. Data o kapacitě 4 TB při této teoretické rychlosti budou načteny za 10 000 s, tedy zhruba 3 hodiny. Pokud zapojíme 100 samostatných strojů s polovičním výkonem, bude stejné množství dat načteno přibližně za 3 minuty. A právě na této filozofii je postaven Hadoop. Jak již víme z kapitoly 5, stejnou filozofii razí Google, kde se Hadoop inspiroval.[4] [25]

Hadoop se skládá ze dvou hlavních částí – pro ukládání a pro zpracování. Pro ukládání dat se využívá distribuovaný souborový systém **HDFS** (Hadoop Distributed File Systém) a zpracování probíhá paralelně na více uzlech ve dvou základních krocích. Jde o princip **MapReduce**. Funkce Map nejdříve úlohu rozdělí a funkce Reduce použije výstupy z funkce Map, udělá výpočet a spojí výsledky.

HDFS obsahuje **NameNode**, **Secondary NameNode** a **DataNode.** NameNode spolu se Secondary NameNode běží na hlavním PC (označován jako master) a DataNodes běží na všech ostatních počítačích zapojených do clusteru (takovýto počítač se označuje jako slave). Secondary NameNode slouží jako záloha NameNode, kdyby došlo k selhání hlavního

master uzlu. MapReduce obsahuje **JobTracker** a **TaskTracker**. První ze jmenovaných – JobTracker je obdobou NameNode s tím rozdílem, že má na starost zpracování dat a stejně tak je uložen na master uzlu. Tasktrackery jsou spuštěné na slave uzlech a jejich úkolem je provádět instrukce z JobTrackeru. Text z tohoto odstavce graficky znázorňuje Obrázek 3.



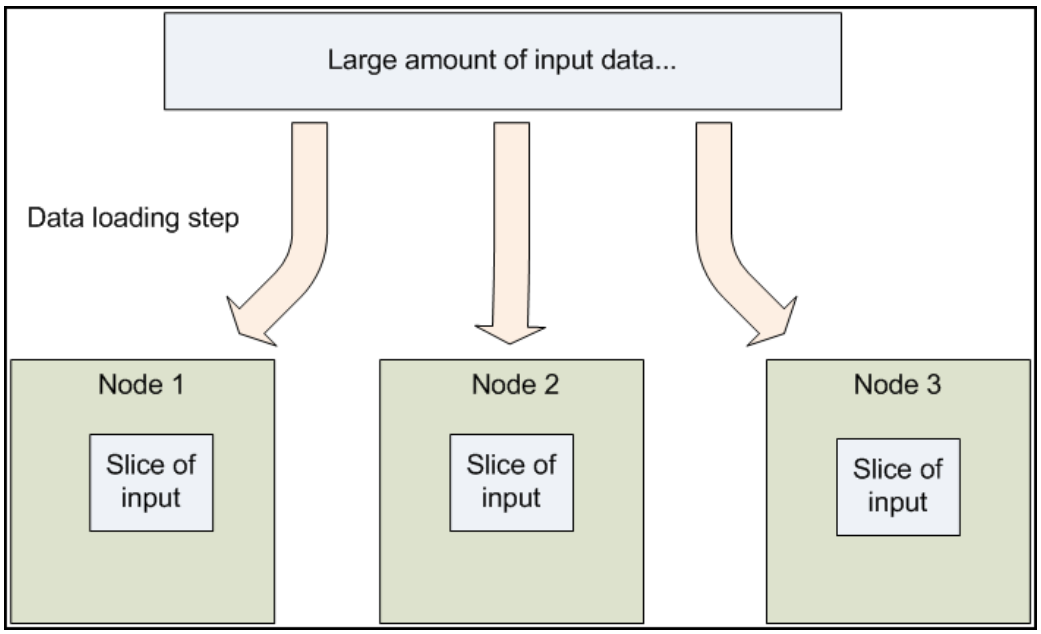
Obrázek 3: Role jednotlivých uzlů clusteru

zdroj: [56]

7.1 Hadoop Distributed File System

Jak již bylo zmíněno v předchozí kapitole, jedná se o distribuovaný souborový systém navržený pro práci na běžných počítačích. Do jisté míry je velice podobný běžným distribuovaným souborovým systémům, ovšem existuje zde několik rozdílů. HDFS soubory jsou ukládány redundantním způsobem, čímž je HDFS vysoce odolný vůči chybám. Je určen k nasazení na běžném hardwaru, je navržený pro držení velkého objemu dat a umožňuje práci paralelně spuštěným aplikacím.[56]

HDFS rozděluje velký objem dat do bloků. Každý blok je replikován na určitý počet bloků (podle nastavení replikace). Jednotlivé bloky jsou ukládány na různé uzly v clusteru. Díky tomuto opatření, selhání jednoho bloku nezpůsobí ztrátu přístupu k datům. Vhodné datové rozmístění snižuje vytížení šířky pásma, zabraňuje zbytečným síťovým přenosům a zvyšuje tak samotný výkon při zpracování. Schéma rozdělení datového souboru vystihuje Obrázek 4.



Obrázek 4: Rozdělení dat na bloky

zdroj: [4]

NameNode

Úkolem NameNode je řízení a správa ukládání dat na jednotlivé DataNodes. NameNode ukládá metadata, která obsahují informace o struktuře souborů a adresářů. Metadata jsou uloženy trvale na lokálním disku. Dále poskytuje informace o tom, ve kterých DataNodes jsou uloženy jednotlivé bloky dat. Stanice, na které NameNode běží, se nazývá HDFS master, protože podřízeným stanicím, na kterých běží DataNode, zadává příkazy. HDFS master zpravidla neobsahuje žádná data pro zpracování a tudíž se na ní neprovádí žádné náročné výpočty. Ovšem v případě malého clusteru je možné master stanici využít i jako slave stanici a zvýšit tak výpočetní výkon clusteru. Stanice, na které je NameNode spuštěn, je jediné místo v clusteru, kvůli kterému může dojít k selhání Hadoopu. NameNode také zajišťuje uživatelské rozhraní pro přístup k HDFS.[25]

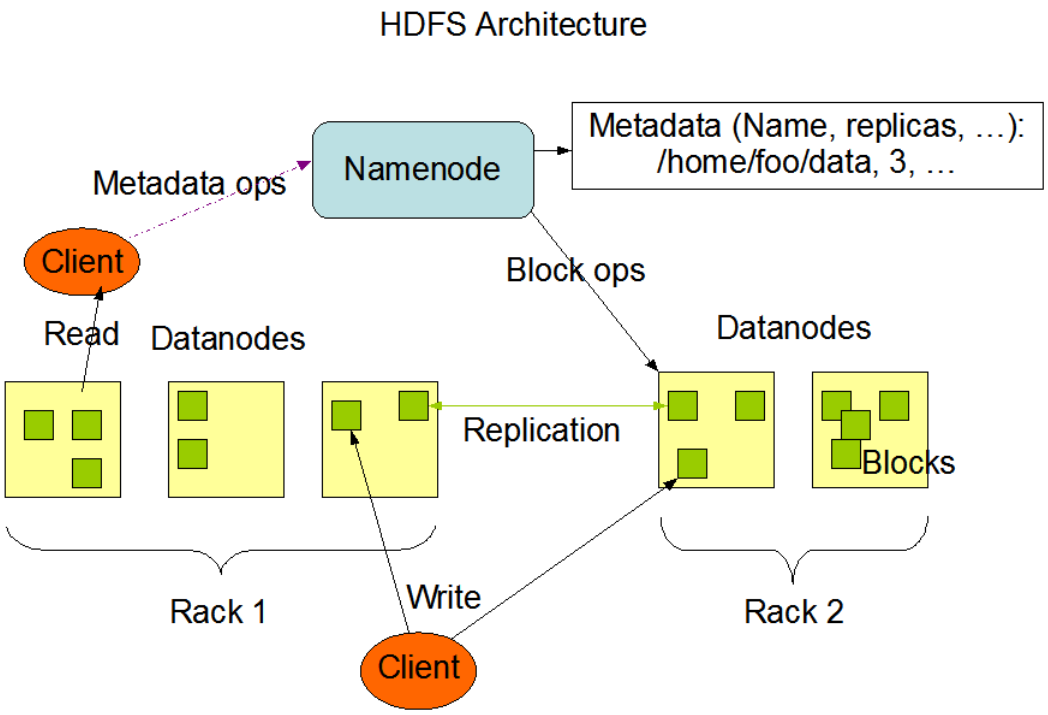
Secondary NameNode

Slouží jako záloha NameNode, kdyby došlo k jeho selhání. Seconadary NameNode (SNN) se v pravidelných intervalech připojuje na NameNode a provádí jeho zálohu. Velikost tohoto intervalu je možno nastavit v konfiguraci clusteru. Jak již bylo zmíněno selhání uzlu, na kterém je spuštěn NameNode a SNN, vede k výpadku celého Hadoop clusteru, tudíž je vhodné tento uzel zaopatřit kvalitním hardwarem.[25]

DataNode

Jeho úkolem je ukládat data do sdíleného souborového systému. Každý DataNode spravuje určitý blok úložiště systému HDFS. DataNode periodicky zasílá zprávu na NameNode se seznamem bloků, které obsahuje. Každá stanice vždy obsahuje pouze jeden DataNode. Jednotlivé DataNodes spolu vzájemně komunikují především kvůli replikaci dat, kde si posílají jednotlivé bloky dat. Díky replikaci je zajištěn bezproblémový provoz HDFS.[25]

Architekturu HDFS znázorňuje Obrázek 5.

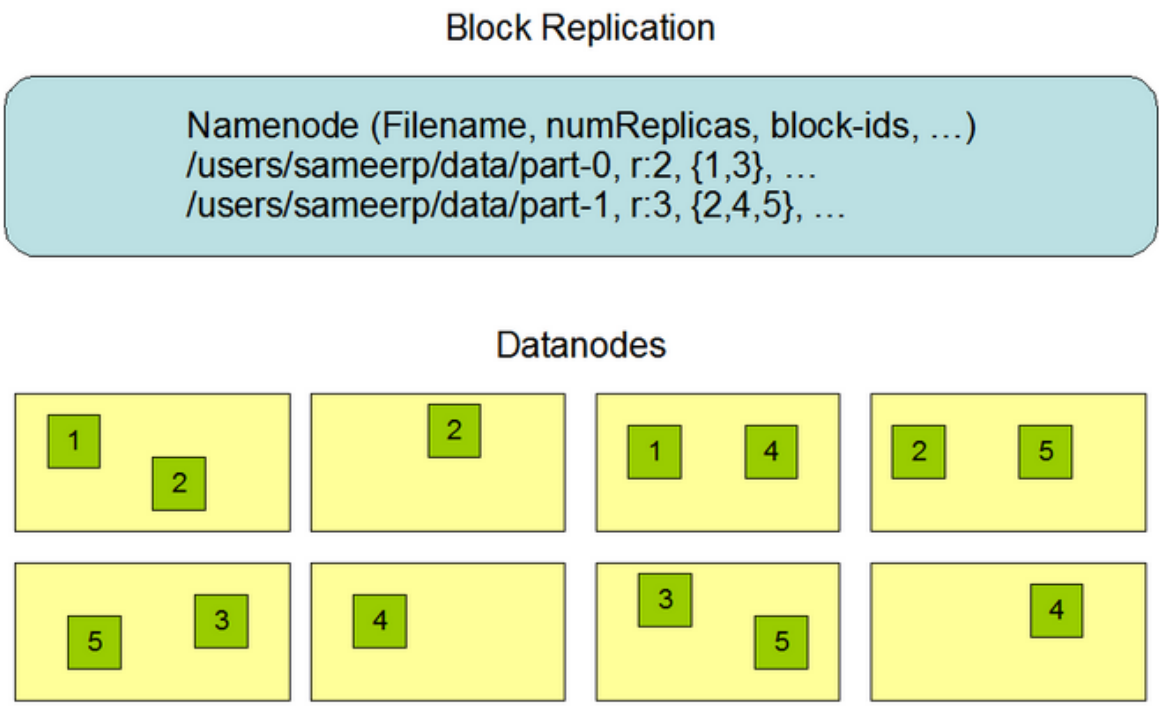


Obrázek 5: HDFS architektura

zdroj: [25]

Replikace dat

Pro předcházení problémům, spojených se selháváním jednotlivých uzlů, jak již bylo zmíněno, HDFS využívá replikace jednotlivých bloků. Replikační strukturu vystihuje Obrázek 6.



