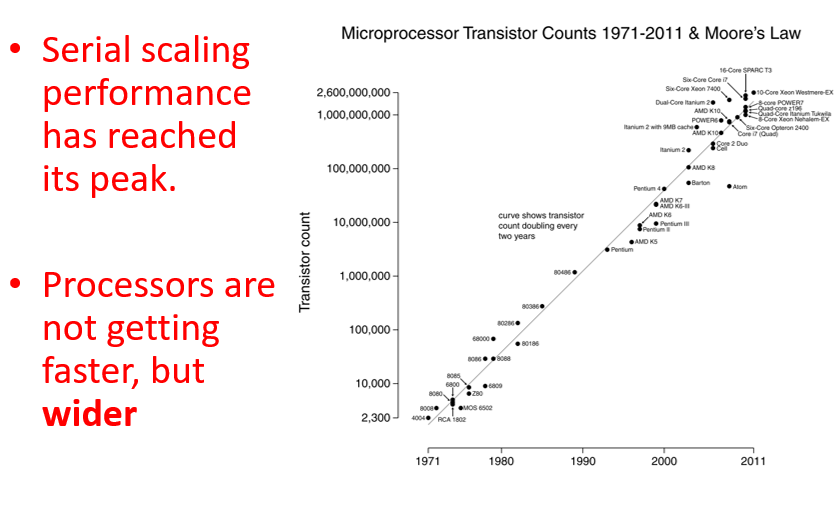
Moore’s Law: 摩尔定律即将终结

CPU的处理性能每两年提升一倍



Scientific computing demand：new application和more performance之间相互促进

Flynn’s Classical Taxonomy: 根据data stream和instruction stream的multiplicity进行分类

分为SISD(traditional sequential order)，SIMD(比如array processor)，MISD(并不常见，例子有systolic array-脉动阵列)和MIMD(distributed memory或者SMP架构)

Modern Classification: Sima, Fountain, Kacsuk

Classification based on how Parallelism is achieved---又可以分为data parallelism和function parallelism

Data parallelism: Vector processors, SIDM(or array processors), systolic arrays

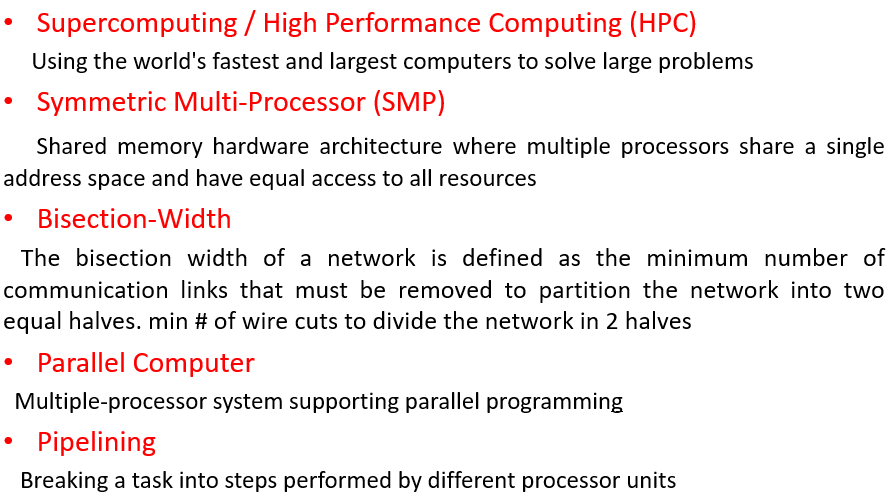
Function parallelism:

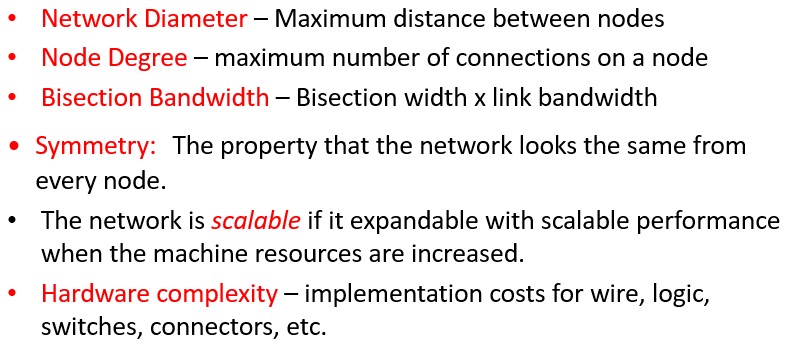
Instruction Level Parallelism(ILP) Architecture，细分为pipelined processors, VLIWS, 超标量

