

C2115

Praktický úvod do superpočítání

XIII. lekce / Modul 4

Petr Kulhánek

kulhanek@chemi.muni.cz

Národní centrum pro výzkum biomolekul, Přírodovědecká fakulta,
Masarykova univerzita, Kotlářská 2, CZ-61137 Brno

Paralelní úlohy

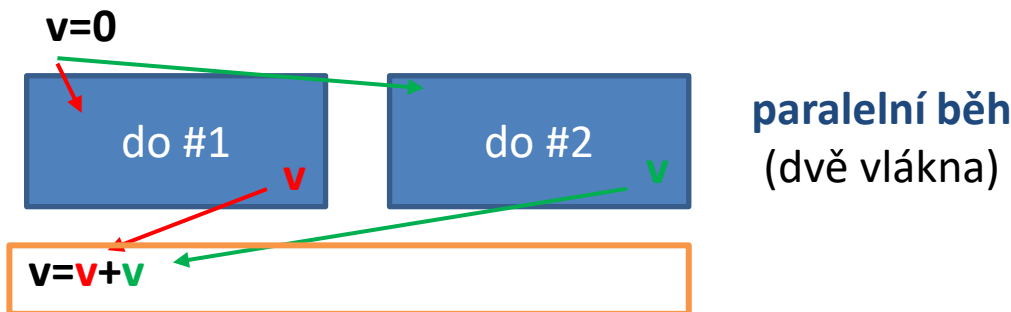
Souběh chyb, numerické chyby, spouštění aplikace,
efektivita paralelizace (Amdahlův zákon)

Úskalí paralelizace úloh, chyby souběhu

Paralelní úlohy jsou náchylné na **chyby souběhu** (race condition). Při návrhu paralelních aplikací je proto nutné již při návrhu aplikace tomuto úskalí věnovat značnou pozornost.

```
v = 0.0d0
!$omp do private(i,x,y), reduction(+:v)
do i=1,n
  x = (i-0.5d0)*h + r1
  y = 4.0d0/(1.0d0+x**2)
  v = v + y*d
end do
!$omp end do
```

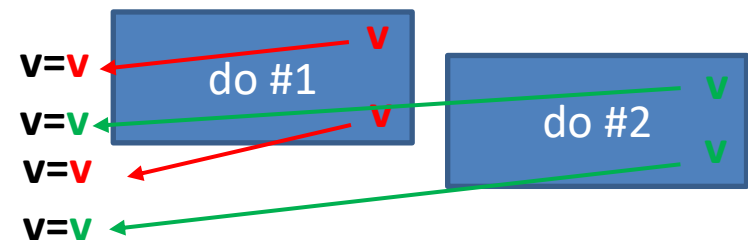
!!! klíčové pro funkčnost !!!



bariéra (nejdříve se počká na doběhnutí všech vláken a teprve poté se vypočítá výsledek)

Bez bariéry se bude proměnná "v" používat v nedefinovaném pořadí s možností přepisování mezivýsledků.

Zjednodušeně (pro ilustraci):



Úskalí paralelizace úloh, numerické chyby

Pořadí vykonávání aritmetických operací se může v čase měnit, proto můžete získat **různý výsledek** za použití různého počtu CPU a či při opakování stejné paralelní úlohy (typické pro úlohy s dynamickým vyvažováním zátěže).

Úlohy s dynamickým vyvažováním zátěže mění rozdělení problému na jednotlivé CPU (např. atomy, které bude CPU zpracovávat v molekulové dynamice) tak, aby úloha běžela co nejrychleji.

```
[kulhanek@wolf openmp]$ OMP_NUM_THREADS=1 ./integral
Number of threads =          1
integral =      3.1415926535894521
[kulhanek@wolf openmp]$ OMP_NUM_THREADS=2 ./integral
Number of threads =          2
integral =      3.1415926535888206
[kulhanek@wolf openmp]$ OMP_NUM_THREADS=4 ./integral
Number of threads =          4
integral =      3.1415926535898944
```

Spouštění paralelních úloh

- U **nových úloh vždy ověřujte, že skutečně spouštíte paralelní verzi aplikace** (programu).
- Pokud úloha neběží paralelně, ověřte, že spouštíte správnou verzi programu a správným způsobem (OpenMP vs MPI).

Užitečné příkazy:

- **ldd** vypíše seznam dynamických knihoven, které program využívá
- **top** vypíše přehled běžících procesů
- **ps tree** vypíše přehled běžících procesů

Spouštění paralelních úloh, pokr.