Obrázek 6: Princip replikace dat

zdroj: [25]

Obrázek 6 znázorňuje replikace souborů part-0 a part-1. Soubor part-0 je rozdělen do bloků 1 a 3. Počet replikací je nastaven na hodnotu 2. Soubor part-1 je rozdělen do bloků 2, 4 a 5. Replikační faktor je nastaven na hodnotu 3. Na jednotlivých DataNodes je znázorněno rozložení replikovaných bloků na 8 uzlů.

Jednomu bloku v HDFS ve výchozím nastavení je přidělena maximální velikost 64 MB. Všechny bloky v souboru, vyjma posledního, mají stejnou velikost a pokud to množství uzlů dovoluje, každý blok je umístěn na jiném uzlu. Při začátku ukládání nejprve NameNode ukládá data do lokálního souboru a jakmile velikost tohoto souboru dosáhne 64 MB, vytvoří nový soubor ve vybraném DataNodu a data z lokálního souboru vloží sem. Pokud bude HDFS nastaven replikační faktor tři, DataNode, který již obsahuje ukládaný soubor, začne kopírovat po malých dávkách (4 kB) soubor na další DataNode. Ten zapíše tuto část na svůj lokální disk a už nikam neposílá. Po dokončení kopírování nahlásí každý změněný DataNode svoje aktualizované parametry NameNodu.[25]

7.2 MapReduce v programu Hadoop

Základní princip funkce Map Reduce byl již uveden v kapitole 5, z tohoto důvodu zde nebude dále opakován a autor zde popíše funkčnost Map Reduce v prostředí Hadoop.

Vstupní soubory do MapReduce se načítají přímo z HDFS. Soubory jsou umístěny rovnoměrně na všech clusterech. Dále se spustí funkce Map, poté Reduce (stejným způsobem jako v kapitole 5). Pokud se poškodí některý uzel v clusteru, zpracovávaná úloha se opět spustí znovu.[25]

Spouštění úloh a provádění kontrol (především kontrol umístění vstupních a výstupních souborů) provádí metoda Driver. Provádí se pomocí několika hlavních argumentů, jako jsou inputPath, outputPath, JobConf (informace o konfiguraci) a inputFormat.[4]

Závěrem této kapitoly budou ještě popsány jednotlivé úlohy Map Reduce deamonů – tedy JobTracker a TaskTracker.

JobTracker

Zajišťuje spojení mezi uživatelskou aplikací a Hadoop clusterem. Po spuštění MapReduce úlohy JobTracker provádí kompletní řízení a plánování prováděných úloh. Určuje tedy, která data budou zpracována, úlohy přiřadí jednotlivým stanicím a průběžně monitoruje jejich průběh. Pokud dojde k poruše během zpracování v některém uzlu, JobTracker úlohu restartuje a zapojí jiný uzel, na kterém je vše v pořádku. JobTracker je spuštěn na uzlu master spolu s Name Node.[25]

TaskTracker

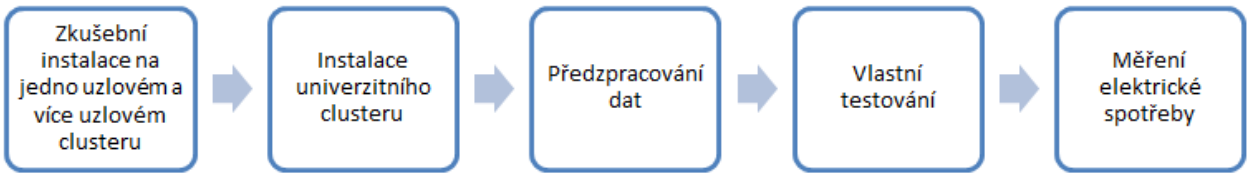
Je druhým deamonem MapReduce metody, který se spouští na uzlech slave. TaskTracker provádí jednotlivé úlohy, které mu zadává JobTracker. Úlohy vykonává na stanici, ve které je spuštěn. TaskTracker také pravidelně zasílá JobTrackeru aktuální zprávy o prováděné činnosti na zadaném úkolu. Na stanici může být vždy spuštěna pouze jedna instance TaskTrackeru, stejně tak jak je tomu u všech deamonů.[26]

Shrnutí sedmé kapitoly

V této kapitole se čtenář seznámil s programem Apache Hadoop, který byl vybrán jako vhodné řešení pro ukládání velkých objemů prostorových dat v podmínkách univerzitního clusteru. Hadoop vychází z řešení společnosti Google. Principem Hadoopu a současně také Googlu, je využití všech výpočetních kapacit, které se skládají z velkého množství počítačů a tím tak dochází k obrovskému urychlení práce. Pro ukládání slouží distribuovaný souborový systém (HDFS) a pro výpočty a zpracování funkce MapReduce.

8 TESTOVÁNÍ PROGRAMU APACHE HADOOP

Testování programu bude provedeno dle schématu, které zachycuje Obrázek 7. Hadoop bude nejdříve zkušebně nainstalován na vlastním PC, poté bude nainstalován i na druhý vlastní autorův počítač a následně se oba počítače propojí, nakonfigurují a vytvoří tak počítačový cluster ze dvou PC. Na tomto clusteru si autor práce odzkouší instalaci a princip fungování programu. Následně bude provedena instalace programu na počítače v univerzitní laboratoři a následné vytvoření malého počítačového clusteru. Dalšími kroky budou předzpracování dat, vlastní testování a měření elektrické spotřeby. Předzpracování dat je určeno především pro potřeby následného zpracování dat. V nejrozsáhlejším kroku bude provedeno vlastní testování programu, kde se autor pokusí o konfiguraci clusteru pro nejrychlejší možné ukládání. Z důvodu přehledu o spotřebě elektrické energie, bude v posledním kroku tato veličina měřena.



Obrázek 7: Schéma postupu při testování programu Hadoop

zdroj: vlastní zpracování

8.1 Instalace jedno uzlového clusteru

Aby bylo možné začít s testováním softwaru Apache Hadoop, musí se nejdříve systém nainstalovat a správně nakonfigurovat. Z počátku byl Hadoop nainstalován, nakonfigurován a odzkoušen na jediném počítači. Takové zapojení se označuje jako jedno uzlový cluster. Pro účely testování byl použit notebook Asus x53e s následujícími parametry:

Operační systém: Ubuntu 12.10 / Windows 7 Profesional; **Procesor**: Intel Core i3 2310M @ 2.10 Ghz; **Operační paměť**: 4 GB Single-Channel DDR3 @ 665 Mhz; **Grafická karta**: Intel HD Graphics Family; **Pevný disk**: 512 GB Seagate.

Nutno doplnit, že na tomto počítači byla nainstalována bezplatná linuxová distribuce – Ubuntu 12.10, která měla vyhrazených 30 GB paměti z pevného disku. Hadoop je podporován pouze na operačním systému Linux. Po absolvování všech kroků (instalace programu, konfigurace a spuštění ukázkového příkladu) lze tvrdit, že počítač s již zmíněnými

parametry neměl žádný problém s během programu. A jelikož se jedná o běžný počítač, lze také potvrdit fakt, že Hadoop je koncipován svou náročností na obyčejné počítačové sestavy.

8.2 Instalace více uzlového clusteru

Pro sestavení více uzlového clusteru, bylo potřeba nainstalovat Hadoop na další počítač. Tímto počítačem byl desktopový počítač s podobnými parametry:

Operační systém: Ubuntu 12.10 / Window 7 Profesional; **Procesor**: AMD AthlonNeo MV-26 1.60 GHz; **Operační paměť**: 2 GB Single-Channel DDR3 @ 665 Mhz; **Grafická karta**: Intel HD Graphics Family; **Pevný disk**: 512 GB Seagate.

Počítač měl opět nainstalovaný druhý operační systém – Ubuntu 12.10 s 30GB prostorem na pevném disku. I na tomto, o něco slabším, počítači fungoval Hadoop bez problémů. Opět bylo potvrzeno pravidlo, že Hadoop je koncipován na obyčejné počítačové součástky.

Následně byly počítače propojeny pomocí síťového kabelu a nakonfigurovány na více uzlový cluster. I v tomto případě zapojení pracoval Hadoop bez problémů.

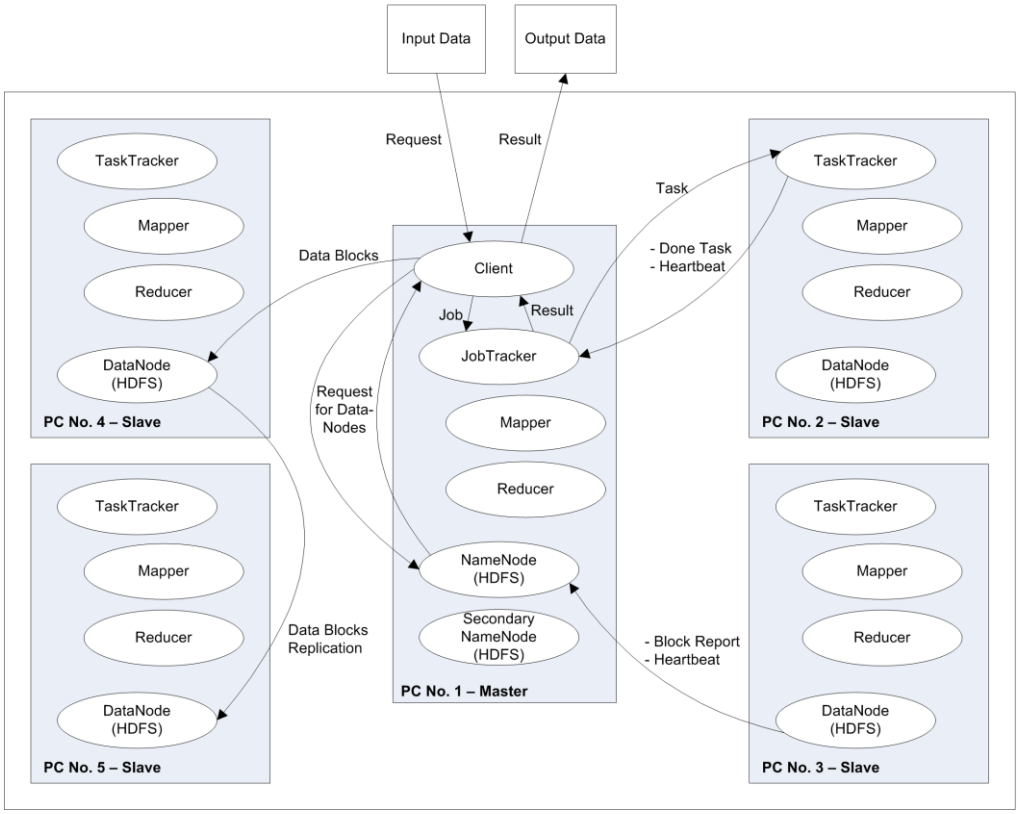
8.3 Instalace univerzitního clusteru

Rozsáhlejší testování následovalo v univerzitní laboratoři, která se skládá z běžných PC (Obrázek 8). Znovu se zde provedla instalace a konfigurace, nyní na pěti počítačích. Tuto část popisuje Příloha A a diplomová práce spolužáka Jakuba Špidlena - zdroj [53] . V laboratoři bylo vybráno pět počítačů, na které byl Hadoop nainstalován a nakonfigurován. Model univerzitní architektury zachycuje Obrázek 9. Všechny počítače byly stejně výkonné. Parametry sestav jsou následující:

Operační systém: Ubuntu 12.10 / Windows 7; **Procesor**: Intel Dual Core i3-3220 3.30 GHz; **Operační paměť**: 4 GB Single-Channel DDR3 @ 665 Mhz,; **Grafická karta**: Intel HD Graphics Family; **Pevný disk**: 512 GB Seagate.

