Java Grid Computing aneb jak určit velká prvočísla
Java Grid Computing or how to find large primes

Bakalářská práce
Petr Papež
Vedoucí bakalářské práce: RNDr. Jaroslav Icha
Jihočeská univerzita v Českých Budějovicích
Pedagogická fakulta
Katedra informatiky
2010

Prohlášení

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

Prohlašuji, že v souladu s § 47b zákona č. 111/1998 Sb. v platném znění souhlasím se zveřejněním své bakalářské/diplomové práce, a to v nezkrácené podobě pedagogickou fakultou elektronickou cestou ve veřejně přístupné části databáze STAG provozované Jihočeskou univerzitou v Českých Budějovicích na jejích internetových stránkách.

V Českých Budějovicích dne 20.4.2010

Anotace

Bakalářská práce se zabývá hledáním Mersennových prvočísel za pomoci technologie Grid Computing.

Cílem práce je zprovoznění frameworku pro provádění síťových výpočtů a implementace algoritmu pro hledání velkých Mersennových prvočísel.

Součástí práce je teorie hledání Mersennových prvočísel a historické techniky hledání Mersennových prvočísel, popis projektu GIMPS a dosažené výsledky, popis technologie Grid Computing a možnosti programu Mathematica 7 pro řešení problému hledání Mersennových prvočísel.

Annotation

This Bachelor work contains methods of searching for Mersenne primes using Grid Computing technology.

The goal of this work is to implement a network-based computing framework and an implementation of an algorithm for searching for large Mersenne primes.

Part of this work is a theory of searching for Mersenne primes and historical methods of searching for such numbers, GIMPS project description and its results, Grid Computing technology explanation, and Grid Computing capabilities of Mathematica 7 software for searching for Mersenne primes.

Poděkování

Rád bych poděkoval vedoucímu práce, RNDr. Jaroslavu Ichovi, za pomoc při realizaci bakalářské práce.

Obsah

1 Úvod

Hledání Mersennových prvočísel je jedním z oborů stojících za rozvojem technologie Grid Computing. Ověřování prvočísel je časově velmi náročné a technologie umožňující akumulovat výpočetní výkon přes síť zde má své uplatnění.

Mersennova prvočísla jsou prvočísla, jenž odpovídají vzorci $2^n - 1$. V současné době je známo 47 takových čísel. Poslední nalezené číslo obsahuje téměř 13 milionů číslic. [1]

35. až 47. Mersennovo prvočíslo objevil projekt Great Internet Mersenne Prime Search (GIMPS). GIMPS je praktickou aplikací technologie Grid Computing. Každá zapojená pracovní stanice získává z centrálního serveru úlohy, které spočítá a odešle zpět.

V teoretické části práce je popsána historie Mersennových prvočísel, historie Grid Computingu a projekt GIMPS. Dále je popsán standard OGSA a jeho implementace Globus Toolkit od Globus Alliance, software GridGain a Mathematica 7. [2]

Pro stavbu výpočetního gridu bylo použito software GridGain 2.1.1. GridGain je opensource software naprogramovaný v programovacím jazyce Java. Jeho výhodou je relativní jednoduchost tvorby aplikací na bázi technologií Grid Computing a Cloud Computing.

Výsledkem je aplikace pro výpočetní grid pro hledání Mersennových prvočísel. [3]

Další software, který je pro hledání Mersennových prvočísel použit, je matematický software Mathematica 7.[4]

2 Předmět výzkumu

2.1 Charakteristika oblasti výzkumu

Hlavní oblastí výzkumu je technologie Grid Computing. Teoretická část mapuje technologii Grid Computing včetně projektů postavených na tomto řešení, historii technologie a standard OGSA. Jelikož je pro praktické ověření technologie využito hledání Mersennových prvočísel, je teoretická část rozšířena o tuto tématiku a projekty zaměřené na tuto oblast bádání, především projekt GIMPS.

Pro účely bakalářské práce je naprogramována aplikace postavená na technologii Grid Computing pomocí frameworku GridGain v programovacím jazyce Java.

2.2 Specifikace cíle

Cílem bakalářské práce je provést experimentální výpočty pomocí technologie Grid Computing.

Výpočty se skládají z hledání Mersennových prvočísel v zadaném intervalu.

Pro výpočty jsou použity dvě aplikace – vlastní řešení pomocí frameworku GridGain v programovacím jazyce Java a použití matematické aplikace Mathematica 7. Cílem bakalářské práce není

porovnávat efektivitu matematických algoritmů vlastního řešení s efektivitou komerční aplikace Mathematica.

3 Mersennova prvočísla

3.1 Matematický základ Mersennových prvočísel

3.1.1 Co to je Mersennovo prvočíslo

Prvočíslo je takové přirozené číslo, které má právě dva dělitele různé od nuly. Prvním dělitelem musí být číslo 1 a druhým dělitelem je číslo samotné. Například číslo 11 je prvočíslo, protože neexistuje žádné jiné číslo z intervalu <2, 10>, které je dělitelem 11.

Mersennovo prvočíslo je takové prvočíslo, které se rovná $2^n - 1$, kde n je libovolné přirozené číslo. Například při dosazení n = 2, získáme číslo 3. Číslo 3 je prvočíslo, a protože odpovídá danému vzorci, je Mersennovým prvočíslem. [5]

3.1.2 Hledání Mersennova prvočísla

Zatím není znám způsob, jakým předem spolehlivě odhadnout Mersennovo prvočíslo.

Ověřování, zda číslo je prvočíslo, je náročnou početní úlohou.

Řešením jsou specializované algoritmy pro hledání prvočísel, z nichž nejrychlejší jsou pravděpodobnostní algoritmy. Pravděpodobnostní algoritmy jsou algoritmy, jejichž správnost výsledku odpovídá určité pravděpodobnosti.

Pomalejšími algoritmy jsou deterministické algoritmy, jejich výsledek je vždy pravdivý. [5]

3.2 Historie Mersennových prvočísel

3.2.1 Historie hledání Mersennových prvočísel

První Mersennova prvočísla nalezli již antičtí matematici. Jedním z nejznámějších řeckých matematiků, Euklides, nalezl spojitost Mersennových prvočísel a tzv. dokonalých čísel.[6] Dokonalé číslo je takové číslo, které je součtem všech svých dělitelů mimo sebe sama. Například číslo 6 je dokonalé číslo, protože 6 = 3 + 2 + 1.

Ve starověku a raném středověku se mělo za to, že všechna Mersennova prvočísla jsou všechna čísla odpovídající vzorci $2^n – 1$, kde n je prvočíslo. V roce 1536 však Hudalricus Regius zjistil, že 2^{11}-1 takovému stavu neodpovídá. $2^{11} - 1 = 2047$, 2047 není prvočíslo.

Počátkem 17. století italský matematik Pietro Antonio Cataldi nalezl další Mersennova prvočísla, $2^{17} - 1$, $2^{19} - 1$ a $2^{31} - 1$. Nalezl i další Mersennova prvočísla, ovšem o 37 let později francouzský právník a úředník Pierre de Fermat prokázal, že Cataldi se v nich mýlil. Další korekce Cataldiho čísel provedl v 18. století švýcarský matematik Leonhard Euler.

Francouzský mnich Marin Mersenne, podle kterého získala čísla své jméno, ve své knize *Cogitata Physica-Mathematica* (publikována v roce 1644) uvedl, že tato čísla odpovídají vzorci 2^n-1, kde n = 2, 3, 5, 7, 13, 17, 19, 31, 67, 127 a 257 a dále složeniny kladných celých čísel větších než 257. O 303 let později, v roce 1947 se konečně podařilo spočítat všechny exponenty z intervalu 0 až 258. Správná řada měla znít n = 2, 3, 5, 7, 13, 17, 19, 31, 61, 89, 107 a 127. [5]

Velký rozvoj hledání Mersennových prvočísel zažilo ve dvacátém století díky příchodu počítačů. Dnešní počítač dokáže násobit a hledat prvočísla výrazně rychleji než člověk. Od poloviny 20. století se využíval jeden počítač pro výpočty, zlom nastal v 90. letech, kdy vznikl výpočetní grid GIMPS. Ten v roce 1996 objevil 35. Mersennovo prvočíslo. Od té doby se tato výpočetní síť zasloužila o nalezení dalších 14 Mersennových prvočísel. [7]

3.2.2 Marin Mersenne

Marin Mersenne (8. 9. 1588 – 1. 9. 1648) se narodil v Oizé na řece Maine, Francie. Studoval na Collège du Mans, o několik let později na jezuitské koleji v La Flèche. Po dokončení školy se přestěhoval do Paříže, kde se na přání svého otce věnoval náboženství a připojil se k řádu Minimů, v Paříži studoval na Francouzské královské koleji.

Řád Minimů je postaven na jednoduchosti a členové řádu jsou odhodláni zasvětit život modlitbám a studiu. Po dvou letech se Marin Mersenne stal představeným řádu v klášteře královského paláce v Paříži, ve kterém zůstal, s výjimkou vyjížděk a cest, až do své smrti v roce 1648.

Mersenne se zabýval především akustikou a rychlostí zvuku. Jeho jméno však zůstalo zapsáno u prvočísel. Mersenne se pokusil najít vzorec pro generování prvočísel odpovídajících vzorci $2^n - 1$. V roce 1644 sestavil řadu takových prvočísel menších než 258, n = 2, 3, 5, 7, 13, 17, 19, 31, 67, 127 a 257. Mersenne se v pěti číslech z této řady zmýlil a tři jiná, 61, 89, 107, vynechal. Byl však natolik blízko, že daná prvočísla získala název Mersennova prvočísla.[8]

4 Grid Computing

4.1 Co to je Grid Computing

Grid Computing je technologie pro akumulaci a sdílení počítačových zdrojů. Grid Computing umožňuje sdílet hardwarové a softwarové prostředky pomocí sítě. Množina počítačů spojená do jedné jednotky se v angličtině nazývá grid (v českém překladu mřížka nebo volněji síť). Grid může být multiplatformní, tj. výpočetní jednotky zapojené v gridu nemusí běžet na stejné platformě. Výpočetní jednotky také nemusí být umístěny na stejném místě, podmínkou však je, že tyto jednoty musí být spojeny počítačovou sítí.

Grid je virtuální superpočítač, kterému je možné zadávat příkazy a kde ústřední server gridu distribuuje příkazy svým klientům. Klienti mohou být dedikované stanice pro výpočty, ale častým jevem jsou i gridy, kde jednotlivé stanice jsou vázány velmi volně a jejichž primární účel není poskytovat výpočetní výkon gridu. [9][10]

4.1.1 Výpočetní grid

Grid, jehož primárním účelem je akumulovat výpočetní výkon jednotek zapojených do tohoto gridu a jim, ideálně pomocí zabezpečené komunikace, rozdělovat výpočty. [11]

4.1.2 Datový grid

V tomto typu gridu se sdílí a uchovávají data. Data jsou distribuována mezi klienty tohoto gridu, grid se však zvenčí jeví jako jednotné datové úložiště. [11]

4.1.3 Znalostní grid

Jedná se o rozšíření typu datového gridu o možnosti katalogizace dat. [11]

4.2 Slovníček Grid Computing

4.2.1 CPU Scavenging

CPU Scavenging (významem slova *scavenger* v češtině je mrchožrout) je technologie zabývající se využíváním nevyužitého výkonu procesoru.

Cílem technologie je využít výpočetní výkon procesoru tak, aby případný uživatel nezaznamenal úbytek výkonu a mohl plně pracovat.

Velmi vhodným nástrojem je systém priorit procesů operačního systému. Priority ovlivňují přidělování výpočetního výkonu procesoru procesům. Proces klienta aplikace se spustí s nižší než standardní prioritou a v případě, že procesy s vyšší prioritou zcela nevyužívají výpočetní výkon procesoru, je mu zbylý výpočetní čas přiřazen.

CPU Scavenging se používá především u gridů, jejichž výpočetní jednotky nejsou primárně určeny jako klienti gridu, ale jako pracovní stanice. Příkladem jsou například počítače na univerzitách, v podnicích apod. [12]

4.2.2 Kernel

Název kernel využívá software Mathematica 7. Jedná se o označení jednoho výpočetního vlákna. V textu je překládán jako výpočetní jádro, které je významově nejbližší původnímu anglickému pojmu.

4.2.3 Node

Node, z angličtiny uzel nebo uzlový bod, je instance aplikace - klienta. V této práci jej překládám z angličtiny pojmem klient, který je, dle mého názoru, nejbližší původnímu anglickému pojmu.

Na jedné pracovní stanici může pracovat více aktivních klientů, například z důvodu využití dalších procesorových jader nebo procesorů.

4.3 Historie Grid Computingu a ukončené projekty

Grid Computing vzešel jako evoluce Metacomputingu. Jeho základy položili Ian Foster a Carl Kesselman v publikaci nazvané „The Grid: Blueprint for a New Computing Infrastructure“.
[http://portal.acm.org/citation.cfm?id=296209]

4.3.1 Multics

Nejstarším předkem gridů by mohla být paralelizace výpočtů. Nejstarším operačním systémem, který umožňoval zapojení více procesorů, byl přímý předchůdce UNIXu, Multics. [13] Multics je operační systém, jehož plánování a vývoj začal v roce 1964 a poslední fungující instalace byla ukončena v roce 2000 v Kanadě. [14]

4.3.2 Metacomputing

Metacomputing je termín používaný přibližně od roku 1990, popularizovaný především Larry Smarrem, bývalým ředitelem amerického Národního centra pro použití superpočítačů.[15]

Metacomputing je síťový virtuální superpočítač vytvářený dynamicky z geograficky odlišných oblastí spojených vysokorychlostní sítí.

Metacomputing byl vytvořen z několika důvodů:

- Přístup k datům umístěných na více počítačích
- Ekonomické hledisko; levnější a postupný upgrade oproti skutečným superpočítačům
- Může obsahovat unikátní části – např. databáze

Mezi projekty Metacomputing patří například Fafner, I-Way nebo Legion. [16]

4.3.2.1 Fafner

Factoring via Network-Enabled Recursion (ve zkratce Fafner) je již ukončený projekt, který se pomocí výpočtů distribuovaných přes síť snažil vyřešit rozklad čísla RSA-130.

Číslo RSA je přirozené číslo, jenž je výsledkem násobení dvou prvočísel. Číslo RSA-130 bylo rozloženo v roce 1996 týmem vedeným Arjen K. Lenstrou. Projekt Fafner úspěšný nebyl. [17]

Projekt běžel v roce 1995, oproti dnešním gridům bylo potřeba zasahovat do distribuce a získávání výsledků od výpočetních jednotek. Fafner přišel s technologiemi pro dělení výpočtů a jejich distribuci, které později ovlivnily Grid Computing – především projekt pro hledání mimozemské aktivity Seti@Home. [15][18]

4.3.2.2 I-Way

Information-Wide-Area-Year síť byla výkonná experimentální síť založená na technologii společnosti ATM. Síť vnikala v roce 1995 a měla přenosovou rychlost 155 Mbps.

Síť sloužila k propojení více než tuctu nejvýkonnějších superpočítačů a pokročilých virtuálních stanic. Cílem nebylo vytvořit novou architekturu, ale spojit dohromady již existující sítě běžící pod různými protokoly s různými parametry a různým hardwarovým zázemím.

