NÁVRH AKCELERACE ČASOVĚ NÁROČNÝCH OPERACÍ PŘI TVORBĚ 3D MODELU POVRCHU Z LIDAROVYCH DAT ZA POMOCÍ DISTRIBUOVANÝCH VÝPOČTŮ

ACCELERATION OF TIME-CONSUMING OPERATIONS IN CASE OF 3D MODEL CREATION BY UTILIZATION OF LIDAR TECHNOLOGY AND DISTRIBUTED COMPUTATIONS

Jan Hovad

Abstract: *This article is focused on optimization of the processing of large volumes of data sets obtained by LIDAR technology. Data are previously cleansed, transformed into a square grid, which resolution is adaptable to the terrain slope factor. Intermediate interpolated data sets are qualitatively compared by statistical methods and transformed into the form of realistic model of the terrain. In case that the entire process is performed by a single PC, its implementation is limited to the relatively small spatial areas. The size of growing input data gradually reduces the possibility to create 3D terrain model and to be successful, must be handled by means of distributed computing. Implementation of this process is difficult and involves some critical tasks. The most common issues are directed into the area of jobs planning, thread management, optimization of the available hardware resources, solving of critical situations and controlling the data flow throughout the network. Simplification lies in the abstraction of the above mentioned problems. The focus is directed into the programming phase and into the explanation of distributed principles. This article is aimed to prepare the implementation of computationally intensive tasks that are computed by the distributed computations. Proposed distributed solution is based on the principles of Google File System.*

Keywords: *GFS, Distributed computing, Functional approach, LIDAR.*

JEL Classification: *C61, C63, C88.*

Úvod

Light detection and ranging (LIDAR) je široce využitelná technologie, která poskytuje informace o vzdálenostech snímaných objektů, jejich začlenění do skupin a také informuje o jejich poloze. Nasnímané objekty jsou obvykle zachyceny připevněním skeneru k pohybujícímu se objektu, který při svém pohybu zaznamenává území pod sebou. Těmito prostředky jsou například letadla (letecké skenování) nebo auta (mobilní skenování). Druhou variantou jsou statické skenery (modely budov, analýzy povrchů ad.). Surový datový záznam je tvořen tzv. mračnem bodů, jehož struktura je značně nepravidelná. Mračno bodů má obvykle enormní datové rozměry a i na poměrně malých zájmových oblastech, například 10×20 km, může obsahovat i 20 milionů zachycených bodů. Pomocí takto zhuštěného záznamu je možné vytvořit velmi detailní model jak samotného terénu, libovolných objektů na povrchu, tak i objektů okem neviditelných, například pod vodní hladinou. Zpracování záznamu je však velmi náročné a to ze všech možných pohledů (ukládání dat v reálném čase, archivace, klasifikace, přenos dat, využití dat jako vstup pro dílčí zpracování, atd.) [2].

O těchto i mnoha dalších problémech se autor přesvědčil při návrhu široce využitelného modelu terénu, který bodovou strukturu laserového záznamu transformoval pomocí interpolačních technik do pravidelné čtvercové sítě. Tato síť však nereprezentovala pouze samotné body, ale byla převedena na ucelenou polygonovou strukturu, která atributy bodových mračen pouze dědila. Tato struktura se může dále využít například pro simulace, 3D tisk, návrh komunikací a architektonických prvků, využití v navigacích a mnoha dalších odvětvích. Dle výzkumu ostatních zainteresovaných osob v oboru laserového skenování je patrné, že se většina autorů vyhýbá využití technologie pro rozlehlejší oblasti. Typické použití je limitováno na konkrétní objekt (budova, interiér, povrch zkorodovaného materiálu, …), případně velmi omezené zájmové území (park v centru města, sídliště, …). V tomto článku je však díky navržené polygonové struktuře možné zpracování dat razantně rozšířit, což dokazují i současné výstupy z modelu o rozměru 10×20 km.