Obrázek 8: Univerzitní laboratoř

Zdroj: vlastní zpracování



Obrázek 9: Architektura univerzitního clusteru

zdroj:[35]

Na akademické půdě Univerzity Pardubice se pro obsluhu prostorových dat používá systém ArcGIS for desktop. Data z tohoto systému lze získat ve formátu shapefile, nebo z geodatabáze ve formátu dbf, případně gdb. Aby bylo možné uložená prostorová data v HDFS dále zpracovávat pomocí Map Reduce streaming, je potřeba je převést do textového formátu. Vstupy i výstupy do Map Reduce streaming jsou vždy reprezentovány textovým formátem.

Poskytnutá data z ArcGISu Univerzity Pardubice pro testování systému nedosahují takových velikostí, aby splňovali kritéria pro označení velkých objemů dat. Z tohoto důvodu pro účely testování vedoucí práce doc. Ing. Jitka Komárková, Ph.D. spojila více mapových listů s pravidelně rozmístěnými body. Body jsou reprezentovány souřadnicemi x, y a z. Celkem bylo sjednoceno 71 mapových listů, což odpovídá rozloze přibližně 1.300 km^2. Tato rozloha představuje zhruba jednu třetinu Pardubického kraje.

8.4 Předzpracování dat

Pro potřebu dalšího zpracování prostorových dat by nebylo možné ukládat prostorová data ve formátu shapefile či databázovém formátu (dbf, gdb, mdb). Z tohoto důvodu byl využit program MS Access pro konverzi databázového formátu na formát textový. Načtená data v tomto programu se dají ukládat ve formátu txt. Během předzpracování bylo vytvořeno několik textových souborů, většina ovšem nedosahovala potřebnou velikost pro další výsledky práce při ukládání. Ze všech předzpracovaných souborů, byl dále pro účely testování použit pouze jeden soubor, který byl označen pro jednoduchou orientaci jako soubor TXT. Parametry tohoto souboru popisuje Tabulka 3.

Tabulka 3: Parametry souboru TXT

Velikost souboru [kB]	607 313 920
Počet záznamů	38 749 386
Počet souřadnic	3
Počet mapových listů	71
Rozloha [km^2]	1.300
Název souboru	71vse.txt

zdroj: vlastní zpracování

8.5 Popis ukládání

Před samotným testováním je potřeba popsat jakým způsobem se soubory do HDFS ukládají. Tato kapitola se dělí na tři části. První popisuje postup a především příkaz, jak soubor do HDFS uložit. Druhá podkapitola popisuje princip ukládání s názornými obrázky. Třetí kapitola objasňuje princip replikace při ukládání.

8.5.1 Postup ukládání dat

Kapitola popisuje postup pro ukládání dat do HDFS. Předpřipravená data se nahrají do vybraného adresáře na lokálním disku jednoho z počítačů patřícího do clusteru (časy potřebné pro kopírování na lokální disk jsou popsány v kapitole 8.6.6). Následuje spuštění Hadoopu, viz Příloha B – Start Hadoopu. Podle pokynů z této přílohy se spustí Hadoop a požadovaný soubor či složka souborů se uloží do HDFS pomocí příkazu:

```
bin/hadoop dfs –copyFromLocal /home/hduser1/71vse.txt /user/71.txt
```

bin/hadoop dfs - představuje cestu, kde se spouštěcí soubor nachází,

–copyFromLocal - je příkaz pro kopírování z lokálního disku počítače do HDFS,

/home/hduser1/71vse.txt - je cesta ukládaného souboru, který je uložen na lokálním disku,

/user/71.txt - představuje cílovou cestu, kam se soubor v HDFS ukládá.

HDFS nejdříve vstupní soubor TXT rozdělí do jednotlivých bloků o nastavené velikosti (primární nastavení je na 64 MB). Při velikosti souboru 607 MB a počtu replikací nastaveného na číslo 1 se soubor TXT rozdělí do deseti částí.

8.5.2 Distribuce bloků při ukládání

Existují dva případy jak HDFS může rozdělovat bloky mezi uzly. První případ je vhodný pro větší clustery, tedy přibližně 20 a více uzlů. Záleží také na spolehlivosti jednotlivých uzlů. Na uzlu master (hlavní uzel celého clusteru) je spuštěn v HDFS pouze NameNode, který řídí ostatní DataNodes a ukládá metadata. V tomto případě se jednotlivé bloky dat ukládají rovnoměrně mezi všechny uzly slave.

Druhý případ je vhodný pro menší clustery. Ukázkovým příkladem je cluster počítačů na Univerzitě Pardubice, který je složen z pěti počítačů. Při takto nízkém počtu počítačů se vyplatí, když je uzel master nastaven i jako pomocný slave, tudíž poskytuje své úložné

místo, na kterém je spuštěn DataNode. Distribuce bloků mezi jednotlivé uzly je závislá na počtu uzlů v clusteru a na velikosti ukládaného souboru. V případě opakovaného ukládání jednoho bloku dat (64 MB), se data ukládají na uzel master do té doby, než bude z větší části zaplněn. Poté bloky začnou ukládat i na ostatní uzly. Při ukládání dvou a více bloků se ukládá vždy na uzel master a rovnoměrně mezi ostatní uzly slave (uvažujeme případ, kdy není žádný uzel zcela zaplněn). Tedy uzel master bude v tomto případě vždy nejvíce zaplněn (takto vzniklou situaci zachycuje Obrázek 12 a Obrázek 13). Stav úložného prostoru clusterů po uložení 3 bloků dat zachycuje Obrázek 10.

Node	Last Contact	Admin State	Configured Capacity (GB)	Used (GB)	Non DFS Used (GB)	Remaining (GB)	Used (%)	Used (%)	Remaining (%)	Blocks
master1	0	In Service	16.78	0	6.01	10.77	0		64.18	1
slave2	0	In Service	16.78	0	3.26	13.53	0		80.6	1
slave3	0	In Service	16.78	0	3.09	13.69	0		81.6	0
slave4	1	In Service	16.78	0	3.14	13.64	0		81.27	1
slave5	1	In Service	16.78	0	2.8	13.98	0		83.31	0

Obrázek 10: Stav HDFS po uložení 3 bloků dat

zdroj: vlastní zpracování

Při opětovném uložení stejného souboru se jeden blok uloží opět na uzel master a zbytek na ostatní uzly. V tomto případě se soubor uložil na stejné uzly jako při prvním uložení, viz Obrázek 11.

Node	Last Contact	Admin State	Configured Capacity (GB)	Used (GB)	Non DFS Used (GB)	Remaining (GB)	Used (%)	Used (%)	Remaining (%)	Blocks
master1	2	In Service	16.78	0.01	6.01	10.75	0.08		64.09	2
slave2	2	In Service	16.78	0.01	3.26	13.51	0.08		80.52	2
slave3	2	In Service	16.78	0	3.09	13.69	0		81.6	0
slave4	0	In Service	16.78	0.01	3.14	13.62	0.08		81.19	2
slave5	0	In Service	16.78	0	2.8	13.98	0		83.31	0

Obrázek 11: Stav HDFS po uložení 6 bloků dat

zdroj: vlastní zpracování

Do ostatních slave uzlů se začne zapisovat v dalším kroku. Uložení 51 bloků znázorňuje Obrázek 12 a uložení 149 bloků Obrázek 13 – zde je vidět princip ukládání HDFS nejlépe.

Node	Last Contact	Admin State	Configured Capacity (GB)	Used (GB)	Non DFS Used (GB)	Remaining (GB)	Used (%)	Used (%)	Remaining (%)	Blocks
master1	0	In Service	16.78	0.62	6.01	10.14	3.71		60.44	17
slave2	0	In Service	16.78	0.32	3.26	13.2	1.93		78.66	11
slave3	2	In Service	16.78	0.28	3.12	13.39	1.64		79.78	8
slave4	0	In Service	16.78	0.43	3.14	13.21	2.57		78.7	9
slave5	0	In Service	16.78	0.22	2.8	13.76	1.29		82.02	6

Obrázek 12: Stav HDFS po uložení 51 bloků dat

zdroj: vlastní zpracování

Node	Last Contact	Admin State	Configured Capacity (GB)	Used (GB)	Non DFS Used (GB)	Remaining (GB)	Used (%)	Used (%)	Remaining (%)	Blocks
master1	0	In Service	16.78	2.43	6.08	8.27	14.5		49.25	49
slave2	2	In Service	16.78	1.21	3.26	12.31	7.24		73.35	27
slave3	1	In Service	16.78	1.13	3.12	12.53	6.73		74.68	22
slave4	2	In Service	16.78	1.45	3.15	12.19	8.62		72.63	27
slave5	2	In Service	16.78	1.2	2.8	12.77	7.17		76.12	24

Obrázek 13: Stav HDFS po uložení 149 bloků dat

zdroj: vlastní zpracování

Z předchozích obrázků je patrné primární ukládání do master uzlu a rovnoměrné rozdělování mezi ostatní uzly slave. Popis jednotlivých sloupců na obrázcích zachycuje Tabulka 4.

Tabulka 4: Popis sloupců u přehledu clusteru

Název sloupce	Popis
Node	Název uzlu zapojeného v clusteru.
Last Contact	Počet kontaktů vyslaných od slave uzlu na master uzel, o tom, že je v pořádku.
Admin State	Stav uzlu (pokud uzel funguje je dán stav „In Service“).
Configured Capacity	Konfigurace kapacity clusteru v GB.
Used [GB]	Využitá kapacita v GB.
Remaining [GB]	Popisuje počet volných GB.
Used [%]	V procentech vyjádřeno, kolik bylo využito paměti.
Remaining [%]	V procentech vyjadřuje, kolik volné paměti ještě zbývá.
Blocks	Počet uložených bloků.

zdroj: vlastní zpracování

8.5.3 Vytížení uzlů při ukládání

Při zápisu do HDFS bylo dále sledováno vytížení jednotlivých počítačů v clusteru během ukládání. Nejvíce vytížen byl logicky uzel master. Během ukládání není možné počítač používat pro jakoukoli jinou běžnou práci. Dle sledování vytížení aplikace htop, ostatní uzly během ukládání nejsou tolik vytíženy a je možné je bez problémů při ukládání používat. I při přenastavení clusteru tak, aby uzel master nebyl využíván zároveň jako pomocný uzel slave

pro ukládání, se vytížení nezmění a není možné počítač v době ukládání využívat pro běžnou práci.