I-Way spojila dohromady sítě vBNS, ESNet, ATDNet, AAInet, CalREN, MREN, NREN, CASA, MAGIC a satelit ACTS. [19]

Vývoj I-Way silně ovlivnil vývoj standardu Globus. Přínosem bylo především zavedení tzv. „resource broker“. To je část, která sjednocuje přístup ke gridu. [15][20]

4.3.2.3 Legion

Projekt Legion vznikal na půdě americké University of Virginia od roku 1993. Legion je open-source software.

Cílem projektu bylo vytvořit jeden celosvětový virtuální počítač, k němuž lze přistupovat jako k jednomu objektu a skrze něj přistupovat k datům a sdílet je pomocí vysokorychlostní sítě.

Legion byl silně objektově orientovaný. Každá část tohoto systému je objekt se striktně stanovenými funkcemi. Objekty jsou data, počítače spojené v tomto projektu atd. [21][22]

Pokračováním tohoto projektu je spin-off Applied Meta, dnes známá jako Avaki Corporation. Avaki Corporation poskytuje komerční Grid Computing.[23]

4.3.2.4 CODINE

COmputing in DIstributed Networked Environments byl projekt, jenž začal v roce 1992 koupí Distributed Queuing System společností Genias od Florida State University. Genias dále projekt vylepšil a vydal v roce 1993 pod názvem Codine.

V roce 1997 se balík stal součástí balíku GRD od společnosti Raytheon a Codine se prodával samostatně i jako součást GRD. O tři roky později se společnost Genias spojila se společností Chord a vytvořili společnost Gridware.

V témže roce je společnost Gridware odkoupena společností Sun Microsystems a produkt je přejmenován na Sun Grid Engine. Sun odkoupil práva na GRD a společnost Raytheon se v rámci výměny stává prodejcem nově vzniklého produktu Sun Grid Engine. [24]

20. dubna 2009 došlo k akvizici společnosti Sun softwarovým gigantem Oracle, jméno produktu zůstalo zachováno. [25]

4.4 Příklady Grid Computing frameworků / toolkitů

4.4.1.1 Condor

Jedná se o jeden z nejstarších software pro Grid Computing a Metacomputing, začal již v roce 1988 na americké University of Wisconsin. Projekt je stále ve vývoji.

Původním cílem projektu bylo sjednotit výpočetní výkon jednoho oddělení této univerzity. Projekt využívá technologie CPU Scavenging.

Primárním médiem tohoto projektu byla místní síť LAN, nikoliv globální síť. Projekt se však od původního projektu značně změnil. V dnešní

době má v sobě zapojené některé části z Globus Toolkitu (varianta Condor-G) pro transport prací do gridu. [21][26]

4.4.1.2 Globus Toolkit

Viz. kapitola 7.1.

4.4.1.3 Nimrod

Nimrod vznikl na Monash University v Austrálii v roce 1994.

Cílem projektu je distribuovat téměř shodné výpočty přes síť – např. výkon křídla letadla pod různými úhly působení vzduchu.

Nimrod využívá Globus Toolkit pro distribuci výpočtů po síti (varianta Nimrod/G). [21][27]

Nimrod Toolkit má řadu využití na akademické půdě a pro vědecké účely. Prakticky jej využívá například společnost Axceleon pro svůj produkt EnFuzion. Zajímavé na tomto produktu je množství podporovaného software, ve kterém lze výpočty paralelizovat po síti. Jedná se především o 3D rendery podporované EnFuzion 3D (3Ds MAX, Maya, Cinema 4D a množství dalších). [28][29]

4.4.1.4 Unicore

Unicore je německý open-source software pro stavbu gridů aplikující standard OGSA a WS-RF.

Projekt začal v roce 1997 jako iniciativa pro spojení výpočetního výkonu superpočítačů v Německu, tedy rok před vydáním průlomové knihy o Grid Computingu. [30]

4.5 Příklady aplikací Grid Computing

Aplikace technologie Grid Computing se dočkal mnoha zajímavých projektů. Odkazy na některé projekty se často objevují i v českých populárně-vědeckých médiích.

4.5.1 Boinc

Boinc je projekt spadající pod University of California, vznikl v roce 2004. Boinc je platformou pro další projekty, tzn. že v rámci Boinc je možné se přihlásit k počítání různých projektů. Mezi nejznámější, nikoliv však jediné, patří:

- SETI@home – hledání stop po mimozemských civilizacích především pomocí radiových signálů
- Einstein@home – hledání neutronových hvězd (pulsarů)
- LHC@home – počítání dat pro největší urychlovač části na světě - CERN
- Malariacontrol.net – výpočty pro hledání ideální ho rozmístění sítí proti komárům v Africe, výzkum chemoterapie nebo testování nových vakcín
- Rosetta@home – vyhledávání nových variant bílkovin pro boj s nemocemi AIDS, malárie, rakovina apod.
- GPUGrid.net – výpočetní grid pro biomedicínu speciálně upravený pro běh na grafických kartách NVIDIA.

[31][32]

4.5.2 Folding@Home

Projekt Standfordské Univerzity. Folding@Home se zaměřuje na skládání bílkovin pro boj s nemocemi.

Zajímavostí jsou platformy, na kterých projekt pracuje – mimo běžných procesorů to jsou i grafické karty ATI a NVIDIA a herní konzole PlayStation 3. [33]

5 Great Internet Mersenne Prime Search

5.1 Popis GIMPS

Great Internet Mersenne Prime Search (zkratka GIMPS) je veřejný výpočetní grid pro hledání Mersennových prvočísel.

GIMPS byl založen Američanem Georgem Woltmanem v roce 1996. Síť začala v době procesorů Intel Pentium a postupně narůstala nejenom výpočetním výkonem, ale i počtem členů. Zatímco v únoru 1996 bylo do výpočtů zapojeno 50 počítačů[34], k dnešnímu datu je do gridu zapojeno 9579 počítačů s 107172 výpočetními jednotkami s potenciálním výpočetním výkonem 71,487 TFlops, tj. se skutečným výkonem 40,988 TFlops. [35]

Výpočetní grid má jméno PrimeNet. V současné době ve verzi 5. PrimeNet si dělí počítače podle několika kritérií a podle nich přiřazuje klientu práci. Pro určení správné kategorie PrimeNet využívá softwarového klienta. Kritériem je výkon procesoru, důvěryhodnost procesoru a bezporuchovost stroje, bohužel pro poslední dvě kritéria GIMPS nespecifikuje, jak je měří, či co dané hodnoty znamenají.

První kategorií jsou klienti pro počítání doposud nespočítaných exponentů., výkonnostně musí být nejméně na úrovni výkonu Intel Pentia 4 2000 MHz, důvěryhodnost procesoru vyšší než 2.0 a bezporuchovost nižší než 0.7.

Slabší stroje dostávají kontrolu výsledků již spočítaných exponentů. Zde je potřeba nejméně počítač na úrovni Intel Pentia 4 1000 MHz s důvěryhodností procesoru 2.0 a spolehlivostí nižší než 0.5.

Pokud ani tyto požadavky stroj nesplňuje, je mu přiřazena jedna ze dvou prací, buď prvočíselný rozklad a to za předpokladu, že výkon klienta je

nejméně na úrovni Intel Pentia 4 50 MHz, nebo Pollardovo P – 1 rozklad a to za předpokladu, že klient disponuje alespoň 300 MB volné paměti.

Posledním omezujícím parametrem je velikost vyrovnávací paměti procesoru. Procesory s L2 cache menší než 128 kB dostávají exponenty pod 10 000 000, procesory s L2 + L3 cache menší než 256 kB získávají čísla pod 20 000 000. [35] [36]

5.2 Výsledky GIMPS

Od roku 1996 se podařilo GIMPS nalézt 14 Mersennových prvočísel. V tabulce je vždy uvedeno jméno osoby, na jejímž počítači bylo číslo nalezeno. [7]

Kdy	Kdo	Číslo
1. 11. 1996	Joel Armengaud	$2^{1398269}-1$
24. 8. 1997	Gordon Spence	$2^{2976221}-1$
27. 1. 1998	Roland Clarkson	$2^{3021377}-1$
1. 6. 1999	Nayan Hajratwala	$2^{6972593}-1$
14. 11. 2001	Michael Cameron	$2^{13466917}-1$
17. 11. 2003	Michael Shafer	$2^{20996011}-1$
15. 5. 2004	Josh Findley	$2^{24036583}-1$
18. 2. 2005	Dr. Martin Nowak	$2^{25964951}-1$
15. 12. 2005	Curtis Cooper a Steven Boone	$2^{30402457}-1$
4. 9. 2006	Curtis Cooper a Steven Boone	$2^{32582657}-1$
23. 8. 2008	Edson Smith	$2^{43112609}-1$
6. 9. 2008	Hans-Michael Elvenich	$2^{37156667}-1$
12. 8. 2009	Odd Magnar Strindmo	$2^{42643801}-1$

Tabulka č. 1 – Úspěšní účastníci projektu GIMPS

6 Open Grid Forum a standard OGSA

Global Grid Forum (zkr. GGF) vzniklo v roce 2001 na setkání v Amsterodamu a bylo následníkem prvního Grid fóra z roku 1999. Open Grid Forum (zkr. OGF) vzniklo v roce 2006 sloučením GGF a Enterprise Grid Alliance. [37]

V roce 2008 vzniklo OGF – Europe, evropská větev OGF. [38]

Cílem OGF je vytvoření komunity pro tvorbu a aplikaci distribuovaných systémů.

OGF má tzv. rady, každá má svůj specifický úkol:

Standards Council – má na starosti vývoj architektur, specifikací, budoucího vývoje a všechny aktivity spojené se standardy a součinností.

Enterprise Council – má na starosti komunikaci s podniky a prodejci. Identifikuje jejich požadavky a organizuje setkání a propaguje produkty fóra.

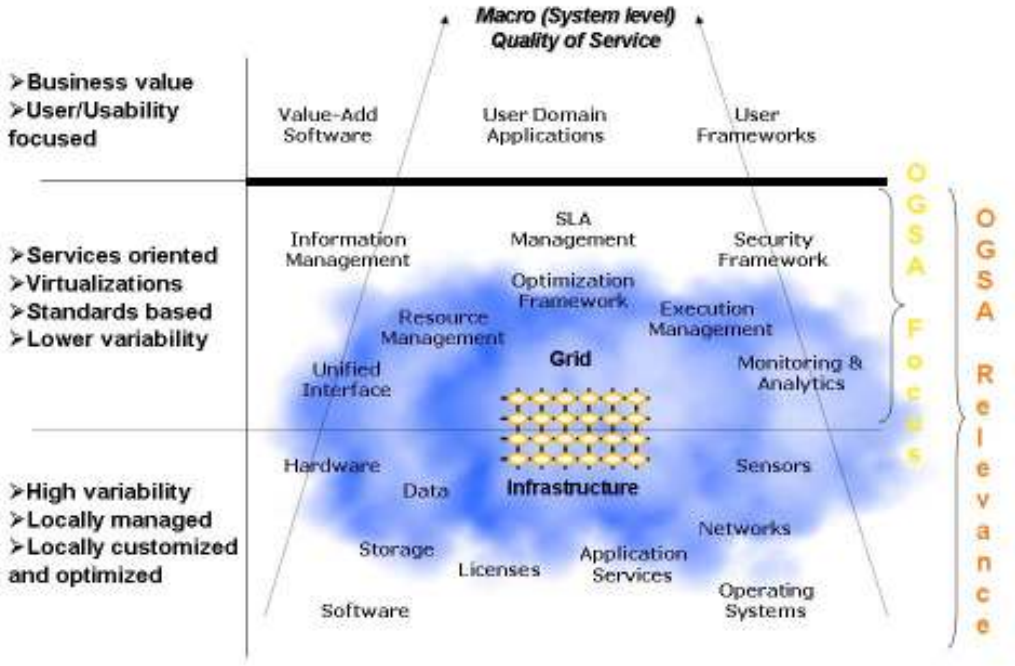
e-Research Council – komunikuje s vědci, vývojáři a pedagogy a identifikuje jejich požadavky a pořádá pro ně semináře a setkání. [39]

6.1 Standard OGSA

Open Grid Services Architecture (OGSA) je standard pro architekturu gridů od fóra OGF. Vznikl mimo jiné na základě práce „The Physiology of the Grid“ z roku 2002. [40]

Cílem standardu je umožnit Grid Computing ve velmi diverzifikovaném prostředí, kde jednotlivé části mohou být hostovány na různých platformách (např. J2EE, .NET), různých OS (např. UNIX, Linux, Windows, Solaris), mohou být zcela odlišné (např. počítače, senzory, datová

úložiště, databáze, sítě) a mohou poskytovat různé služby od různých poskytovatelů. [41]



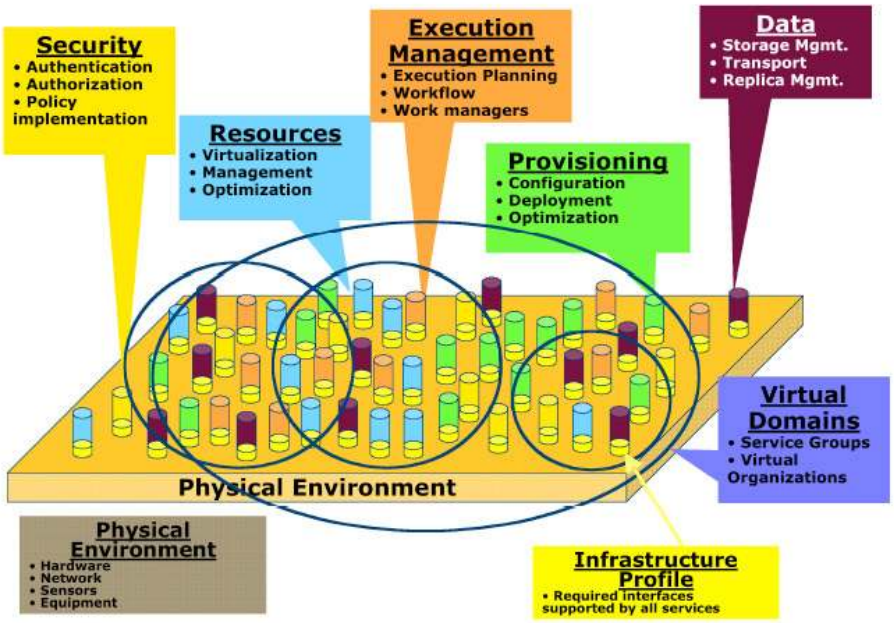
Obrázek č. 1 - Ilustrace služeb gridu a standardu OGSA [41]

Výše uvedený obrázek zobrazuje, o jaké části gridu se stará OGSA, které mají co dočinění s tímto standardem a ty, jenž nejsou s OGSA spojeny.

Oblasti standardu: informační management, management zdrojů, unifikované rozhraní / přístup, SLA management, optimalizační framework, bezpečnostní framework, management spouštění prací, monitoring a analýza.

Mimo oblast standardu, ale stále relevantní: hardware, data, úložiště, software, licence, služby aplikací, senzory, sítě, operační systémy.

