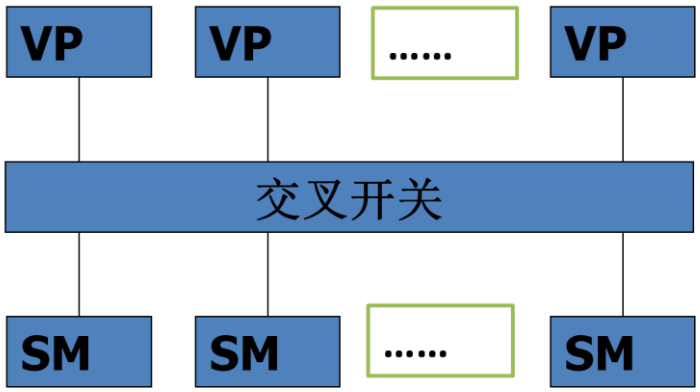
目录

第一章 并行计算体系结构

1.1 并行计算机系统结构模型

1.1.1 并行向量处理机（PVP）

- 属于 MIMD（Multiple-Instruction Multiple-Data）、UMA （Uniform Memory Access）型的细粒度并行计算机
- 少量的高性能向量处理器，处理能力≧1G flops
- 专用高宽带交叉开关实现存储器之间的互联
- 大量的共享存储器模块（SM）
- 大量向量寄存器和指令缓冲器,不使用高速缓存。
- 机型 Cray C-90/T-9,NECSX-4,Gralaxy-1,Cray-1
- 典型结构：



- 实例 Cray-1,组成如下：
  - 中央处理器，含运算控制部件，指令缓冲器，指令控制部件和寄存的功能部件。
  - 存储器（内存）

- 交互通道，连接诊断维护控制机，磁盘存储器（SM）前端机（用户机）
- 向量流水部件，含 8*64 个向量寄存器，但 V0-V7 配向量加和浮点加部件，标量寄存器组 S0-S7

- 4 种向量运算指令
  - 源向量取自两个向量寄存器组

    V_j op V_k
  - 源操作数之一取自标量寄存器组

    V_j op S
  - 主存储与向量寄存器之间数据传送

    Mem op V_j

    V_j op mem
- 并行要求
  - 无向量冲突

    V4→V1+V2　　V1 发生源向量冲突

    V5→V1*V3
  - 无功能部件冲突

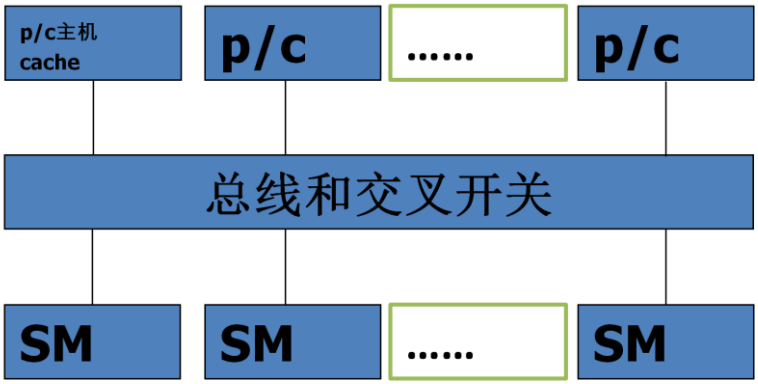
    V4→V1*V3　　发生乘部件冲突

    V5→V2*V6

1.1.2 对称多机系统（SMP）

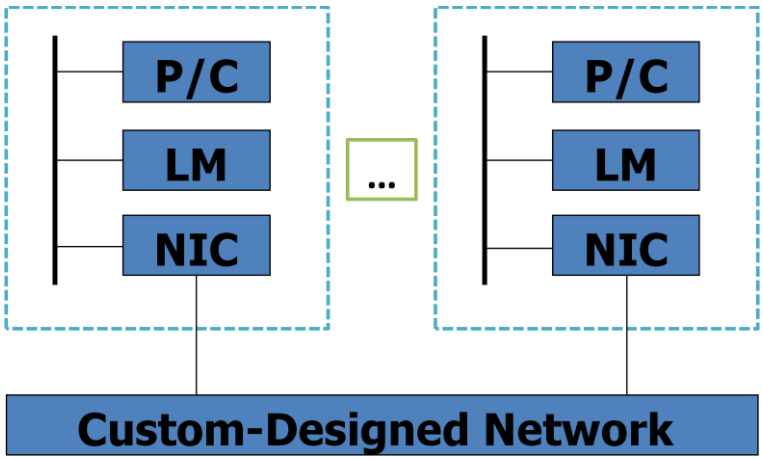
- 属于 MIMD,UMA,中粒度，高级别并行多机系统
- 具有可插拔的 Cache 芯片的商用多机系统
- 集中式共享存储器
- 对称性：每个处理机关于 I/O 操作，OS 服务和 SM 的权限是等同的
- 可扩展行受到 SM 和总线的限制