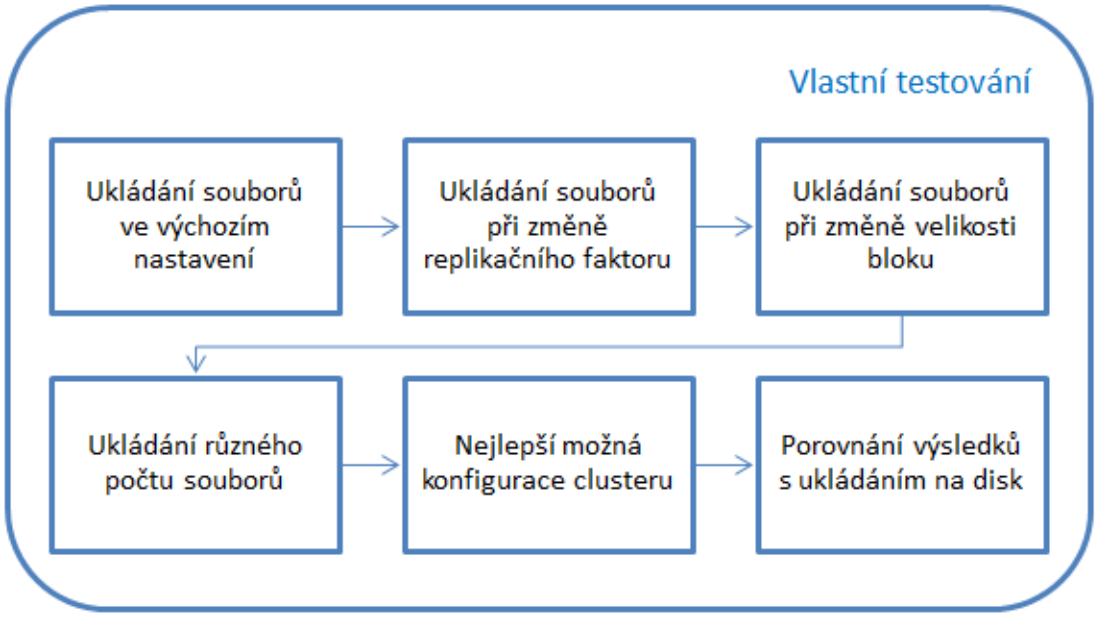
8.6 Vlastní testování

Tato část práce se zabývá především měřením rychlosti ukládání dat do HDFS. Měření bude probíhat podle postupu, který zachycuje Obrázek 14. Testovat se bude ukládání dvou typů souborů o různých velikostech a při různém nastavení HDFS. Textový **soubor TXT** byl již popsán (Tabulka 3). Jak již bylo zmíněno, ostatní předzpracované textové soubory nedosahují potřebných velikostí a z tohoto důvodu byl použit k testování shapefile soubor, označen jako **soubor SHP** i přes fakt, že další práce s tímto formátem po uložení do HDFS není možná. Parametry souboru SHP popisuje Tabulka 5.

Tabulka 5: Parametry souboru SHP

Velikost souboru [kB]	1 413 285
Počet záznamů	12 921 454
Počet souřadnic	3
Název souboru	merge.shp

zdroj: vlastní zpracování



Obrázek 14: Schéma vlastního testování

zdroj: vlastní zpracování

8.6.1 Ukládání souborů ve výchozím nastavení HDFS

Cílem tohoto měření bylo zjistit, jak dlouho se soubory o různých velikostech ukládají do HDFS v případě, že je cluster nastaven na výchozí nastavení. Tedy v nezměněném nastavení po instalaci. Výchozí nastavení clusteru má následující parametry:

- **replikační faktor** nastaven na hodnotu 1 – replikační faktor představuje počet vytvořených replik při uložení. Při nastavení hodnoty na číslici 1 se uloží pouze ukládaný soubor bez replik. Maximální hodnota replikačního faktoru může být rovna počtu PC zapojených v clusteru. Pokud uzel master neslouží současně také jako uzel slave, je tato hodnota o jednu jednotku menší.
- **počet PC v clusteru se** rovná 5 – vyjadřuje počet zapojených počítačů v clusteru.
- **velikost bloku** nastavena na hodnotu 64 MB – jedná se o nastavení velikosti bloků, do kterých se ukládaný soubor rozděluje.

V následujících kapitolách se budou tyto parametry měnit za účelem testování nejlepšího možného nastavení clusteru. U textového souboru TXT, který je tvořen 71 mapovými listy a souboru SHP (shapefile) lze čas ukládání pozorovat a sledovat odchylky při různém nastavení clusteru. Čas byl měřen pomocí log souborů, které v čase zachycují veškerou aktivitu HDFS.

Soubor SHP byl ve výchozím nastavení opakovaně ukládán do HDFS v následujících časech: **2:41, 2:33, 2:53, 2:37, 1:31, 0:55, 0:57, 1:07, 1:05 min**. Významná časová změna se vyskytla při pátém uložení. Do té doby se veškeré bloky ukládaly pouze na uzel master. HDFS takto ukládá v případě, že je replikační faktor nastaven na hodnotu 1. Při větším replikačním faktoru zapojuje slave uzly do ukládání již od počátečního ukládání. Od pátého uložení se už bloky ukládaly rovnoměrně mezi všechny uzly. Díky spolupráci všech pěti uzlů při ukládání dochází k více jak **dvojnásobnému zrychlení.** Dále je z měření pozorováno, že dochází poměrně k velkým časovým výkyvům ať už při ukládání pouze do uzlu master, nebo mezi všechny PC.

Soubor TXT, který obsahuje 607 MB dat, je zhruba poloviční oproti předchozímu souboru. Je možné, že určitou roli v rychlosti ukládání zde bude hrát to, že se jedná o jiný typ souboru. Při stejné konfiguraci jako v předchozím případě bylo provedeno šestnáct časových měření s následujícími časy: **0:58, 0:57, 1:03, 1:07, 1:10, 1:11, 0:55, 0:58, 0:59, 1:05, 0:31, 0:28, 0:32, 0:26, 0:41, 0:31 min**. Jelikož se jedná o menší soubor, cluster začal ukládat

na všechny uzly až od jedenáctého uložení, což se projevilo i na čase (0:31 min.). K nejrychlejšímu uložení došlo při čtrnáctém uložení (0:26 min.), kde všechny bloky byly ukládány pouze na uzly slave.

Shrnutí prvního měření

Díky počátečnímu měření bylo zjištěno, že pro dosažení nejlepšího možného ukládání, musí HDFS ukládané bloky rovnoměrně rozložit mezi uzly, nikoli ukládat pouze na jeden master uzel. Takto ukládá pouze v případě, když je replikační faktor nastaven na hodnotu 1 a to do určité doby, než se uzel master do jisté míry zaplní. Poté začne ukládat i na ostatní uzly slave. Nejlepší časy byly zaznamenány při ukládání, kdy byly bloky rozloženy pouze mezi uzly slave. Při ukládání v tomto konkrétním případě mohou nastat dvě možnosti ukládání. První možností je ukládání na uzel master a na dva uzly slave. Jedná se také o nejčastější uložení. Druhou možností je ukládání na tři uzly slave. Toto ukládání je málo časté, dochází k němu až při téměř zaplněném uzlu master, ovšem dochází k nejrychlejšímu ukládání.

8.6.2 Ukládání souborů při změně replikačního faktoru HDFS

Prvním zásahem do konfigurace clusteru byla změna replikačního faktoru. Tato změna se nastavuje v konfigurační složce Hadoopu, přímo v souboru hdfs-site.xml. Nutno poznamenat, že maximální velikost replikačního faktoru závisí na počtu uzlů v clusteru. V tomto případě je maximální replikační faktor 5. Cílem tohoto měření bylo změřit časy ukládání souborů TXT a SHP. Každý soubor byl celkem desetkrát uložen do HDFS. Pětkrát při replikačním faktoru nastaveném na 3 a poté na 5. Naměřené časy zachycují následné tabulky: Tabulka 6 a Tabulka 7. Časy měření pro 1 repliku jsou brány až od chvíle, kdy byly ukládány na všechny uzly, aby bylo možné porovnat časové hodnoty.

Tabulka 6: Doba uložení 1,4GB souboru do HDFS při změně repl. faktoru

Počet replik	Doba ukládání [minut] *Soubor SHP (1,4 GB)*						
		První měření	Druhé měření	Třetí měření	Čtvrté měření	Páté měření	Průměr
	1 replika	1:31	0:55	0:57	1:07	1:05	**1:07**
	3 repliky	2:54	2:36	2:44	2:50	2:40	**2:44**
	5 replik	3:16	3:30	3:26	3:39	3:15	**3:25**

zdroj: vlastní zpracování

Tabulka 7: Doba uložení 607MB souboru do HDFS při změně repl. faktoru

Počet replik	Doba ukládání [minut] *Soubor TXT (607 MB)*						
		První měření	Druhé měření	Třetí měření	Čtvrté měření	Páté měření	Průměr
	1 replika	0:31	0:28	0:32	0:26	0:41	**0:31**
	3 repliky	1:06	1:20	1:14	1:16	1:08	**1:12**
	5 replik	1:38	1:30	1:46	1:36	1:32	**1:36**

zdroj: vlastní zpracování

Je zde patrné, že s přibývajícími hodnoty replikačního faktoru roste doba ukládání. Ovšem časy ukládání jsou stále malé v porovnání, že u replikačního faktoru 5 se ukládají pětinásobně větší data než při replikačním faktoru 1.

8.6.3 Ukládání souborů při změně velikosti bloku HDFS

Změna velikosti bloku se nastavuje v konfigurační složce v souboru hdfs-conf.xml. Pro změnu výchozího nastavení se vloží do souboru následující kód:

```
<property>
    <name>dfs.block.size</name>
    <value>134217728</value>
    </property>
```

Hodnota mezi páry value představuje velikost bloku v bitech. V tomto případě je velikost bloku nastavena na 134217728 b (128 MB).

V předešlých měřeních byla vždy hodnota bloku ve výchozím nastavení, tedy 64 MB. V této kapitole budou měřeny hodnoty při různé velikosti bloku. Postupně budou nastavené na 8 MB, 64 MB (výchozí) a 128 MB, což je maximální hodnota velikosti bloku pro HDFS. Aby bylo zajištěno ukládání na všechny uzly, replikační faktor byl nastaven na hodnotu 5. Při postupných změnách velikosti bloku byly ukládány soubory pro testování. Naměřené časové hodnoty doby uložení zachycují tyto tabulky: Tabulka 8 a Tabulka 9.

