Paralelní algoritmus pro MHD počítačové modelování ve sluneční fyzice

M. Bárta, Astronomický ústav AV ČR, Ondřejov, barta @asu.cas.cz

Abstrakt

Sluneční fyzika zažívá v posledních letech strmý nárůst kvality i objemu pozorovacích dat, jejichž interpretace je často možná pouze v rámci poměrně komplikovaných modelů. Kvůli nelinearitě rovnic modelujících v různých přiblíženích zkoumané problémy je analytické řešení obvykle nemožné a tak jediným možným přístupem zůstávají numerické simulace. V počítačovém modelování musí být jinak spojité parametry modelovaného prostředí i čas nahrazeny jejich diskrétní reprezentací, kdy veličiny jsou přesně definovány pouze v určitých časo-prostorových bodech, jejichž počet by měl kvůli věrohodnosti simulací být co největší. Tím samozřejmě narážíme na limity dané použitou výpočetní technikou, které se nejmarkantněji projeví především v modelování úloh s nízkou symetrií, kde je nutný obecný třírozměrný (3D) popis. Řešením tohoto problému je paralelizace algoritmu pro numerickou integraci příslušných (parciálních) diferenciálních rovnic a jeho spouštění na výpočetních systémech s mnoha procesory – superpočítačích. Kromě klasických superpočítačů využívajících paměť sdílenou jednotlivými procesory (např. Cray) se od 90. let začínají objevovat i systémy s distribuovanou pamětí – tzv. počítačové clustery, které jsou finančně dostupné i pro střední a malé vědecké instituce. V příspěvku je po obecnější úvodní pasáži týkající superpočítání (High-Performance Computing, HPC) podán popis algoritmu pro numerickou integraci soustavy magnetohydrodynamických (MHD) rovnic ve 3D, princip jeho paralelizace metodou rozložení problému do podoblastí (domain decomposition), odhady efektivity provedené paralelizace, různá úskalí týkající se především otázek průběžného výstupu výsledků a první výsledky 3D modelu rekonexe magnetického pole spočítaného na loni vybudovaném počítačovém clusteru v Ondřejově.

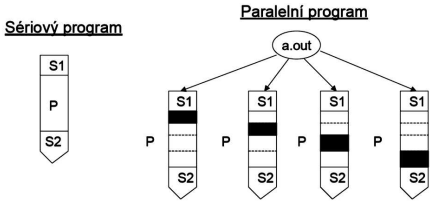
1. ÚVOD

Fyzikální modelování procesů sluneční aktivity metodou numerických simulací s sebou přináší vysoké nároky na výpočetní zdroje. Především globální simulace poměrně velkorozměrových jevů, jako jsou sluneční erupce a výrony koronální hmoty — mají-li být simulovány alespoň poněkud věrně — vyžadují velký počet bodů diskretizační sítě, tzv. *gridů*, ve kterých jsou vzorkovány hodnoty stavových veličin. S počtem gridů ovšem rostou i nároky na výpočetní čas (CPU time) a také na obsazení paměti. Vzhledem k tomu, že výpočetní možnosti jednotlivého procesoru (dnes spíše procesorového jádra) budou vždy limitovány, řešením vysokých výpočetních nároků je rozdělení úlohy na více výpočetních zdrojů současně, tzv. paralelizace. Principem paralelizace je, že nezávislé operace, prováděné obvykle nad různými daty, mohou být vykonány s použitím více CPU současně. Tuto základní ideu schematicky ukazuje Obr. 1. Po vykonání seriové části kódu $S1$ (obvykle inicializace dat) jsou na čtyřech procesorech současně prováděny instrukce, které mohou být paralelizovány (obvykle hlavní cyklus numerického kódu) následované seriovou částí $S2$ (korektní ukončení programu, „úklid" datových struktur). Ze schematu je patrné, že vzhledem k nutné přítomnosti seriových částí nelze program urychlovat do nekonečna zvyšováním počtu CPU, neboť čas strávený v seriových operacích je na počtu procesorů nezávislý. Horní odhad efektivity paralelizace tak poskytuje Amdahlovo pravidlo (Hlavička, 1994)

$$t_n = \frac{t_1 - s}{n} + s \qquad (1)$$

kde t_1 a t_n je trvání výpočtu na jednom, resp. n procesorech a s je celkové trvání seriových (principiálně neparalelizovatených) částí.