Zcela mimo oblast standardu: přidaný software, aplikace na straně klienta, frameworky na straně klienta.



Obrázek č. 2 - Ilustrace uspořádání OGSA [41]

Důležité poznámky k OGSA:

- OGSA není vrstvená architektura, ani objektově-orientovaná. Dala by se spíše definovat jako modulární.
- Ne všechny části standardu musí být implementovány nebo přítomny, aktivní může být i podmnožina všech součástí.
- OGSA reprezentuje služby a jejich rozhraní, chování a interakci. OGSA nedefinuje, jakým způsobem mají být jednotlivé implementovány.

Architektura se skládá z tzv. služeb (angl. services).

6.1.1 Infrastructure Services

Insfrastructure Service je komponenta sloužící pro infrastrukturu, tj. komunikaci mezi jednotlivými složkami gridu (např. komunikace mezi příjemcem a poskytovatelem služby).

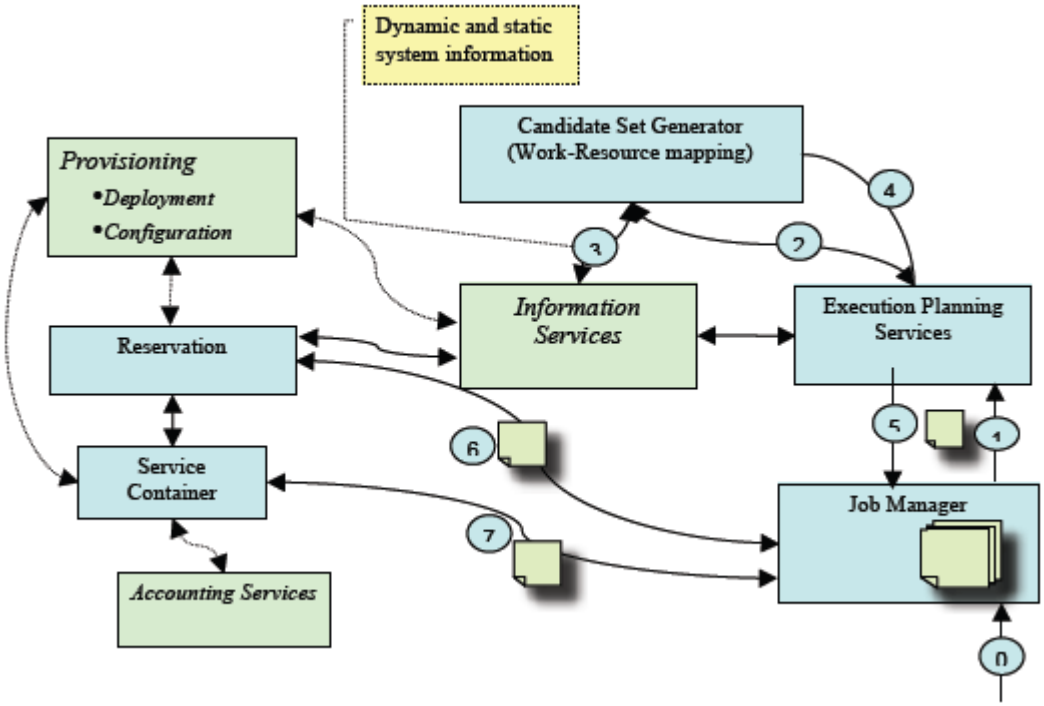
OGSA spoléhá na Web Services Architecture (WS-Architecture), na jejíž vývoj přispívá. V praxi to znamená, že infrastruktura OGSA je založena na této architektuře a zároveň využívá rozhraní definovaná v Web Services Description Language (WSDL) 1.1. Dorozumívacím jazykem je jazyk XML, pro výměnu zpráv mezi službami OGSA slouží SOAP.

Jelikož WSDL neposkytuje všechny služby, které grid poskytuje, OGSA spolupracuje na vývoji nové WSDL 2.0. Jednou z hlavních motivací pro tvorbu nového WSDL je zlepšení bezpečnosti WS-Security. OGSA nevyvíjí vlastní infrastrukturu a zcela spoléhá na WS-Architecture. [41]

6.1.2 Execution Management Services

Tato komponenta se zaměřuje na vytváření, dokončení a správu jednotek prací. EMS hledá zdroje, které dokážou splnit daný požadavek. Poté se z vhodně zvolených kandidátů vybere zdroj, jenž požadavek splní. Nakonec EMS připraví práci pro zdroj, spustí práci a monitoruje a spravuje, dokud není práce dokončena.

EMS se dělí do třech tříd: zdroje, správa prací a služby pro výběr zdroje.



Obrázek č. 3 – Execution Management Services [41]

V grafu je EMS znázorněno na bodech 1 až 4.

1. Definice práce a její požadavky
2. Nelezení vhodného zdroje a vybrání vhodného zdroje pro spuštění práce
3. Zajištění zdroje a vše co s tím souvisí
4. Monitorování práce po celou dobu průběhu a reakce v případě problémů, např. v případě selhání, předávání práce jinému zdroji [41]

6.1.3 Data Services

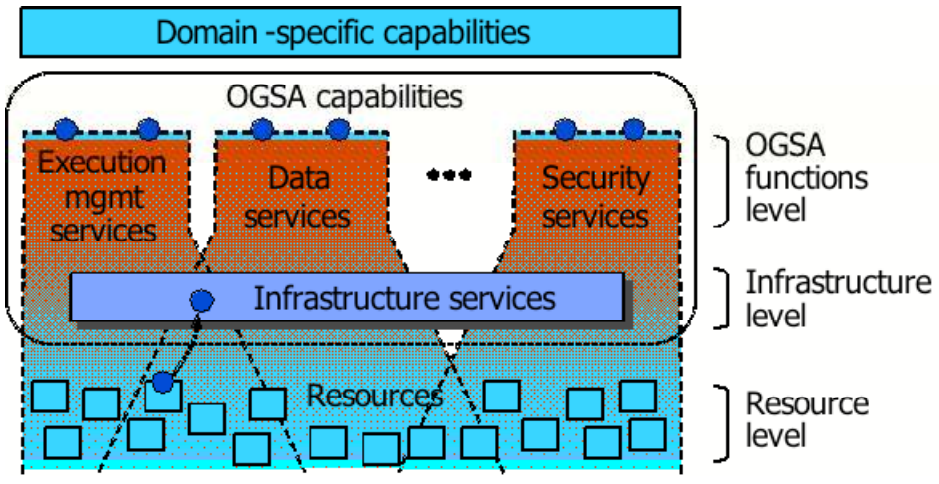
Služby DS slouží ke správě, přístupu a aktualizaci datových zdrojů. Datové služby zajišťují jiným komponentám, například EMS, přístup k datům jak lokálně, tak vzdáleně. V případě několika kopií, replik, datové služby udržují data konzistentní, tj. data se nesmí lišit a jsou aktualizována na všech místech dle potřeby.

Mezi datové zdroje patří především soubory, streamy, databáze, katalogy a nakonec i samy datové služby také mohou obsahovat další data. [41]

6.1.4 Resource Management Services

Služby zdrojů jsou rozděleny do třech typů :

1. Správa fyzických a logických zdrojů
2. Správa zdrojů gridu náležejících pod OGSA (např. rezervace zdrojů, posílání a monitoring prací)
3. Správa infrastruktury gridu.



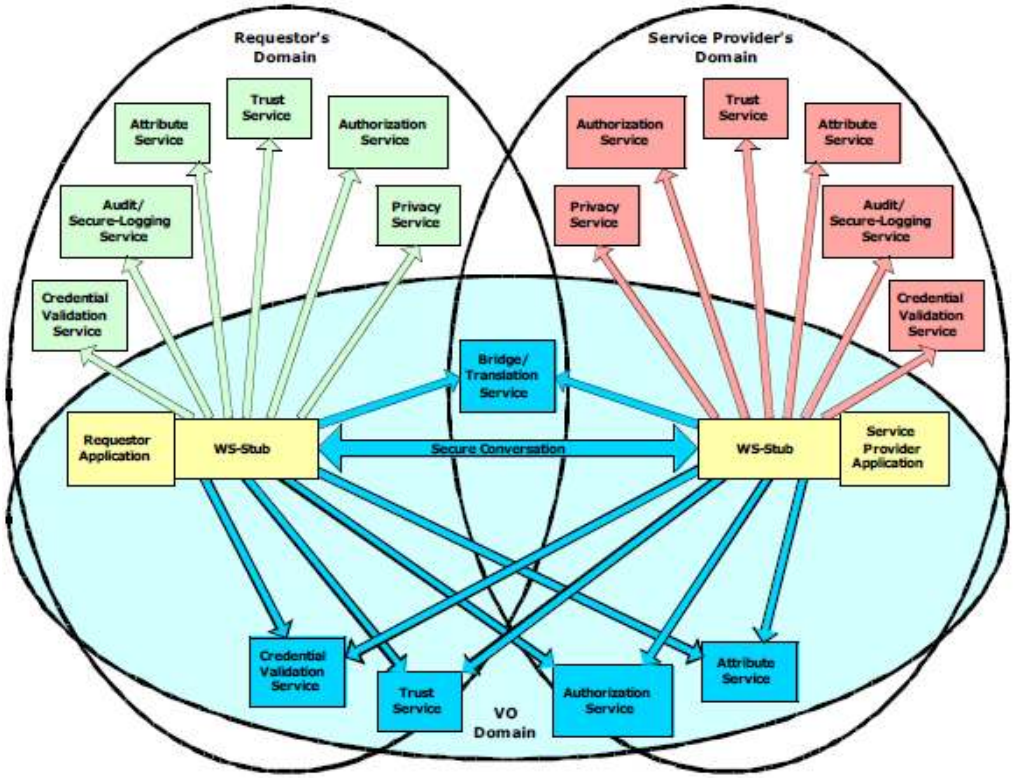
Obrázek č. 4 – Resource Management Services [41]

Typ 2 a 3 patří pod „OGSA functions level“, tzn. že tyto správy zdrojů operují na úrovni OGSA, typ 1 operuje zároveň na úrovni infrastruktury a zdrojů – využívá například protokoly SNMP nebo JMX. [41] [42] [43]

6.1.5 Security Services

Grid musí být v komerčním prostředí velmi dobře zabezpečen. V akademické prostředí nemusí bezpečnost hrát tak důležitou roli, jelikož grid může být postaven pouze pro místní akademické účely bez přístupu zvenčí.

Security Services slouží k definici bezpečnostní politiky OGSA. Cílem je vytvořit standard, jenž je schopen odolat útokům a zároveň ho lze integrovat do již existující bezpečnostního řešení. [41]



Obrázek č. 5 – Security Services [41]

Ilustrace zobrazuje komunikaci a Security Services. SS se dělí na dvě domény – na doménu poskytovatele služby a doménu klienta požadujícího službu. Komunikace mezi oběma subjekty probíhá pomocí zabezpečené WS-Stub.

6.1.6 Self-Management Services

Jedná se o služby, které nejsou ve finální fázi a jsou stále ve vývoji. Self-Management Services se zaměřuje na snížení nákladů a náročnosti na stavbu IT infrastruktury. Self-management (do češtiny lze přeložit jako samo-správcovství) má za cíl co nejmenší vliv lidí na chod hardwarové infrastruktury (sítě, procesory, úložiště) a softwarové infrastruktury (operační systémy, aplikace), tzn. automatickou konfiguraci, automatické řešení problémů a automatickou optimalizaci.

Self-Management Services silně spolupracuje s ostatními komponentami například při automatickém hledání vhodné zdroje při provádění práce, při rezervaci zdrojů pro provádění práce apod. [41]

6.1.7 Information Services

Information Services (česky informační služby) je komponenta starající se o přístup a manipulaci s informacemi o aplikacích, zdrojích a službách. IS využívají další služby pro získávání statických i dynamicky měnících se informací a pro přístup k logům. Typicky jde o zjišťování informací o zdrojích a vyhledávání zdrojů, monitoring apod. Protože jde o zásadní informace pro grid, IS je navrženo tak, aby bylo schopno překonat menší selhání.

7 Globus Alliance

Globus Alliance je uskupení univerzit a institucí, které se zabývá vývojem Grid Computingu. Jejich hlavním produktem je toolkit splňující kritéria daná standardem OGSA Globus Toolkit.

Globus Alliance sídlí v americké Argonne National Laboratory. Aliance se skládá z Univerzity Jižní Karolíny, Univerzity v Chicagu, skotské Univerzity v Edinburghu, švédského Centra pro paralelní počítače a amerického Národního centra pro aplikace superpočítačů (NCSA).

Globus, Globus Alliance a Globus Toolkit jsou obchodními značkami Univerzity v Chicagu. [44]

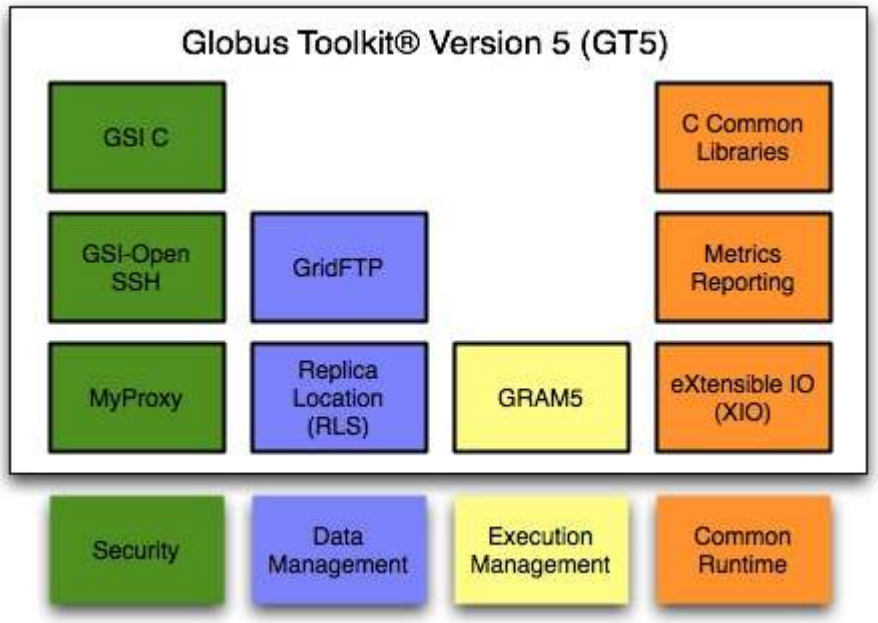
Organizace získává peníze od sponzorů mezi něž, mimo státní organizace, patří i soukromé společnosti IBM Corporation, Microsoft Research a Cisco Systems Corporation. [45]

Na tvorbě Globus toolkitu se, mimo mnohých dalších, podílí i „vynálezci“ technologie Grid Computing Ian Foster a Carl Kesselman. [46]

7.1 Globus Toolkit

7.1.1 Historie

První verze toolkitu vyšla v roce 1998. Toolkit od svého vzniku získal množství ocenění a pochvalného přijetí. V roce 2002 toolkit získal mezi 100 technologiemi ocenění "Most Promising New Technology" časopisu R&D. V současné době je k dispozici verze 5.0.1. [47][48]