Tabulka 8: Doba uložení 1,4GB souboru do HDFS při změně velikosti bloku

Velikost bloku	**Doba ukládání [minut]** *Soubor SHP (1,4 GB)*						
		První měření	Druhé měření	Třetí měření	Čtvrté měření	Páté měření	Průměr
	8 MB	3:54	4:01	3:59	3:52	3:51	**3:55**
	64 MB	3:16	3:30	3:26	3:39	3:15	**3:25**
	128 MB	3:22	3:30	3:31	3:17	3:25	**3:25**

zdroj: vlastní zpracování

Tabulka 9: Doba uložení 607MB souboru do HDFS při změně velikosti bloku

Velikost bloku	**Doba ukládání [minut]** *Soubor TXT (607 MB)*						
		První měření	Druhé měření	Třetí měření	Čtvrté měření	Páté měření	Průměr
	8 MB	1:36	1:29	1:45	1:40	1:33	**1:36**
	64 MB	1:38	1:30	1:46	1:36	1:32	**1:36**
	128 MB	1:20	1:26	1:35	1:22	1:29	**1:26**

zdroj: vlastní zpracování

Nejdříve byla nastavena velikost bloku na **8 MB**. U souboru TXT se jedná téměř o stejné výsledky jako při výchozím nastavením, dokonce průměr vychází stejně. U souboru SHP byly časy v porovnání s výchozím nastavením už patrně větší. Při této velikosti souboru již lze konstatovat, že při ukládání do menších bloků dochází ke zpoždění.

Dále byla nastavena hodnota bloku na maximální hodnotu, tedy **128 MB.** Ostatní parametry zůstaly nezměněny. Při ukládání souboru TXT dochází ke zrychlení v průměru o deset sekund. V případě souboru SHP není patrno změn v rychlosti ukládání, průměr je stejný jako při ukládání ve výchozím nastavení velikosti bloku.

8.6.4 Ukládání různého počtu souborů stejné velikosti

Dalším cílem měření bylo zjistit, jak rychle se budou ukládat data o stejné velikosti při rozdělení do několika souborů. Data ze souboru TXT byly nejdříve rozděleny do šesti menších souborů, které dohromady tvořily stejnou velikost jako jeden soubor TXT. V dalším kole měření byl soubor TXT rozdělen do 45ti souborů. Velikost jednoho souboru vycházela na 13,5 MB. Replikační faktor byl nastaven na hodnotu 3, velikost bloku na výchozí hodnotu 64 MB. Výsledky obou měření zachycuje Tabulka 10.

Tabulka 10: Doba uložení různých počtů souborů do HDFS

Počet souborů	**Doba ukládání [minut]** *Soubor TXT (607 MB)*						
		První měření	Druhé měření	Třetí měření	Čtvrté měření	Páté měření	**Průměr**
	1 soubor	1:06	1:20	1:14	1:16	1:08	**1:12**
	6 souborů	1:48	1:26	1:31	1:28	1:33	**1:33**
	45 souborů	1:55	2:30	2:15	1:57	2:23	**2:12**

zdroj: vlastní zpracování

Z výsledků měření je patrné, že větší počet menších souborů se ukládá delší dobu než jeden soubor o stejné velikosti.

8.6.5 Nejlepší možná konfigurace clusteru

Tato kapitola zkoumá měření doby ukládání při nejlepším možným nastavením clusteru za účelem nejkratší doby ukládání. Konfigurace clusteru byla nastavena podle nabytých znalostí z předchozích měření. Nutno poznamenat, že se nejedná o praktické řešení, ale o nastavení pro dosažení co nejrychlejšího ukládání, pomocí kterého se zjistí, v jakém nejkratším čase zvládne univerzitní cluster uložit soubory SHP a TXT do HDFS.

Z předchozího měření bylo ověřeno, že pro dosažení rychlejšího ukládání není nutné soubory rozkládat do menších celků, ba naopak je to spíše kontraproduktivní. Oba soubory tedy zůstaly nerozděleny. Velikost bloku HDFS především u textového souboru by měla být nastavena na maximální hodnotu, tedy na 128 MB. V případě shapefilu může zůstat velikost bloku výchozím nastavení. Replikační faktor bude nastaven na nejnižší možnou hodnotu, tedy na číslo 1. Poslední potřebnou konfigurací je, aby na uzlu master byl spuštěn pouze NameNode. Tímto krokem budou data ukládána pouze na uzly slave, čímž dojde též k urychlení doby ukládání. Tato konfigurace se provede tak, že z konfiguračního souboru slave uzlů se odebere uzel master. Naměřené hodnoty zachycují tyto dvě tabulky: Tabulka 11 a Tabulka 12.

Tabulka 11: Porovnání doby ukládání 607MB souboru po optimalizaci

Konfigurace clusteru	**Porovnání doby ukládání [minut]** *Soubor TXT (607 MB)*						
		První měření	Druhé měření	Třetí měření	Čtvrté měření	Páté měření	**Průměr**
	Výchozí	0:31	0:28	0:32	0:26	0:41	**0:31**
	Optimalizovaná	0:22	0:24	0:25	0:27	0:31	**0:25**

zdroj: vlastní zpracování

Tabulka 12: Porovnání doby ukládání 1,4GB souboru po optimalizaci

Konfigurace clusteru	**Porovnání doby ukládání [minut]** *Soubor SHP (1,4 GB)*						
		První měření	Druhé měření	Třetí měření	Čtvrté měření	Páté měření	**Průměr**
	Výchozí	1:31	0:55	0:57	1:07	1:05	**1:07**
	Optimalizovaná	0:49	0:51	0:51	0:54	0:53	**0:51**

zdroj: vlastní zpracování

Při nejlepší možné konfiguraci clusteru se soubor TXT, během pěti měření, uložil v průměrném čase 0:25 min. Větší soubor SHP se ve stejném počtu měření ukládal v průměrném čase 0:51. Při porovnání s rychlostmi ukládání ve výchozím nastavení clusteru, v době, kdy se data začala ukládat na všechny uzly rovnoměrně, se podařilo dosáhnout průměrného zrychlení o 6 sekund v případě menšího 607MB souboru a o 16 sekund v případě 1,4GB souboru. Z výsledku je patrné, že u větších souborů je možné dosáhnout větších

rozdílů, zajímavé by určitě byly naměřené hodnoty na několika násobně větším clusteru, který bohužel není k dispozici.

8.6.6 Porovnání rychlosti ukládání na HDFS s ukládáním na disk

Pro porovnání rychlosti ukládání do HDFS byla změřena také rychlost uložení stejného souboru na lokální disk. Kopírování se provádělo z flash disku Kingston DTSE9 16 GB s maximální přenosovou rychlostí 5 MBps, přes rozhraní USB 2.0. Postupně byl ukládán soubor SHP a následoval ho soubor TXT. U každého souboru byly změřeny tři doby uložení a vybrána ta nejrychlejší. Stejně tak tomu bylo při ukládání do HDFS, kde se vybralo pouze nejrychlejší uložení. Naměřené hodnoty zobrazuje Tabulka 13. Měření bylo prováděno pomocí stopek, tudíž mohlo dojít k nepřesnosti výsledného času v rámci několika setin.

Tabulka 13: Časový rozdíl při ukládání souborů do HDFS a na pevný disk

	Pevný disk -> HDFS	Flash disk -> Pevný disk	**Časový rozdíl**
607 MB *(Soubor TXT)*	0:22 [min]	1:19 [min]	**0:57 [min]**
1,4 GB *(Soubor SHP)*	0:49 [min]	2:01 [min]	**1:12 [min]**

zdroj: vlastní zpracování

Díky výsledkům porovnání je patrné, o kolik dokáže být rychlejší ukládání při zapojení pěti počítačů v porovnání ukládání z flash disku na jeden počítač.

8.7 Měření elektrické spotřeby

Jelikož se poslední dobou neustále poukazuje na energetickou spotřebu v datacentrech a s tím spojené i získávání elektrické energie z obnovitelných zdrojů, je cílem této kapitoly změřit hodnotu spotřeby počítačového clusteru Univerzity Pardubice. Komponenty, ze kterých jsou sestaveny jednotlivé počítače v clusteru, jsou uvedeny v předchozích kapitolách. Hodnoty byly měřeny nejdříve v klidovém stavu a poté při zátěži, tedy při ukládání dat v programu Hadoop. Každé měření probíhalo po dobu jedné minuty. Hodnoty byly měřeny pomocí měřiče energetických nákladů EKM 30. Měřen byl elektrický proud a výkon.

Nejdříve bylo provedeno měření na uzlu master. V klidu tento uzel odebírá 0,16 – 0,17 A a spotřebovává v průměru 34.33 W. V době, kdy uzel master ukládal data do HDFS, byly změřeny odběry v průměru o 0,04 A a 6 W více.

Jednotlivá měření na uzlu master zachycuje Tabulka 14. Veličina I představuje hodnotu odběru elektrického proudu v klidném stavu počítače a veličina $\mathbf{I_z}$ při zátěži. Podobně je tomu tak s veličinami P a P_z. První představuje výkon počítače v klidu a veličina P_z při zátěži.

Tabulka 14: Naměřené hodnoty el. proudu a výkonu na uzlu master

Naměřené hodnoty v intervalu jedné minuty na uzlu master						
	0:10	**0:20**	**0:30**	**0:40**	**0:50**	**1:00**
I [A]	0,17	0,16	0,16	0,17	0,16	0,16
I_z [A]	0,19	0,22	0,18	0,17	0,21	0,23
P [W]	34,30	34,60	34,69	34,15	33,91	34,22
P_z [W]	38,23	40,14	37,55	36,29	46,85	42,66

zdroj: vlastní zpracování

Další měření bylo provedeno na uzlu slave. Postup měření byl stejný jako u uzlu master. Jednotlivé časy měření na uzlu slave 2 popisuje Tabulka 15. Proud se zvýšil při zátěži v průměru o **0,02 A a** výkon o **5 W**.

Tabulka 15: Naměřené hodnoty el. proudu a výkonu na uzlu slave

Naměřené hodnoty v intervalu jedné minuty na uzlu slave 2						
	0:10	**0:20**	**0:30**	**0:40**	**0:50**	**1:00**
I [A]	0,16	0,16	0,16	0,17	0,16	0,16
I_z [A]	0,18	0,17	0,18	0,19	0,16	0,18
P [W]	32,55	32,68	32,50	31,90	32,28	32,41
P_z [W]	38,22	34,36	37,14	39,89	36,50	41,55

zdroj: vlastní zpracování

Z naměřených hodnot vyplývá, že uzel master má větší energetickou spotřebu, než uzel slave. Ovšem rozdíly jsou nepatrné. Dále bylo vypozorováno, že při práci clusteru se energetická spotřeba zvýší o 8,5 % v případě elektrického proudu a o 14,5 % výkonu v porovnání s klidovým odběrem počítače.

ZÁVĚR

Efektivním a přitom levným způsobem řešení nárůstu velkých objemů prostorových dat se stalo distribuované ukládání a paralelní zpracování dat. Toto řešení je založeno na spolupráci velkého počtu počítačů, které jsou propojeny v počítačovém clusteru. Takovýto způsob jako první masivně zavedla společnost Google, která jako jedna z prvních musela začít řešit problematiku velkých objemů prostorových i neprostorových dat.