1 Formulace problematiky

Při celém procesu je možné najít výpočetně náročné procesy, které jsou s rychle rostoucí velikosti vstupu n i při lineárním vzestupu složitosti prakticky nespočitatelné. Celou řadu problémů je však možné atomicky rozdělit a pomocí distribuovaného výpočtu rapidně urychlit. Distribuované řešení tak rozšiřuje možnost využití laserových bodových mračen na mnohem větší zájmová území. Při rozloze ČR rovné cca 79 000 km^2 je přibližná datová velikost zachyceného záznamu rovna objemu 421 GB dat. Tento objem je z praktického pohledu na jednom PC nezpracovatelný. Pro množinu PC však nepředstavuje až tak velký problém a to ani v mnohem větších objemech (např. Evropa, cca 55 TB dat). Datová velikost je přibližná a je ovlivněna celou řadou faktorů (rychlost a výška letu, rychlost záznamu dat a frekvence rotace zrcadla, ad.). Tento příklad však poskytuje zcela zjednodušený avšak hmatatelný pohled na problematiku rostoucího množství a objemu datových souborů. V současné době jsou distribuované výpočty na velkém vzestupu a běžný člověk se s výsledky setkává čím dál častěji například v médiích. Aktuální demonstrace výstupu z časově náročných operací byla k vidění při prezidentské volbě v roce 2013, kdy bylo předvedeno, že lze během velmi krátké doby (řádově hodiny), zanalyzovat prakticky celý mediální obsah českého internetu a pomocí přehledných statistik zobrazit aktuální názor obyvatelstva. Prakticky se jedná o stejný problém, jako v případě zpracování laserového záznamu - analýza terabajtů textových dat stažených z internetových deníků, sociálních sítí nebo diskuzních fór. Problematika zůstává stejná, ale implementace může být rozdílná v závislosti na charakteru potřebných dat. Úloh, které mohou být zpracovány distribuovaným způsobem, je mnohem více, například simulace tekutin, vykreslování obrazu, analýzy shluků, korelací a grafových příkladů, indexování velkých souborů nebo například i hraní PC her. V tomto článku bude však navržen konkrétní postup pro časové urychlení výpočtů a rozšíření oblasti zájmu při tvorbě 3D modelu terénu [9] [10].

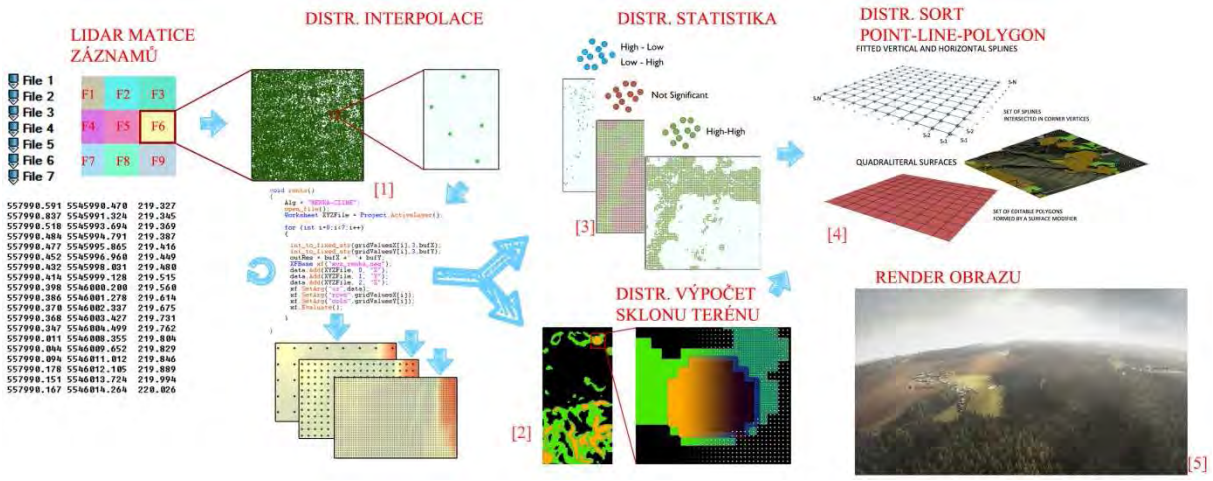
2 Metody

Tato kapitola popisuje znázornění aktuálního stavu modelu a vypisuje použité metody nutné k jeho sestrojení. Vybrány jsou procesy, které mohou být rozděleny a vypočteny distribuovaným způsobem. Dílčí kroky nutné k tvorbě modelu jsou abstrahovány a brány jako součást předešlého výzkumu. Laserová bodová data zpracována do polygonového modelu pomocí stanice Intel i7 2600K (HTT, 4/8), 16 GB RAM, SSD SATA III (cca 450 MB/s), GeForce 460 GTX. Zpracováván je soubor o velikosti 800MB, který obsahuje 22 milionů záznamů. Každý záznam nese souřadnice bodu XYZ. Atribut X a Y je reprezentován hodnotou v souřadnicovém systému WGS 1984 a atribut Z představuje