Thread Level Parallelism Architecture

Process Level Parallelism Architecture(即MIMD), 细分为distributed Memory MIMD & Shared Memory MIMD

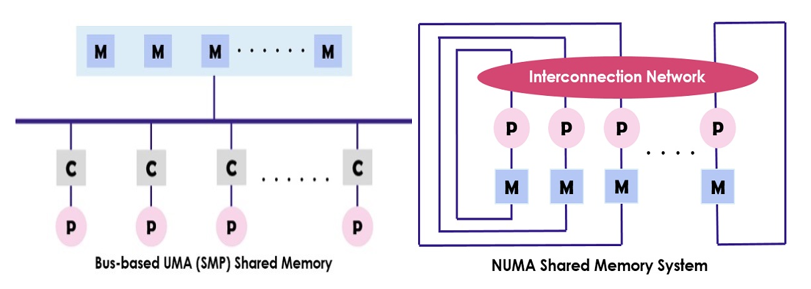
一些术语：第二道题考术语





Parallel Computer Memory Architectures：

Shared memory---UMA(Uniform Memory Access) & NUMA(Non-Uniform Memory Access)



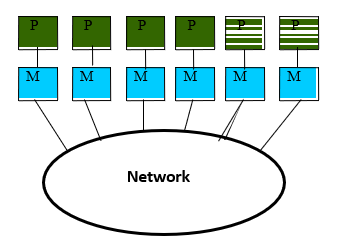
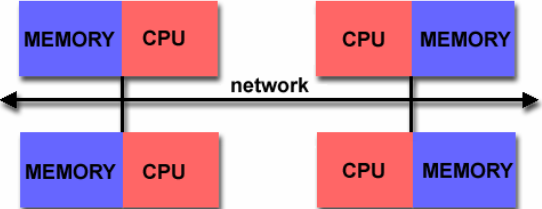
Symmetric Multi-Processor(SMP): Shared memory hardware architecture where multiple processors share a single address space and have equal access to all resources, UMA的一种

Cache coherent的意思是如果一个processor更新了shared memory中的一个位置，则所有processor都可以知道这个update。如果能够保证cache coherent，则可以称为CC-UMA和CC-NUMA

Shared memory的好处在于对于编程者来说非常方便，因为可以使用global address；data sharing between task is fast and uniform due to proximity of memory to CPUs

缺点在于memory和CPU之间的scalability：增加CPU会极大地增加共享的memory-CPU path上的traffic；访问数据时的consistency需要由编程者自己保证；贵

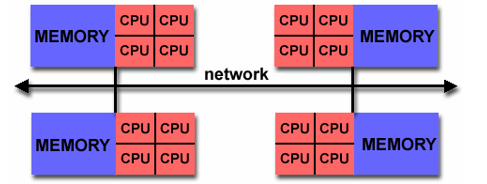
Distributed Memory:

Distributed memory的优点是：memory大小和processor的数量成比例增长---scalable；processor访问内存的速度很快，并且不需要担心和别的处理器之间的interference和cache coherence；可以使用商品化的处理器，便宜