- 机型：SGI 工作站，DEC 服务器 8400，Dawning-1 等
- 典型结构：



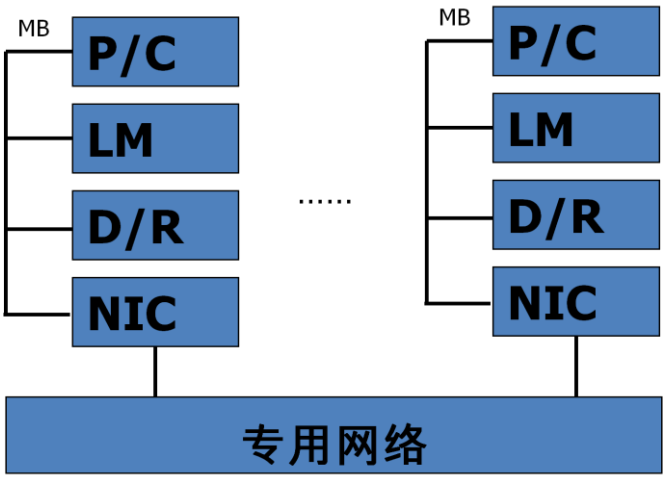
1.1.3 大规模并行处理机（MPP）

- 属于 MIMD ，NUMA 中/大粒度多处理机
- 采用专用的高宽带低延时的通信网络
- 物理上分布的存储器
- 进程间采用阻塞报文交互操作（同步） 处理机级、任务级（异步）
- 机型：Intel Paragon，IBM SPQ，Dawning 1000
- 典型结构：



1.1.4 分布式共享存储器多机系统（DSM）

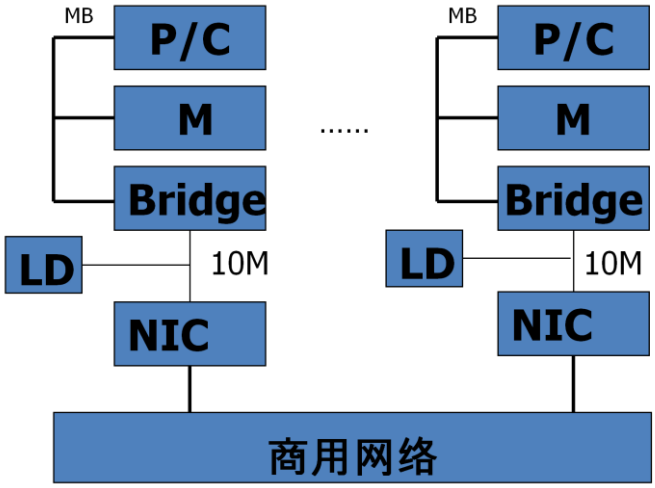
- 属于 MIMD，NUMA，NORMA，大粒度多机系统（No-Remote Memory Access）
- 对用户来讲，是一个物理上分布的，逻辑上是共享的一个由硬件支持的单一的存储器地址空间。
- 采用基于 DIR（Cache 目录）的 Cache 一致性机制
- 采用专用通信网络
- 可使用共享存储器编程模式
- 机型 Stanford DASH, Gray T30
- 典型结构：



1.1.5 工作站机群（COW）

- 属于 MIMD NUMA 粗粒度多机系统
- 分布式存储器
- 每个节点是一套完整的计算机系统（SMP 或 PC）
- 采用低成本的商品网络互连结构
- 每个节点拥有本地磁盘和完整的 OS（MPP 只有内核）
- 机型：Berkeley NoW，Alpha Farm, FXCOW 等