Kromě toho, i paralelně vykonávané instrukce musí být ve většině programů čas od času vzájemně koordinovány — obvykle v určité fázi výpočtu vyžadují data, nad kterými operuje konkurenční proces běžící na druhém CPU. Mechanismy této koordinace jsou ale hardwarově závislé a je proto třeba se krátce zmínit o dvou koncepcích paralelizace odpovídajících dvěma hlavním typům architektur superpočítačů.

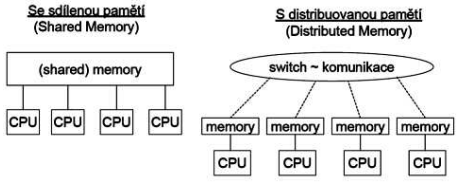


Obr. 1. ***Schema ukazující rozdíl mezi seriově a paralelně spuštěným kódem. V paralelním programu se operace prováděné nezávisle (obvykle nad jinými daty) mohou vykonat současně, což přináší značné urychlení výpočtu.***

2. ARCHITEKTURY SUPERPOČÍTAČŮ

Komunikace mezi jednotlivými paralelními procesy jednoho výpočtu se v praxi technicky řeší dvěma způsoby (Obr. 2). Tradiční řešení je založeno na architektuře se sdílenou pamětí (*shared memory systems*, též *Symmetric Multi-Processing systems, SMP*) — k výměně dat dochází prostřednictvím paměti, kterou mají k dispozici všechny procesory, na nichž běží paralelní zpracování výpočtu. O koordinaci přístupu k paměti se stará operační systém (semafory, na abstraktnější úrovni vlákna – threads). Takovou architekturu mají poměrně nákladné systémy např. typu *Cray*, i když dnes s rozvojem mnohajádrových (*multi-core*) procesorů ceny tohoto typu počítačů klesají — např. čtyř-procesorový stroj osazený dual-core procesory AMD Opteron (obsahující tedy osm výkonných procesorových jader) lze pořídit i pod deset tisíc euro. Výhodou těchto systémů je poměrně snadná paralelizace již dříve vytvořených seriových programů. K té je třeba – ke stávajícímu překladači použitého programovacího jazyka – nainstalovat softwarový balík OMP (*Open Multi-Processing*). Vlastní paralelizace je pak u typických numerických úloh realizována drobným zásahem do zdrojového kódu – v podstatě pouhým označením začátku a konce úseku kódu, jenž může být zpracován paralelně.

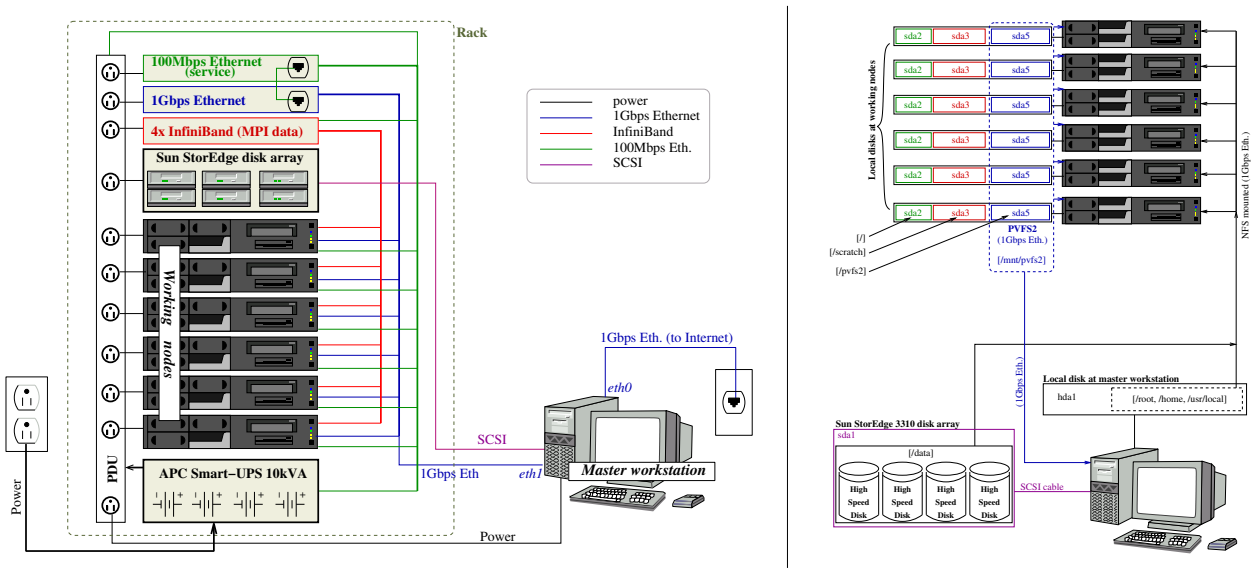
V devadesátých letech se s rozvojem rychlého síťového propojení mezi počítači začala objevovat al-



