实验四 集群系统性能测试

严昕宇 20121802

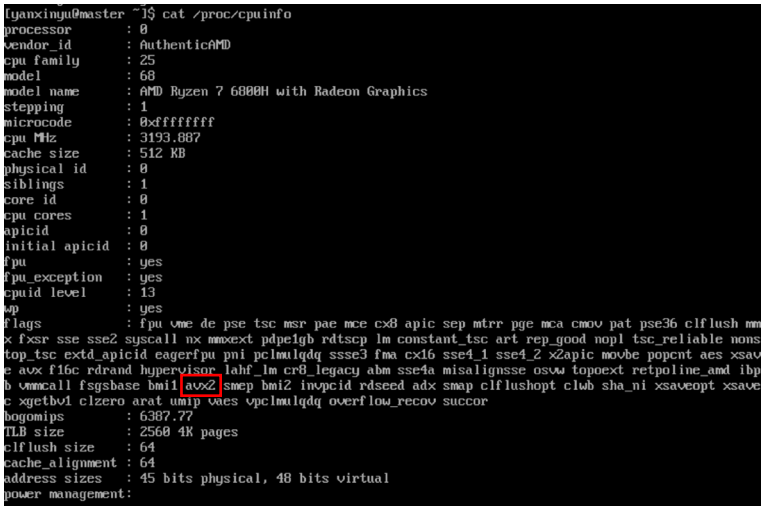
一、实验步骤：

1. 计算计算机峰值速度

CPU 主频：查看/proc/cpuinfo 文件，将看见 cpu 的详细信息，其中 cpu MHz 是主频值

网上查找资料计算峰值速度

cpuinfo 文件：



对于 Linpack 测试需要的双精度数据(64bits)来说，AVX1/2 每次计算可以支持 4 个双精度数据操作，而每个时钟周期可以执行一次浮点乘法和一次浮点加法运算，所以理论上每个时钟周期最多支持 8 次浮点计算的能力。对 AVX2 来说，因为支持所谓的 FMA 融合乘加也就是在一次计算里就能同时完成一次乘法和一次加法，所以理论上每个时钟周期的浮点计算能力得到再次翻倍到 4 次。

因此，本 CPU 所支持的指令集中有 AVX2，则 CPU 每个时钟周期执行双精度浮点运算次数=4。

理论浮点峰值＝CPU 主频×CPU 每个时钟周期执行浮点运算次数×CPU 数量，则本设备搭载的 CPU（AMD Ryzen 7 6800H）的计算峰值速度=3.2G×4×4=51.2GFlops。

2. 性能测试

使用 gcc 编译器的情况下测试，并将最佳测试结果填写下面表格

进程个数	4	6	8	32
HPL 测试峰值速度（Gflops）	1.388e+01	4.182e+00	4.297e+00	2.884e-01
效率	27.11%	8.17	8.39%	0.56%

参与运算主机名	进程数	N	NB	P	Q	Time	HPL 测试的 Gflops	效率
master	4	2048	80	2	2	0.37	1.555e+01	30.37%
master	6	1960	80	2	2	0.94	5.370e+00	10.49%
master	8	1960	80	2	2	1.32	3.805e+00	7.43%
master	32	2048	80	2	2	28.20	2.033e-01	0.40%

master, slave1	4	2048	80	2	2	0.49	1.171e+01	22.87%
master, slave1	6	2048	60	2	2	0.50	1.143e+01	22.32%
master, slave1	8	2048	80	2	2	0.54	1.056e+01	20.63%
master, slave1	32	2048	80	2	2	5.51	1.040e+00	2.03%
master, slave1,slave2	4	2048	80	2	2	2.78	2.059e+00	4.02%
master, slave1,slave2	6	2048	80	2	2	2.88	1.991e+00	3.89%
master, slave1,slave2	8	2048	80	2	2	3.08	1.861e+00	3.63%
master, slave1,slave2	32	2048	80	2	2	11.93	4.806e-01	0.94%