Obrázek č. 6 – rozdělení Globus Toolkitu [47]

7.1.2 Globus Toolkit

Globus Toolkit je implementace standardu OGSA. Toolkit je modulární a stává se z komponent. [47]

7.1.2.1 Bezpečnostní komponenty

7.1.2.1.1 GSI C

GSI C poskytuje autentizaci a autorizaci před komunikací přes webové služby. Autentizace probíhá pomocí veřejného klíče, ku příkladu pomocí certifikátu X.509.[49]

X.509 je certifikát navrhnutý již v roce 1988, dnes je verzi 3. Tento certifikát využívá množství protokolů a standardů, např. HTTPS, SSH, LDAP a WS-Security. [50][51]

7.1.2.1.2 GSI - OpenSSH

GSI – OpenSSH je upravená verze OpenSSH rozšířená o podporu certifikátu X.509, vzdálené přihlášení a přenos souborů. [52][53]

OpenSSH je nástroj pro zabezpečení přenosu dat po síti. Tento nástroj šifruje veškerou komunikaci včetně hesel, poskytuje zabezpečené přihlášení a zabezpečuje data při cestě po síti. [54]

7.1.2.1.3 MyProxy

MyProxy je nástroj pro správu certifikátů X.509. MyProxy je úložiště těchto certifikátů a vydává tyto certifikáty, je-li o to požádáno. [55]

7.1.2.2 Komponenty správce dat

7.1.2.2.1 GridFTP

Komponenta pro přenos souborů postavená na protokolu FTP rozšířená o některé specifické požadavky gridů. Mezi specifické požadavky patří požadavky na bezpečnost, komunikace mezi jedním klientem a větším počtem serverů (FTP je komunikace klient-server), možnost přerušení stahování a stahování pouze části souboru, podpora velkých souborů, podpora datových kanálů v případě přenosu mezi stejnými body – není potřeba nového přihlášení atd. [56]

7.1.2.2.2 RLS

Replica Location Service je komponenta pro vytváření a vyhledávání replik. V rámci této komponenty jsou dvě služby LRC (katalog) a RLI (index). Index uchovává jména logická jména souborů, katalog jejich logická jména spolu s umístěním. [57]

7.1.2.3 Komponenty správce prací

7.1.2.3.1 GRAM5

Grid Resource Allocation and Management je komponenta pro správu prací ve zdrojích gridu. GRAM má na starosti spouštění vzdálených prací a jejich správu, rozhraní pro správu místních zdrojů, práci se souboru a kontrolu a monitoring prací. [58]

7.1.2.4 Common Runtime

7.1.2.4.1 C Common Libraries

C Common Libraries poskytuje abstraktní vrstvu pro datové typy a struktury a systémové požadavky. [59]

7.1.2.4.2 Metrics Reporting

Informace pro vývojáře Globus Toolkitu. Odesílá obecné informace o tom, kolik lidí služby využívá, kolik běží prací, jaké služby jsou spuštěny apod. [60]

7.1.2.4.3 eXtensible IO (XIO)

Globus XIO je knihovna pro jednotný přístup k protokolům. Toto API poskytuje příkazy open, close, read a write pro protokoly CP, UDP, file, HTTP, GSI, GSSAPI_FTP, TELNET a práce s frontou. [61]

8 GridGain

8.1 Společnost GridGain Technologies

GridGain Technologies je americká společnost založená v roce 2005 ve městě Pleasanton, Kalifornie. Zakladatelem společnosti je Nikita Ivanov a spoluzakladatelem Dmitriy Setrakyan.

Cílem společnosti je vytvořit dostupný a uživatelsky přívětivý produkt na bázi technologie Grid Computing, který by umožnil snadné propojení s programovacím jazykem Java a JEE. Takovým produktem by měl být jejich produkt GridGain™ framework. [3][62]

8.1.1 Produkty společnosti GridGain Technologies

GridGain Technologies nabízí produkt GridGain™ ve verzi Community Edition, Professional Edition a Enterprise Edition. Dále nabízí podporu a trénink k těmto produktům a odborné konzultace. [63]

8.1.1.1 GridGain™ Community Edition

Varianta GridGain zdarma. Této verzi chybí především privátní podpora vývojářů. Lze však využívat veřejné diskusní fórum a email podpory. [63]

8.1.1.2 GridGain™ Professional Edition

Profesionální edice stojí od $2000 ročně. V rámci této edice lze vyžadovat reakci podpory během dvou dní a máte kontakt na jednoho zástupce

firmy, ten vám v případě vážnějších problémů pomůže až pětkrát měsíčně. Navíc se lze obrátit na produktové specialisty. [63]

8.1.1.3 GridGain™ Enterprise Edition

Nejvyšší varianta produktu Vám zpřístupňuje za částku začínající na $3000 kontakt na 3 zástupce firmy, reakci během 2 hodin a neomezenou podporu v případě problémů. Kromě produktových specialistů je k dispozici i Account manažer. [63]

8.2 GridGain™ ve verzi Community Edition

Community Edition se z programovacího hlediska nijak neliší od placených variant, je ochuzena pouze o podporu vývojářů. [63]

Aplikace je šířena zdarma pod licencemi LGPL a Apache 2.0. [64]

V současné době je k dispozici verze GridGain 2.1, GridGain 3.0 je v současné době k dispozici pouze registrovaným uživatelům v uzavřené beta verzi. [65]

8.2.1 Architektura SPI

Architektura SPI je důležitou součástí GridGain. Service Provider Interface (SPI) poskytuje množství důležitých funkcí a nastavení. Každý SPI může mít několik zcela odlišných implementací. [66]

SPI	Popis
Discovery	Zodpovědný za vyhledávání nových klientů a detekování odpadlých klientů
Communication	Umožňuje komunikaci mezi klienty
Collision	Umožňuje plánování práce a řešení v případě problémů
Failover	Umožňuje přidat vlastní řešení problému v případě selhání klienta
Topology	Umožňuje přidat vlastní řízení topologie
Load Balancing	Umožňuje přidat vlastní řešení vyrovnávání výkonu mezi stanicemi
Checkpoint	Umožňuje přidat ukládat výpočty v průběhu počítání (tzv. checkpoint)
Deployment	Zodpovědný za zpracování úloh
Event Storage	Úložiště pro údálosti
Tracing	Umožňuje přidat vlastní externí měřící nástroje
Metrics	Poskytuje aktuální data o klientech

Tabulka č. 2 – Přehled SPI

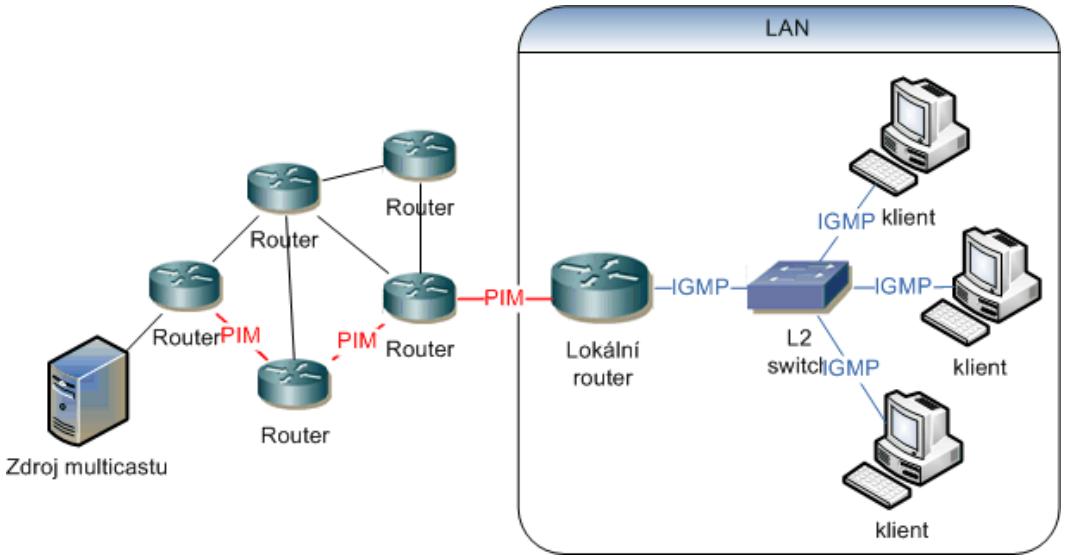
8.2.2 Klíčové vlastnosti a funkce

8.2.2.1 Odesílání dat klientům

Standardně GridGain pro odesílání dat klientům využívá technologie IP-Multicast.

Standardní komunikace na síti probíhá navázáním spojení s každým klientem a odesíláním dat, v takovém případě však dochází k nárůstu duplikace odesílaných dat a značnému zatížení sítě.

IP-Multicast je komunikace mezi jedním odesílatelem a množstvím příjemců, kterým se odesílají stejná data. Odesílatel odesílá data pouze jednou pro každou větev sítě a k vytvoření kopie dat dochází pouze při větvení. [67][68]



Obrázek č. 7 – IP-Multicast [67]

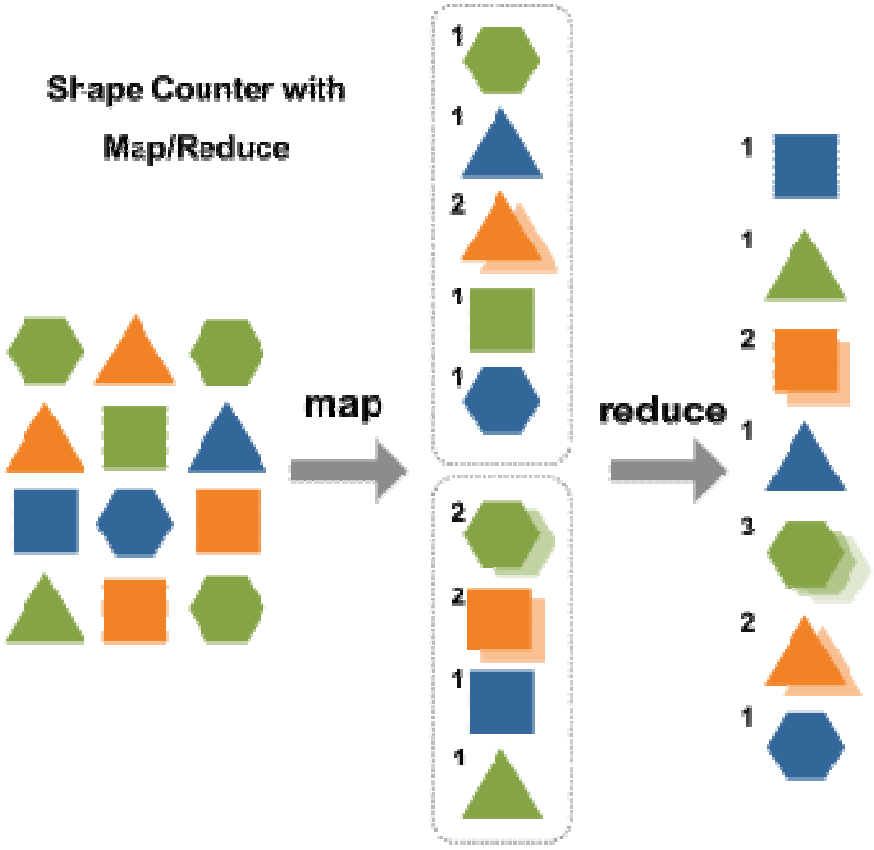
IP-Multicast využívá Communication SPI.[69]

8.2.2.2 MapReduce paradigma

Model MapReduce je u GridGain využíván pro rozdělení a získávání výsledků.

S tímto programovacím modelem přišel v roce 2003 Google, který jej také má patentovaný. Inženýři Google se inspirovali v programovacím jazyce Lisp. Veškerá práce se převádí na práci se seznamy a operace map a reduce. Google technologii využívá pro distribuci dat a získávání výsledků z vlastních gridů.

GridGain MapReduce umožňuje rozdělit práci na menší části na základě určitého klíče a výsledky opět složit. [69][70][71][72]



Obrázek č. 8 - Ukázka použití MapReduce [73]

Tuto technologii má implementována třída `GridTask`. Třída má tři metody:

- `map(List<GridNode> subgrid, @Nullable T arg)` - slouží k rozdělení úlohy na jednotlivé práce.
- `result(GridJobResult result, List<GridJobResult> received)` - získává v průběhu výpočtu výsledky jednotlivých prací a rozhoduje co s nimi, zda je vyhodnotí jako úspěšné a provede s nimi další operace nebo zda je předá jinému klientu.

- `reduce reduce(List<GridJobResult> results)`- vrací doposud vypočítané výsledky.

9 Wolfram Mathematica 7

9.1 Představení Wolfram Mathematica 7

Wolfram Mathematica 7 je poslední verze matematického software od společnosti Wolfram Research. Balík Mathematica obsahuje množství funkcí nejen pro matematické výpočty, ale i pro vizualizaci či pro jiné vědní obory, například chemii, ekonomii či optiku. [74]

Možnosti software Mathematica 7 lze rozšířit pomocí placených modulů. Pro paralelizaci výpočtů Wolfram Research nabízí dva produkty - Wolfram Lightweight Grid Manager a gridMathematica. [75]

9.2 Paralelní výpočty bez placených modulů

Standardní edice software Mathematica 7 bez modulů má omezený počet výpočetních vláken na 4. [76]

9.2.1 Parallelize

Bez placených modulů lze v základním balíku Mathematica 7 využít metody `Parallelize`. Tento příkaz automaticky distribuuje dílčí části výpočtu mezi výpočetní jádra / procesory na dané stanici. Metoda není určena k distribuci výpočtů přes síť nebo jiné kooperaci mezi dvěma a více stanicemi.

Metoda má několik několik možností nastavení z nichž základní je Automatic. Další možnosti jsou rozdělení výpočtu na co nejmenší části (FinestGrained), rozdělení na počet částí odpovídající počtu výpočetních jednotek (CoarsestGrained), na přesně definovaný počet částí na jednu výpočetní jednotku (EvaluationsPerKernel) a rozdělení výpočtu do intervalu přesně stanovené délky (ItemsPerEvaluation). [77]

9.2.2 ParallelTry

Další příkaz, `ParallelTry`, se od `Parallelize` liší tím, že nejsou rozkládány samotné příklady na dílčí části, ale jedná se o paralelní zpracování celých příkladů a to takovým způsobem, že první pravdivý výsledek metodu ukončí. Touto metodou tedy hledáme, zda pro dané příklady existuje řešení. Pro vypsání vyššího množství výsledků je nutné nastavit parametr `k`. [78]

9.2.3 ParallelEvaluate

`ParallelEvaluate` funguje stejně jako příkaz `Parallelize` jen s tím rozdílem, že je možné pomocí parametru ve volání `ParallelEvaluate[výraz, výpočetní jádro]` specifikovat výpočetní jednotku, na které úloha poběží. [79]

9.3 gridMathematica

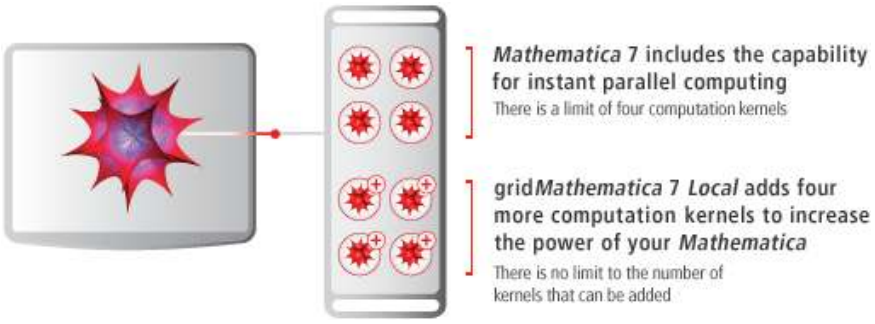
Doplněk gridMathematica je k dispozici ve dvou verzích gridMathematica Local a gridMathematica Server. Navíc je k dispozici správce Wolfram Lightweight Grid Manager.