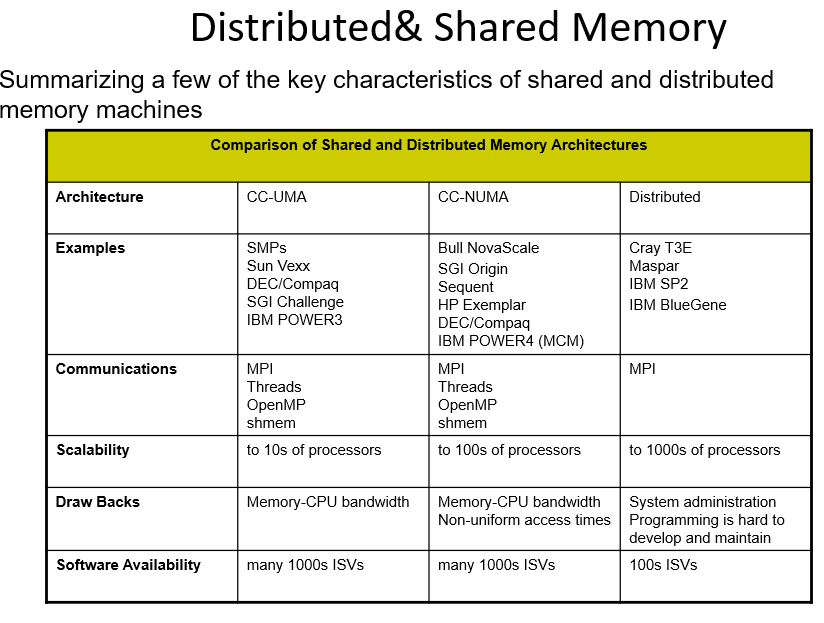
缺点是：编程者需要负责进程之间的通信；从现存的global memory到distributed memory的数据结构的mapping可能会很复杂；memory access time is not uniform

Hybrid distributed-shared memory machine:



Granularity: 粒度，描述的是计算和通信的比值

Embarrassingly Parallel: 解决类似的但是independent的tasks，不太需要task之间的合作

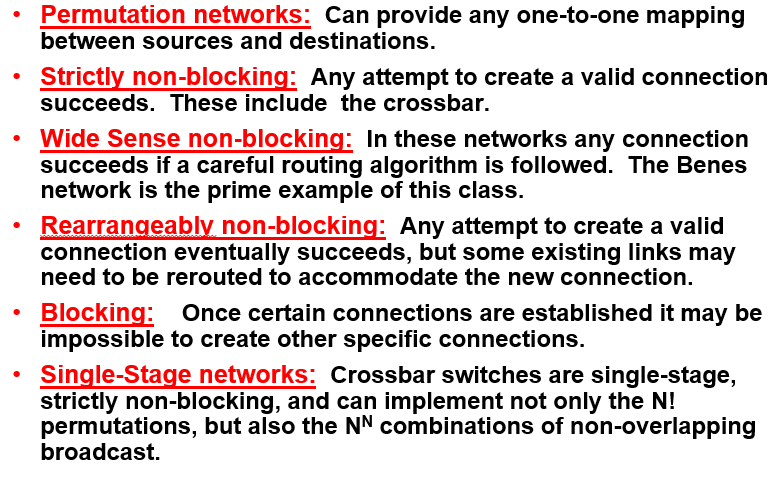


Parallel architecture performance metrics:

通过MFLOPS来测量，分为peak performance(通过硬件测量)和sustained performance(通过benchmark进行测量)，记efficiency rate为sustained/peak

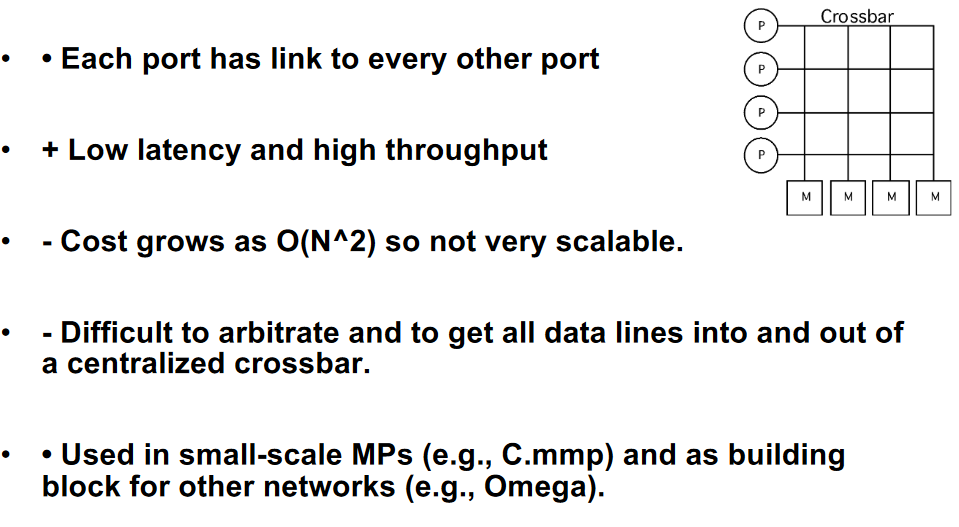
Benchmarks：用于测量性能的program---LINPACK(Linear System Package)