nadmořskou výšku. Data jsou nejdříve očištěna (odlehlé hodnoty, duplicity, vadné body) a zregistrována (transformační vztahy dílčích výstupů). Poté je proveden výpočet převýšení pro každou buňku rastru ve zvoleném rozlišení. Na základě tohoto výpočtu je provedena hromadná interpolace (lokální i globální metody) v C++, která informaci o sklonu terénu využívá pro adaptivní vytvoření sady zjednodušených bodů. Interpolované modely jsou analyzovány pomocí statistických metod (Moranův test - analýza shluků a prostorové autokorelace, Spearmanův pořadový test - síla vztahu) a pomocí modifikované metody Rozdělení vstupního souboru je interpolovaný model korigován. Korekce spočívá v aplikaci opravné spojité matice, která eliminuje zahrnuté chyby. Pravidelně uspořádané body jsou v 3D prostředí uloženy do polí (X/Y). Pole jsou seřazena a skrze každý bod v odpovídajícím offsetu od počátku souřadnicového systému je proložena přímka. Iterativně je pokryta celá mřížka bodů. Linie slouží jako vstupy pro tvorbu polygonové sítě. Rozlišení polygonového modelu roste s rostoucím sklonem terénu - tento fakt je podložen korelací hodnot odchylek interpolovaného modelu a reálných hodnot. Terén je dále pomocí základních množinových operací segmentován do tříd (orné půdy, lesy, louky, zástavba). Pro každou třídu je vytvořena sada detailních modelů. Atributy modelů (výška, hustota,…), jsou založeny na laserovém záznamu. Na základě pravděpodobnosti jsou poté odpovídající skupiny modelů rozmístěny na korespondující oklasifikované povrchy (jehličnatý les = smrk, borovice, modřín, atp.). Řadu z těchto dílčích procesů lze zpracovat hromadně pomocí více stanic.

V tomto článku bude navržen směr zpracování pro časově nejnáročnější operace (Obr. 1) Mezi ně patří interpolace bodů do pravidelné čtvercové struktury (1), interpolace terénu pro každou buňku rastru (např. 1×1 metr) vůči svému okolí (2), výpočet geostatistiky pro komparaci modelů vůči reálným hodnotám (3), aplikace distribuovaného seřazení a proložení přímek (4), distribuované vykreslení obrazu z vytvořeného modelu (5). Všechny uvedené operace mohou být zpracovány plně nebo částečně distribuovaným způsobem.

Obr. 1: Proces tvorby polygonového modelu s důrazem na dělitelné operace.



Zdroj: vlastní zpracování autora

3 Distribuované a paralelní výpočty při tvorbě 3D modelu terénu

V posledních několika desítkách let docházelo velmi často k citování Gordona Moora a jeho zákona o zdvojnásobování počtu tranzistorů v 18 měsíčních cyklech. Toto tvrzení je známo jako tzv. Moorův zákon. V období několika posledních měsíců je však patrná drobná odchylka od, do nedávné doby, stálého trendu. Faktorů je celá řada. Většina procesorů se dnes pohybuje ve frekvenčním rozmezí 2-4 GHz, toto číslo už se dramaticky nemění

a i pokud by frekvence stoupla na dvojnásobnou hodnotu, nebyla by plně využita z důvodů pomalého vyčítání dat z paměti. Velikost tranzistorů téměř klesla na extrémně nízkou úroveň a další zmenšování v současné době není možné. Moorův zákon tak poodkrývá oblast distribuovaných výpočtů a klade otázku, zda je lepší 18 měsíců vyčkat na řádově výkonnější PC, což už nemusí být pravda, nebo spojit několik současných strojů za účelem urychlení výpočtu.

3.1 Paralelní a distribuované systémy

Z historického pohledu lze v tomto směru narazit na dva pojmy. Paralelní a distribuované výpočty. Paralelní výpočty byly první tohoto typu a jsou populární až do dnešní doby. Z hlediska architektury jsou zaměřené buď vektorově (1 dimenzionální pole), nebo vláknově. Mezi známé stanice tohoto druhu se řadí například super PC Cray (SPC). Paralelní výpočty jsou charakteristické zpracováním fyzicky na 1 PC, případně na 1 PC a více vláknech. Distribuované výpočty jsou zpracovávané na N PC, přičemž každé PC může mít více CPU. Spojení několika PC přidává do problematiky další faktor a to síťové spojení. V současné době je oproti investici do SPC ekonomicky mnohem více dostupné řešení distribuované a to v podobě propojení 1-N běžně dostupných kancelářských PC .

3.2 Paralelizace

Paralelizace je možná v případě, kdy lze zpracování rozdělit do N nezávislých skupin. Například při vykreslení obrazu je prakticky jedno, co se děje na pravé straně, protože to nikterak neovlivňuje stranu levou. Zároveň není překážkou ani chaotické přidělování obrazových výseků jednotlivým vláknům. Tento případ je demonstrován na vykreslení snímku z vytvořeného modelu pomocí 4 vláken. Dílčí výseky, které jsou vláknům přidělované, mají nastavenou velikost X×X pixelů (Obr. 2).