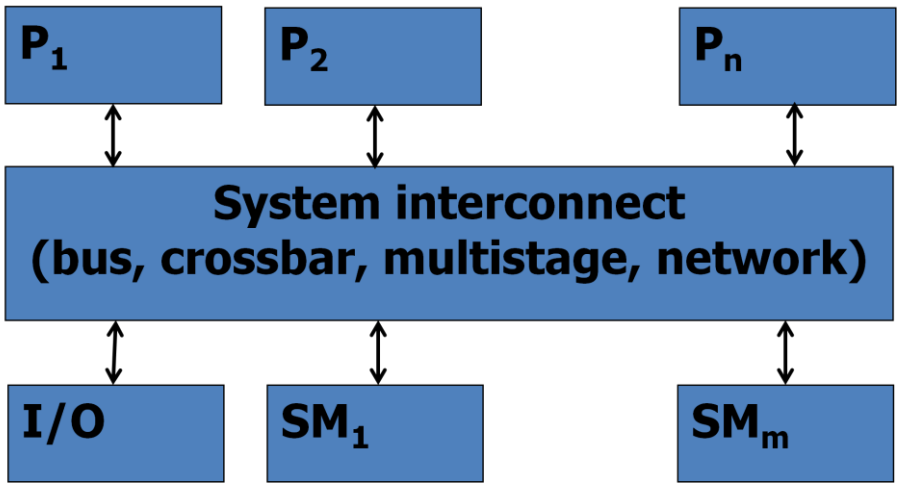
- 典型结构：



1.2 访存模型

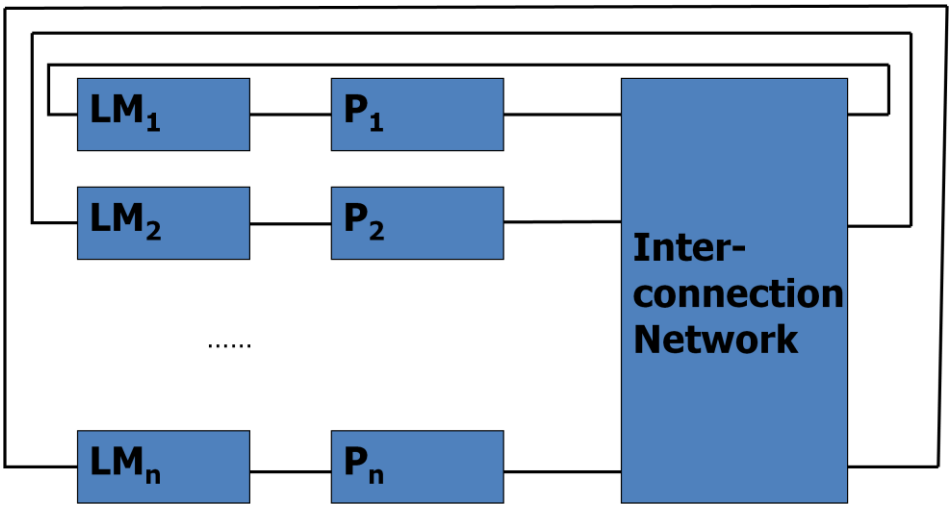
1.2.1 均匀访存模型（UMA）

- 物理存储器被所有处理机均匀共享
- 所有处理机访存时间相同
- 适于通用的或分时的应用程序类型
- 模型：



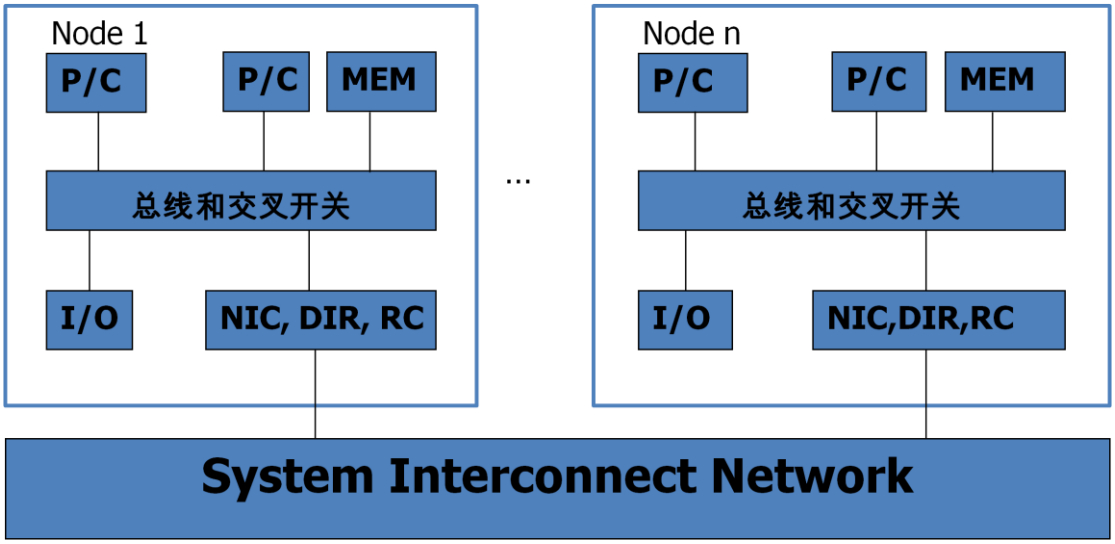
1.2.2 非均匀访存模型（NUMA）

- 是所有处理机的本地存储器的集合
- 访问本地 LM 的访存时间较短
- 访问远程 LM 的访存时间较长
- 模型：



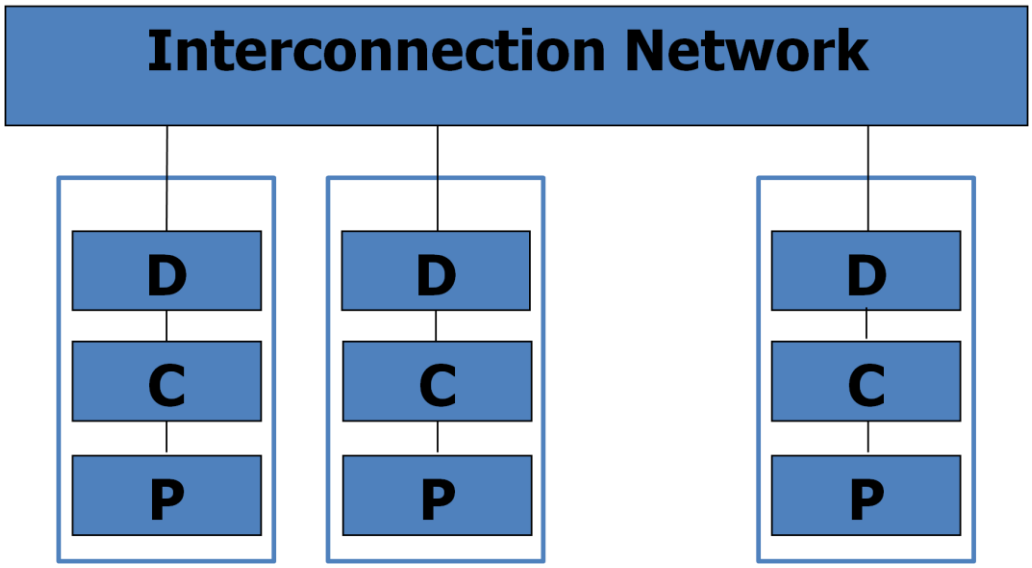
