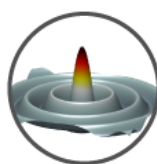
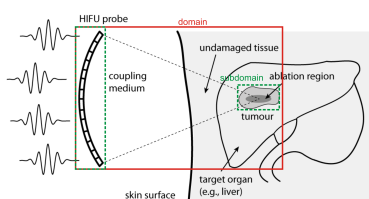


Optimalizace distribuovaného I/O subsystému projektu k-Wave

Ondřej Vysocký*



k-Wave
A MATLAB toolbox for the time-domain
simulation of acoustic wave fields

Abstrakt

Práce se zabývá řešením efektivního paralelního zápisu dat pro nástroj k-Wave, zabývající se simulací šíření ultrazvuku. Tento nástroj je superpočítačovou aplikací, proto je spouštěn na souborovém systému Lustre a vyžaduje paralelní zpracování pomocí MPI a zápis ve formátu vhodném pro velké množství dat (HDF5). V rámci této práce byl navržen způsob zefektivnění zápisu dat dle potřeb k-Wave, pomocí kumulace dat a přerozdělování, který je částečně implementován a otestován.

Klíčová slova: optimalizace — I/O — HDF5 — MPI — k-Wave — HPC

Příložené materiály: N/A

*xvysoc01@stud.fit.vutbr.cz, Fakulta informačních studií, Vysoké učení technické v Brně

1. Introduction

Rychlost vývoje pevných disků výrazně zaostává za vývojem procesorů [1], tím pádem se I/O stává úzkou částí a způsobuje výrazné zpomalení celé aplikace. Proto je nutné systém čtení a zápisu vyladit pro maximální zmenšení rozdílu mezi I/O a výpočetní částí. S tímto problémem se setkává mnoho paralelních aplikací, které generují obrovské množství dat a je pro ně proto nutné tento problém řešit.

Nástroj k-Wave [2] slouží k simulaci šíření ultrazvuku, čehož se využívá především v medicíně nebo při zkoumání materiálů. Z důvodu vysoké výpočetní a paměťové složitosti těchto simulací musí výpočet probíhat na superpočítačích. Prostředí počítačových clusterů je značně specifické, především z důvodů sdílení prostředků s ostatními uživateli, dávkového zpracovávání úloh nebo pro tento případ zásadní zápis dat na RAID disková pole se souborovým systémem lustre. V této práci se zabývám MPI verzí nástroje k-Wave, který zapisuje výsledky simulace do HDF5 souboru.

Uvedené nástroje a prostředí byly vyvinuty nebo upraveny pro clustery a paralelní aplikace, kterým umožňují rychlý paralelní zápis velkého množství dat. Problémem je však rychlost zápisu malého množství dat, jelikož přístup k pevným diskům je v tomto prostředí spojen s velkou režii, která zcela degraduje rychlost malých paralelních zápisů.

Rychlost zápisu je pro k-Wave zásadní, jelikož v průběhu simulace mohou být vygenerovány stovky gigabajtů dat. Rozložení zapisovaných dat mezi jednotlivé procesy však může způsobit nízkou rychlost zápisu, která zdaleka nedosahuje maximální možné rychlosti, a spolu s tím roste doba celé simulace a tedy i její cena.

Mým řešením nízké rychlosti zápisu malého množství dat je kumulace dat před samotným zápisem. V případě aplikací zapisujících data opakovaně v průběhu provádění je přirozeným způsobem zvětšení množství dat na proces kumulace dat přes jednotlivé iterace a zápis až po dosažení dostatečného množství dat. Pro jed-

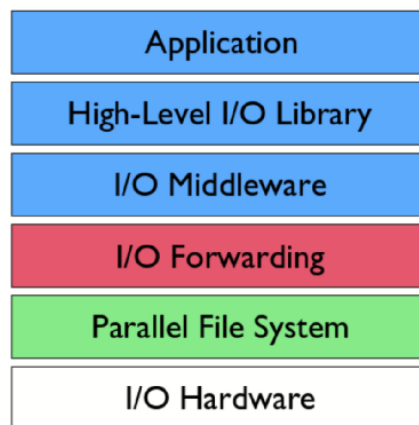
norázový zápis (který však z hlediska celého běhu aplikace nebývá kritický), nebo nerovnoměrné rozložení dat mezi jednotlivé procesy je možné provést přeuspořádání dat s cílem zvětšit množství dat na zapisující proces.