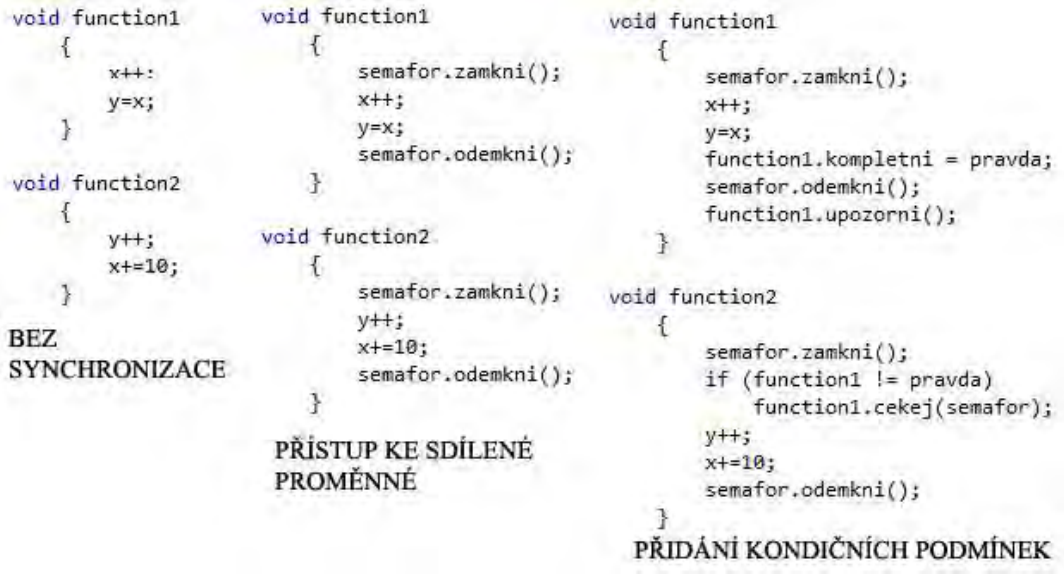
Obr. 2: Paralelizace vykreslení obrazu z 3D modelu.

Zdroj: vlastní zpracování autora

3.3 Paralelizace při výpočtech

V některých případech ale předchozí postup možný není. Například u závislých výpočtů. V tomto případě je tedy nutné vyřešit celou komunikaci vláken a to tak, aby nebyly spouštěny náhodně - nepředvídatelné chování. V případě paralelizace je tedy nutné vymyslet funkční synchronizační systém. Například v případě datové struktury seznam s hlavou by jedno vlákno bez takového systému přidávalo prvek, zatímco druhé vlákno počítalo počet prvků seznamu. Výsledek by nebyl správný. Řešením tohoto problému je přístup ke sdílené proměnné pomocí semaforů (binární semafor). Místo sdílené proměnné lze substituovat například vlakový tunel s jednou kolejí. Pomocí stavů zamknout/odemknout lze omezit přístup k proměnným, ale stále nedochází k řízení vláken. Řešením jsou tak kondiční podmínky, které upozorňují jednotlivá vlákna. Tento problém je demonstrován na následujícím pseudo příkladu, kde funkce 1 představuje práci pro vlákno č. 1, funkce 2 pro vlákno č. 2 (Obr. 3).

Obr. 3: Přístup ke sdílené proměnné.

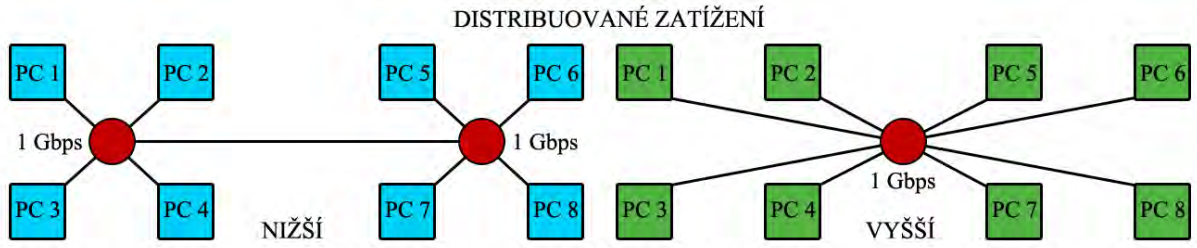


Zdroj: vlastní zpracování autora

3.4 Distribuované systémy

Distribuované výpočty rozšiřují výpočty z 1 PC do množiny PC, které jsou umístěné v síti. Objevuje se tak další problém a to řízení síťové komunikace na síťové vrstvě (TCP/IP). Kromě komunikace je ale důležité i navržení samotné infrastruktury sítě, která by měla být optimalizovaná v závislosti na samotném řešení síťových přenosů. Tato problematika může být zobrazena pomocí odlišného zapojení síťových prvků, které pomáhá snižovat, případně urychlovat datové toky při výpočtech (Obr. 4). Využití tohoto problému je popsáno v další části článku věnující se přímé implementaci distribuovaných výpočtů na bázi Google® File Systému.

Obr. 4: Rozložení zátěže toku dat v síti.