Dynamic network definitions：



包括：Bus systems，Multi-stage Interconnection Networks(MINs)---Omega network, Baseline Network, Butterfly Network，Single-Stage Crossbar Switch Networks

Crossbar：相当于一个N×N的switch：可以通过打开不同的点实现不同的port相连



根据不同的实现方法复杂度也不同：正常的是O(N2)，如果是implemented in stages(通过多个小的switch分步实现)则为O(N LogN)

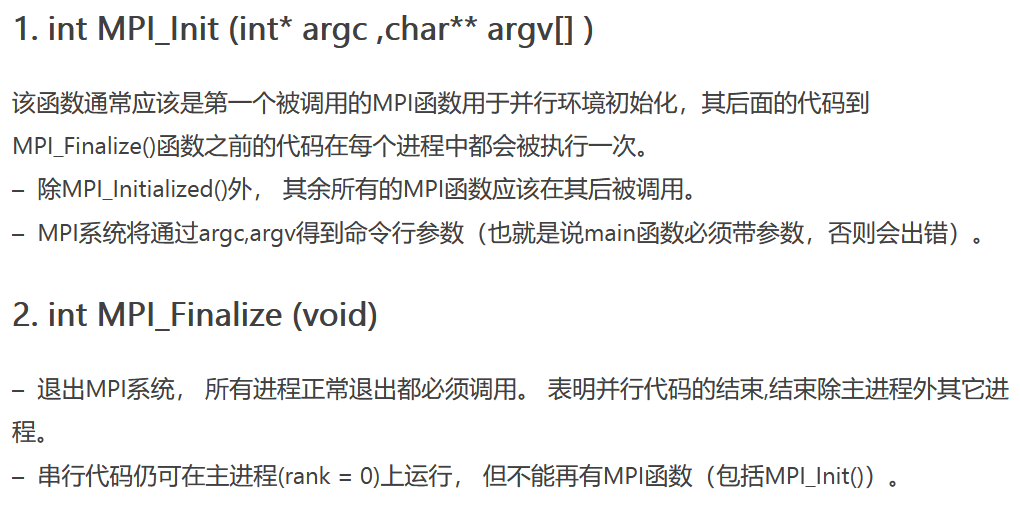
HIPPI(High Performance Parallel Interface):