OpenMP

```
bash(17479) — integral(21331) — {integral}(21332)
                                     {integral}(21333)
                                     {integral}(21334)
```

1 proces a 4 vlákna

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
21331	kuľhanek	20	0	29284	1592	1452	R	399.3	0.0	1:40.14	integral
3469	root	20	0	0	0	0	S	0.0	0.0	4:53.43	afsd

MPI

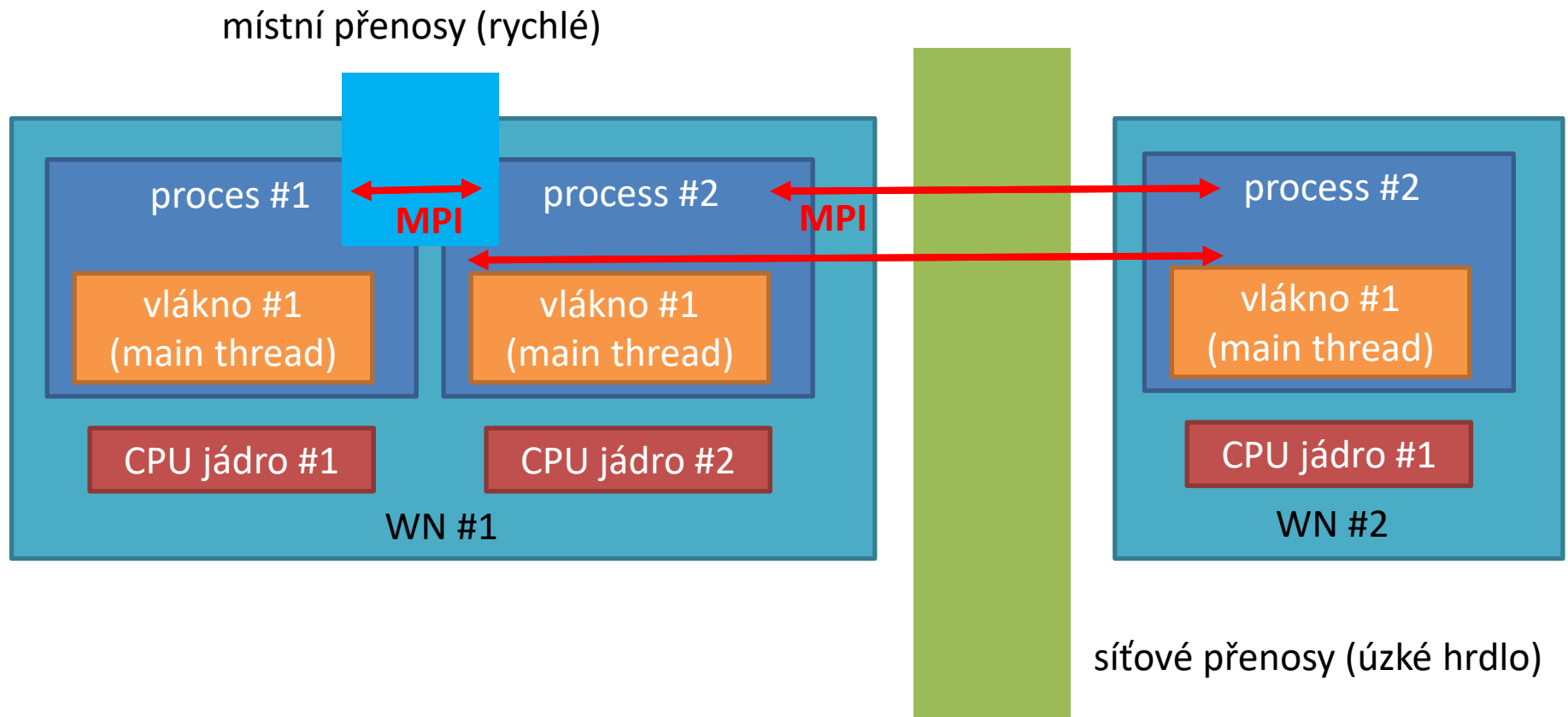
```
bash(17479) — mpirun(23938) — integral(23942) — {integral}(23946)
                                     {integral}(23948)
                                     {integral}(23956)
                                     integral(23943) — {integral}(23949)
                                     {integral}(23952)
                                     {integral}(23955)
                                     integral(23944) — {integral}(23947)
                                     {integral}(23950)
                                     {integral}(23957)
                                     integral(23945) — {integral}(23951)
                                     {integral}(23953)
                                     {integral}(23954)
                                     {mpirun}(23939)
                                     {mpirun}(23940)
                                     {mpirun}(23941)
```

4 procesy (+ servisní vlákna, které využívá MPI)

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
23942	kuľhanek	20	0	375740	15796	12576	R	100.0	0.1	0:15.48	integral
23943	kuľhanek	20	0	375740	15680	12456	R	100.0	0.1	0:15.48	integral
23944	kuľhanek	20	0	375740	15724	12508	R	100.0	0.1	0:15.48	integral
23945	kuľhanek	20	0	375740	15736	12512	R	100.0	0.1	0:15.48	integral