Navržený způsob zápisu je vyvíjen a optimalizován pro aplikaci k-Wave, ale výsledky této práce jsou použitelné pro libovolné paralelní aplikace s distribuovanou pamětí a častým zápisem.

2. I/O subsystém na superpočítačích

Vstupně/výstupní systém se skládá z několika vrstev, přičemž každá z nich zásadně ovlivňuje výkonnost paralelního I/O.

- **I/O hardware**
Vstupně/výstupní hardware je tvořen poli pevných disků, která jsou propojena s výpočetními uzly skrze storage servery. Některé superpočítače poskytují také datová úložiště tvořená magnetickými páskami, určená pro zálohu uživatelských dat.
- **Paralelní souborový systém**
Paralelní souborový systém spravuje datový hardware. Typickým příkladem takového systému je Lustre [3], který zcela odděluje metadata od zbytku dat. Umožňuje nám především rozložit data přes disky diskového pole (pomocí nastavení stripování), čímž je zvýšena režie, ale především průchodnost paralelního zápisu a čtení.
- **I/O forwarding**
I/O forwarding směřuje vstupně/výstupní požadavky na dedikované I/O uzly, které je přesměrovávají na paralelní souborový systém. Požadavky jsou zde přeskládány a následně agregovány s cílem snížit latenci systému [4]. Tato vrstva je však přítomná jen na některých z největších superpočítačových systémech a je typicky úzce spojená hardwarem [4][5].
- **I/O middleware**
V případě využívání Message Passing Interface (MPI) a paralelního I/O, je na této úrovni MPI-I/O standart. Stejně jako MPI standart, také MPI-I/O má mnoho implementací, nejrozšířenější z nich je ROMIO [6].
- **Vysokoúrovňová I/O knihovna**
Pro paralelní zápis je možné využívat některou z implementací MPI-I/O, avšak nad touto vrstvou jsou postaveny ještě vysoko-úrovňové I/O knihovny, které kromě zápisu pomocí MPI-I/O přináší navíc také správu metadat, optimalizaci pro zápis do překrývajících se datových oblastí nebo



Obrázek 1. Vrstvy I/O subsystému na superpočítačích [5].

současný zápis a čtení různými procesy a mnoho dalších. Příkladem takovýchto knihoven jsou netCDF [7] nebo HDF5 [8].

Je nezbytné mít na mysli, že každá uvedená vrstva má mnoho nastavení a uživatel by měl, dříve než začne optimalizovat svůj kód, najít vhodné nastavení vrstev předchozích.

2.1 HDF5

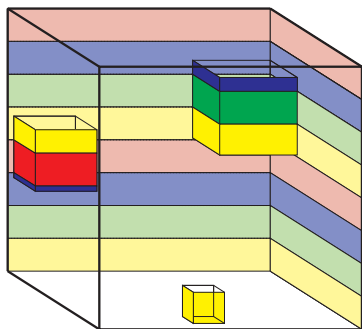
The Hierarchical Data Format (HDF) je samo-popisný souborový formát pro usnadnění práce a uložení výzkumných dat nezávisle na operačním systému a architektuře počítače.

V projektu k-Wave je využívána knihovna HDF5 pro svou komplexnost. Využíváme toho, že HDF5 soubor je kontejnerem pro HDF5 objekty, což znamená, že v jednom souboru může být uloženo několik samostatných datových celků (datasetů) a především že poskytuje jednoduchý efektivní paralelní zápis do souboru pomocí hyperslabů, které vymezují prostor v datasetu pro data každého zapisujícího procesu.