Obr. 2. ***Dva základní typy architektur superpočítačů. Vlevo systém se sdílenou pamětí (SMP), vpravo distribuovaný systém (cluster).***

Obr. 3. ***Ondrejov Cluster for Astrophysical Simulations — OCAS.***

ternativa k tradiční SMP architektuře superpočítačů, tzv. distribuované systémy neboli *počítačové clustery*. Jedná se o sadu nezávislých počítačů, každý z nichž ovládaný vlastním operačním systémem, které jsou schopny koordinovaného výpočtu díky vzájemnému síťovému propojení. Hlavní výhodou clusterů oproti stejně výkonným SMP systémům je nízká cena vyplývající z použití standardních komponent (PC) vyráběných ve velkých sériích. Komunikace mezi jednotlivými procesy je pak zajištěna prostřednictvím vzájemného posílání zpráv – *messages* obsahujících požadovaná data v elektronických obálkách opatřených služebními údaji. Posílání zpráv stejně jako koordinovaný start a řízení výpočtu na všech zúčastněných strojích – tzv. uzlech clusteru je zajištěno v rámci prostředí MPI – *Message Passing Interface* (http://www.mpi.org). Toto řešení přináší zvýšené nároky na programátora, neboť výměna dat musí být specifikována přímo ve zdrojovém kódu pomocí standardních funkcí protokolu MPI (typicky `MPI_send()` a `MPI_recv()`). Náročnější par-



Obr. 4. ***Schema zapojení sítí (vlevo) a přiřazení logických disků fyzickým zařízením (vpravo) u ondřejovského clusteru OCAS.***

alelizace může být vnímána jako určitá nevýhoda oproti jednoduššímu konceptu OMP pracujícího ale na druhé straně na mnohem dražších systémech SMP.

