Revize 2

C2115 Praktický úvod do superpočítání

12. lekce / Modul 4

Petr Kulhánek

kulhanek@chemi.muni.cz

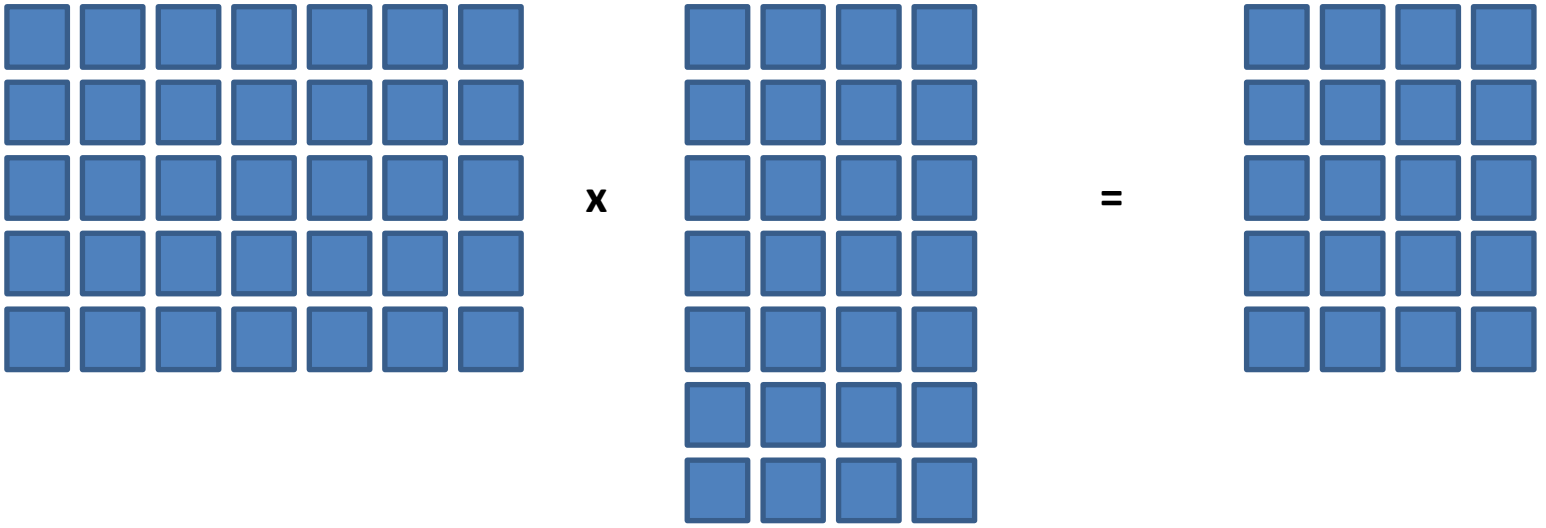
Národní centrum pro výzkum biomolekul, Přírodovědecká fakulta,
Masarykova univerzita, Kotlářská 2, CZ-61137 Brno

Násobení matic

Obsah

- **Násobení matic**
  implementace, komplexita, výpočetní výkon, cvičení
- **Vysvětlení získaných výsledků**
  architektura počítače a její úzká hrdla
- **Použití optimalizovaných knihoven**
  BLAS, LAPACK, LINPACK, porovnání výsledků, cvičení

Násobení matic

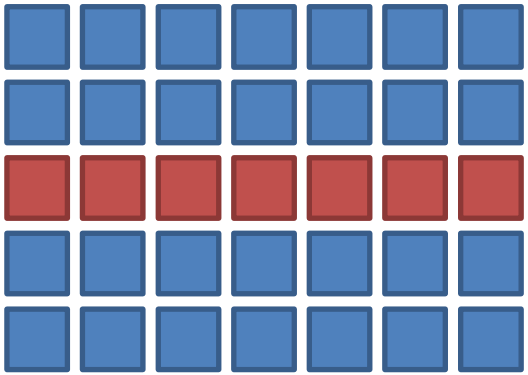


A(n,m) **B**(m,k) **C**(n,k)