3. Projekt k-Wave

V k-Wave se pracuje s tří-dimenzionálním prostorem, v kterém se vypočítává průběh šíření ultrazvuku. Tato doména je rozdělena mezi jednotlivé procesy, a to rozdělením jedné z dimenzí tak, jak je tomu znázorněno na obrázku 2. Jelikož není nutné ukládat veškerá data, definujeme takzvanou senzorovou masku, která nám udává pozice v doméně, které budeme ukládat. Tímto způsobem došlo k nerovnoměrnému rozložení dat na zápis mezi jednotlivé procesy, kdy některé procesy nezapisují vůbec, jiné procesy naopak mohou spravovat (a zapisovat) několikanásobně více dat než jiné.

Jelikož kolektivní paralelní zápis je blokující operací, není možné pokračovat ve výpočtu, dokud neza-



Obrázek 2. Simulační doména rozdělená mezi procesy s 3 subdoménami.

píše data nejpomalejší z procesů. A protože pro analýzu výsledků simulace je nutné uložit data z velkého množství iterací, stává se zápis dat významnou časově náročnou operací, která vyžaduje optimalizaci.

V mnoha simulačních nástrojích je I/O velmi pomalé (vstupně/výstupní operace nedosahuje maximální možné rychlosti), ač všechny procesy zapisují (respektive čtou) velké množství dat. Tento problém vzniká u aplikací s vysokou dimenzionalitou dat, kde je paměťová oblast rozdělena podle nejvyšší dimenze, tudíž na místo jediného požadavku na velké množství dat vzniká mnoho menších požadavků.

4. Paralelní zápis v k-Wave

V paralelních aplikacích existuje trojice vzorů pro zápis. Prvním z nich je zápis jediným z procesů. Dále je zde možnost, kdy každý proces zapisuje do samostatného souboru, a třetím způsobem je paralelní zápis do jediného sdíleného souboru. Každá z variant má své výhody i nevýhody, my se však budeme zabývat pouze poslední variantou, která nám umožňuje snadný post-processing a správu souborů (v porovnání s variantou generující N souborů v každém běhu aplikace), ale vyžaduje synchronizaci procesů při nastavení přístupu k souboru a vykazuje nízkou propustnost při zápisu malých datových celků.

V k-Wave simulace probíhá v třídímním prostoru, ve kterém se vypočítává průběh šíření ultrazvuku. Tato doména je distribuována mezi jednotlivé procesy rozdělením jedné z dimenzí. Jelikož není nutné ukládat data v celé doméně, definujeme senzorovou masku, což je seznam indexů udávající pozice obalující oblasti zájmu.

V každém kroku simulace se pak ukládají nové hodnoty na pozicích označených senzorovou maskou, což z hlediska zápisu znamená, že rozložení zapisovaných dat mezi procesy není rovnoměrné a některé

procesy nezapisují vůbec. Problémem jsou především některé tvary subdomén, přiděluující na proces malé množství dat na zápis. Tyto malé zápisy vykazují vysokou režii, která způsobuje velmi pomalý zápis.

Jelikož kolektivní zápis je blokující operací, není možné pokračovat ve výpočtu, dokud nezapiše data nejpomalejší z procesů. A protože pro analýzu výsledků simulace je nutné uložit data z velkého množství iterací, stává se zápis dat významnou časově náročnou operací, která vyžaduje optimalizaci.

5. Optimalizace

Nový optimalizovaný I/O subsystém projektu k-Wave je založen na přeuspořádání dat mezi procesy a na kumulaci přes jednotlivé iterace, což jsou způsoby zvětšení množství dat na zápis. V případě nerovnoměrného rozložení dat navíc dochází k přeuspořádání dat s cílem minimalizovat tyto rozdíly.