1.2.3 Cache 一致性非均匀访存模型（CC-NUMA）

- DSM 结构
- 模型：



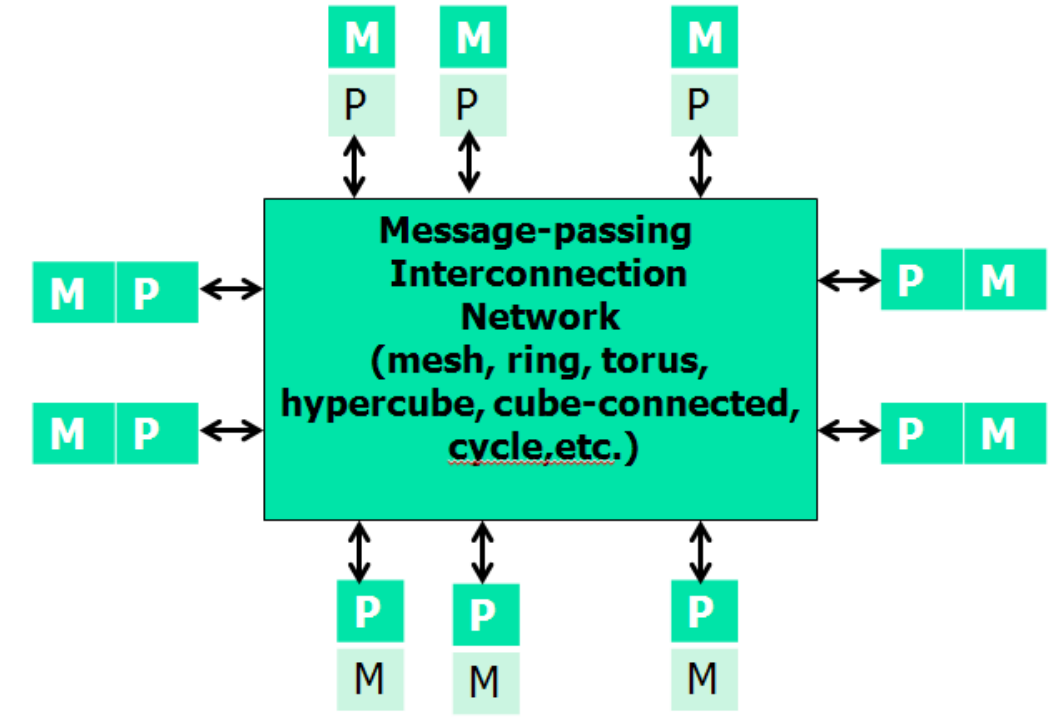
1.2.4 全局 Cache 访存模型（COMA）

- 是 NUMA 的一种特例，是采用各处理机的 Cache 组成的全局地址空间
- 远程 Cache 的访问是由 Cache 目录支持的
- 模型：



1.2.5 非远程访存模型（NORMA）

- 在分布式存储器多机系统中，如果所有存储器都是专用的，而且只能被本地存储机访　问，则这种访问模型称为 NORAM
- 绝大多数的 NUMA 支持 NORAM
- 在 DSM 中，NORAM 的特性被隐匿的
- 模型：

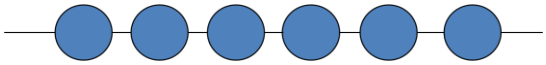