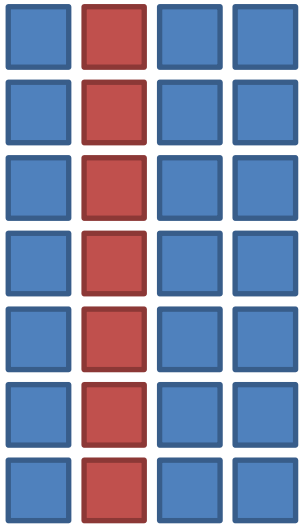
Využití:

- hledání vlastních čísel a vektorů čtvercových matic (kvantová chemie)
- řešení soustavy lineárních rovnic (QSAR, QSPR)
- transformace (posunutí, rotace, škálování - zobrazení a grafika)

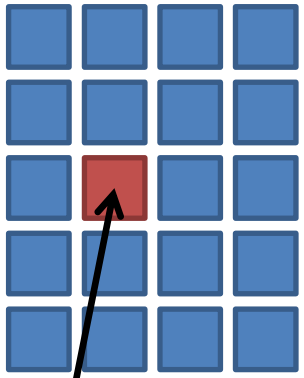
Násobení matic



x



=



A(n,m)

B(m,k)

C(n,k)

$$C_{ij} = \sum_{l=1}^{m} A_{il} B_{lj}$$

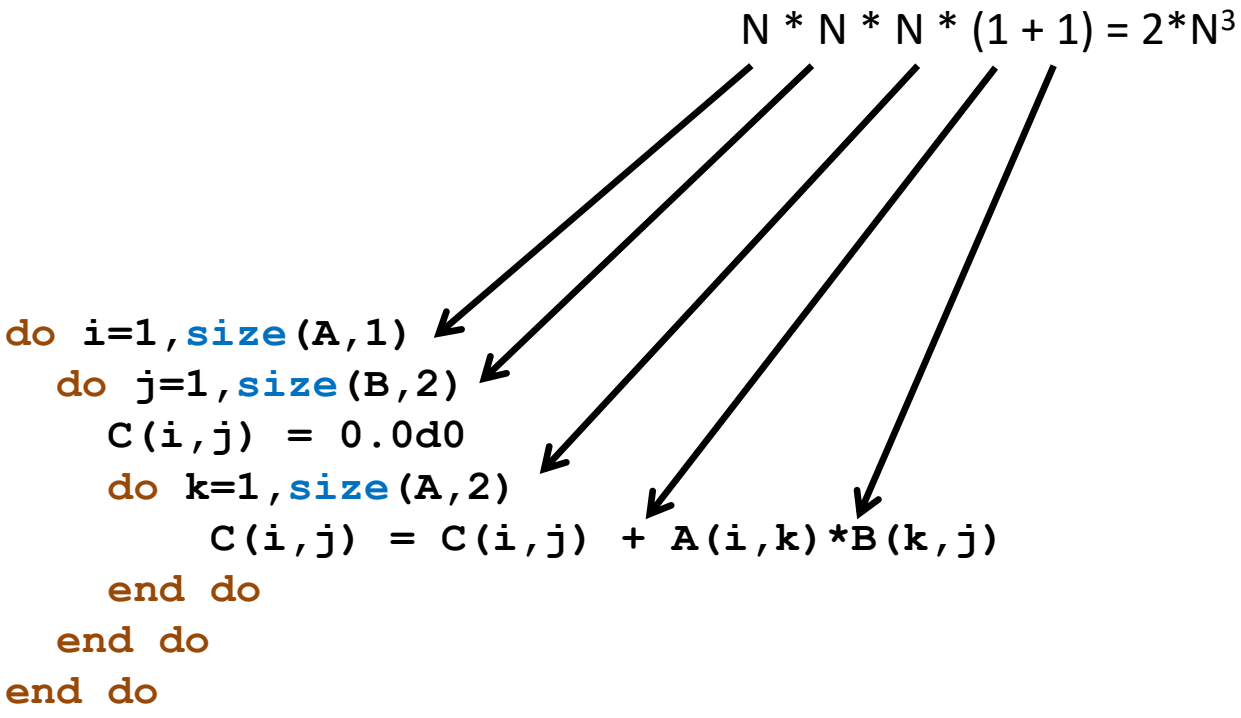
prvek výsledné matice **C** je skalárním součinem vektorů tvořených řádkem *i* matice **A** a sloupcem *j* matice **B**