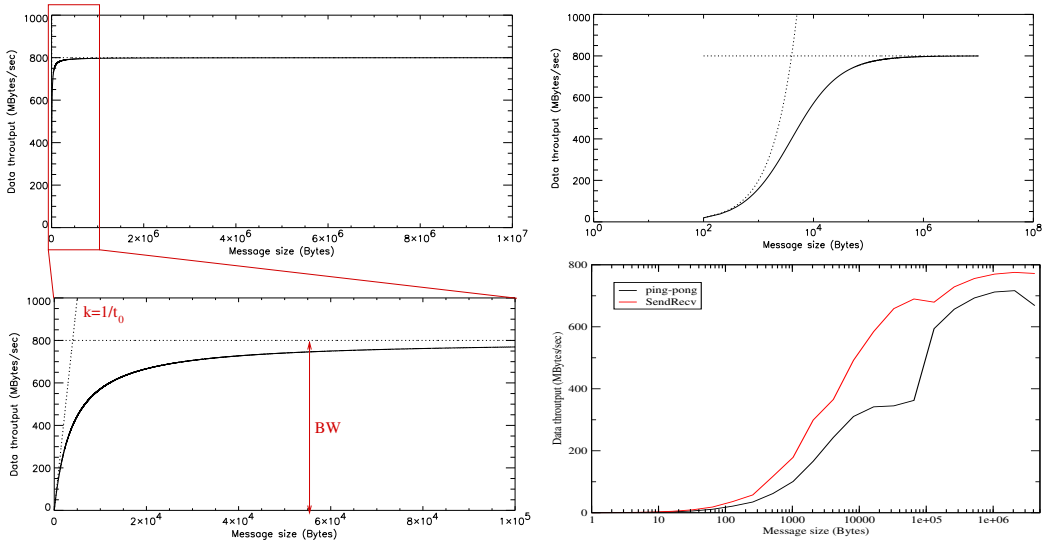
3. CLUSTER NA ASÚ AVČR V ONDŘEJOVĚ

V druhé polovině roku 2005 byl na AsÚ v Ondřejově realizován koncept počítačového clusteru pro náročné vědecké výpočty — *Ondrejov Cluster for Astrophysical Simulations, OCAS* (Obr. 3). Cluster se skládá z řídícího uzlu (master node) poskytujícího uživatelské rozhraní pro komunikaci s clusterem a zároveň hostícího servery pro obsluhu clusteru (NFS, PVFS2, daemon fronty úloh atd.) a šestnácti identických pracovních stanic (working nodes) SunFire V20z. Každá ze stanic je osazena dvěma 64-bitovými procesory AMD Opteron 252 s taktem 2.6 GHz, 4 GB DDR pamětí a 80 GB SCSI diskem. Další diskové kapacity jsou k dispozici na lokálním harddisku řídícího uzlu a na externím diskovém poli připojeném k řídícímu uzlu přes rozhraní SCSI. Uzly jsou propojeny třemi typy sítí — 1Gb Ethernet zajišťuje obsluhu clusteru a vstupně výstupní operace, 100Mb Ethernet tvoří služební síť pro nízkoúrovňové řízení a kontrolu systému (v této síti jsou zapojeny i administrativní porty některých obslužných zařízení, jako UPS), a konečně pro přenos dat mezi spolupracujícími procesy v protokolu MPI je používána vysokorychlostní síť *InfiniBand* (viz dále). Schema prosíťování a přiřazení logických disků fyzickým zařízením je na Obr. 4. Co se týče softwaru, na clusteru je kromě standardních překladačů pro C/C++ a Fortran instalováno prostředí MPI v implementaci MVAPICH2 podporující přenos zpráv přes rozhraní InfiniBand, a pro správu úloh a jejich front sytém Sun Grid Engine (SGE). Více informací o ondřejovském clusteru lze získat na stránkách s online dokumentací na http://wave.asu.cas.cz/ocas.

Síť pro MPI komunikaci Naprosto zásadní komponentou pro efektivní běh paralelního programu na clusteru je rychlé síťové propojení zabezpečující tok MPI zpráv mezi uzly. Rychlost propojení lze charakterizovat dvěma parametry – maximální datovou propustností (maximum bandwidth) $bw(\infty)$ udávající rychlost datového toku (např. v MB/s) při limitně nekonečně velké datové velikosti zprávy, kdy se už neuplatňuje vliv tzv. latence t_L — druhého zásadního parametru charakterizujícího síťové propojení. Latence udává dobu, která uplyne od požadavku na komunikaci přes síťové rozhraní do skutečného zahájení této komunikace. Celkový čas $t(s)$ pro odeslání zprávy o velikosti s tak můžeme odhadnout jako $t(s) = t_L + s/bw(\infty)$ a rychlost datového přenosu $bw(s)$ při předávání zpráv velikosti s pak je

$$bw(s) \equiv \frac{s}{t(s)} = \frac{bw(\infty)s}{bw(\infty)t_L + s} \qquad (2)$$