Cílem diplomové práce byl návrh vhodného způsobu pro ukládání velkých objemů prostorových dat a následná volba vhodného programového řešení pro malý univerzitní cluster složený z běžných počítačů. Jako vhodný způsob ukládání velkých objemů prostorových dat, na základě rešerše, vyšlo distribuované ukládání, které je rychlé a v dnešní době velice populární. Pomocí metod vícekriteriálního rozhodování a následné volbě řešení byl vybrán jako vhodné programové řešení Apache Hadoop. Tento program byl na univerzitním clusteru nainstalován a otestován. Součástí testování bylo dosažení konfigurace clusteru pro dosažení nejvyšší možné rychlosti ukládání, které bylo podložené měřením. Vybraný software byl také změřen z hlediska energetické náročnosti při ukládání a bylo také pozorováno vytížení jednotlivých stanic clusteru při ukládání. V příloze nalezne čtenář shrnutí postupu instalace, který může posloužit jako návod pro instalaci programu Apache Hadoop v českém jazyce. Tuto práci lze brát, jako v praxi ověřený způsob distribuovaného ukládání na běžný počítačový cluster prostřednictvím programu Apache Hadoop. Ze zmíněných hledisek lze tedy cíl práce posoudit jako splněný.

Díky ověřeným výsledkům při ukládání byl dokázán ohromný potenciál tohoto řešení a to nejen z hlediska rychlosti ukládání velkých objemů dat, ale také z hlediska, že tento program nevyžaduje žádný speciální hardware. Hadoop je i nadále rozvíjen a aktualizován, vznikají další rozšíření, které mohou společnosti bezplatně v Hadoopu využívat. Zajímavých výsledků by se určitě dosáhlo při testování na větším počítačovém clusteru, kde se princip distribuovaného ukládání projeví nejvíce. Ovšem rychlost ukládání vybraného řešení byla patrná i na pětiuzlovém clusteru, ve kterém byl Hadoop testován.

Ukládání velkých objemů prostorových dat úzce souvisí se zpracováním, kterým se práce z hlediska rozsáhlosti tématu nezabývala. Zpracování je popsáno v diplomové práci spolužáka Jakuba Špidlena, se kterým autor práce spolupracoval především v počítačové laboratoři.

POUŽITÁ LITERATURA

[1] 7 key Consideration for Big Data Adoption in Government. *Infosys* [online]. 2013 [cit. 2014-3-1]. Dostupné z: http://www.infosys.com/bigdataedge/resources/Documents/7-key-considerations.pdf

[2] 16 Top Big Data Analytics Platforms. *InformationWeek* [online]. 2014 [cit. 2014-04-14]. Dostupné z: http://www.informationweek.com/big-data/big-data-analytics/16-top-big-data-analytics-platforms/d/d-id/1113609pen

[3] Apache Hadoop. *Hadoop*. [online]. 2013 [cit. 2013-03-10]. Dostupné z: http://wiki.apache.org/hadoop/

[4] Apache Hadoop: Tutorial. *Yahoo!* [online]. 2009 [cit. 2013-8-1]. Dostupné z: http://developer.yahoo.com/hadoop/

[5] Architektura databází. *Databáze* [online]. 2010 [cit. 2013-8-1]. Dostupné z: http://www.databaze.chytrak.cz/architektura.htm

[6] Bigdata 2013. *COMPUTERWORLD* [online]. 2013 [cit. 2014-3-1]. Dostupné z: http://data.computerworld.cz/file/specialy/BigData_2013.pdf

[7] Bigdata pro začátečníky a pokročilé. *COMPUTERWORLD* [online]. 2013 [cit. 2014-3-1]. Dostupné z: http://data.computerworld.cz/file/BigData/BigData-2012-web.pdf

[8] Big data. *SystemOnLine* [online]. 2011 [cit. 2013-8-1]. Dostupné z:http://www.systemonline.cz/clanky/big-data.htm

[9] Big data. *wikipedie* [online]. 2013 [cit. 2013-8-1]. Dostupné z: http://cs.wikipedia.org/wiki/Big_data

[10] Big data: na velké objemy dat musíme jinak. *živě.cz* [online]. 2013 [cit. 2013-07-26]. Dostupné z:http://www.zive.cz/clanky/big-data-na-velke-objemy-dat-musime-jinak/sc-3-a-167624/default.aspx

[11] BUHL, Hans, et al. Big Data. *Business & Information Systems Engineering* [online]. 2013, [cit. 2013-07-26]. ISSN 1867-0202. DOI: 10.1007/s12599-013-0249-5. Dostupné z: http://link.springer.com/10.1007/s12599-013-0249-5

[12] Business Problems Suited to Big Data Analytics. *DataInformed* [online]. 2012 [cit. 2014-03-24]. Dostupné z: http://data-informed.com/business-problems-suited-to-big-data-analytics/

[13] CENTRUM INFORMAČNÍCH TECHNOLOGIÍ. *Co je to GIS* [online]. 2011 [cit. 2011-11-08]. Dostupné z:http://cit.osu.cz/gis/pages/coJeToGis.php

[14] Cloud computing. *CHIP.cz*[online]. 2009 [cit. 2013-8-1]. Dostupné z: http://www.chip.cz/clanky/trendy/2009/5/cloud-computing

[15] Datové centrum. *CHIP.cz*[online]. 2009 [cit. 2013-8-1]. Dostupné z: http://www.chip.cz/clanky/trendy/2009/11/datove-centrum

[16] Datový model.*KARTOGRAFIE*[online]. 2011 [cit. 2011-10-29]. Dostupné z: http://kartografie.fsv.cvut.cz/2-1-0-datovy-model.php

[17] distributed file system. *webopedia.* [online]. 2013 [cit. 2013-03-10]. Dostupné z: http://www.webopedia.com/TERM/D/distributed_file_system.html

[18] Distributed File System. *Windows Server*.[online]. 2013 [cit. 2013-03-10]. Dostupné z: http://technet.microsoft.com/en-us/library/cc753479(v=ws.10).aspx

[19] ESRI Shapefile Technical Description. *ESRI* [online]. 2010 [cit. 2013-8-1]. Dostupné z: http://www.esri.com/library/whitepapers/pdfs/shapefile.pdf

[20] Gartner Says Solving Big Data Challenge Involves More Than Just Managining Volume sof Data. *Gartner* [online]. 2011 [cit. 2013-8-1]. Dostupné z: http://www.gartner.com/newsroom/id/1731916

[21] Geodatabáze.*Managementmania* [online]. 2011 [cit. 2011-10-29]. Dostupné z: http://managementmania.com/index.php/component/content/article/841

[22] Geografické informační systémy. *redreaper.eu* [online]. 2009 [cit. 2013-8-1]. Dostupné z: http://www.redreaper.eu/school/PDB/ar01s01.html

[23] GHEMAWAT, Sanjay, HOWARD, Gobioff, SHUN-TAK, Leung. The Google File system [online]. 2003 [cit. 2011-10-29]. Dostupný z: http://static.Google Inc.usercontent.com/external_content/untrusted_dlcp/labs.Google Inc..com/cs//papers/gfs-sosp2003.pdf

[24] Google servery tvoří 2 % celého internetu. *zive* [online]. 2009 [cit. 2011-10-29]. Dostupné z: http://www.zive.cz/clanky/Google Inc.-servery-tvori-2--celeho-internetu/sc-3-a-148632/default.aspx

[25] HDFS Architecture. *Hadoop* [online]. 2013 [cit. 2013-03-10]. Dostupné z: http://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-hdfs/HdfsDesign.html

[26] How Map and Reduce operations are actually carried out. *Hadoop Wiki* [online]. 2013 [cit. 2013-03-10]. Dostupné z: http://wiki.apache.org/hadoop/HadoopMapReduce

[27] HP umožňuje rychlejí zhodnotit velké objemy dat.*hp* [online]. 2013[cit. 2013-07-26]. Dostupné z: http://www8.hp.com/cz/cs/hp-news/press-release.html?id=1404722#.UfKmaNJ9De8

[28] HPCC Data Tutorial: Boca Raton Documentation Team. *HPCC Systems* [online]. 2014 [cit. 2014-04-14]. Dostupné z: http://cdn.hpccsystems.com/releases/CE-Candidate-4.2.4/docs/HPCCDataTutorial-4.2.4-1.pdf

[29] InfiniDB Documentation. *InfiniDB* [online]. 2014 [cit. 2014-04-14]. Dostupné z: http://infinidb.co/resources

[30] Installing Hadoop on Debian. *Informatics Bridging Team*. [online]. 2011 [cit. 2013-03-10]. Dostupné z: https://beagle.whoi.edu/redmine/projects/ibt/wiki/Installing_Hadoop_on_Debian

[31] Jak moc zelený může být telehouse. *LUPA.CZ* [online]. 2008[cit. 2013-8-1]. Dostupné z: http://www.lupa.cz/clanky/jak-moc-zeleny-muze-byt-telehouse/

[32] JEDLIČKA, Karel. Úvod do GIS: *Atributy a jejich vztahy k prostoru* [online] 2012 [cit. 2013-04-14]. Dostupné z:http://www.mendelu.org/upload//05-AtributyAJejichVztahKProstoru.pdf

[33] KEUDELL, Fabian. Cloud vyžaduje opravdu hodně energie. *CHIP*. 11/2012. 36 s. [cit. 2013-03-10].

[34] KLEIN, Dominik, et al. Big Data.*Informatik-Spektrum* [online]. 2013 [cit. 2013-8-1]. Dostupné z: http://link.springer.com/content/pdf/10.1007%2Fs12599-013-0249-5.pdf

[35] KOMÁRKOVÁ, Jitka, et al. *Distributed processing of elevation data by means of Apache Hadoop in a small cluster*. Univerzita Pardubice. 2013, 5 s.

[36] KOMÁRKOVÁ, Jitka, KOPÁČKOVÁ, Hana. *Geografické informační systémy. Automatická identifikace*. Pardubice, 2008. 55 s. ISBN 978-80-7395-120-7.