Násobení matic, implementace

```
subroutine mult_matrices(A,B,C)

  implicit none
  double precision    :: A(:,:)
  double precision    :: B(:,:)
  double precision    :: C(:,:)
  !-------------------------------------
  integer             :: i,j,k
  !-------------------------------------------------------------

  if( size(A,2) .ne. size(B,1) ) then
    stop 'Error: Illegal shape of A and B matrices!'
  end if

  do i=1,size(A,1)
    do j=1,size(B,2)
      C(i,j) = 0.0d0
      do k=1,size(A,2)
          C(i,j) = C(i,j) + A(i,k)*B(k,j)
      end do
    end do
  end do

end subroutine mult_matrices
```

Počet operací

Za předpokladu, že matice **A** a **B** jsou čtvercové o rozměrech NxN:



Ve výpočetní technice se pro posuzování výpočetního výkonu používá hodnota udávaná ve **FLOPS (FLoating-point Operations Per Second)**, která vyjadřuje kolik operací v pohyblivé čárce dané zařízení vykoná za sekundu.

Výsledky

wolf21: gfortran 4.6.3, optimalizace O3, Intel(R) Core(TM) i5 CPU 750 @ 2.67GHz

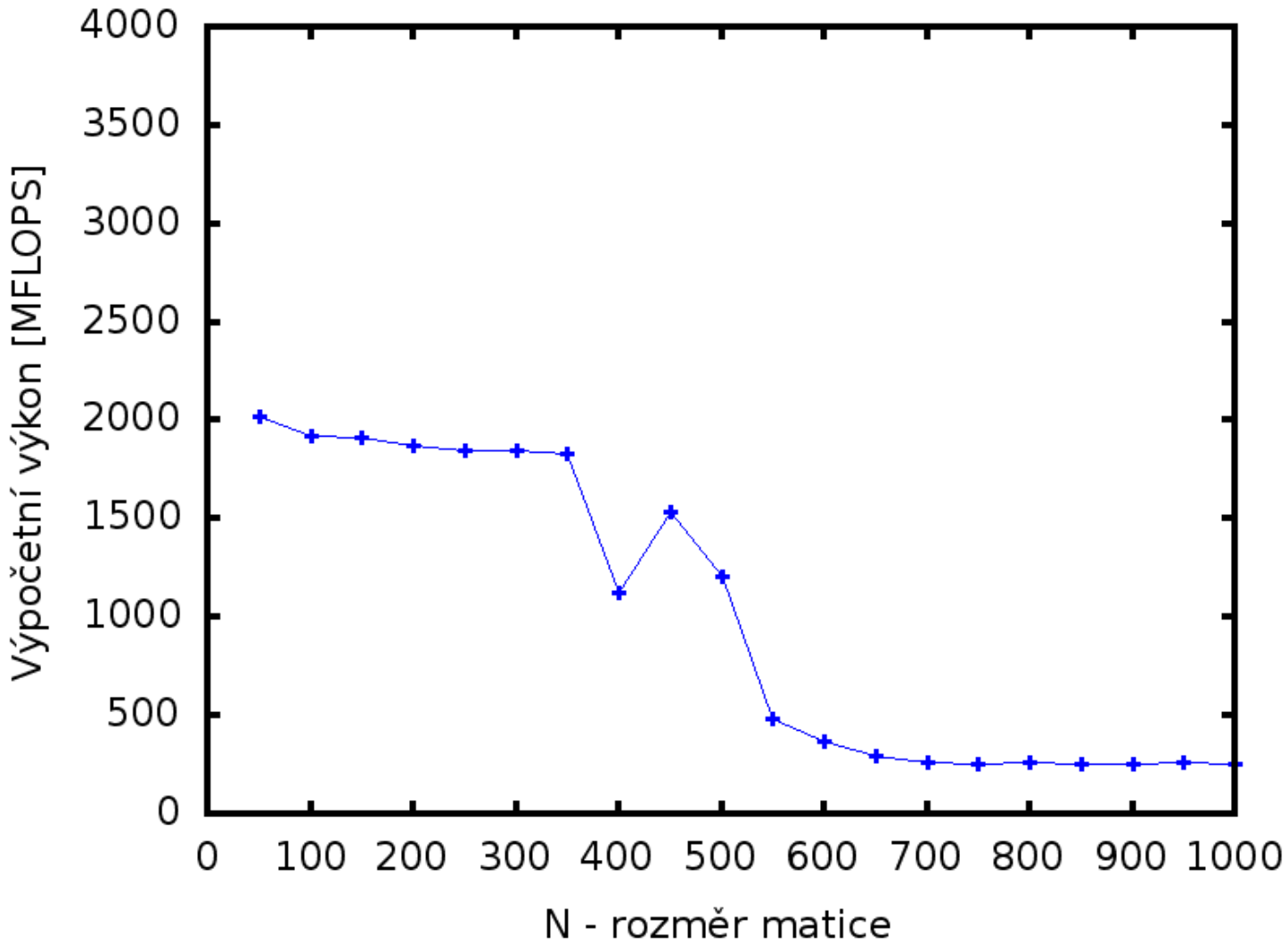
N	NR	NOPs	Time	MFLOPS
50	50000	12500000000	6.1843858	2021.2
100	500	1000000000	0.5200334	1923.0
150	50	337500000	0.1760106	1917.5
200	50	800000000	0.4280272	1869.0
250	50	1562500000	0.8440533	1851.2
300	50	2700000000	1.4640903	1844.1
350	50	4287500000	2.3441458	1829.0
400	50	6400000000	5.7083569	1121.2
450	50	9112500000	5.9363708	1535.0
500	50	12500000000	10.3366470	1209.3
550	1	332750000	0.6880417	483.6
600	1	432000000	1.1600723	372.4
650	1	549250000	1.8601189	295.3
700	1	686000000	2.5881615	265.1
750	1	843750000	3.2762032	257.5
800	1	1024000000	3.8522377	265.8
850	1	1228250000	4.7883034	256.5
900	1	1458000000	5.6963577	256.0
950	1	1714750000	6.5044060	263.6
1000	1	2000000000	7.9444962	251.7

Legenda:

N – rozměr matice
NR – počet opakování
NOPs – počet operaci v FP
Time – doba vykonávání v s
MFLOPS – výpočetní výkon