Jednotlivé verze se liší počtem podporovaných výpočetních vláken a podporou síťových výpočtů.

Pro výpočet není potřeba nijak upravovat příkazy, metoda `Parallelize` pracuje shodně jako v případě paralelizace bez tohoto doplňku.

9.3.1 gridMathematica Local

Produkt přidává podporu dalších 4 výpočetních vláken na místní pracovní stanici. [80]



Obrázek č. 9 - Ilustrace produktu gridMathematica Local [80]

9.3.2 gridMathematica Server

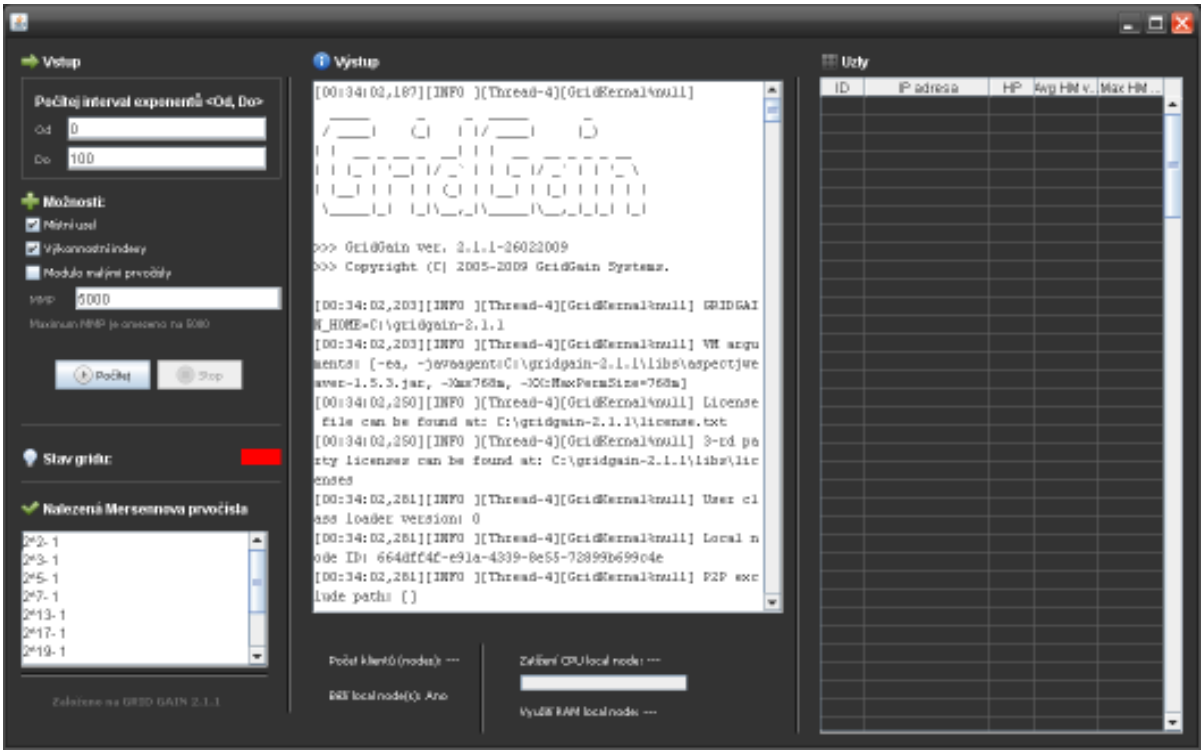
gridMathematica Server přidává podporu výpočtů po síti a podporu pro dalších 16 výpočetních vláken. [81]

9.3.3 Wolfram Lightweight Grid Manager

Wolfram Lightweight Grid Manager umožňuje správu připojených výpočetních vláken pomocí webového rozhraní. [82]

10 Implementace výpočtů

10.1 Program GridGain



Obrázek č. 10 – GUI vlastní aplikace

Pro potřeby bakalářské práce bylo implementováno grafické uživatelské rozhraní, viz. obrázek č. 10.

10.1.1 Algoritmy

V aplikaci bylo pro počítání a ověřování Mersennových prvočísel využito třídy `java.math.BigInteger`. Vlastní řešení nebylo implementováno, protože třída `BigInteger` a konkrétně metoda

`isProbablePrime()` je implementována a obsažena ve standardní edici programovacího jazyku Java v balíku `java.math`.

10.1.2 Umocnění čísla

Umocnění využívá metody `pow(int exponent)` ze třídy `BigInteger`. [83][84]

10.1.3 Ověřování prvočísla

Metoda `isProbablePrime(int certainty)` je metoda, která vrací `true` v případě, že se pravděpodobně jedná o prvočíslo a `false` v případě, že se zcela jistě nejedná o prvočíslo.

Výpočetní část je uložena v metodách `primeToCertainty(int certainty)`, `passesMillerRabin(int rounds)` a `passesLucasLehmer()`. Třída `BigInteger` pro ověření prvočísla využívá dvou algoritmů, pravděpodobnostního Rabin-Miller algoritmu a deterministického Lucas-Lehmer algoritmu. Parametr `int certainty` dvou výše uvedených metod vyjadřuje pravděpodobnost, s jakou se algoritmus Miller-Rabin může mýlit. Čím vyšší hodnota parametru, tím vyšší pravděpodobnost, že výsledek bude pravdivý. Pravděpodobnost odpovídá vzorci $1 - 1/2^{\text{parametr certainty}}$.

Nejdříve se volá algoritmus Rabin-Miller a je-li vyhodnocen jako pravdivý, je potenciální prvočíslo ověřeno deterministickým algoritmem Lucas-Lehmer. Metoda `isProbablePrime(int certainty)` se nemůže za běžných okolností mýlit. [83] [84]

10.1.4 Implementace software

Implementace probíhala na platformě x86 s operačním systémem Microsoft Windows XP Verze 2002 SP3 CZ.

Nainstalovaná verze Java Development Kit Java JDK 1.6.0.18.

Pro klientskou aplikaci se v práci využívá standardně dodávaného klienta realizovaného souborem gridgain.bat. Konfigurace GridGain probíhala pomocí Spring XML frameworku. Zde se měnilo množství paralelně zpracovávaných prací pomocí vlastnosti `executorService` v JavaBean `java.util.concurrent.ThreadPoolExecutor`. Hodnota byla snížena z továrního nastavení 100 na hodnotu 10, není-li dále řečeno jinak. Dalším nastavením klienta bylo rozšíření paměti Java Heap Space. Standardní nastavení –Xms128m a –Xmx128m se změnilo na –Xms800m a –Xmx800m. Parametr Xms vyjadřuje počáteční velikost paměti v MB paměti Java Heap Space, parametr Xmx je maximální velikost Java Heap Space.[85] Tato změna je nutná především z hlediska počtu paralelně zpracovávaných prací.

Serverová část aplikace byla naprogramována v programovacím jazyce Java na téže sestavě v prostředí NetBeans IDE 6.5.

10.1.5 Zrychlení výpočtu

10.1.5.1 Dělení práce dle výkonu klienta

Aplikace má naprogramované dva módy rozdělování prácí, rozdělení počítaného intervalu na stejně dlouhé části dle počtu klientů a rozdělení intervalu na různě dlouhé části dle výkonu jednotlivých klientů s tou vlastností, že zbytek je distribuován nejvýkonnějšímu stroji v gridu. Jednotliví klienti jsou zapsáni v seznamu dvouprvkových polí, kde prvním prvkem pole je IP adresa klienta a druhým jeho výkon při hledání Mersennových prvočísel 2^{n-1}, kde n je

interval <0, 2500> v porovnání se strojem č. 2. Tyto hodnoty se zapisují manuálně do zdrojového kódu aplikace.

Tato technologie byla pro potřeby bakalářské práce nazvána výkonnostní indexy.

10.1.5.2 Zvýšení výkonu pomocí dělení malými prvočísly

Dalšího zvýšení výkonu se dosáhlo pomocí dělení malými prvočísly.

Metoda `isProbablePrime(int certainty)` ze třídy `BigInteger` okamžitě podrobí zkoumané číslo Rabbin-Millerovu a Lucas-Lehmerovu algoritmu. Zvýšení výkonu jsem dosáhl tím, že zkoumané číslo nejdříve dělím prvočísly. Pokud je zkoumané číslo modulo libovolné prvočíslo 0, potom je toto prvočíslo dělitelem. Odstraní se tak velmi rychle například sudá čísla a není třeba toto číslo zkoumat náročnými algoritmy.

Může dojít i k negativnímu vlivu použitím této optimalizace a to v případě, obsahuje-li zadaný interval převážně prvočísla. V takovém případě je výše popsané dělení provedeno, ale protože nebyl nalezen žádný dělitel, je dále toto prvočíslo zkoumáno pokročilejšími algoritmy. K tomuto případu však dojde velmi zřídka, i rychlé odstranění byť jen jediného čísla může znamenat, převážně u větších prvočísel, značný výkonnostní nárůst. Dělení je operace velmi rychlá v porovnání s pokročilejšími algoritmy, jak je dokázáno níže.

Tato technologie byla pro potřeby bakalářské práce nazvána modulo malými prvočísly.

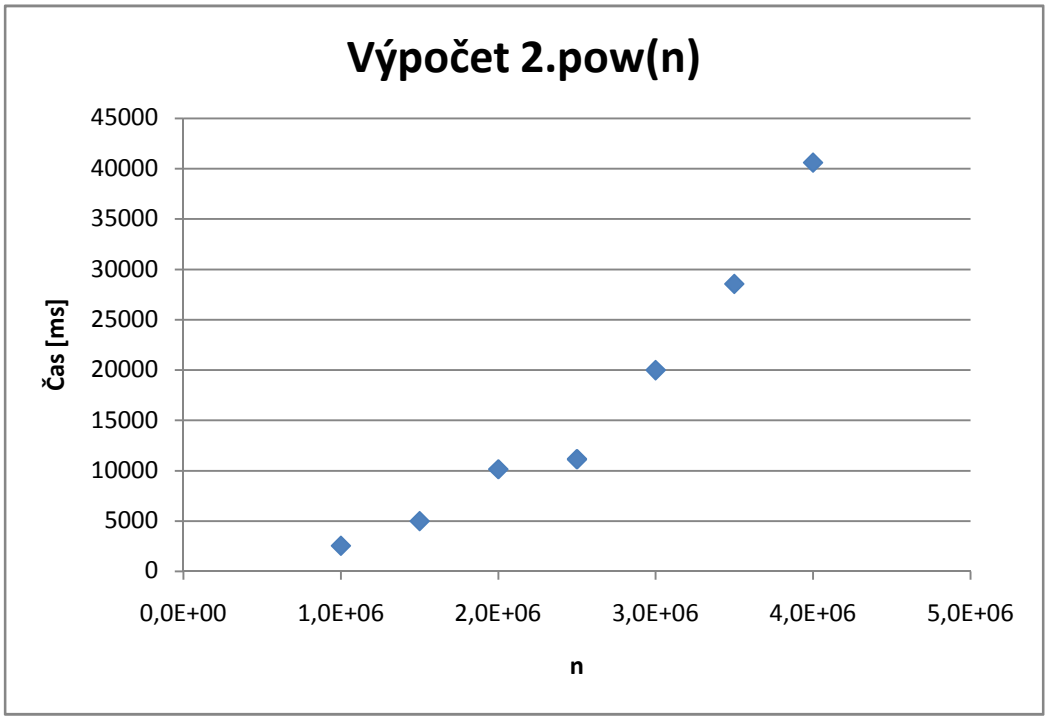
10.1.6 Časová náročnost využitých algoritmů

Níže uvedené výsledky byly otestovány na sestavě č.1, hardwarové specifikace viz. příloha č. 1.

10.1.6.1 Umocnění čísla

BigInteger.valueOf(2).pow(n)		
n	Čas [ms]	Nárůst náročnosti oproti předchozí hodnotě
1,0E+06	2547	
1,5E+06	4984	95,7%
2,0E+06	10140	103,5%
2,5E+06	11141	9,9%
3,0E+06	19985	79,4%
3,5E+06	28547	42,8%
4,0E+06	40594	42,2%

Tabulka č. 3 – Umocnění čísla pomocí třídy BigInteger

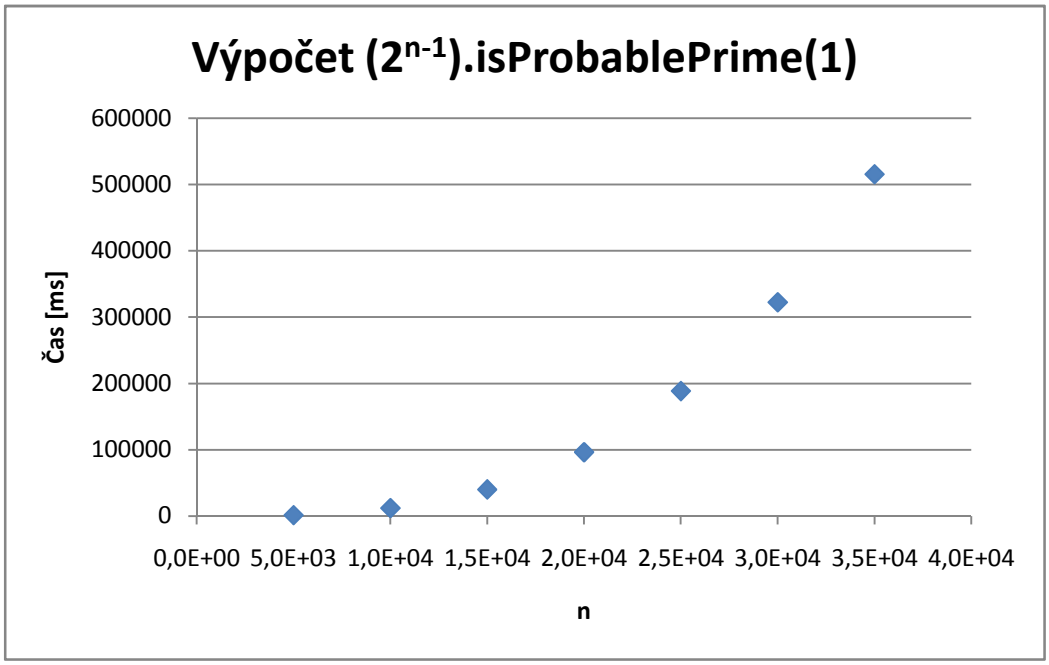


Graf č. 1 – Umocnění čísla pomocí třídy BigInteger

10.1.6.2 Ověření prvočísla

(2n-1).isProbablePrime(1)		
n	**Čas [ms]**	**Nárůst náročnosti oproti předchozí hodnotě**
5000	1578	
10000	12125	668,4%
15000	40235	231,8%
20000	96406	139,6%
25000	188547	95,6%
30000	322391	71,0%
35000	515266	59,8%