3. 完成上述测试后比较和分析上面的测试结果，特别是如何能够得到高的性能测试值

注意到，上述几组测试数据中获取的效率最佳值的 P/Q 值大多为 2/2 组合。由此可以得出，P/Q 值越相近，获取的测试结果会越佳。对于 N 而言，接近内存容量的取值会更佳。

但同时，还需注意到上述结果实际上是不符合预期的，多机效果显著低于预期。且在实验过程中，每次测试得到的结果都不尽相同，甚至差异较大。个人认为，这是由于实验环境搭建于虚拟机之上的缘故，CPU 资源取决于当前系统中的任务调度等。故得到的实验结果有较大的随机性，难以实现所预期结果。**特别是进程数如果大于 CPU 支持的最大进程数（本实验中为 4），那么会触发操作系统的进程调度，会大幅度降低性能，所以一般不要超过最大进程数，因此实验结果中往往是第一条进程数为 4 时的效率最高。**

而 Linpack 性能优化的理论参数设置如下所述：

- **矩阵规模 N**

矩阵的规模 N 越大，有效计算所占的比例也越大，系统浮点处理性能也就越高；但与此同时，矩阵规模 N 的增加会导致内存消耗量的增加，一旦系统实际内存空间不足，使用缓存，性能会大幅度降低。因此，对于一般系统而言，要尽量增大矩阵规模 N 的同时，又要保证不使用系统缓存。

考虑到操作系统本身需要占用一定的内存，除了矩阵 A（N×N）之外，HPL 还有其它的内存开销，另外通信也需要占用一些缓存（具体占用的大小视不同的 MPI 而定）。一般来说，矩阵 A 占用系统总内存的 80%左右为最佳，即 N×N×8＝系统总内存×80%。

- **矩阵分块大小 NB**

分块的大小对性能有很大的影响，NB 的选择和软硬件许多因数密切相关。NB 值的选择主要是通过实际测试得到最优值。但 NB 的选择上还是有一些规律可寻，如：NB 不可能太大或太小，一般在 256 以下；NB×8 一定是 Cacheline 的倍数等等。具体 N 最优选择还跟实际的软硬件环境密切相关。当整个系统规模较小、节点数较少、每个节点的内存较大时，N 可以选择大一点。当整个系统规模较大、节点数较多、每个节点的内存较小时是，N 可以选择大一点。

- **二维处理器网格 P×Q**

P×Q=系统 CPU 数=进程数。一般来说一个进程对于一个 CPU 可以得到最佳性能。对于 Intel Xeon 来说，关闭超线程可以提高 HPL 性能。P≤Q；一般来说，P 的值尽量取得小一点，因为列向通信量（通信次数和通信数据量）要远大于横向通信。P=2n，即 P 最好选择 2 的幂。HPL 中，L 分解的列向通信采用二元交换法（Binary Exchange），当列向处理器个数 P 为 2 的幂时，性能最优。例如，当系统进程数为 4，且问题规模较小的时候，P×Q 选择为 1×4

的效果要比选择 2×2 好一些，但当问题规模较小的时候，二者相差并不大，因为此时节点内的计算开销相比通信开销要大很多，所以网格的分布方式对整个性能的影响就比较小了。在集群测试中，P×Q=系统 CPU 总核数。

二、设计思考实验

1. 还有什么技术会影响测试结果，例如 sse2、超线程等，请设计实验。并详细书写实验的采用的库文件、Makefile 文件、测试结果、数据分析等。

- **超线程**

在 CPU 所采用的技术中，存在着一个相似的名词——超线程(HT, Hyper-Threading)。那么超标量与超线程是同一个概念吗？两者是不同的。

超线程是 Intel 公司提出的一种提高 CPU 性能的技术，可以将一个物理 CPU 当作两个逻辑 CPU 使用，使 CPU 可以同时执行多重线程，从而发挥更大的效率。超线程技术通过利用特殊的硬件指令，把两个逻辑内核模拟成两个物理芯片，让单个处理器都能使用线程级并行计算，进而兼容多线程操作系统和应用软件，减少 CPU 的闲置时间，提高 CPU 的运行效率。超线程技术原先只应用于 Xeon 处理器中，当时称为“Super-Threading”。之后陆续应用在 Pentium 4 HT 中。如今，几乎所有的 CPU 都是使用了这项技术。