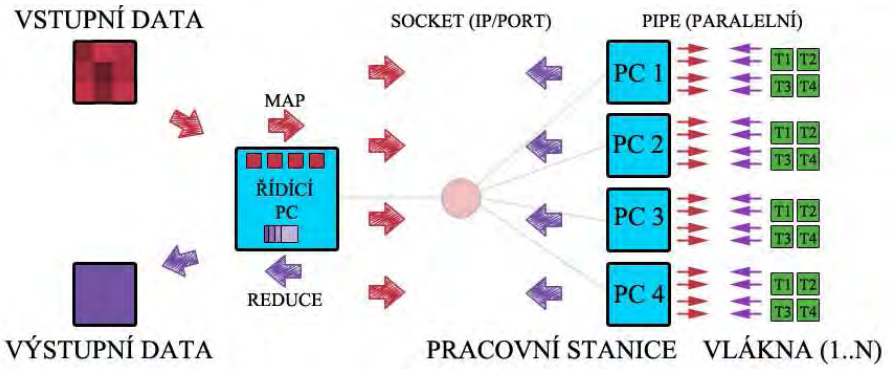


Zdroj: vlastní zpracování autora

Distribuované systémy jsou z velké části založeny na tzv. Funkcionálním programování, které je známé především u uživatelů LISPU (implementovaný např. v aplikaci Autodesk Autocad). Funkcionální programování je založené na předpokladu tvorby kopie zpracovávaného datového souboru. Nedochází tak k modifikaci vstupních dat, ale je vytvořena jejich kopie. Pořadí zpracování každé kopie může být náhodné. V případě vložení prvku do datové struktury seznam s hlavou, je nejprve vytvořena kopie a poté je do této kopie prvek vložen. Vstupní data nejsou modifikována. Rozdělení vstupního souboru na části je zpravidla označeno jako Map fáze. Naopak seskupení vypočtených výstupů jako Reduce či Fold fáze. Dílčí paralelizovatelné části jsou odeslány řídícím prvkem k ostatním pracovním stanicím, které je zpracovávají. Těmito stanicemi může být tzv. Cluster nebo Grid. Cluster reprezentuje obdobné PC stanice zapojené v jedné síti (např. výpočtové

centrum), Grid reprezentuje PC stanice rozdílného druhu v několika sítích (např. SETI a jemu podobné projekty). Princip LISTP Map/Reduce je zobrazen na následující ilustraci (Obr. 5) [11].

Obr. 5: Funkcionální přístup, Map a Reduce.