Při nastavování parametrů nejprve dojde k rozdělení procesů do skupin. Skupiny tvoří procesy se stejným množstvím dat na zápis rozšířené o okrajové procesy, které mohou mít dat méně. Dále dojde k vyhledání skupiny, v níž připadá na jednotlivé procesy největší množství dat. Ve snaze minimalizovat rozdíly v množství dat na zapisující procesy je ve zbylých skupinách určen počet procesů z dané skupiny, které by měly sbírat data od okolních procesů ze skupiny pomocí funkce `MPI_Gatherv()` a získat tak podobné množství jako procesy s nejvíce dat.

Pokud ve všech skupinách stále připadá na zapisující procesy malé množství dat (do stovek kilobajtů) a je zároveň možné ve všech skupinách zredukovat počet zapisujících procesů na polovinu až osminu, pak dochází k této redukci zapisujících procesů a tady ke zvětšení množství dat na tyto procesy.

Dle počtu zapisujících procesů ve skupinách jsou nastaveny parametry funkce `MPI_Gatherv()` a je vytvořen nový MPI komunikátor pojící procesy z dané skupiny, které posílají svá data jedinému zapisujícímu procesu. Další komunikátor je vytvořen také pro zapisující procesy.

Přeposílání dat mezi procesy během simulace probíhá na závěr každé iterace, ale zápis nikoli, jelikož je prováděna také kumulace dat přes několik iterací. Základem pro určení ideálního počtu kumulujících iterací je testovací aplikace, která vyzkouší na daném výpočetním clusteru zápis různého množství dat na proces. Původním předpokladem bylo určit, při jakém množství dat dosáhneme s daným počtem procesů maximální rychlosti, a poté kumulovat data přes jednotlivé iterace, dokud nedosáhneme odpovídajícího množství. To se však ukázalo jako mylné, jelikož zápis je zásadně

ovlivněn dimenzionalitou dat (kumulací v čase rozšiřujeme druhou dimenzi). Proto testovací aplikace musela být modifikována tak, že pro jednotlivé množství dat provádí kumulaci přes jednotlivé iterace, stejně jako je tomu v optimalizované aplikaci. Počet kumulujících iterací se zvětšuje, dokud vede k rychlejšímu zápisu.

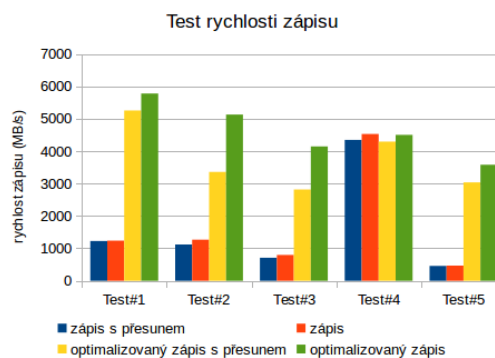
Výsledná rychlost zápisu a ideální počet kumulujících iterací je zaznamenán do xml souboru, který je vstupem k-Wave¹. Dle vstupní analýzy každá skupina procesů určí ideální počet kumulujících iterací, poté dojde k nalezení minimálního počtu mezi skupinami, aby tento parametr byl synchronizován a byl umožněn kolektivní paralelní zápis.

Tímto způsobem byl optimalizován zápis výstupních dat označených pomocí sensorové masky. Využití sensorové masky však způsobuje problémy při následném zpracování výstupů, kdy se data načítají po jednotlivých indexech daných maskou. Pro zlepšení rychlosti zpracování výstupních souborů je proto nutné nahradit sensorovou masku takzvanými kuboidy, subdoménami ve tvaru kváдру obalujícími oblast zájmu. Toto řešení označení dat pro zápis vede obecně k většímu množství ukládaných dat, protože nejsme schopni definovat libovolný tvar subdomény, ale umožní načítání dat po blocích, což je výrazně rychlejší.