Z této závislosti je patrné, že latence ovlivňuje především oblast asymptoticky malých datových velikostí zpráv, zatímco maximální propustnost je důležitá pro zprávy o velikosti blížící se limitně k nekonečnu. Jak je vidět na Obr. 5, který ukazuje teoretický průběh závislosti (2) pro parametry uváděné výrobcem v porovnání se skutečným testem (pravý dolní panel), pro oblast velikosti zpráv typicky



Obr. 5. ***Analýza rychlosti datového přenosu pro síť InfiniBand použitou v ondřejovském clusteru v závislosti na velikosti zasílaných MPI zpráv. Panel vlevo nahoře ukazuje teoretický průběh závislosti podle vztahu (2) s výrobcem udávanými hodnotami*** $bw(\infty) = 800$ ***MB/s,*** $t_L = 5\mu$***s, panel pod ním pak detail z výřezu označeným obdélníkem. Čárkované přímky ukazují asymptotickou závislost pro malé a velmi velké datové velikosti zpráv. Vpravo nahoře je stejná teoretická závislost jen vyjádřená v log-lin stupnici (zakreslená včetně asymptot) a konečně vpravo dole je skutečné měření na ondřejovském clusteru za použití standardního testu IMB (dříve Pallas benchmark) rovněž v log-lin škále.***

používaných v numerických paralelních kódech (1kB — 1MB) jsou důležité oba parametry. Porovnání ukazuje, že hodnot latence a maximální propustnosti uváděných výrobcem InfiniBandu bylo na našem clusteru plně dosaženo.

4. MHD KÓD PRO MODELOVÁNÍ VE SLUNEČNÍ FYZICE

Dynamika plazmatu a magnetického pole je v rámci magnetohydrodynamického (MHD) modelu popsána následující soustavou parciálních diferenciálních rovnic:

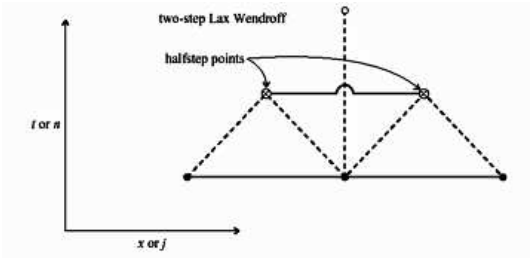
$$\begin{aligned} \frac{\partial \rho}{\partial t} + \boldsymbol{\nabla}\cdot(\rho \boldsymbol{u}) &= 0 \\ \rho\frac{\partial \boldsymbol{u}}{\partial t} + \rho(\boldsymbol{u}\cdot\boldsymbol{\nabla})\boldsymbol{u} &= -\boldsymbol{\nabla}p + \boldsymbol{j}\times\boldsymbol{B} + \rho\boldsymbol{g} \\ \frac{\partial \boldsymbol{B}}{\partial t} &= \boldsymbol{\nabla}\times(\boldsymbol{u}\times\boldsymbol{B}) - \boldsymbol{\nabla}\times(\eta\boldsymbol{j}) \\ \frac{\partial U}{\partial t} + \boldsymbol{\nabla}\cdot\boldsymbol{S} &= 0, \end{aligned} \qquad (3)$$

kde tok energie $\boldsymbol{S}$ a pomocné veličiny (tlak plazmatu p a hustota elektrického proudu $\boldsymbol{j}$) jsou vyjádřeny pomocí standardních vztahů (např. Kliem et al., 2000). Pro účely numerického řešení musí být tato soustava diskretizována, což znamená, že stavové veličiny popisující plazma a magnetické pole jsou reprezentovány vzorkovými hodnotami v uzlech diskretizační sítě a definovány pouze v diskrétních časových okamžicích. Parciální derivace v soustavě (3) jsou pak nahrazeny diferencemi podle zvoleného numerického schematu.