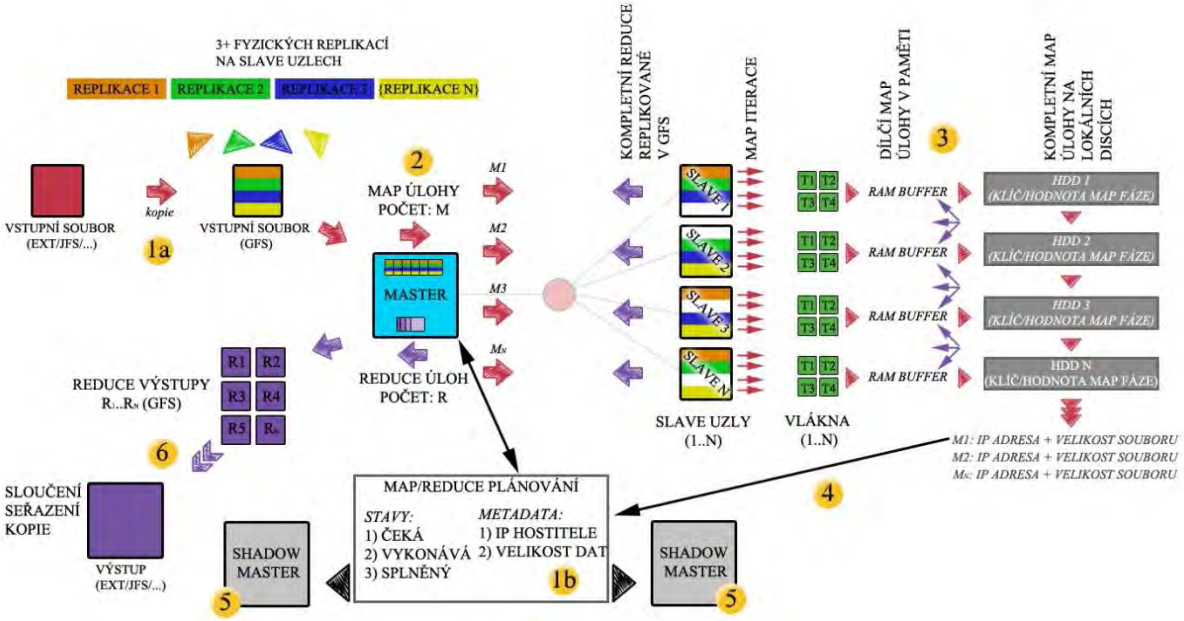


Zdroj: vlastní zpracování autora

3.5 Implementace na bázi Google® File Systému

Z výše popsaného popisu je patrné, že implementace distribuovaného výpočtu je poměrně složitou záležitostí, v které samotné řešení konkrétního problému, zabírá jen malou část celého procesu. Tento problém byl řešen firmou Google®, která se snažila o vývoj souborového systému vhodného pro zpracování nadměrného množství dat vyprodukovaného vyhledávacím systémem. Hlavním cílem bylo využití běžně dostupných kancelářských PC pro distribuované výpočty v clusteru. Abstrakce řešení problémů s přístupy k vláknům, segmentace a redukce vstupních dat, řízení přidělování Map/Reduce operací centrálním prvkem, minimalizace datových přenosů, řešení kritických okamžiků - selhání pracovních stanic, selhání hardwaru a mnoha dalších problematických faktorů. Uživateli tak odpadla nutnost řešit celou řadu oddělených problémů a byla tak zpřístupněna možnost zaměřit se na naprogramování samotné Map a Reduce fáze zpracování dat. V současné době je GFS implementován i v open source alternativě, kterým je Framework Apache Hadoop® využívající Hadoop® File Systém (HDFS). Výpočtové prostředí je v současné době ve vývojové verzi dostupné výhradně pro Unixové operační systémy (Linux). Alternativní implantace pro platformu Windows je přístupná pouze ve fázi beta testování na platformě Windows Azure jako dílčí část mnoha cloudových služeb. Princip funkčnosti frameworku spočívá ve funkcionálním přístupu vysvětleném v předchozí kapitole (Obr. 5). Je však rozšířen o ošetření všech kritických míst. Struktura GFS, která bude použita v dalším postupu viz. Obr. 6 [5] [7].

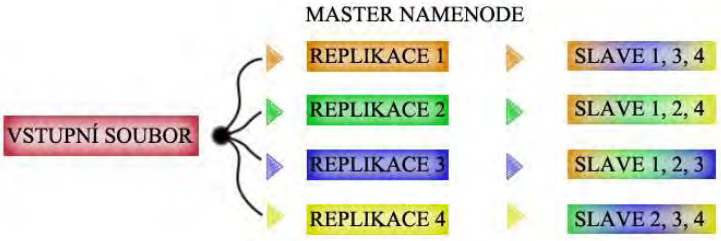
Obr. 6: GFS schéma.



Zdroj: vlastní zpracování autora

Schopnost zpracovávat paralelizovatelné operace na definované struktuře předpokládá běh frameworku na Unixovém operačním systému, korektní nastavení SSH pro přístup mezi všemi uzly, nastavení vlastnických práv uživatele a také správně nakonfigurovaný Java 1.6+ (Sun), kořenový adresář. V případě správné konfigurace celý proces začíná v bodě 1a, Obr. 6 při sběru vstupních dat. V tomto případě mají data podobu LIDAR bodového mračna, tedy textových souborů se záznamy oddělenými novým řádkem. Dílčí hodnoty jsou separované mezerami. Data jsou získána pomocí leteckého přeletu nad zájmovým územím, nicméně vstupní soubory mohou být získány plně automaticky například přímo z webu. Vstupní soubor (nebo množina souborů) je uložen na lokálním file systému, např. EXT3/4, JFS, Raiserfs ad. Datový soubor je překopírován do prostoru vymezeného GFS/HDFS, přičemž je replikován na tzv. Data node, datové uzly. Datový uzel, označovaný též jako Slave (otrok) představuje zároveň v pozdější fázi pracovní stanici vykonávající potřebné dílčí výpočty. Replikace zajišťuje redundanci dat a to v minimálním/základním počtu 3 datových bloků (méně důležité soubory) nebo obvykle vyšším (více důležité soubory). Tímto je zajištěna bezpečnost a dostupnost dat v případě ztráty spojení s datovým uzlem. Informace ohledně rozložení replikovaných částí souboru jsou uchována v tzv. Name node na straně Master PC (řídící PC). Tato struktura tzv. metadat (informace o datech), má například pro ilustrovaný soubor (Obr. 6) následující podobu (Obr. 7) [1] [6].

Obr. 7: Master NameNode, replikační metadata.



Zdroj: vlastní zpracování autora

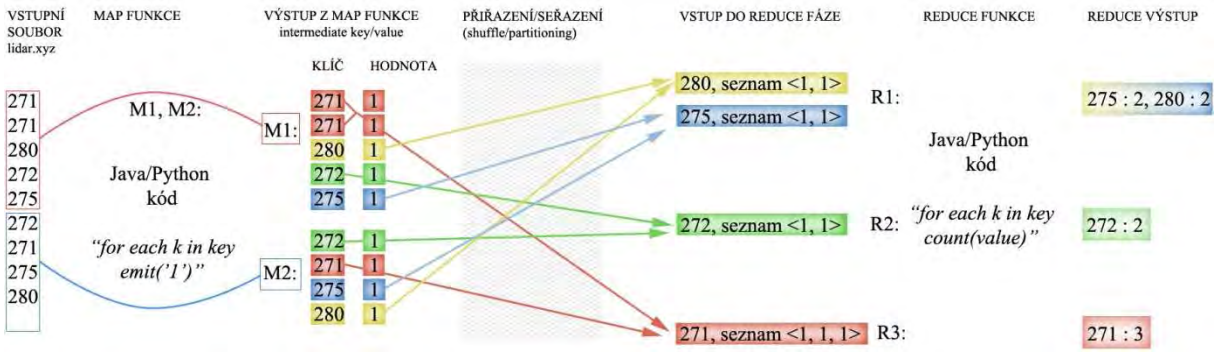
V další části procesu je vstupní soubor rozdělen na M Map úloh (Obr. 6, bod 2). Jedná se o podobné rozdělení jako u segmentace obrazu, v tomto případě jsou však dělena