MPI和OpenMP的区别---OpenMP使用的是threads model；MPI使用的是message passing model

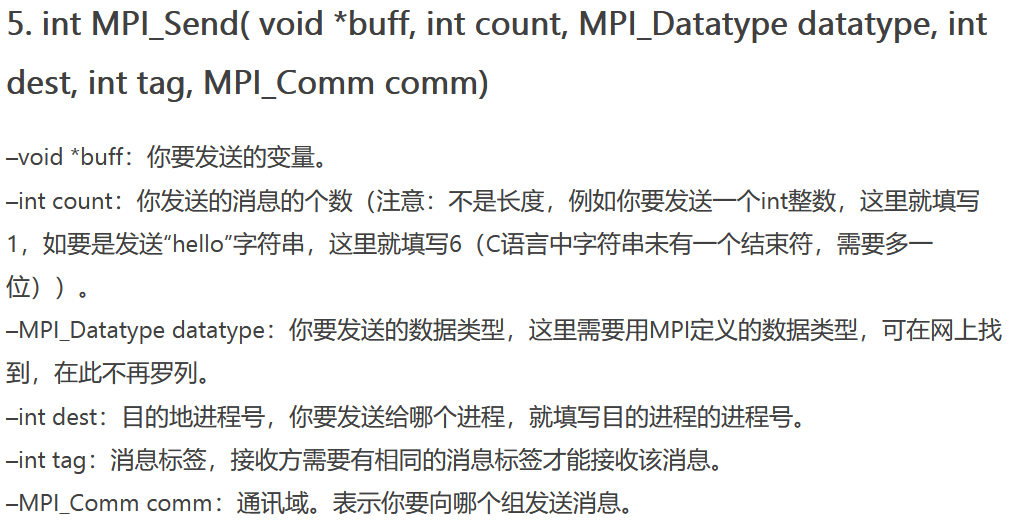
MPI：

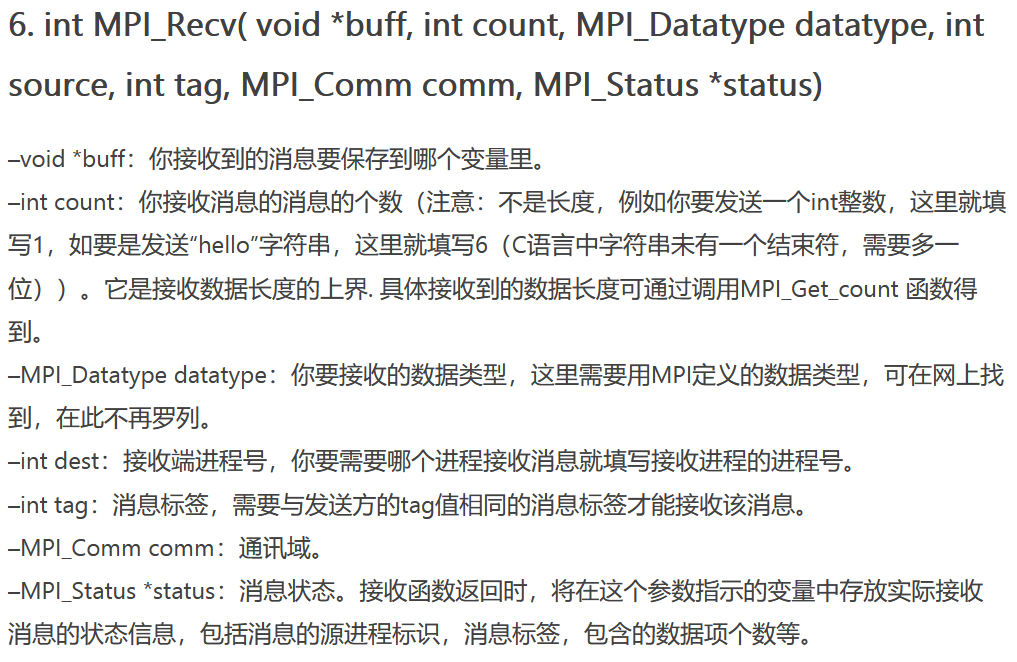
一个process通常来讲包含address space和program counter，其中可以包含多个共享address space的threads。MPI提供的是进程间的通信，inter process communication主要包括两部分：synchronization和address space之间的数据传递