Numerické schema Numerické schema, které definuje jakým způsobem jsou derivace vyskytující se v diferenciální rovnici jež se má numericky integrovat reprezentovány příslušnými diferenčními formulemi bývá určeno pro nějakou poměrně širokou třídu rovnic. Nejčastěji jsou numerická schemata definována pro rovnice ve formálním tvaru zákona zachování

$$\frac{\partial \boldsymbol{u}(\boldsymbol{x},t)}{\partial t} + \frac{\partial \boldsymbol{F}(\boldsymbol{u}(\boldsymbol{x},t))}{\partial \boldsymbol{x}} = 0 \qquad (4)$$

kde $\boldsymbol{u}$ je obecně n-rozměrný stavový vektor závislý na pozici a čase. Druhý člen formálně představuje divergenci toku $\boldsymbol{F}$ veličiny $\boldsymbol{u}$. Pro tento typ rovnice je konstruováno nejen Lax-Wendroffovo (LW) numerické schema použité v rozebíraném MHD kódu, ale i řada modernějších integrátorů MHD rovnic. Aby bylo možno LW schema aplikovat na sadu rovnic (3), musí tato být nejprve převedena do tvaru (4). První a čtvrtá rovnice soustavy, vyjadřující zákon zachování hmoty (rovnice kontinuity), resp. energie již v tomto tvaru jsou. Druhý a třetí řádek (pohybovou a indukční rovnici) lze na tento tvar převést algebraickou manipulací, např. pohybová rovnice může být s využitím rovnice kontinuity a Maxwellovy rovnice $\boldsymbol{\nabla}\cdot\boldsymbol{B} = 0$ převedena na tvar



Obr. 6. ***Dvoustupňové Lax-Wendroffovo numerické schema pro integraci parciálních diferenciálních rovnic ve tvaru (4). V prvním kroku jsou spočteny pomocné „interpolované" veličiny v bodech mezi uzly diskretizační sítě a v polovině časového kroku. Tyto pomocné veličiny jsou využity jen pro výpočet nového stavu v druhém kroku a pak mohou být zapomenuty.***

$$\frac{\partial \rho u_i}{\partial t} = -\nabla_j . \left(\rho u_i u_j - \frac{B_i B_j}{\mu_0} + \delta_{ij} (\frac{B^2}{2\mu_0} + p) \right) \quad (5)$$

a podobně upravíme i indukční rovnici s využitím vztahu pro proudovou hustotu v kvazistacionárním přiblížení $\boldsymbol{\nabla} \times \boldsymbol{B} = \mu_\circ \boldsymbol{j}$. Takto je celá soustava (3) vyjádřena ve tvaru (4) se stavovým vektorem

$$\boldsymbol{u} = \begin{pmatrix} \rho \\ \rho v_x \\ \rho v_y \\ \rho v_z \\ B_x \\ B_y \\ B_z \\ U \end{pmatrix} \quad (6)$$

a tokem $\boldsymbol{F}$ složeným z odpovídajících členů pravých stran soustavy (3) vyjádřených pomocí komponent vektoru (6).

Samotné schema pro řešení diskretizované rovnice ve tvaru (4) spočívá ve výpočtu stavového vektoru ve zvoleném bodě prostoru v j následujícím časovém okamžiku $(n+1)$ z hodnot stavových vektorů v tomto bodě a jeho okolí $(j \pm 1)$ ve dvou krocích popsaných (pro zjednodušení v 1D případě) diferenčními rovnicemi (Press et al., 1992, viz též Obr.6)

