程序实际性能及性能优

当前高性能计算机的主要特征

- 计算速度与访存速度越来越失配 (*Moore's*
- 多级存储、非均匀存储、大规模并行处理
- 消息传递并行 (MPI) 与共享存储并行 (Op
- 串行程序性能: 访存局部化
- 并行程序性能: 数据相关性、负载均衡、通

程序性能

- 由于计算机 CPU 与内存性能间的日益失
  的实际性能远远低于 CPU 的峰值性能。
  - ⋆ HPC Linpack: 50%–80%
  - ⋆ FFTW: $<$ 50%
  - ⋆ 普通应用程序: 1% $\sim$ 20%
- 典型应用程序的实际性能通常在 CPU 峰
  以下。
- 高性能计算机的峰值性能，甚至 HPC Lin

影响程序单机性能的主要因素

- 处理器/内存结构
- 算法结构
- 程序结构
  - ⋆ 数组分配方式与数据对齐
  - ⋆ 循环顺序, 运算组织
- 编译优化
- 性能优化工具

影响程序并行性能的主要因素

计算机软硬件特性

通信、计算性能，通信的 *CPU* 占用率，通

算法复杂性

冗余计算、数据访问模式变化、内存开销

算法并行度

数据相关性造成处理器空闲等待。

通信开销

通信量、通信粒度、及通信模式 (通信图案

负载均衡

程序性能实例 1：LSSC–II 性能统计

LSSC–II 部分程序性能统计

应用程序实例 2: 矩阵乘 — 循环顺序

```
DO I = 1, N
   DO J = 1, N
      DO K = 1, N
         C(I,J) = C(I,J) + A(I,K) * B
      ENDDO
   ENDDO
ENDDO
```

考查不同循环顺序: IJK, IKJ, JIK, ...

- 源程序: *matmul0.m4*
- 编译器: Intel Fortran 8.0
- 编译选项: `-O3 -align all -unroll4`

800MHz P3, 256KB 二级 cache

2.0GHz P4 Xeon, 512KB 二级 cache, Intel For

2.4GHz P4 Xeon, 512KB 二级 cache

应用程序实例 3: 矩阵乘 — 数组维数

```
DIMENSION A(LD, N), B(LD, N), C(LD, N)
... ...
DO I = 1, N
   DO J = 1, N
      DO K = 1, N
         C(I,J) = C(I,J) + A(I,K) * B
      ENDDO
   ENDDO
ENDDO
... ...
```

N = 1024, P4/2.4G, Intel Fortran 8.0

编译选项：ifort -O3 -align all -unroll4

循环顺序	实际性能 (Mflops)	
	LD=1024	LD=1025
I J K	23.25	195.23
I K J	9.35	67.19
J I K	24.63	148.20
J K I	489.18	484.76
K I J	9.36	62.57
K J I	229.43	231.91

应用程序实例 4: Fortran Linpack

在具有 P 个处理器的机器上同时运行 p 份
pack 程序 ($p = 1, 2, \ldots, P$).

每组数据运行数次，性能按 最小/平均/最
出。

每种情况第一行数据对应于 LDA=N，第二
LDA=N+1 (LDA 为数组首维维数，即: DIMENSION

	$N = 100$	$N = 400$	$N = 1000$	
1	728/728/728	880/880/880	413/413/413	4
	728/728/728	967/967/967	411/411/411	4
2	728/728/728	880/885/889	412/413/414	3
	728/728/728	967/967/967	288/289/290	2
4	728/730/734	889/889/889	410/414/416	4
	767/774/778	947/954/967	411/414/420	2
8	718/730/739	880/885/889	408/413/420	2
	767/771/778	947/957/967	283/353/422	2
16	723/730/739	880/885/898	272/349/416	2

	$N = 100$	$N = 400$	$N = 1000$	
1	524/524/524	716/716/716	338/338/338	4
	524/524/524	661/661/661	336/336/336	4
2	497/510/524	614/637/661	324/326/329	3
	497/510/524	537/599/661	323/326/329	4
4	524/545/553	537/596/661	278/297/315	3
	524/539/553	537/587/661	278/295/315	3
8	524/542/553	505/552/661	219/238/290	2
	524/531/553	505/542/573	219/236/291	2
16	497/526/553	286/457/573	197/204/212	1
	497/524/553	296/428/537	186/202/225	1

	$N = 100$	$N = 400$	$N = 1000$	
1	639/639/639	749/749/749	520/520/520	5
	765/765/765	749/749/749	518/518/518	5
2	643/645/648	742/742/742	520/520/521	5
	765/771/777	742/749/756	519/524/528	5
4	635/644/648	736/747/756	517/523/527	5
	759/771/777	742/747/756	521/525/527	5
8	652/652/652	736/747/756	516/522/526	5
	771/777/783	736/743/756	519/522/526	5
16	635/645/652	729/736/742	512/522/528	5

程序性能优化

- 高速 (多通道) 内存、多级大容量 Cache、基
  线的向量/伪向量技术。缺点: 大大增加硬
  对许多大规模 (大内存量) 计算效果有限。
- 高性能数学库的使用: BLAS, FFTW, LAP
  PACK, ...
- 编译优化技术
- 性能优化工具
- 人工优化: 辅助工具

程序性能优化实例：循环展开

Fortran 程序 (DGEMM)

```
DO J = 1, N
   DO K = 1, N
      TEMP = B( K, J )
      DO I = 1, N
         C( I, J ) = C( I, J ) + TEMP*A
      ENDDO
   ENDDO
ENDDO
```

当 N 为偶数时步长为 2 的三重循环展开结果:

```
DO J=1,N,2
  DO K=1,N,2
    T00=B(K,J)
    T01=B(K+1,J)
    T10=B(K,J+1)
    T11=B(K+1,J+1)
    DO 140 I=1,N,2
      C(I,J)=C(I,J)+T00*A(I,K)+T01*A(I,K+1)
      C(I+1,J)=C(I+1,J)+T00*A(I+1,K)+T01*A(I
      C(I,J+1)=C(I,J+1)+T10*A(I,K)+T11*A(I,K
      C(I+1,J+1)=C(I+1,J+1)+T10*A(I+1,K)+T11
    ENDDO
```

M4 宏语言 (*unroll.m4*)

```
m4_local(t, j, l)
m4_do( j, 1, n, 1, nn_2, {
 m4_do( k, 1, n, 1, nn_3,{
  m4_expand(j, l, {t = alpha*b( l, j )}
  m4_do( i, 1, n, 1, nn_1,{
   m4_expand( i, j, {
    c(i,j)=c(i,j) m4_expand(k, {&+a(i,k
   })
  })
 })
})
```

转换 (*ftp://ftp.cc.ac.cn/pub/home/zlb/mfor/*)

```
m4 -P -Dnn_1=2 -Dnn_2=2 -Dnn_3=2 \
    defs.m4 unroll.m4 | m4post > unroll
```

(转换结果)

演示：搜索最佳展开步数