Zápis pomocí kuboidů přináší do problematiky několik změn. Každý kuboid je v rámci výstupního souboru zapsán do samostatného datasetu a pro umožnění paralelního zápisu do několika nezávislých kuboidů je nutné vytvořit pro každý kuboid samostatný MPI komunikátor.

Z hlediska optimalizace zápisu v tomto případě nemusíme vyhledávat skupiny procesů, jelikož ty jsou nyní dány jednotlivými kuboidy. Zároveň je možné využít toho, že všechny kuboidy mají vlastní komunikátor. Proto není nutné synchronizovat zápis všech subdomén na shodné iterace, ale je minimálně vhodné nastavit počet kumulujících iterací tak, aby subdomény s delší periodou kumulace byly zapisovány v násobcích period subdomén s častějším zápisem, aby nedocházelo k zbytečnému vzájemnému blokování.

Pro zápis optimalizovaný pomocí kumulace dat pře jednotlivé iterace je nutné spolu s bufferovanými daty rozšířit také HDF5 chunk, definující nejmenší zapisovatelnou jednotku do HDF5 souborů. Malý chunk tedy dělí výsledný zápis do řady menších a degraduje tak veškerý přínos získaný kumulací dat.



Obrázek 3. Graf srovnávající rychlost zápisu v nativní podobě a po optimalizaci.

6. Testy

Následující testy byly provedeny na superpočítači Salomon v Ostravě a zkoumají možné případy subdomén uvnitř domény o velikosti 512 x 512 x 512, která byla zpracovávána pomocí 256 procesů. Každá ze simulací probíhala po dobu 1000 iterací, ve kterých byla generována data pro zápis. Veškeré testy byly založeny výhradně na označení subdomén pomocí sensorové masky, jelikož implementace kuboidů nebyla dosud dokončena. Volba podoby subdomén vychází z běžných požadavků aplikace k-Wave, přičemž se jedná především o případy vykazující nízkou rychlost zápisu.

V uvedených případech byl změřen čas nejpomalejšího procesu provádějícího zápis a přepočten na rychlost. Jelikož je v této implementaci zápis spojen s přesunem dat z pole vypočtených hodnot do pole pro zápis, jsou změřeny také časy včetně času na přesun a právě tyto časy jsou porovnány při výpočtu zrychlení. Rychlost zápisu v jednotlivých testech je vyobrazena na grafu 3.

V testu #1 jsou definovány oblasti zájmu jako tři rovnoběžné desky (512 x 512 x 1), které přidělují data na zápis všem procesům, na které tak připadá 12 kB. Došlo proto k redukci zapisujících procesů na 32 a následně se kumulovala data po dobu 15 iterací, což vedlo k 4,3násobnému zrychlení zápisu z rychlosti 1227 MB/s na 5254 MB/s.

Jestliže došlo k otočení subdomén tak, že data byla rozdělena pouze mezi tři procesy (Test#2), na které tak připadl 1 MB dat, získáváme zcela rozdílnou situaci. Takovéto rozložení dat není nijak upravováno a dochází pouze ke kumulaci přes 9 iterací. Tato optimalizace vedla ke zvýšení rychlosti z 1118 MB/s na pouhých 3359 MB/s (3násobné zrychlení), avšak samotný optimalizovaný zápis proběhl rychlostí 5129 MB/s.

Další test obsahoval opět tři desky, avšak tentokrát každá z nich byla umístěna v polovině jedné z dimenzí domény (Test#3). Jediný z procesů tak zapisuje 1 MB, zatímco zbylé procesy pouze 8 kB. Jedná se

¹ Při spuštění simulace bez I/O analýzy daného počítače není zápis nijak optimalizován.

o velmi výrazné nerovnoměrné rozdělení dat, proto optimalizací dochází k redukci počtu zapisujících procesů na tři, které zapisují každou devátou iteraci. Tímto způsobem bylo z původní rychlosti 708 MB/s dosaženo 4násobného zrychlení a rychlosti 2816 MB/s (samotný zápis pak 4146 MB/s).