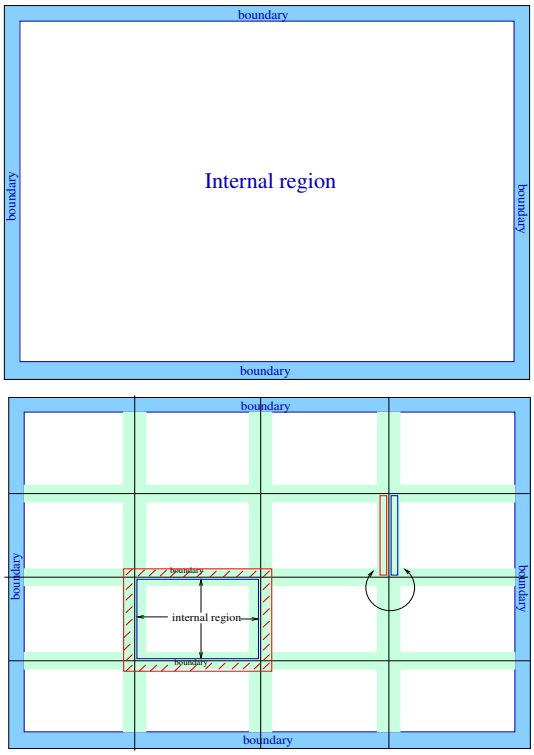
$$\boldsymbol{u}_{j+1/2}^{n+1/2} = \frac{1}{2}(\boldsymbol{u}_{j+1}^n + \boldsymbol{u}_j^n) - \frac{\Delta t}{2\Delta x}(\boldsymbol{F}_{j+1}^n - \boldsymbol{F}_j^n) \quad (7)$$

pro výpočet pomocných „mezistavových" veličin v prvním kroku a

$$\boldsymbol{u}_j^{n+1} = \boldsymbol{u}_j^n - \frac{\Delta t}{\Delta x}\left(\boldsymbol{F}_{j+1/2}^{n+1/2} - \boldsymbol{F}_{j-1/2}^{n+1/2} \right) \quad (8)$$

pro vyčíslení nového stavu v čase $n+1$.

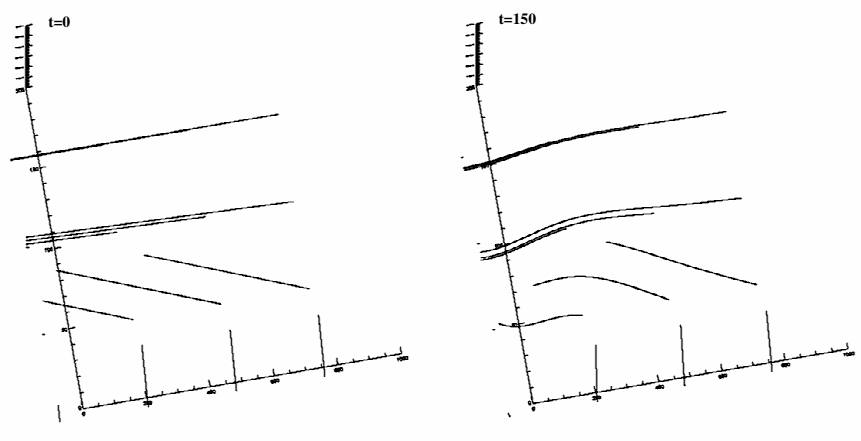
Hodnoty $\boldsymbol{u}_{j\pm1}^n$ v sousedních bodech nutné pro výpočet nového stavu $\boldsymbol{u}_j^{n+1}$ dle vztahů (7) a (8) nejsou definovány, dospějeme-li k hranici oblasti v níž provádíme výpočet. U „klasického" neparalelního kódu je způsob prodloužení stavových veličin z oblasti výpočtu do oblasti hranice dán diskrétní reprezentací zvolené hraniční podmínky (Obr. 7, nahoře) — např. von Neumanovu hraniční podmínku $\partial\rho/\partial\boldsymbol{n} = 0$ pro hustotu numericky implementujeme tak, že že hodnotu hustoty v hraničním bodě doplníme hodnotou v nejbližším (ve směru kolmém k hranici) vnitřním bodě.



Obr. 7. ***Paralelizace metodou rozložení do podoblastí (domain decomposition).***

Metoda paralelizace Metoda paralelizace spočívá v rozdělení výpočtové oblasti do mnoha podoblastí, z nichž každá je řešena jedním procesorem (Obr. 7, dole). Tím ovšem dostáváme dva typy hranic: vnitřní mezi oblastmi řešenými jednotlivými procesy, a vnější – skutečné hranice úlohy. S vnějšími hranicemi zacházíme stejně jako u sekvenčního kódu, zatímco prodloužení dat v jedné podoblasti za její vnitřní hranici je dáno stavem ve vnitřní výpočtové oblasti sousedního procesu. Potřebná data získáme ze sousedního procesu pomocí funkcí protokolu MPI.