[37] KOVACS, Kristof. Cassandra vs MongoDB vs CouchDB etc.comparison. *kkovacs.eu*. [online]. 2013 [cit. 2013-03-10]. Dostupné z: http://kkovacs.eu/cassandra-vs-mongodb-vs-couchdb-vs-redis

[38] Large-scale computing ve stylu Google. *Google developer day 2008 Prague.*[online]. 2008 [cit. 2011-10-29]. Dostupné z: https://sites.google.com/site/developerday2008prague/google-developer-day-prague-2008/large-scale-computing-ve-stylu-google-mapreduce-bigtable-hadoop-hdfs-a-dalsi

[39] LUO, Min, YOKOTA, Haruo. Comparing Hadoop and Fat-Btree Based Access Method for Small File I/O Applications. *Tokyo Institute of Technology*. 2012 [cit. 2013-08-05]. Dostupné z: http://link.springer.com/chapter/10.1007/978-3-642-14246-8_20

[40] Metoda párového srovnávání. *ROZHODOVACÍ PROCESY* [online]. 2011 [cit. 2014-04-14]. Dostupné z: http://www.rozhodovaciprocesy.cz/vicekriterialni-rozhodovani/2-1-metody-stanoveni-vah-kriterii.html

[41] Nástroje pro práci se snímky z MSG. *EUMETSAT* [online]. 2007 [cit. 2013-07-26]. Dostupné z: http://old.chmi.cz/meteo/sat/msg/msg05a.html

[42] NOLL, Michael. Tutorials. *Michael-noll* [online]. 2010 [cit. 2013-04-14]. Dostupné z: http://www.michael-noll.com/tutorials/

[43] Open Source Tools. *Bigdata-startups.cz* [online]. 2014 [cit. 2014-04-14]. Dostupné z: http://www.bigdata-startups.com/open-source-tools/

[44] Oracle TimesTen In-Memory Database. *ORACLE* [online]. 2014 [cit. 2014-04-14]. Dostupné z: http://www.oracle.com/technetwork/database/database-technologies/timesten/overview/index.html

[45] PIKE, Rob, DORWARD, Sean, GRIESEMER, Robert, et al. Interpreting the Data: Parallel Analysis with Sawzall. [online]. 2004 [cit. 2013-03-10]. Dostupné z: http://static.googleusercontent.com/external_content/untrusted_dlcp/research.google.com/cs//archive/sawzall-sciprog.pdf

[46] Products. *GridGain* [online]. 2014 [cit. 2014-04-14]. Dostupné z: http://www.gridgain.com/products/

[47] Quick Start. *hbase.apache.org*. [online]. 2013 [cit. 2013-03-10]. Dostupné z: http://hbase.apache.org/book/quickstart.html

[48] RAPANT, Petr. Úvod do geografických informačních systémů [online]. 2009 [cit. 2013-07-26]. Dostupné z:http://gis.vsb.cz/pan-old/Skoleni_Texty/TextySkoleni/SkolGIS4.pdf

[49] RAPANT, Petr. Úvod do geografických informačních systémů. *Skripta PGS*. 2002 [cit. 2004-09-11]. Dostupné z:http://gis.vsb.cz/Publikace/Skripta_sylaby/U_GIS/UGIS.pdf

[50] Raster Image Size: Of Megapixels, Resolution and Inches. *Creative Progression* [online]. 2011 [cit. 2013-8-1] Dostupné z:http://www.creativeprogression.com/resolution/

[51] Sector/Sphere Manual - version 2.5. *Sector/Sphere* [online]. 2010 [cit. 2014-04-14]. Dostupné z: http://sector.sourceforge.net/doc/index.htm

[52] Spark Documentation. *Spark* [online]. 2014 [cit. 2014-04-14]. Dostupné z: http://spark.apache.org/documentation.html

[53] ŠPIDLEN, Jakub. Zpracování velkých objemů prostorových dat. *Diplomová práce.* Pardubice. 2013 [cit. 2013-08-05].

[54] The Coda Distributed Filesystem for Linux. *LinuxPlanet.* [online]. 2013 [cit. 2013-03-10]. Dostupné z: http://www.linuxplanet.com/linuxplanet/tutorials/4481/1/

[55] TVRDÝ, Dalibor. Shapefile. *ESRI* [online]. 2011[cit. 2013-8-1]. Dostupné z: http://wiki.cs.vsb.cz/images/b/b9/Gis_tvr038.pdf

[56] Understanding hadoop clusters and the network. *Brad Hedlung* [online]. 2011 [cit. 2013-04-14]. Dostupné z: http://bradhedlund.com/2011/09/10/understanding-hadoop-clusters-and-the-network/

[57] What is DBMS. *wiseGEEK*.[online]. 2011 [cit. 2011-10-29]. Dostupné z: http://www.wisegeek.com/what-is-dbms.htm

[58] What is the difference between a file system and a database. *Quora* [online]. 2011 [cit. 2013-04-14]. Dostupné z: http://www.quora.com/File-Systems/What-is-the-difference-between-a-file-system-and-a-database

[59] What is the difference between a file system and a database management system http://wiki.answers.com/Q/What_is_the_difference_between_a_file_system_and_a_database_management_system

[60] YONGGANG, Wang, SHENG, Wang, DALIANG Zhou. Cloud computing: *Retrieving and Indexing Spatial Data in the Cloud Computing Environment.* [online]. 2013 [cit. 2013-03-10]. Dostupné z: http://link.springer.com/chapter/10.1007/978-3-642-10665-1_29

[61] Zpracování dat LIDARových měření v ArcGIS. *Vysoká škola Báňská* [online]. 2009 [cit. 2013-04-14]. Dostupné z: http://gis.vsb.cz/vojtek/content/seminars/files/2010_LIDAR/LIDAR.pdf

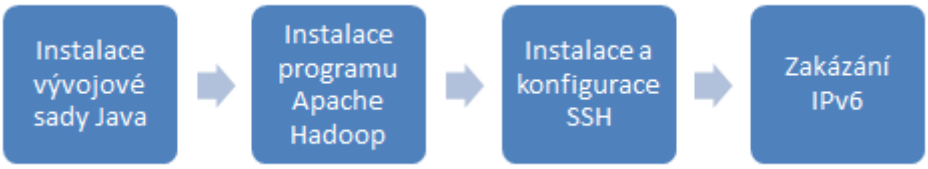
[62] Životnost pevných disků dle Google. *CZC.cz* [online]. 2007 [cit. 2013-04-14]. Dostupné z: http://diit.cz/clanek/zivotnost-pevnych-disku-dle-google

SEZNAM PŘÍLOH

Příloha A

Instalace Apache Hadoop

V této části bude nejdříve popsána instalace vývojové sady Java, která je nutná pro správné fungování Hadoopu. Poté bude následovat popis instalace samotného programu Hadoop. Po instalaci programu Apache Hadoop je potřeba nainstalovat a nakonfigurovat SSH spojení. Následně bude potřeba zakázat IPv6. Po splnění těchto kroků bude možné tvrdit, že Hadoop byl správně nainstalován. Postup zachycuje obrázek.



Zdroj: vlastní zpracování

Instalace vývojové sady Java

Aby Hadoop správně fungoval, musí se na každý počítač, na kterém má být spuštěn, nainstalovat vývojová sada Java. Důvodem je, že v tomto programu byl Hadoop vytvořen. Dle zdroje [30] Hadoop vyžaduje použití Sun Java 1.6.0_20 a novější. Ubuntu od verze 11.10 neposkytuje balíčky pro Sun Java 6 JDK. Součástí instalace je z licenčních důvodů pouze Open JDK, ve kterém ovšem některé programy Hadoopu nepracují správně.

V prvním kroku se do repozitáře Ubuntu přidá nový zdroj. Do souboru source.list, který se nachází ve složce **/etc/apt/sources.list** se přidají dva řádky následujícího kódu:

```
deb http://debian.lcs.mit.edu/debian/ squeeze main non-free contrib
deb-src http://debian.lcs.mit.edu/debian/ squeeze main non-free contrib
```

Kód obsahuje zdrojovou webovou adresu vedoucích k repozitáři. Repozitář obsahuje softwarové balíčky připravené pro instalaci, ze kterého si je operační systém přímo stahuje.

Pro aktualizace nově přidaného repozitáře se v konzoli zadá příkaz:

```
apt-get update
```

Nyní je možné přejít k instalaci balíku. Zadá se příkaz:

```
apt-get install sun-java6-jdk
```

Po odsouhlasení dvou hlášek se java úspěšně nainstaluje.

Instalace samotného programu Apache Hadoop

Instalace probíhá obdobně jako instalace javy. Opět se přidají zdroje vedoucí k repozitáři aktualizací souboru source.list, do kterého se přidá kód:

```
deb http://archive.cloudera.com/debian squeeze-cdh3 contrib
deb-src http://archive.cloudera.com/debian squeeze-cdh3 contrib
```

kód obsahuje zdrojovou cestu repozitáře obsahující instalaci hadoop. Opět se aktualizuje databáze přidaných repozitářů pomocí příkazu:

```
apt-get update
```

Nyní se může začít instalovat hadoop pomocí příkazu:

```
apt-get install hadoop
```

Pro tuto práci práci byl použit Hadoop 1.0.4. Po instalaci hadoopu se vytvoří uživatel pro práci s hadoopem. Tento krok se dělá především proto, aby uživatel Hadoopu byl oddělený od ostatních uživatelů, kteří využívají stejný počítač a na něm ostatní programy. Postup pro vytvoření je následující. Nejdříve se vytvoří skupina Hadoop do které se přidá uživatel hduser. Skupina se vytvoří příkazem:

```
addgroup hadoop
```

uživatele hduser se vytvoří zadáním příkazu

```
adduser-ingroup hadoop houser
```

Databáze HDFS ukládá data ve strukturovaně ve formě souborů a adresářů. Komunikace uživatele s HDFS probíhá pomocí klientské aplikace. Pokud uživatel projevuje dobrou orientaci v příkazech linuxu, vystačí si s terminálovým rozhraním. Někdy se nazývá jako Shell. Aby byla zajištěna synchronizace příkazů mezi Shell a Hadoopem upraví se o následující řádky systémový konfigurační soubor .bashrc, který je umístěn ve složce uživatele, tedy **/hduser/.bashrc**:

```
# Nastavení domácího adresáře Hadoop - adresář, kde je Hadoop nainstalován
export HADOOP_HOME=/usr/local/hadoop
```

```
# Nastavení domácího adresáře Java = adresář, kde je Java nainstalována
export JAVA_HOME=/usr/lib/jvm/sun-java6-jdk


# Přidání několika praktických aliasů a funkcí pro spuštění Hadoop příkazů
unalias fs &> /dev/null
alias fs="hadoop fs"
unalias hls &> /dev/null
alias hls="fs -ls"


# Přidání Hadoop bin/ adresáře do proměnné PATH
export PATH=$PATH:$HADOOP_HOME/bin
```

Instalace a konfigurace SSH

Hadoop vyžaduje zabezpečenou komunikaci mezi jednotlivými uzli. K tomu využívá komunikační protokol Secure Shell (SSH). Cílem je vygenerování RSA klíče mastera a následné jeho přidání mezi autorizované klíče na všech slave uzlech tak, aby master mohl otevírat SSH spojení ke všem slave uzlům a nevyžadoval ověření pomocí hesla. Postup na jednotlivých pracovních jednotkách se provede následovně, nejdříve se nainstaluje openssh server:

```
apt-get install openssh-server
```

Dále se vytvoří veřejný a soukromý klíč s prázdným heslem a uloží se do nově vzniklé složky **hduser**. Příkaz vypadá následovně:

```
ssh-keygen -t rsa -P ""
```

Pokud v konzoli uživatel nevystupuje jako hduser, přihlásí se k tomuto účtu pomocí příkazu:

```
su houser
```

po zadání hesla dojde k přepnutí účtů. V dalším příkazu:

```
cat $HOME/.ssh/id_rsa.p >> HOME/.ssh/authorized_keys
```

se přiřadí vzniklé klíče mezi klíče autorizované. Ovšem je potřeba je nakopírovat do adresářů autorizovaných klíčů na ostatních uzlech pomocí příkazu:

```
ssh-copy-id -i $HOME/.ssh/id_rsa.pub hduser2@slave2
ssh-copy-id -i $HOME/.ssh/id_rsa.pub hduser3@slave3
ssh-copy-id -i $HOME/.ssh/id_rsa.pub hduser4@slave4
ssh-copy-id -i $HOME/.ssh/id_rsa.pub hduser5@slave5
```

Příkazem ssh hduserX@slaveXje možné ověřit správnost spojení a přepínat se mezi jednotlivými účty.