Test#4 je založen na subdoméně tvořené kostkou 256 x 256 x 256 přidělující polovině procesů 512 kB. Tento případ vykazuje vysokou rychlost zápisu již při nativním zápisu (4348 MB/s) a optimalizací se tak nepodařilo tento případ zrychlit (4295 MB/s), ale důležité je, že nedošlo ani k negativnímu ovlivnění rychlosti.

Poslední Test#5 je tvořen subdoménami o rozměrech 64 x 64 x 64, 33 x 65 x 33 a 10 x 40 x 41 rozmístěnými různě po doméně, tudíž tvoří tři skupiny procesů s různým množstvím dat (32, 16 a 3 kB). Vyrovnání rozdílů v množství dat a kumulací přes 15 iterací vedlo k rychlosti 3036 MB/s, což je v porovnání s předchozí rychlostí 457 MB/s zrychlení 6,6 násobné.

7. Závěr

Výsledkem této práce je způsob optimalizace paralelního zápisu pro superpočítačové aplikace, které opakovaně zapisují malé množství dat na proces, tudíž nedosahují maximální možné rychlosti zápisu z důvodu vysoké míry režie, která je s tímto zápisem spojena. Poměr režie vůči celému zápisu je snížen pomocí zvětšení množství zapisovaných dat jejich kumulací přes jednotlivé iterace běhu aplikace. Počet iterací, přes které se data kumulují, je určen na základě výstupu vytvořeného programu provádějícího I/O analýzu cílového počítače.

Na uvedených testech byla představena různá potenciální rozložení dat, způsobující pomalý zápis. Kromě případu, který vykazuje vysokou rychlost již při nativním způsobu zápisu, došlo ve všech zbylých případech k razantnímu zrychlení zápisu. Ve všech testech bylo dosaženo minimálně rychlosti 2,75 GB/s, přičemž zrychlení bylo vždy 3 až 6,6násobné.

Dosud neimplementovanou optimalizací je zavedení označování subdomén pomocí kuboidů, které budou optimalizovány podobně jako při využití sensorové masky, ale navíc by měly přinést především rychlejší post-processing.

Vytvořený způsob optimalizace by mohl být v budoucnu doplněn neblokujícím kolektivním zápisem, avšak tento typ zápisu je novinkou v MPI a v HDF5 ještě není implementován. Alternativně by mohl být tento způsob zápisu doplněn o delegované I/O uzly (případně procesy), což je vhodné řešení pro architektury počítačů s počtem jader na procesor, který nedělí velikost simulační domény.

Poděkování

Tato práce byla podporována Ministerstvem školství, mládeže a tělovýchovy z projektu „IT4Innovations National Supercomputing Center – LM2015070“ pro podporu velkých výzkumných infrastruktur, experimentální vývoj a inovativní projekty.

Literatura

- [1] Steve Fingerhut. Does storage break moore's law? [online]. [cit. 2016-04-07].
- [2] Treeby, B., Cox, B. k-wave a matlab toolbox for the time-domain simulation of acoustic wave fields. [online]. [cit. 2016-04-07].
- [3] EOFS OpenSFS. Lustre. [online], 2015. [cit. 2016-04-07].
- [4] Ohta, K., Kimpe, D., Cope, J., Iskra, K., Ross, R., Ishikawa, Y. Optimization techniques at the i/o forwarding layer. 2010.
- [5] Latham, R., Ross, R., Welch, B., Antypas, K. Parallel i/o in practice. [online].
- [6] Romio. [online]. [cit. 2016-04-07].
- [7] Unidata. netcdf. [online]. [cit. 2016-04-07].
- [8] The HDF Group. The hdf5 home page. [online], 2015-11-12. [cit. 2016-04-07].