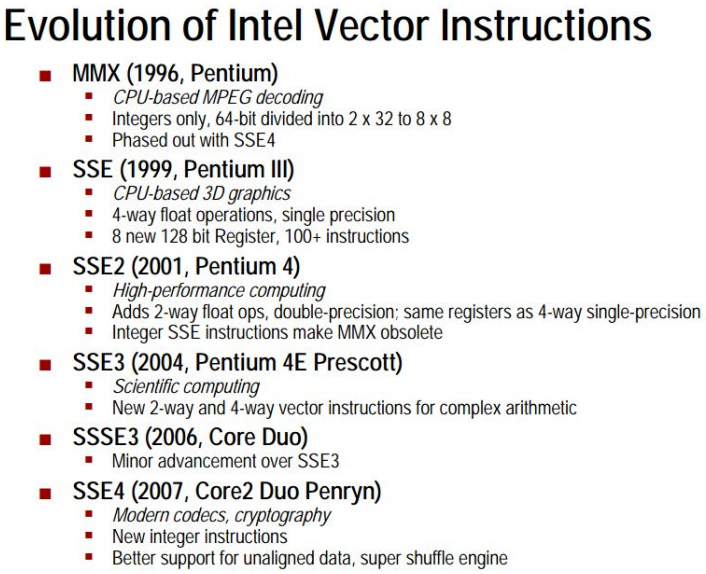
但是超标量并不是超线程。超线程要求同一个核心有两套执行部件的同时，还有两套保存线程状态的寄存器。超标量是没有的，所以它不能同时执行两个线程上的两个指令，只能执行同一个线程上的两个指令。这两个指令不必是连续，可以是 CPU 的硬件分析出来的一段代码中，两个互相不依赖结果的任意指令。也就是说，例如 ABCD 四条指令，要求 CPU 能够分析出 C 指令的执行不依赖 AC 的结果。

率。

- **SSE2**

SSE2 指令集(Streaming SIMD Extensions 2，Intel 官方称为单指令多数据流技术扩展 2 或单指令多数据流扩展指令集 2)是 Intel 公司在 SSE 指令集的基础上发展起来的。相比于 SSE，SSE2 使用了 144 个新增指令，扩展了 MMX 技术和 SSE 技术，这些指令提高了广大应用程序的运行性能。

随 MMX 技术引进的单指令多数据流整数指令从 64 位扩展到了 128 位，使 SIMD 整数类型操作的有效执行率成倍提高。双倍精度浮点（实数）单指令多数据流指令允许以 单指令多数据流格式同时执行两个浮点（实数）操作，提供双倍精度操作支持有助于加速内容创建、财务、工程和科学应用。除 SSE2 指令之外，最初的 SSE 指令也得到增强，通过支持多种数据类型（例如，双字和四字）的算术运算，支持灵活并且动态范围更广的计算功能。



SSE2 指令可让软件开发员极其灵活的实施算法，并在运行诸如 MPEG-2、MP3、3D 图形等之类的软件时增强性能。Intel 是从 Willamette 核心的 Pentium 4 开始支持 SSE2 指令集的，而 AMD 则是从 K8 架构的 SledgeHammer 核心的 Opteron 开始才支持 SSE2 指令集的。它有两个部分组：SSE 部分和 MMX 部分.SSE 主要负责处理浮点数，MMX 则专门计算整数.SSE2 的寄存器容量是 MMX 的两倍。寄存器存储的数据量也增加了两倍。在指令处理器速度保持不变的情况下，通过 SSE2 优化过的程序和软件运行速度也能提升两倍。由于 SSE 指令集和 MMX 指令集相兼容。因此，被 MMX 优化过的程序很容易被 SSE2 进行更深层次的优化，达到更好的效果。