Zakázání IPv6

Hadoop používá ve svém softwaru IPv4 adresy. Z tohoto důvodu se musí na Ubuntu zakázat IPv6 adresy. Dělá se jednoduchým přidáním tří řádků kódu do systémového souboru sysel. Conf, který se nachází ve složce konfiguračních souborů aplikací /etc/sysctl.conf. Na konec souboru se napíše následující text doplněný o komentář:

```
# disable ipv6
net.ipv6.conf.all.disable_ipv6 = 1
net.ipv6.conf.default.disable_ipv6 = 1
net.ipv6.conf.lo.disable_ipv6 = 1
```

Po restartu počítače je možné ověřit, zda jsou IPv6 zakázané následujícím příkazem:

```
cat /proc/sys/net/ipv6/conf/all/disable_ipv6
```

Pokud je výstupem konzole „0“, zakázání se nepodařilo. Žádaným výstupem je „1“, tedy zakázání IPv6.

Příloha B

Vytvoření počítačového clusteru

Univerzitní počítačový cluster bude sestaven z pěti počítačů. Na jednom z nich bude nainstalován master. Tedy hlavní počítač, který bude řídit ostatní počítače v clusteru a bude označován jako master1. Ostatní počítače v clusteru budou označeny jako slave2, slave3, slave4 a slave5. Hlavní část při vytváření clusteru se provádí na pracovní jednotce mastera. Počítače představují síťové uzly, které jsou identifikovány IP adresami. V konfiguračním souboru mastera - **hostname** ve složce **etc se** zapíše IP adresa mastera a pod ní IP adresy všech ostatních pc v clusteru. Pro jednodušší práci v komunikaci a nastavení jednotlivých uzlů je možné používat překlady místo IP adres. Překlad se zapisuje za konkrétní IP adresu. V tomto případě bude hostname vypadat následovně:

10.84.230.153 master1

10.84.230.152 slave2

10.84.230.154 slave3

10.84.230.156 slave4

10.84.230.155 slave5

Jednou za čas může dojít ke změně přidělení IP adres. Pokud se stane, že některý slave nekomunikuje s masterem, bývá to nejčastější důvod. Pro vyřešení tohoto problému stačí přepsat IP adresy aktuálními a vše opět funguje tak jak má.

V předchozí části – instalace Hadoopu bylo možné si všimnout vytvoření hdusera pro každý počítač. Nyní je vhodná chvíle popsat role hduserů. Uživatel houser se vytváří především proto, aby nedošlo k nežádoucím změnám na administrátorském účtu a dále také z důvodu přehlednosti. Na každém počítači byl tedy vytvořen uživatelský účet hduserx, kde x představuje číslo od jedné do pěti, představující číslo počítače. Jednoznačné identifikátory uživatelských účtů na uzlech v clusteru Hadoopu, poté vzniknou spojením uživatelských účtů se jmény uzlů. Jsou to tedy: hduser1@master1, hduser2@slave2, hduser3@slave3, hduser4@slave4 a hduser5@slave5.

Konfigurace clusteru

V první řadě je potřeba správně nadefinovat cestu nově vzniklé java instalace, aktualizací v souboru hadoop-env.sh, který se nachází v instalaci hadoopu v konfigurační složce (/**conf/hadoop-env.sh**). Správně nadefinovaná cesta vypadá takto:

```
export JAVA_HOME=/usr/lib/jvm/java-6-sun
```

Dále je nutné změnit několik vlastností v souborech core-site.xml, hdfs-site.cml a mapred-site.xml. V souboru **core-site.xml** se definuje adresa master uzlu a cesta pro ukládání datových bloků.

- **fs.default.name** – slouží k přesné specifikaci master uzlu NameNode. Všechny DataNodes běžících na slave uzlech poskytnou svá data nastavené adrese. Dále se na tuto adresu připojují klientské programy pro zjištění umístění skutečných souborových bloků. Nastavení bude vypadat následovně:

```
<property>
    <name>fs.default.name</name>
        <value>hdfs://master1:54310</value>
</property>
```

- **hadoop.tmp.dir** – obsahuje cestu lokálního úložiště, kam NameNode a DataNodes ukládají svá data. Možno nastavit pro NameNode a DataNodes různé cesty. Cesta bude následující:

```
<property>
    <name>hadoop.tmp.dir</name>
    <value>/app/hadoop/tmp</value>
</property>
```

V souboru **hdfs-site.xml** se nastavují parametry pro změnu v HDFS následovně:

- **dfs.replication** – parametr, který nastavuje počet replik ukládaného souboru. Hodnota by neměla přesáhnout hodnotu počtu uzlů. V případě univerzitního clusteru, který se skládá z kvalitních PC, může být hodnota nastavena klidně na 1. Jelikož se zde nepředpokládají poruchy. Nízký parametr značně urychlí ukládání do HDFS. Xml kód vypadá následovně:

```
<property>
    <name>dfs.replication</name>
    <value>1</value>
</property>
```

- **dfs.premissions** – nastavení HDFS povinnosti vyžadující informaci o právech. V případě malého clusteru není potřeba, aby HDFS vyžadoval zadaná práva, tedy hodnota zde bude false:

```
<property>
    <name>dfs.permissions</name>
    <value>false</value>
</property>
```

- **dfs.http.address** - ve webovém rozhraní je možné sledovat základní informace o spuštěném HDFS. Zadáním správného portu tuto funkci zpřístupníme. Pro spuštění webové aplikace stačí, když se do adresního řádku zadá adresa http://master:40012. Tato hodnota bez http:// se zadává jako hodnota tohoto parametru, viz:

```
<property>
    <name>dfs.http.address</name>
    <value>master1:40012</value>
</property>
```

Parametry pro MapReduce se nastavují v souboru mapred-site.xml. Pro potřeby malého clusteru je nutné nastavit dva parametry.

- první parametr se nastavuje v souboru **mapred.job.tracker.** Definuje hostitele a port pro spouštění MapReduce aplikací. Kód vypadá následovně:

```
<property>
    <name>mapred.job.tracker</name>
    <value>master1:54311</value>
</property>
```

- **mapred.reduce.tasks** – tento parametr se dá dle zdroje [3] vypočítat následovně:

mapred.reduce.tasks = počet uzlů * (počet jader – 1) * <0,95; 1,75>, z intervalu bylo vybráno číslo 0,95 z toho důvodu, že cluster obsahuje všechny počítače na stejné výkonnostní úrovni. Výsledkem pro univerzitní cluster je číslo 15, viz:

```
<property>
        <name>mapred.reduce.tasks</name>
        <value>15</value>
</property>
```

Až na konfiguraci souboru **hdfs-site.xml**, se veškeré změny provádí na všech uzlech clusteru. Soubor hdfs-site.xml se konfiguruje pouze v master uzlu.

Dalšími soubory, které se musejí nakonfigurovat, jsou **masters** a **slaves**. Tyto dvě konfigurace se provádějí pouze na master uzlu opět v konfigurační složce **conf**. Soubor masters musí obsahovat identifikátor master uzlu, tedy:

```
hduser1@master1
```

Soubor slaves obsahuje všechny uzly v clustery, jelikož i master může plnit roli slave. Což je obzvlášť vhodné pro malý cluster u kterého se nepočítá tolik s poruchovostí. Soubor vypadá následovně:

```
hduser1@master1
hduser2@slave2
hduser3@slave3
hduser4@slave4
hduser5@slave5
```

Start Hadoopu

Ještě před spuštěním samotného Hadoopu, se musí naformátovat NameNode. NameNode se tak spojí s DataNodes na ostatních uzlech a přidělí jim identifikační čísla a oznámí název jmenného prostoru. Veškeré příkazy se provádějí z účtu hduser1@master1. Formátování se spustí následujícím příkazem ve složce Hadoopu:

```
/bin/hadoop namenode –format
```

Při opětovném spuštění tohoto příkazu je nutné ověřit, zda jsou zformátovány i všechny DataNodes na uzlech, aby jim mohl NameNode při každém prvním startu HDFS přidělit nový ID a jmenný prostor. Po úspěšném naformátování NameNodu se spouští Hadoop. Postupně se spouští HDFS a poté MapReduce. Je možné jednotlivé části spouštět postupně pomocí

příkazu **start-dfs.sh** pro start HDFS a **start-mapred.sh**pro start MapReduce. Druhou možností, jak Hadoop spustit je pomocí příkazu **start-all.sh**. Tímto příkazem se spustí HDFS a následně hned MapReduce ve všech uzlech. Na master uzlu se těmito příkazy spouští NameNode, Secondary NameNode, JobTracker, TaskTracker a DataNodes (v případě, že master je nastaven i jako slave – soubor slaves). Na slave uzlech se spouští pouze DataNode a TaskTracker. Jednotlivé aplikace jsou popsány v předchozí kapitole a architekturu vystihuje Obrázek 9.

Pro ukončení Hadoopu platí obdobné příkazy. Příkaz **stop-dfs.sh** zastaví HDFS, **stop-mapred.sh** zastaví MapReduce. Příkaz stop-all.sh zastaví jak HDFS tak MapReduce.

Příloha C

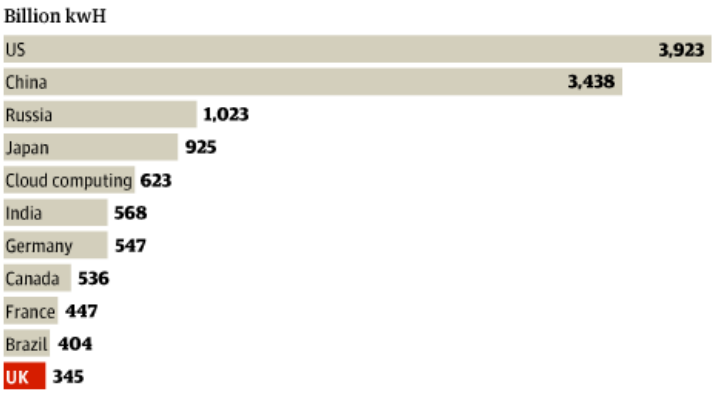
	PUE	CEI	Poloha USA	Poloha ostatní	Stupeň
YAHOO	1,08	56,4	New York Nebraska Virginie Washington	Švýcarsko Singapur	zelená
FACEBOOK	1,07	36,4	Oregon Severní Karolína Kalifornie Virginie	Švédsko Irsko	zelená
GOOGLE INC.	1,14	39,4	Oregon Severní Karolína Georgia	Belgie Nizozemsko Singapur Hongkong Tchajwan Irsko Finsko	zelená
AMAZON	1,45	13,5	Virginie Kalifornie	Japonsko Brazílie	zelená
TWITTER	1,85	21,3	Kalifornie Atlanta		žlutá
APPLE	1,08	15,3	Severní Karolina Kalifornie Oregon		oranžová

MICROSOFT	1,25	13,9	Illinoa Virginie Washington Texas Iowa	Hongkong Irsko	oranžová
IBM	1,19	12,1	Colorado Severní Karolína	Nový Zéland Singapur Německo Irsko	červená

zdroj: vlastní zpracování - upraveno na základě [33]

Hodnota PUE = Ukazuje podíl energie, kterou počítačové centrum vyžaduje pro provoz serverů, vzhledem k celkově spotřebované energii. Například hodnota 2,0 znamená, že pro každý watt, který server spotřebovává na výpočty, je nutný další watt pro jeho chlazení. Optimální hodnota je proto teoreticky hodnota 1,0.

CEI (Clean energy index) = Propojení PUE údajů s environmentálními daty, například využíváním obnovitelných zdrojů energie. Sto procent zde znamená „zelené“ datové centrum, zcela využívajících obnovitelné zdroje, viz zabarvení tabulky.



Celosvětová spotřeba energie

zdroj: [33]