Tabulka č. 4 – Ověření prvočísla pomocí třídy BigInteger



Graf č. 2 – Ověření prvočísla pomocí třídy BigInteger

10.1.6.3 Modulo prvočíslem

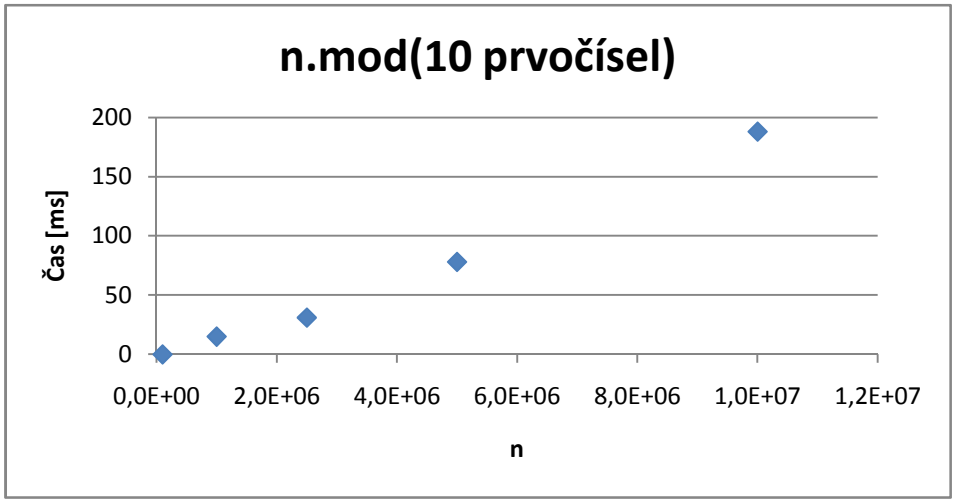
V této sekci je využito metody `n.mod(prvocislo)`, kde `n` je potenciální prvočíslo a `prvocislo` jsou pomocí cyklu `for` vkládána různá prvočísla.

První příkaz první řádky následující tabulky tak vypadá následovně: $2^{100000-1}$ % 2.

10.1.6.3.1 Prvních 10 prvočísel

BigInteger.mod(prvních 10 prvočísel)		
n	Čas [ms]	Nárůst náročnosti oproti předchozí hodnotě
100000	0	
1000000	15	
2500000	31	106,7%
5000000	78	151,6%
10000000	188	141,0%

Tabulka č. 5 – Modulo prvočíslem pomocí prvních 10 prvočísel

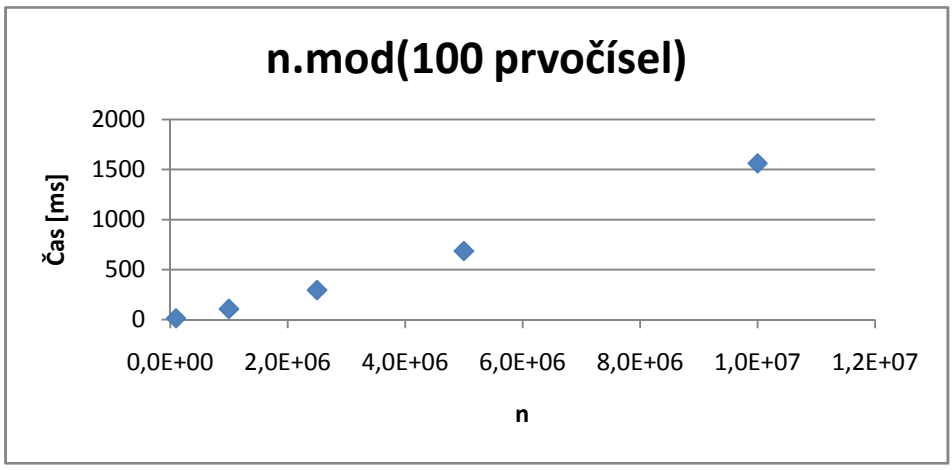


Graf č. 3 – Modulo prvočíslem pomocí prvních 10 prvočísel

10.1.6.3.2 Prvních 100 prvočísel

BigInteger.mod(prvních 100 prvočísel)		
n	Čas [ms]	Nárůst náročnosti oproti předchozí hodnotě
100000	16	
1000000	109	581,3%
2500000	297	172,5%
5000000	687	131,3%
10000000	1562	127,4%

Tabulka č. 6 – Modulo prvočíslem pomocí prvních 100 prvočísel

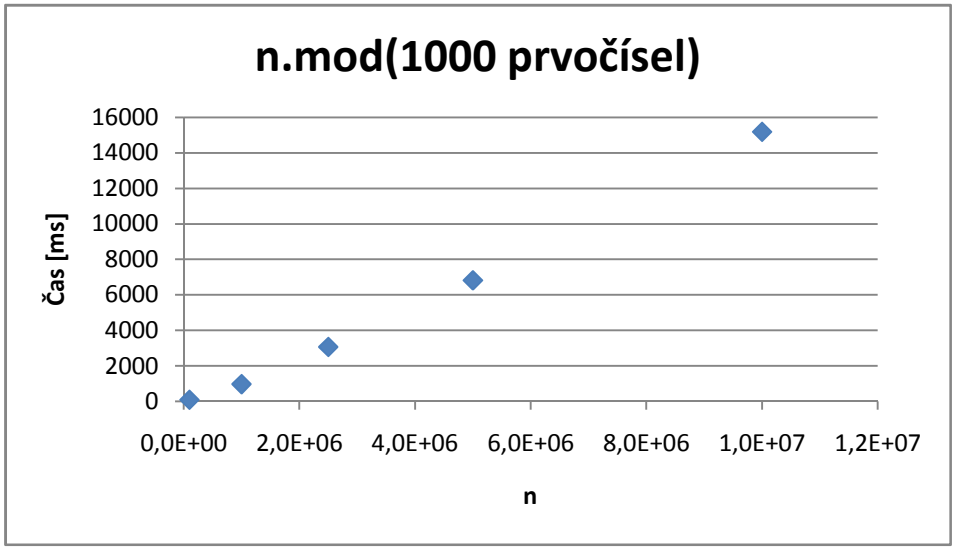


Graf č. 4 – Modulo prvočíslem pomocí prvních 100 prvočísel

10.1.6.3.3 Prvních 1000 prvočísel

BigInteger.mod(prvních 1000 prvočísel)		
n	Čas [ms]	Nárůst náročnosti oproti předchozí hodnotě
100000	93	
1000000	969	941,9%
2500000	3062	216,0%
5000000	6813	122,5%
10000000	15172	122,7%

Tabulka č. 7 – Modulo prvočíslem pomocí prvních 1000 prvočísel



Graf č. 5 – Modulo prvočíslem pomocí prvních 1000 prvočísel

10.1.6.4 Parametr Certainty pro Rabin-Miller algoritmus

Metoda `isProbablePrime(int certainty)` má parametr `int certainty`. V praktickém použití je však vliv parametru na dobu výpočtu zanedbatelný.

	Výp. jednotky					Hodnota paremtru aplikovaná na vzorec 2^{15001}-1				
	C1	C2	C3	C4	C5	1	3	5	100	1000
Běh 1						1578	1594	1625	1609	1594
Běh 2						1610	1578	1625	1594	1593
Běh 3						1610	1578	1610	1594	1672
Běh 4						1593	1578	1672	1609	1594
Běh 5						1579	1641	1578	1594	1625
Průměr						1594	1594	1622	1600	1616
Pravděpodobnost						$1 - 0{,}5^1$	$1 - 0{,}5^3$	$1 - 0{,}5^5$	$1 - 0{,}5^{100}$	$1 - 0{,}5^{1000}$

Tabulka č. 8 – Vliv parametru pravděpodobnosti na dobu výpočtu

Výsledek se liší pouze statistickou chybou v řádu desítek ms. Pravděpodobnost se ovšem liší velmi výrazně a to od 50% ($1 – 0{,}5^1$) až po téměř 100% ($1 – 0{,}5^{1000}$).

10.1.7 Interpretace výsledků

Z tabulek lze vypozorovat, že i exponenty pohybující se v jednotkách milionů jsou umocněny poměrně rychle. Nejvyšší testovaná hodnota, $2^{7000000}$, je vyřešena pomocí metody `pow(int exponent)` pod dvě minuty.

Naopak jasně vyplývá, že ověření prvočísla je výpočetně významně náročnější než umocnění. Pro porovnání jsem zvolil hodnotu 2^{35000}-1, kde umocnění trvá 15 ms, zatímco ověření prvočísla 515266 ms.

10.2 Program Mathematica 7

10.2.1 Umocnění čísla

Umocnění využívá příkazu `2 ^ proměnná - 1`, kde znak `^` je umocnění.

10.2.2 Ověřování prvočísla

Pro ověření, zda je číslo prvočíslo využívám metody `PrimeQ[expr]`.

Metoda `PrimeQ[expr]` nejdříve výraz v závorkách vydělí malými prvočísly, poté výraz podrobí testu Miller-Rabin a nakonec použije Lucas-Lehmer algoritmus. [86]

11 Experimentální výpočty

V následující kapitole jsou pro ušetření místa využívány dvě zkratky. VI jsou výkonnostní indexy, MMP je modulo malými prvočísly.

11.1 Výsledky vlastní aplikace

Aplikace serveru běžela na stroji č. 1.

11.1.1 Výpočty bez gridu

Následující tabulka je rozdělena do 8 sekcí. Vždy se jedná o výpočet vyznačeného intervalu ve vzorci $2^n - 1$ na vyznačené pracovní stanici (podrobný hardwarový popis stanic je v příloze č. 1).

Stanice má k dispozici buď jednu, nebo dvě výpočetní jednotky, což je zvýrazněno barvou buňky ve sloupci „Výp. jednotky“. Počet paralelně zpracovávaných prací na jedné stanici je 10. Vypnutí jádra u sestavy č. 1 probíhalo softwarově pomocí odebrání spřažení procesory s daným procesem ve Správci úloh systému Windows.

Stanice	Výp. jednotky					Interval dosazovaný do 2^n-1				
	C1	C2	C3	C4	C5	0 - 2500	2500 - 3500	3500 - 4000	10000 - 10010	10000 - 10100
č. 1*						166172	359485	333937	131484	1248625
č. 1**						83718	179219	163719	68515	608563
č. 2						419922	914547	856094	342437	3182532
č. 3						161906	393296	369625	158953	1383062
Zapnutá optimalizace MMP s prvními 25 prvočísly										
č. 1*						N/A	90734	76640	48562	276079
č. 1**						36937	45125	38547	23875	138344
č. 2						128938	224344	198234	127250	723813
č. 3						51968	95375	84234	53156	309281
Zapnutá optimalizace MMP s prvními 100 prvočísly										
č. 1*						N/A	61922	51484	24032	183062
č. 1**						31734	31141	25000	11859	79750
č. 2						103328	144922	123750	64890	411391
č. 3						39688	62156	53437	27312	176187
Zapnutá optimalizace MMP s prvními 5000 prvočísly										
č. 1*						N/A	60219	48375	23922	156390
č. 1**						32656	31094	24516	12234	78281
č. 2						112437	116250	104094	64063	352141
č. 3						38359	50656	45094	27078	157922
Poměr časů výpočtu mezi MMP s prvními 25 prvočísly a MMP s prvními 100 prvočísly										
č. 1*						N/A	68,2%	67,2%	49,5%	66,3%
č. 1**						85,9%	69,0%	64,9%	49,7%	57,6%
č. 2						80,1%	64,6%	62,4%	51,0%	56,8%
č. 3						76,4%	65,2%	63,4%	51,4%	57,0%
Poměr časů výpočtu mezi MMP s prvními 100 prvočísly a MMP s prvními 5000 prvočísly										
č. 1*						N/A	97,2%	94,0%	99,5%	85,4%
č. 1**						102,9%	99,8%	98,1%	103,2%	98,2%
č. 2						108,8%	80,2%	84,1%	98,7%	85,6%
č. 3						96,7%	81,5%	84,4%	99,1%	89,6%
Poměr časů výpočtu mezi MMP s prvními 25 prvočísly a MMP s prvními 5000 prvočísly										
č. 1*						N/A	66,4%	63,1%	49,3%	56,6%
č. 1**						88,4%	68,9%	63,6%	51,2%	56,6%
č. 2						87,2%	51,8%	52,5%	50,3%	48,7%
č. 3						73,8%	53,1%	53,5%	50,9%	51,1%
Poměr časů výpočtu mezi neaktivní MMP a MMP s prvními 5000 prvočísly										
č. 1*						N/A	16,8%	14,5%	18,2%	12,5%
č. 1**						39,0%	17,3%	15,0%	17,9%	12,9%
č. 2						26,8%	12,7%	12,2%	18,7%	11,1%
č. 3						23,7%	12,9%	12,2%	17,0%	11,4%

* - jedno výpočetní jádro, ** - dvě výpočetní jádra

Tabulka č. 9 – výpočet zvolených intervalů na jednotlivých stanicích bez MMP

Tabulka č. 9 obsahuje dobu výpočtu intervalů na jednotlivých strojích bez zapojení počítání přes síť. Bez zapojení přes síť lze využít pouze lokálních možností paralelizace.

Z první části tabulky lze vypozorovat přínos další výpočetní jednotky v témže stroji (stroj č. 1* a stroj č. 1**).

Z tabulky je jasně vidět zrychlení výpočtu s pomocí optimalizace MMP. Například na dvoujádrovém stroji č. 3 došlo u nejnáročnějšího intervalu <10000, 10100> k zrychlení výpočtu s pomocí MMP s prvními 5000 prvočísly z 23 minut a 3 sekund na 2 minuty a 38 sekund (snížení doby výpočty oproti původní hodnotě na 11,4%).

S výjimkou intervalu <0, 2500> vždy došlo k nárůstu výkonu s MMP s vyšším počtem prvočísel. U tohoto intervalu se také projevoval problém s nedostatkem paměťového prostoru při softwarovém vypnutí jednoho jádra procesoru, z tohoto důvodu výsledky v tabulce chybí.

11.1.2 Výpočty pomocí Grid Computing bez VI a MMP

	Výp. jednotky					Interval dosazovaný do 2^n-1				
	C1	C2	C3	C4	C5	0 - 2500	2500 - 3500	3500 - 4000	10000 - 10010	10000 - 10100
Grid						128125	308110	286235	125469	1070891
Počet prací č. 1**						833	333	167	3	33
Počet prací č. 2						834	334	167	4	34
Počet prací č. 3						834	334	167	4	34
Změna oproti č. 1**						153,0%	171,9%	174,8%	183,1%	176,0%
Změna oproti č. 2						30,5%	33,7%	33,4%	36,6%	33,6%

Tabulka č. 10 – výpočet zvolených intervalů bez VI a bez MMP

Tabulka č. 10 obsahuje dobu výpočtu na všech strojích při zapojení počítání přes síť. Počítaný interval se rozdělí na stejně dlouhé části, zbytek prací se rovnoměrně distribuuje mezi uzly podle data připojení ke gridu.