textová/číselná/… data, určená pro výpočtové operace. Každá z Map úloh je pomocí Master uzlu naplánována na zpracování Map funkcí na jednom z dostupných Slave uzlů. Master uchovává informace ohledně stavu jednotlivých Map částí v podobě těchto atribut: čeká na přidělení Map úlohy/vykonává Map úlohu, přidělené úlohy splněny, tedy Slave uzel je volný. Map funkce je plně definována uživatelem v některém z podporovaných programovacích jazyků (Java, Python, C++). Z důvodu minimalizace datových přenosů je ze strany Mastera vždy snaha o alokaci Map úlohy přímo na místě výskytu odpovídajících replikovaných dat, nebo v případě nevyhovění této podmínky, v těsném okolí Slavu uzlu. Například v rámci jednoho switche tak, aby nebyla zbytečně zatížena propustnost celé sítě (viz Obr. 4). Slave uzel paralelně dle svých vláknových možností (počet CPU/HTT podpora) vypočtené Map výstupy ukládá do své paměti a následně zapisuje na svůj lokální disk do definovaného dočasného adresáře (bod 3, Obr. 6). Tento adresář však není fyzicky umístěn v GFS/HDFS ale je na nereplikovaném běžném souborovém systému daného Slave uzlu. Toto umístění je uchováno v podobě metadat na straně Master uzlu (bod 4, Obr. 6)

Výstup z Map funkce má strukturu <klíč, hodnota> přičemž náplň těchto dvou hodnot závisí na typu úlohy. V případě LIDAR dat například <nadm. výška (klíč), výskyt (hodnota) >. Pro Master uzel v pravidelných intervalech vysílá ping směrem k Slave uzlům a zjišťuje odezvu. V případě, že uzel nereaguje, stornuje všechny Map úlohy, které naplánoval a naplánuje je znovu. Včetně těch, které již byly vypočteny a to z toho důvodu, že nebyly replikovány v síťovém úložišti, ale pouze na lokálním disku Slave uzlu - nelze se k nim v případě poruchy dostat. Zároveň je upravena NameNode tabulka s metadaty vypočtených Map úloh a případní čekající Slave uzly jsou od Master uzlu informováni o změně - odstranění záznamu o vypočtené Map úloze (Map N: IP, port Slave uzlu). To proto, aby v pozdější fázi zpracování nebyl Slave uzel mylně informován o fyzické přítomnosti Map výpočtu v místě, kde z důvodu nedostupnosti již přítomný není. Společně s ošetřeným přístupem v případě selhání na straně Slave uzlů je nutné ošetřit i selhání samotného Master uzlu. To je provedeno periodickou zálohou celé tabulky metadat. Tato záloha představuje možný bod obnovení, který je aktivní vždy se zpožděním maximálně několika málo sekund. Záloha se provádí na skryté, tzv. Shadow Master uzly. Volně přeloženo, šedé řídící uzly. Pro zajištění spolehlivějšího zpracování dat může být těchto uzlů i více.

Po dokončení Map fáze výpočtů následuje příprava pro Reduce fázi v podobě tzv. rozdělení map úloh Slave uzlům (tzv. partitioning). Slave uzly mohou začít s redukcí až v době, kdy jsou všechny Map úlohy zpracované. Redukce dat probíhá na základě redukční funkce, která je definovaná uživatelem. Vstupem do redukční funkce jsou seřazené hodnoty dle klíče ve tvaru <klíč, seznam(hodnot)> a je zajištěno, že každý Slave uzel dostane k redukci odpovídající seřazený seznam hodnot dle klíče (tzv. shuffle fáze). Pro LIDAR data může být celý proces demonstrován na praktickém příkladu následovně (Obr. 8) [3] [4].

Obr. 8: Map a Reduce v případě zpracování LIDAR dat.