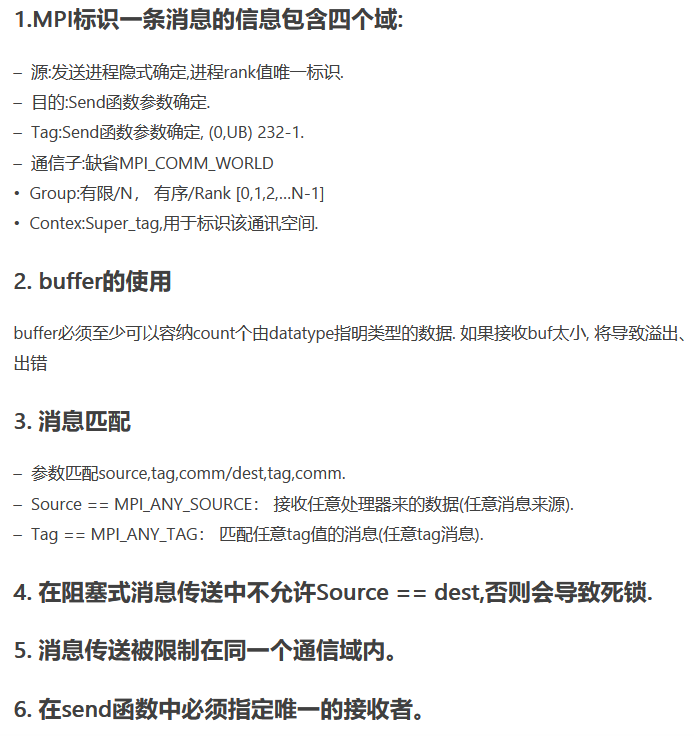
MPI中六个函数的用法：要考手写代码



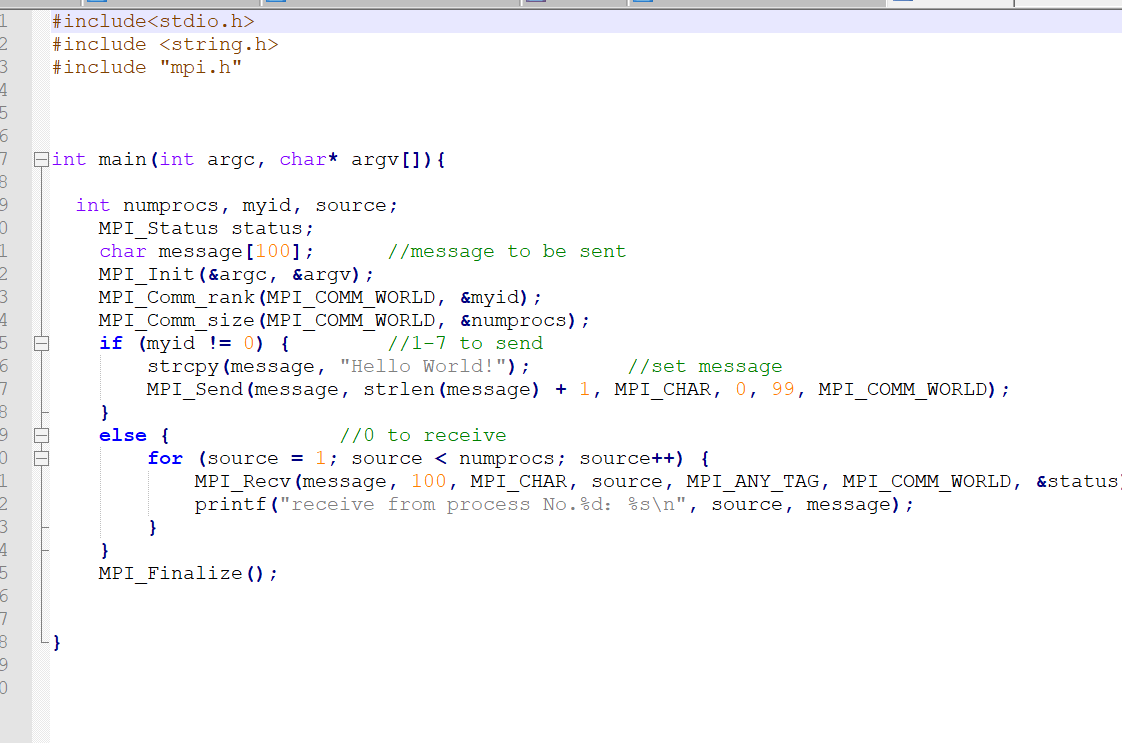








写法参照如下：



Deadlock的出现：

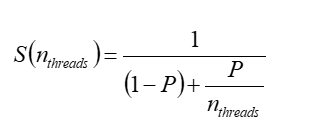
通过几种方法解决：ordering operations；use non-blocking methods；

编译MPICC程序：mpicc test.c -o test

运行：mpirun -np N test

计算speed up ratio的两种方法

Amdahl’s Law：speed up ratio = 加速前时间/加速后时间，P为并行比例



Amdahl主要描述了当对系统的某个部分进行加速时，其对于系统整体性能的影响取决于该部分的重要程度和加速程度。当N趋近于正无穷时，可以化简为1/(1-P)，即被加速的部分速度快到可以忽略，此时加速比取决于并行的程度

Gustafson’s Law：由于我们经常使用高性能的计算机来处理一些large sized problem，阿姆达尔定律显得有些上限不足，因此有了古斯塔夫森定律---把运行的时间当成是常量并且允许问题的大小随着处理器的数量增加

公式记为speedup ratio = p + (1-p) \* N

p为串行的比例，N为并行处理器的个数：当串行比例趋近于0(也就是并行比例接近于1，即并行程度足够)时，加速比接近处理器个数

两者的区别在于Amdahl是根据fixed problem size来计算加速的，而Gustafson是根据fixed time来进行计算的，从不同的角度进行的分析

与speedup ratio相关的概念还有efficiency：将得到的speedup ratio / number of processors并且转换为百分比就可以得到efficiency