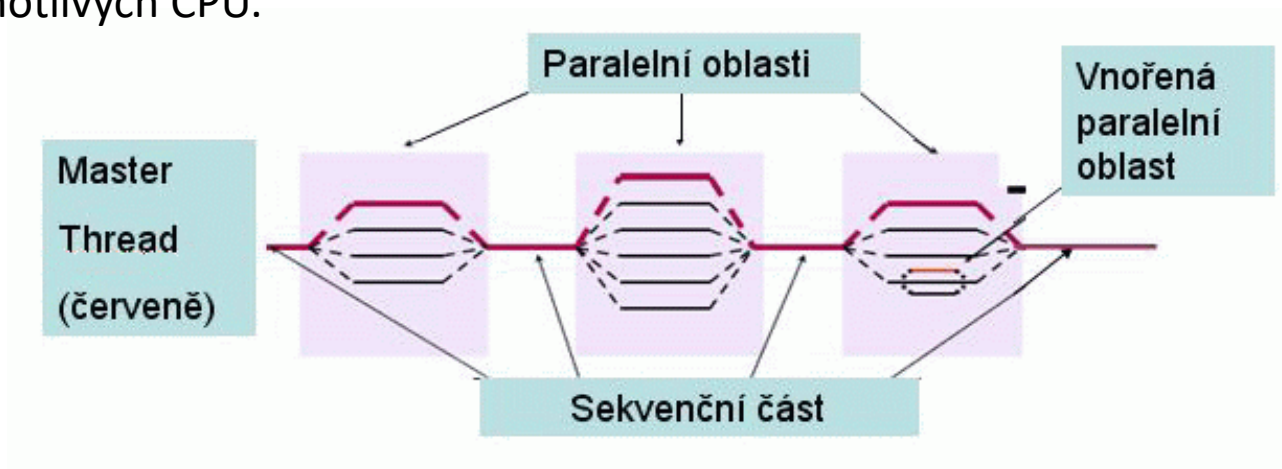
Spouštění paralelních úloh, pokr.

- Vyvarujte se spouštění paralelních úloh na více uzlech (MPI).
- Pokud je to nezbytné, použijte co nejrychlejší síťové propojení (**Infiniband** vs Ethernet)



Efektivita paralelizace

Nedílnou (nicméně nechtěnou) součástí paralelní aplikace jsou sekvenční části. Běh této části není urychlován se zvyšováním počtu CPU, což ve výsledku vede k snižování efektivity využití jednotlivých CPU.



1 CPU



2 CPU



zůstává stejné

zkracuje se

Amdahlův zákon

Amdahlův zákon vyjadřuje maximální předpokládané urychlení paralelní úlohy.

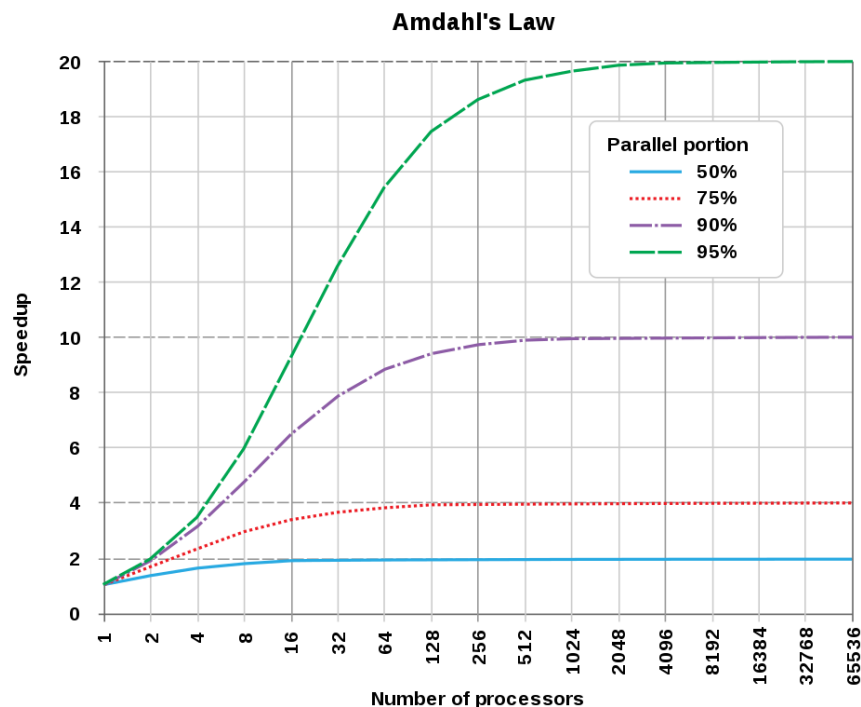
Teoretické urychlení:

$$S_{speedup} = \frac{1}{(1 - p) + \frac{p}{N}}$$

p - paralelní část programu (zlomek)

N - počet CPU

https://en.wikipedia.org/wiki/Amdahl's_law



- **VŽDY ověřujte efektivitu paralelního běhu aplikace** (hlavně u nových problémů či změně jejich velikosti).
- Ověření se dělá podle cvičení L13.M2.1, přičemž akceptovatelná minimální efektivita na jedno CPU je cca 80%.

Cvičení M4.1

Zdrojové kódy:

/home/kulhanek/Documents/C2115/code/integral/openmp

1. Zkompilujte program **integral_rc.f90** s optimalizací **-O3** a podporou OpenMP.
2. Spouštějte program **integral_rc** postupně pro 1, 2, 3, až N vláken, kde N je maximální dostupný počet CPU jader. Jak se mění výsledek a proč?
3. Opakovaně spusťte program **integral_rc** na 2 vláknech. Jak se mění výsledek a proč?