Výsledky

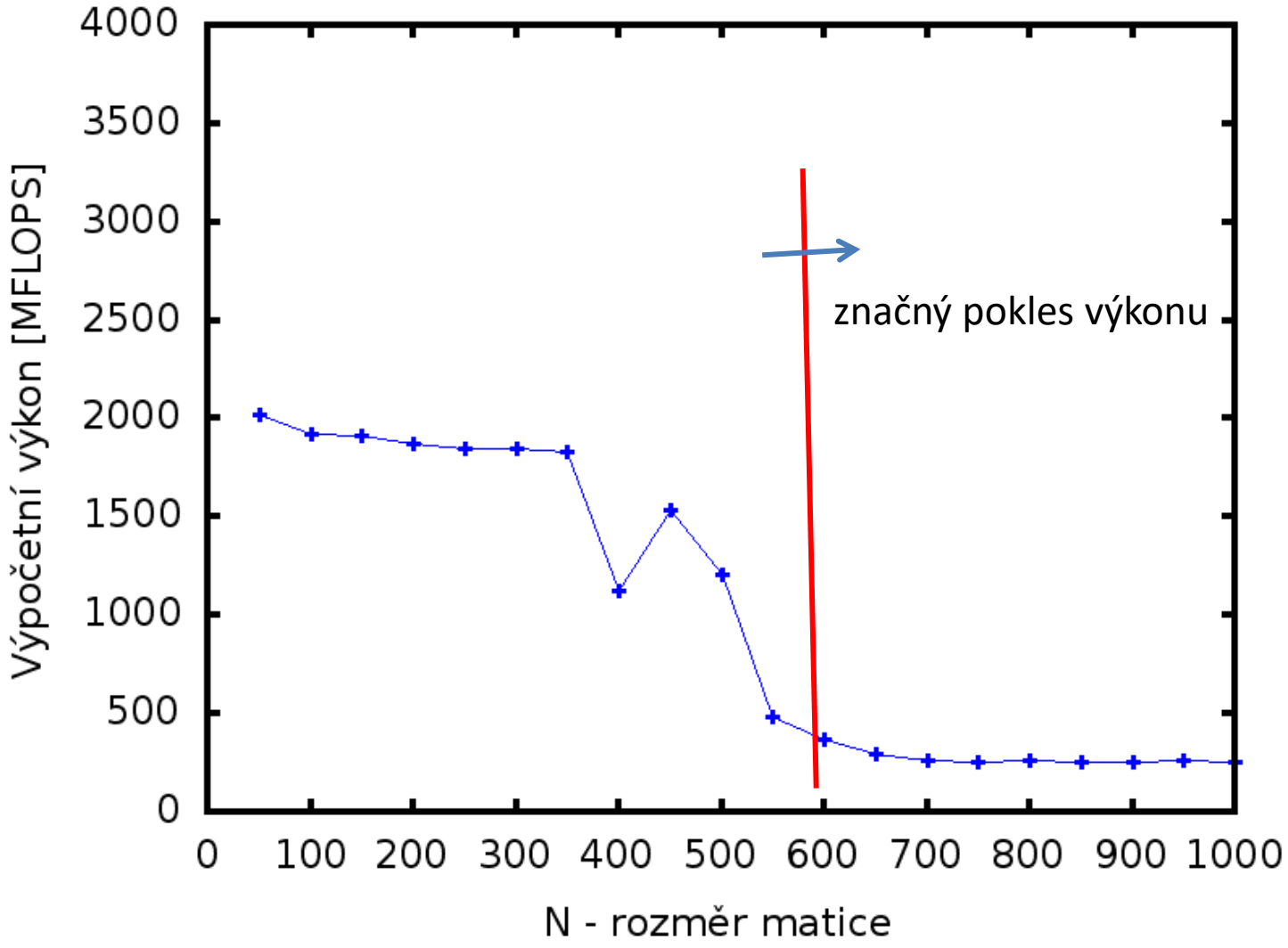
wolf21



Výsledky

wolf21

wolf21



Cvičení M4.1

Zdrojové kódy:
/home/kulhanek/Documents/C2115/code/matrix

1. Zkompilujte program **mult_mat_naive_dp.f90** kompilátorem **gfortran**, použijte **-O3** optimalizaci.
2. Program spusťte a získanou závislost výpočetního výkonu v závislosti na rozměru matice zobrazte ve formě grafu (interaktivní režim gnuplotu).
3. Porovnejte výsledky pro optimalizační úroveň **-O3 a -O0**. Získané závislosti zobrazte v jednom grafu. Graf vložte do protokolu. Nezapomeňte uvést typ CPU (příkaz lscpu).
4. Diskutujte získané výsledky.