První výsledky Kromě 2D a 2.5D verzí MHD kódu, jimiž dosažené výsledky již byly publikovány byly prováděny i první testy 3D kódu. První výsledky těchto testovacích výpočtů jsou na Obr. 8, který ukazuje simulaci 3D rekonexe střižného (*sheared*) magnetického pole.



Obr. 8. ***První výsledky 3D MHD paralelního kódu aplikovaného na problém rekonexe ve střižném (sheared) magnetickém poli. Siločáry magnetického pole v počátečním stavu (vlevo) a po uběhnutí 150 časových jednotek simulace (vpravo).***

Efektivita paralelizace Důležitou otázkou je, zda se paralelizace pro danou úlohu vůbec vyplatí — čili přináší rozdělení výpočtu na mnoho procesorů (kromě netriviální výhody větší úhrnné kapacity paměti – mnoho úloh je na PC neřešitelných právě z důvodů paměťových limitů) podstatné urychlení výpočtu? Měření pro 3D MHD kód spuštěný na různém počtu procesorů ondřejovského clusteru *OCAS* ukazuje Tab. 1.

počet CPU	čas výpočtu
1	8516
2	4867
4	2496
8	1273
16	629
24	415

Tab. 1. ***Efektivita paralelizace měřená pomocí času (v sekundách) nutného pro výpočet standardizované úlohy (3D MHD simulace*** $500 \times 100 \times 20$ ***gridů, 50 časových výpočetních jednotek).***

Je vidět, že i pro 24 použitých procesorů se doba výpočtu zkrátila téměř 21x.

Během testů 3D MHD kódu bylo zjištěno, že efektivita paralelizace může být silně degradována použitím nevhodných metod pro výstup dílčích výsledků na disk společně sdílený jednotlivými procesory — díky konkurenčnímu přístupu mnoha procesů ke stejnému zdroji dochází ke konfliktům, jejichž řešení vyžaduje mnoho režijního času. Naštěstí – MPI verze 2 poskytuje prostředky pro paralelní zápis dat, který tuto degradaci plně odstraňuje – více o této speciální problematice naleznete např. na http://www.mpi-forum.org.

5. ZÁVĚR

Implementace numerického řešení MHD rovnic do počítačového kódu umožňuje modelování mnoha jevů sluneční aktivity (erupce, CME, protuberance). Numerická integrace systému MHD rovnic v realistických případech (rozsáhlé 2D nebo 3D geometrie) je však výpočetně velice náročná (obsazení paměti, procesorový čas) a na standardním hardwaru (PC) prakticky nerealizovatelná. Řešením je rozdělení úlohy na mnoho procesorů – paralelizace. Pro 3D MHD model rekonexe bylo zjištěno, že při jejím rozdělení na 24 procesorů dojde k urychlení výpočtu téměř 21x, obsazení paměti na jeden procesor je téměř 24x menší. Paralelizace MHD algoritmu je tedy velice efektivní nástroj umožňující numerické řešení i poměrně komplikovaných úloh na dostupném hardwaru — počítačovém clusteru.

Poděkování

Výzkum je prováděn za podpory Centra pro teoretickou astrofyziku a grantů *IAA3003202* a *205/04/0358* poskytnutých GA AVČR a GAČR.

LITERATURA

Bagla, J.S. 2004, Khagol 59
(též http://www.mri.ernet.in/~jasjeet/work.html)

Hlavička, J. 1994, Architektura počítačů, Skriptum ČVUT, Praha

Kliem, B., Karlický, M., Benz, A.O. 2000, A&A 360, 715

Nadrchal, J. (ed.) 2004, Proc. of 14^{th} Summer school on computing techniques in physics: Clusters for computing in physics, Třešť u Jihlavy

Press, W.H, Teukolsky, S.A., Vetterling, W.T., Flannery, B.P. 1992, Numerical Recipes in C — The art of scientific computing, Cambridge Univ. Press