Zdroj: vlastní zpracování autora

Reduce funkce je aplikována na Map výstupy, které jsou přes síť vzdáleně načteny přímo z lokálního uložiště Slave uzlu, který je zpracoval. Informace, kde je která Map úloha fyzicky zpracována, je uložena v tabulce metadat u Master uzlu, kterého se Slave uzly na tuto informaci dotazují. Opět platí, že se Master snaží alokovat práci tak, aby nedocházelo ke zbytečným datovým tokům a stejně jako u Map úloh udržuje tabulku stavů. Výstupem Reduce fáze R1 je struktura <seznam(klíč), seznam(hodnota)>. V ilustračním případě předchozího obrázku z důvodu malého množství vstupních dat u R2 a R3 jen <klíč, hodnota>. Pokud je Reduce úloha úspěšně zpracována, není uložena jako v případě Map fáze na lokální disk, ale přímo do GFS/HDFS, kde je výsledek replikován. Výstup je tvořen R soubory, které mohou být považovány za finální výstup nebo dále zpracovány, například seřazeny (Sort), či využity v další iteraci pro navazující MapReduce operaci (bod 6, Obr. 6). Výstup přímo z HDFS/GFS může být buď přímo vypsán například do konzole (cat), nebo zkopírován mimo HDFS/GFS a dále využit. V případě LIDAR dat není potřeba výstupy kombinovat do jednoho souboru, body nesou informaci o souřadnicích a nezáleží na tom, zda jsou načteny z jednoho nebo více individuálních souborů.

4 Diskuze

Na základě struktury popsané v této práci lze implementovat urychlení výpočtů při tvorbě 3D modelu terénu v mnoha směrech. Některé konkrétní úlohy byly demonstrovány (statistika dat, vykreslování obrazu), další budou případně implementovány v dalším výzkumu při inovaci stávajícího řešení. V prvé řadě může být vylepšen proces interpolace nerovnoměrných dat. Současné distribuované řešení problematiky interpolace dat například na bázi organizace OpenTopography zabírá v případě C++ cca 2700 řádků kódu. Pomocí GFS/HDFS může být tento rozsah díky abstrakci mnoha aspektů snížen na cca 700 řádků [8]. V návaznosti na interpolované výstupy ve formě mřížkové struktury lze znatelně urychlit i samotné zpracování této bodové mřížky v 3D prostředí při řazení polí hodnot v osách X a Y (distribuované řazení). Tento výpočet je v řešení autora časově velmi náročný a to z důvodu velkého množství vstupních hodnot, které slouží jako vstup do algoritmu QuickSort, $O(n\log(n)/O(n^2)$. Seřazenými body je poté proložena přímka, která tvoří pravidelný, čtvercový, polygonový povrch terénu.

Závěr

Práce představila konkrétní řešení problematiky časově náročných výpočtů při inovaci postupu tvorby 3D modelů terénu založených na technologii leteckého laserového skenování LIDAR. Využita byla struktura GFS/HDFS přičemž byl detailně popsán i princip paralelních a distribuovaných výpočtů.

Poděkování

Tento příspěvek byl podpořen projektem Inovace a podpora doktorského studijního programu – INDOP, reg. č. CZ.1.07/2.2.00/28.032, financovaným z prostředků EU a ČR.

Reference

[1] Apache Software Foundation, „Hadoop MapReduce Framework“. Available at WWW: <http://hadoop.apache.org/mapreduce/>, 2010

[2] BELKA, L. 2012. Airborne laser scanning and production of the new elevation model in the Czech Republic, *Vojenský geografický obzor*, 55 (1), 19-25.

[3] BLOCH, J. Effective Java. 2008, 2, 346 p., ISBN-10: 0-321-35668-3.

[4] BORTHAKUR, D. 2007 The Hadoop Distributed File System: Architecture and Design. Available at WWW: <http://hadoop.apache.org/core/docs/current/hdfs_design.pdf>

[5] DEAN, J., GHEMAWAT, S. 2004. Map Reduce: Simplified data processing on large clusters, *In OSDI'04 Proceedings of the 6th conference on Symposium on Opearting Systems Design & Implementation,* vol. 6, pp 137-149.

[6] GHEMAWAT, S., GOBIOFF, H. and LUNG, S. 2003 The Google file system. *In 19th Symposium on Operating Systems Principles*, pp. 29-43, Lake George, New York

[7] HOLMES, A. Hadoop in Practice. 2012, 512 p., ISBN 9781617290237

[8] KRISHNAN, S., BARU, CH., CROSBY, CH. 2010. Evaluation of MapReduce for gridding LIDAR data, *Proceedings - 2nd IEEE International Conference on Cloud Computing Technology and Science*, CloudCom, art. no. 5708431, pp. 33-40.

[9] LINZ, P. 2006. *An Introduction to Formal Languages and Automata.* Jones and Bartlett Publishers, 410 p.

[10] SIPSER, M. 2006. *Introduction To The Theory Of Computation.* Thomson Course Technology, 431 p., ISBN: 0-534-95097-3.

[11] THAIN, D., TANNENBAUM, T. and LIVNY, M. 2004 Distributed computing in practice: The Condor experience. *Concurrency and Computation: Practice and Expertise*.

Kontaktní adresa

Ing. Jan Hovad
Univerzita Pardubice, Fakulta ekonomicko-správní
Ústav systémového inženýrství a informatiky
Studentská 84, 532 10 Pardubice, Česká Republika
Email: jan.hovad@upce.cz
Tel. číslo: +420 774 356 593

Received: 22. 11. 2013
Reviewed: 10. 02. 2014, 14. 02. 2014
Approved for publication: 19. 11. 2014