Výsledkem je zpomalení výpočtu oproti stanici č. 1** až o 83,1%. Důvodem je rychlost stroje č. 2, ten je totiž 5,02 krát pomalejší než stroj č. 1. Ke zrychlení výpočtu tak dochází v porovnání se strojem č. 2, nikoliv v porovnání se strojem stroj č. 1**.

Jakmile jsou práce poslány všem výpočetním uzlům, začnou jednotlivé stanice práce zpracovávat. Stanice č. 1** zpracuje svůj interval nejrychleji a čeká na výsledky stanic č. 2 a 3, poté zpracuje své výsledky stanice č. 3 a čeká se na zpracování třetiny intervalu strojem č. 2. Po dokončení jeho části jsou výsledky prezentovány, stroj č. 1 a částečně i stroj č. 3 ovšem „zahálí“ při čekání na výsledky ostatních strojů.

11.1.3 Výpočty pomocí Grid Computing s VI, bez MMP

	Výp. jednotky					Interval dosazovaný do 2^n-1				
	C1	C2	C3	C4	C5	0 - 2500	2500 - 3500	3500 - 4000	10000 - 10010	10000 - 10100
Grid						59422	121390	113266	41500	414078
Počet prací č. 1**						1459	584	292	7	59
Počet prací č. 2						290	116	58	1	12
Počet prací č. 3						752	301	151	3	30
Změna oproti č. 1**						71,0%	67,7%	69,2%	60,6%	68,0%
Změna oproti variantě bez VI						46,4%	39,4%	39,6%	33,1%	38,7%

Tabulka č. 11 – výpočet zvolených intervalů s VI a bez MMP

Tabulka č. 11 obsahuje dobu výpočtu na všech strojích při zapojení počítání přes síť s optimalizací VI. Počítaný interval se rozdělí na různě dlouhé

části dle výkonu jednotlivých stanic. Zbytek prací se distribuuje nejvýkonnější stanici.

Výkon stanic vychází z doby potřebné pro výpočet intervalu <0,2500>.

Zde již dochází k nárůstu výkonu i oproti stanici č. 1**.

11.1.4 Výpočty pomocí Grid Computing s VI a MMP

	Výp. jednotky					Interval dosazovaný do 2^n-1				
	C1	C2	C3	C4	C5	0 - 2500	2500 - 3500	3500 - 4000	10000 - 10010	10000 - 10100
Běh 1						31391	19375	19797	26312	55016
Běh 2						34469	20687	15640	26531	54812
Běh 3						38062	19578	24125	26703	54890
Průměr						34641	19880	19854	26515	54906
Počet prací č. 1**						1459	584	292	7	59
Počet prací č. 2						290	116	58	1	12
Počet prací č. 3						752	301	151	3	30
Změna oproti variantě 1** s MMP 5000						96,1%	62,3%	80,8%	215,1%	70,3%
Změna oproti variantě bez VI						52,8%	16,0%	17,5%	63,4%	13,3%

Tabulka č. 12 – výpočet zvolených intervalů s VI a MMP

Tabulka č. 12 obsahuje dobu výpočtu všech strojů při zapojení počítání přes síť se zapnutými optimalizacemi VI a MMP s prvními 5000 prvočísly.

Tato hodnota je zaokrouhlená hodnota ze tří měření, výsledky měření se mohou velmi lišit a tato hodnota je pouze orientační. Jelikož je číslo, které je úspěšně vyřazeno pomocí MMP optimalizace, vyřazeno velmi rychle a naopak číslo, které není vyřazeno, je vypočítáno výrazně pomaleji, může dojít k různým stavům v gridu. V krajním případě může dojít i ke stavu, kdy nejpomalejší stroj získá pouze čísla, jenž nelze vyřadit pomocí MMP. V takovém případě nedojde k žádnému zrychlení, pouze k mírnému zpomalení

danému potřebou počítat jak MMP optimalizaci, tak ověřování prvočísla. Řešení problému je navrhnuto v následující kapitole.

11.1.5 Hardwarová náročnost aplikace

11.1.5.1 Procesorová náročnost

11.1.5.1.1 Klient

U klienta vytížení procesoru záleží na počtu paralelně zpracovávaných prací a počtu výpočetních jednotek (procesorů / procesorových jader). Jedna práce vytíží jednu výpočetní jednotku na 100% s občasným poklesem při získávání další práce.

11.1.5.1.2 Server

Server je vytížen především při přípravě dat pro grid, větší komunikaci v síti a při sběru výsledků. Server je vytížen spíše nárazově než průběžně.

11.1.5.2 Paměťová náročnost klienta

Paměťová náročnost se odvíjí od několika hledisek a to zejména:

- návrh algoritmů při hledání Mersennových prvočísel
- počet paralelně zpracovávaných prací
- řád počítaných exponentů

Měření probíhalo pomocí statistické třídy `GridNodeMetrics` a konkrétně metody `getHeapMemoryUsed()`, která vrací aktuální hodnotu využité paměti Java Heap Space.

Přesné výsledky zde nejsou uvedeny, jelikož u stejných měření docházelo k různým výsledků, přesto lze vypozorovat některá obecná pravidla.

Čím delší je počítaný interval, tím větší je paměťová náročnost. Vyšší řád exponentů se výrazněji projeví až u použití optimalizace MMP. Vliv paralelně zpracovávaných prací je bez MMP malý, s MMP naopak dochází i k trojnásobnému nárůstu využité paměti na stejném intervalu.

Pro klienta bez optimalizace MMP i server je výhodnější vyšší počet paralelně zpracovávaných prací. Klient s vyšším počtem výpočetních jednotek je efektivněji využit bez výrazné změny v paměťové náročnosti. Server je méně zatížen díky menšímu počtu síťových spojení.

11.2 Výsledky aplikace Mathematica 7

Pro testování je zvolena metoda `Parallelize`, která hledala Mersennova prvočísla v intervalu vytvořeném metodou `Range(i_min, i_max)`.

Výsledný příkaz pro interval <2500, 3500> vypadá `Sequence[st = AbsoluteTime[], Parallelize[Select[Range[2500, 3500], PrimeQ[2^# - 1] &]], ko = AbsoluteTime[], ko - st]`. Proměnné `ko` a `st` slouží pouze pro zjištění aktuálního času, z něhož se poté rozdílem zjistí doba trvání výpočtu.

11.2.1 Výsledky výpočtů

Stanice	Výp. jednotky					Interval dosazovaný do 2^n-1				
	C1	C2	C3	C4	C5	0 - 2500	2500 - 3500	3500 - 4000	10000 - 10010	10000 - 10100
č. 1*						5046	8703	6296	2109	16734
č. 1**						2843	4765	3359	2015	8765
č. 2						10687	18656	13562	4671	35984
č. 3*						10531	18218	13265	4501	35687
č. 3**						5828	10140	7218	4359	19750
Vliv zapojení dalšího výpočetního jádra na sestavě č. 1										
Změna č. 1** oproti variantě č. 1*						56,3%	54,8%	53,4%	95,5%	52,4%
Změna č. 3** oproti variantě č. 3*						55,3%	55,7%	54,4%	96,8%	55,3%

* - jedno výpočetní jádro, ** - dvě výpočetní jádra

Tabulka č. 13 – Výsledky na sestavách pomocí software Mathematica 7

Nárůst výkonu na dvou jádrech oproti jednomu jádru pomocí software Mathematica 7 na intervalu <10000, 10010> jsou pouze jednotky procent. Tento stav vyplývá z toho, že deset čísel je odstraněno velmi rychle a pouze jedno je řešeno složitějšími algoritmy. Využití stroje s dvěma jádry je tak pouze 50%, to odpovídá stroji s jednou výpočetní jednotkou. Ostatní intervaly jsou vypočítány téměř dvakrát rychle.

11.3 Interpretace výsledků

Intervaly jsou vypočteny pomocí software Mathematica výrazně rychleji než v případě aplikace GridGain. Rozdíl je způsobený implementací algoritmů pro ověřování prvočísel, případně dalšími nespecifikovanými algoritmy. Bohužel, Mathematica 7 není open-source a není možné algoritmy porovnat.

11.3.1 Návrhy na zlepšení vlastní aplikace

Aplikace lze dále vylepšovat za účelem zvýšení výkonu.

11.3.1.1 Problém MMP

Z kapitoly lze vypozorovat, že modulo malými prvočísly přidává značně na výkonu. Dalšího výkonu lze dosáhnout lepší distribucí prací. Zpomalení může nastat v případě, jsou-li čísla náročnější na výpočet distribuována slabším klientům.

Příklad na intervalu <1, 10>. Jsou-li zapnuty výkonnostní indexy a stroj č. 1 má výkonnostní index 2 a stroj č. 2 má výkonnostní index 1, potom distribuce může být realizována takto:

Stroj č. 1: 2, 4, 6, 7, 8, 9, 10

Stroj č. 2: 1, 3, 5

Stroj č. 1 získá 6 čísel, stroj č. 2 získá 3 čísla. Jelikož jedno číslo nebylo přiřazeno, je distribuováno nejsilnějšímu počítači v gridu, výsledkem je 7 čísel pro stroj č. 1 a 3 čísla pro stroj č. 2.

Z řady pro stroj č. 1 je pouze jediné číslo prvočíslo, z řady pro stroj č. 2 jsou všechna tři čísla prvočísla. Bylo dokázáno, že ověřování prvočísla je výpočetně významně náročnější a ještě s přihlédnutím k nižšímu výkonu druhého počítače, nedojde téměř k žádnému zrychlení oproti variantě bez modula malými prvočísly.

11.3.1.1.1 Navrhnuté řešení

Problém lze řešit na úrovni distribuce práce před samotným výpočtem nebo, je-li libovolný klient bez práce, pomocí tzv. „job stealing“.

Řešení na úrovni distribuce je problematické, jelikož nelze bez výpočtu odhadnout, zda dané číslo je prvočíslo nebo nikoliv. Řešením by mohlo být vytvořením jiného gridu, jenž by pracoval pouze s algoritmem modulo malými prvočísly a výsledky by poté byly distribuovány většímu gridu, který by ověřoval prvočísla. Vzhledem k rychlosti ověřování MMP by takový grid nemusel být velký. Tato varianta by měla být implementována v gridu s velkým počtem klientů.

Jiným řešením je použití tzv. „job stealing“. Job stealing (česky „kradení práce“) je technologie, při které klient, jehož fronta prací je prázdná a má k dispozici nevyužité výpočetní jednotky, převezme práce od jiných klientů s delší frontou nezpracovaných prací. Framework GridGain obsahuje třídu, jenž takové řešení nabízí. Tato varianta by měla být implementována v gridu s malým počtem klientů. Důvodem je zátěž a rychlost sítě a postradatelnost klienta. Síť by s touto variantou byla vystavována většímu přenosu dat a vyžadovala by větší datovou prostupnost například při kooperaci mezi tisíci klienty.

V malém gridu je klient méně postradatelný než v případě velkého gridu. Vytvořením malého gridu pouze pro optimalizaci MMP by komunikace způsobená Job Stealing technologií odpadla. Ve velkém gridu je několik dedikovaných stanic pro výpočet MMP postradatelných.

11.3.1.2 Lepší odhalování čísel

Vlastní aplikace dokáže na intervalu <10000, 10010> odhalit „pouze“ 7 složených čísel a zbylá 4 jsou poslána složitějším algoritmům. Naopak Mathematica zkoumá složitými algoritmy v tomto intervalu pouze jediné číslo.

Zlepšení odstraňování složených či jinak nevhodných čísel by mohlo významně přispět k výkonu aplikace.

12 Závěr

Bakalářská práce se zabývá teorií okolo technologie Grid Computing. Jelikož je aplikace této technologie prováděna na ověřování Mersennových prvočísel, je práce rozšířena o tuto tématiku.

Pro potřeby bakalářské práce byl úspěšně implementován v programovacím jazyce Java s pomocí frameworku GridGain 2.1.1 výpočetní grid specializovaný na hledání Mersennových prvočísel. V kapitole č. 11 byly prezentovány výsledky paralelních výpočtů této vlastní aplikace a komerční aplikace Mathematica 7. Výkonnostní nárůst odpovídá výkonům jednotlivých pracovních stanic.

Hledání Mersennových prvočísel je výpočetně náročná úloha. Pomocí paralelizace lze toto hledání výrazně urychlit. Metodu pro hledání prvočísel ze třídy `java.math.BigInteger` se podařilo doplnit o modulo malými prvočísly a snížit tak dále potřebnou dobu pro ověřování prvočísel.

Technologie Grid Computing je silným nástrojem pro akumulaci i sdílení zdrojů. Gridy je možné snadno a postupně upgradovat, mohou obsahovat zcela softwarově i hardwarově odlišné uzly a v neposlední řadě mohou být dynamicky sestavovány z geograficky různých oblastí.

13 Reference

[1] Noll, Landon Curt. 242643801-1 is prime [online]. [cit. 27.2.2010]. Dostupný z WWW: http://prime.isthe.com/chongo/tech/math/prime/m42643801/prime-c.html

[2] Mersenne Research, Inc. . PrimeNet 5.0 [online]. [cit. 27.2.2010]. Dostupný z WWW: http://www.mersenne.org/primenet/

[3] GridGain Systems, Inc.. GridGain – Cloud Development Platform [online]. [cit. 27.2.2010]. Dostupný z WWW: http://gridgain.com/

[4] Wolfram Research, Inc. Wolfram Lightweight Grid Manager: The Easy Way to Share Your Hardware [online]. [cit. 27.2.2010]. Dostupný z WWW: http://www.wolfram.com/products/gridmathematica/

[5] University of Tennessee at Martin. Mersenne Primes: History, Theorems and Lists [online]. [cit. 27.2.2010]. Dostupný z WWW: http://primes.utm.edu/mersenne/

[6] Wikipedia. Mersenne prime [online]. [cit. 27.2.2010]. Dostupný z WWW: http://en.wikipedia.org/wiki/Mersenne_prime

[7] Mersenne Research, Inc. . History - GIMPS [online]. [cit. 3.3.2010]. Dostupný z WWW: http://www.mersenne.org/various/history.php

[8] University of St. Andrews. Mersenne biography [online]. [cit. 3.3.2010]. Dostupný z WWW: http://www-history.mcs.st-and.ac.uk/Biographies/Mersenne.html

[9] Mersenne Research, Inc. . GIMPS Home [online]. [cit. 27.2.2010]. Dostupný z WWW: http://www.mersenne.org

[10] Zaglakas, Georgiann. Grid computing: Meeting the challenges of an On Demand world [online]. [cit. 27.2.2010]. Dostupný z WWW:

http://www.ibm.com/developerworks/rational/library/content/RationalEdge/sep04/zaglakas/index.html

[11] EGEE. Úvod do použití gridů [online]. [cit. 27.2.2010]. Dostupný z WWW: http://egee.cesnet.cz/sources/EGEE-II-Gridy-Kmunicek.pdf

[12] Economic Expert. CPU Scavenging [online]. [cit. 20.4.2010]. Dostupný z WWW: http://www.economicexpert.com/a/CPU:scavenging.html

[13] Multicians. Introduction and Overview of the Multics System [online]. [cit. 27.2.2010]. Dostupný z WWW: http://www.multicians.org/fjcc1.html

[14] Wikipedia. Multics [online]. [cit. 27.2.2010]. Dostupný z WWW: http://en.wikipedia.org/wiki/Multics

[15] GridCafé. A brief history of grid computing [online]. [cit. 27.2.2010]. Dostupný z WWW: http://www.gridcafe.org/version1/Gridhistory/ancestors.html

[16] Peking University, CNDS Lab. Globus: A Metacomputing Infrastructure Toolkit [online]. [cit. 27.2.2010]. Dostupný z WWW: http://net.pku.edu.cn/~cnds/readings/Globus.pdf

[17] University of Tennessee at Martin. Factorization of RSA-130 [online]. [cit. 27.2.2010]. Dostupný z WWW: http://primes.utm.edu/notes/rsa130.html

[18] Wikipedia. FAFNER [online]. [cit. 27.2.2010]. Dostupný z WWW: http://en.wikipedia.org/wiki/FAFNER

[19] The National Coordination Office for Networking Information Technology Research and Development. Information-Wide-Area-Year (I-

WAY) [online]. [cit. 27.2.2010]. Dostupný z WWW: http://www.nitrd.gov/pubs/bluebooks/1997/i-way.html

[20] Doe Science Grid. The Grid Resource Broker: Web access to computational grids [online]. [cit. 27.2.2010]. Dostupný z WWW: www.doesciencegrid.org/public/events/GPDW/slides/grb.ppt

[21] GridCafé. Grid-like projects [online]. [cit. 27.2.2010]. Dostupný z WWW: http://www.gridcafe.org/version1/Gridhistory/gridlike.html

[22] University of Virginia. Legion Overview [online]. [cit. 27.2.2010]. Dostupný z WWW: http://legion.virginia.edu/overview.html

[23] University of California, Santa Barbara. AVAKI Grid Software: Concepts and Architecture [online]. [cit. 27.2.2010]. Dostupný z WWW: http://pompone.cs.ucsb.edu/~wenye/majorexam/Architecture/avaki.pdf

[24] Gridengine.info. History of Grid Engine [online]. [cit. 27.2.2010]. Dostupný z WWW: http://gridengine.info/2006/06/22/history-of-grid-engine

[25] Oracle.com. Sun and Oracle [online]. [cit. 27.2.2010]. Dostupný z WWW: http://www.sun.com/third-party/global/oracle/

[26] University of Wisconsin, Madison. Condor Project [online]. [cit. 27.2.2010]. Dostupný z WWW: http://www.cs.wisc.edu/condor/

[27] Monash University. Nimrod Toolkit [online]. [cit. 27.2.2010]. Dostupný z WWW: http://www.messagelab.monash.edu.au/Nimrod

[28] Monash University. Collaborative Projects using Nimrod Tools [online]. [cit. 27.2.2010]. Dostupný z WWW: http://messagelab.monash.edu.au/Nimrod/Projects

[29] Axceleon. Products [online]. [cit. 27.2.2010]. Dostupný z WWW: http://www.axceleon.com/products.html

[30] Unicore. History [online]. [cit. 27.2.2010]. Dostupný z WWW: http://www.unicore.eu/unicore/history.php

[31] University of California, Berkeley. BOINC [online]. [cit. 27.2.2010]. Dostupný z WWW: http://boinc.berkeley.edu

[32] University of California, Berkeley. Choosing BOINC projects [online]. [cit. 27.2.2010]. Dostupný z WWW: http://boinc.berkeley.edu/projects.php

[33] Standford University. Folding@Home [online]. [cit. 27.2.2010]. Dostupný z WWW: http://folding.stanford.edu

[34] Mersenne Research, Inc. . The Mersenne Newsletter, issue #1 [online]. [cit. 3.3.2010]. Dostupný z WWW: http://www.mersenne.org/newsletters/news1.txt

[35] Mersenne Research, Inc. . PrimeNET 5.0 [online]. [cit. 3.3.2010]. Dostupný z WWW: http://www.mersenne.org/primenet/

[36] Mersenne Research, Inc. . Assignment Thresholds [online]. [cit. 3.3.2010]. Dostupný z WWW: http://www.mersenne.org/thresholds/]

[37] Wikipedia. Open Grid Forum [online]. [cit. 27.2.2010]. Dostupný z WWW: http://en.wikipedia.org/wiki/Open_Grid_Forum

[38] Open Grid Forum. About OGF [online]. [cit. 27.2.2010]. Dostupný z WWW: http://www.ogf.org/About/abt_overview.php

[39] Open Grid Forum. An Overview of the Open Grid Forum (v2) [online]. [cit. 27.2.2010]. Dostupný z WWW: http://www.ogf.org/About/OGF-Overview-2009.pdf

[40] Globus Alliance. The Physiology of the Grid [online]. [cit. 27.2.2010]. Dostupný z WWW: http://www.globus.org/alliance/publications/papers/ogsa.pdf

[41] Globus Alliance. The Open Grid Services Architecture, Version 1.5 [online]. [cit. 27.2.2010]. Dostupný z WWW: http://www.ogf.org/documents/GFD.80.pdf

[42] Sun.com. Java Management Extensions (JMX) Technology [online]. [cit. 27.2.2010]. Dostupný z WWW: http://java.sun.com/javase/technologies/core/mntr-mgmt/javamanagement/

[43] Svět sítí. Vznik a principy SNMP [online]. [cit. 27.2.2010]. Dostupný z WWW: http://www.svetsiti.cz/view.asp?rubrika=Tutorialy&clanekID=29

[44] Globus Alliance. About the Globus Alliance [online]. [cit. 27.2.2010]. Dostupný z WWW: http://www.globus.org/alliance/about.php

[45] Globus Alliance. Our Sponsors [online]. [cit. 27.2.2010]. Dostupný z WWW: http://www.globus.org/alliance/sponsors.php

[46] Globus Alliance. The Globus Team [online]. [cit. 27.2.2010]. Dostupný z WWW: http://www.globus.org/alliance/team/

[47] Globus Alliance. About the Globus Toolkit [online]. [cit. 27.2.2010]. Dostupný z WWW: http://www.globus.org/toolkit/about.html

[48] Globus Alliance. Globus Toolkit [online]. [cit. 27.2.2010]. Dostupný z WWW: http://www.globus.org/toolkit/

[49] Globus Alliance. GT 5.0.0 GSI C [online]. [cit. 27.2.2010]. Dostupný z WWW: http://www.globus.org/toolkit/docs/5.0/5.0.0/security/gsic/rn/

[50] Wikipedia. X.509 [online]. [cit. 27.2.2010]. Dostupný z WWW: http://cs.wikipedia.org/wiki/X.509

[51] Lupa.cz. Ukryto pod rouškou X.509 [online]. [cit. 27.2.2010]. Dostupný z WWW: http://www.lupa.cz/clanky/ukryto-pod-rouskou-x-509

[52] Globus Alliance. GT 5.0.0 Release Notes: GSI-OpenSSH [online]. [cit. 27.2.2010]. Dostupný z WWW: http://www.globus.org/toolkit/docs/5.0/5.0.0/security/openssh/rn/

[53] University of Illinois. GSI-Enabled OpenSSH [online]. [cit. 27.2.2010]. Dostupný z WWW: http://grid.ncsa.illinois.edu/ssh/

[54] OpenBSD. OpenSSH [online]. [cit. 27.2.2010]. Dostupný z WWW: http://www.openssh.org

[55] University of Illinois. MyProxy [online]. [cit. 27.2.2010]. Dostupný z WWW: http://grid.ncsa.illinois.edu/myproxy/

[56] Globus Alliance. GT 5.0.0 Release Notes: GridFTP [online]. [cit. 27.2.2010]. Dostupný z WWW: http://www.globus.org/toolkit/docs/5.0/5.0.0/data/gridftp/rn/

[57] Globus Alliance. GT 5.0.0 Release Notes: RLS [online]. [cit. 27.2.2010]. Dostupný z WWW: http://www.globus.org/toolkit/docs/5.0/5.0.0/data/rls/rn/

[58] Globus Alliance. GT 5.0.0 Release Notes: GRAM5 [online]. [cit. 27.2.2010]. Dostupný z WWW: http://www.globus.org/toolkit/docs/5.0/5.0.0/execution/gram5/rn/

[59] Globus Alliance. GT 5.0.0 Release Notes: C Common Libraries [online]. [cit. 27.2.2010]. Dostupný z WWW: http://www.globus.org/toolkit/docs/5.0/5.0.0/common/ccommonlib/rn/

[60] Globus Alliance. Usage Statistics Collection by the Globus Alliance [online]. [cit. 27.2.2010]. Dostupný z WWW: http://www.globus.org/toolkit/docs/5.0/5.0.0/Usage_Stats.html

[61] Globus Alliance. GT 5.0.0 Release Notes: XIO [online]. [cit. 27.2.2010]. Dostupný z WWW: http://www.globus.org/toolkit/docs/5.0/5.0.0/common/xio/rn/

[62] GridGain Systems, Inc.. About company [online]. [cit. 27.2.2010]. Dostupný z WWW: http://www.gridgain.com/company.html

[63] GridGain Systems, Inc.. Services [online]. [cit. 27.2.2010]. Dostupný z WWW: http://www.gridgain.com/services.html

[64] GridGain Systems, Inc.. Product [online]. [cit. 27.2.2010]. Dostupný z WWW: http://www.gridgain.com/product.html

[65] GridGain Systems, Inc.. GridGain Knowledge Base [online]. [cit. 27.2.2010]. Dostupný z WWW: http://www.gridgainsystems.com/wiki/display/GG15UG/Table+Of+Contents

[66] GridGain Systems, Inc.. SPI - Service Provider Interfaces [online]. [cit. 27.2.2010]. Dostupný z WWW: http://www.gridgainsystems.com/wiki/display/GG15UG/SPI+-+Service+Provider+Interfaces

[67] Samuraj-cz.com. TCP/IP - skupinové vysílání IP Multicast a Cisco [online]. [cit. 27.2.2010]. Dostupný z WWW: http://www.samuraj-cz.com/clanek/tcpip-skupinove-vysilani-ip-multicast-a-cisco

[68] GridGain Systems, Inc.. IP-Multicast [online]. [cit. 27.2.2010]. Dostupný z WWW: http://www.gridgainsystems.com/wiki/display/GG15UG/Example+Configuration+And+Setup#ExampleConfigurationAndSetup-IPMulticast

[69] GridGain Systems, Inc.. Product features [online]. [cit. 27.2.2010]. Dostupný z WWW: http://www.gridgain.com/product_features.html

[70] Wikipedia. MapReduce [online]. [cit. 27.2.2010]. Dostupný z WWW: http://en.wikipedia.org/wiki/MapReduce

[71] Majda.cz. MapReduce: Výpočetní model Googlu [online]. [cit. 27.2.2010]. Dostupný z WWW: http://majda.cz/zapisnik/165

[72] Google Labs. MapReduce: Simplified Data Processing on Large Clusters [online]. [cit. 27.2.2010]. Dostupný z WWW: http://labs.google.com/papers/mapreduce.html

[73] GridGain Systems, Inc.. MapReduce Overview [online]. [cit. 27.2.2010]. Dostupný z WWW: http://www.gridgainsystems.com/wiki/display/GG15UG/MapReduce+Overview

[74] Wolfram Research, Inc. Wolfram Mathematica 7 [online]. [cit. 27.2.2010]. Dostupný z WWW: http://www.wolfram.com/products/mathematica/index.html

[75] Wolfram Research, Inc. Built-in Parallel Computing [online]. [cit. 27.2.2010]. Dostupný z WWW: http://www.wolfram.com/products/mathematica/newin7/content/BuiltInParallelComputing

[76] Wolfram Research, Inc. Wolfram gridMathematica: Multiplying the Power of Mathematica Over the Grid [online]. [cit. 27.2.2010]. Dostupný z WWW: http://www.wolfram.com/products/gridmathematica/

[77] Wolfram Research, Inc. Parallelize [online]. [cit. 27.2.2010]. Dostupný z WWW: http://reference.wolfram.com/mathematica/ref/Parallelize.html

[78] Wolfram Research, Inc. ParallelTry [online]. [cit. 27.2.2010]. Dostupný z WWW: http://reference.wolfram.com/mathematica/ref/ParallelTry.html

[79] Wolfram Research, Inc. ParallelEvaluate [online]. [cit. 27.2.2010]. Dostupný z WWW: http://reference.wolfram.com/mathematica/ref/ParallelEvaluate.html

[80] Wolfram Research, Inc. gridMathematica Local [online]. [cit. 27.2.2010]. Dostupný z WWW: http://www.wolfram.com/products/gridmathematicalocal/

[81] Wolfram Research, Inc. gridMathematica Server [online]. [cit. 27.2.2010]. Dostupný z WWW: http://www.wolfram.com/products/gridmathematicaserver/

[82] Wolfram Research, Inc. Wolfram Lightweight Grid Manager: The Easy Way to Share Your Hardware [online]. [cit. 27.2.2010]. Dostupný z WWW: http://www.wolfram.com/products/lightweightgrid/

[83] Sun.com. Class BigInteger [online]. [cit. 27.2.2010]. Dostupný z WWW: http://java.sun.com/j2se/1.4.2/docs/api/java/math/BigInteger.html

[84] KickJava.com. BigInteger [online]. [cit. 27.2.2010]. Dostupný z WWW: http://kickjava.com/src/java/math/BigInteger.java.htm

[85] Caucho.com. JVM Tuning [online]. [cit. 27.2.2010]. Dostupný z WWW: http://www.caucho.com/resin-3.0/performance/jvm-tuning.xtp

[86] Wolfram Research, Inc. Some Notes On Internal Implementation [online]. [cit. 27.2.2010]. Dostupný z WWW: http://reference.wolfram.com/mathematica/note/SomeNotesOnInternalImplementation.html#6849

Rejstřík

Příloha č. 1 – Hardwarové specifikace stanic

Hardware	č. 1	č. 2	č. 3
CPU	Intel Core 2 Duo	Intel Pentium M	Intel Celeron Dual-Core
Kódované označení	E5200	Pentium M 1,86 GHz	E1200
Frekvence jádra	3,5 GHz	1,86 GHz	1,6 GHz
Počet jader	2	1	1
Velikost L2 cache	2 MB	1 MB	512 kB
Typ RAM	DDR2	DDR2	DDR2
Velikost RAM	4096 MB	2048 MB	2048 MB
Frekvence RAM	800 MHz	533 MHz	800 MHz
Chipset	Intel P31	Intel i915	Intel P35
Pevný disk	Western Digital	Hitachi	Seagate
Kapacita HDD	640 GB	80 GB	120 GB
Rychlost HDD	7200 ot./m	5400 ot./m	7200 ot./m
Zdroj	Forton 350W	Integrovaná baterie	EuroCase 350W