1.3 并行计算机系统互连

1.3.1 静态互连网络

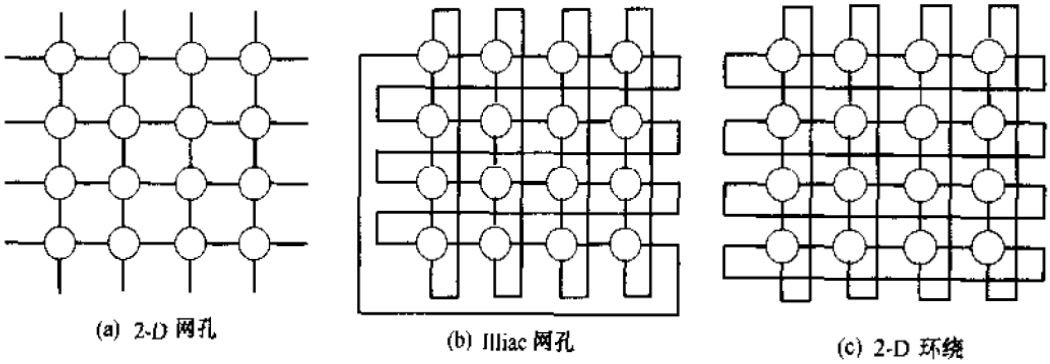
- 定义
  - 静态互连网络：处理单元之间有固定的连接，在程序执行期间，这种点到点的链接保持不变。
  - 节点度：射入或者射出一个节点的边数称为节点度（Node Degree）
  - 网络直径：两个节点之间的最大距离，即最大径数称为~（Network Diametre）
  - 对剖宽度：对分网络各半所必须移去的最小边数称为对剖宽度（Bisection Width）
  - 对称网络：如果从任意节点上观看网络都是一样的，则称为对称网络（Symmetry Network）

- 拓扑结构 1（一维线性阵列）

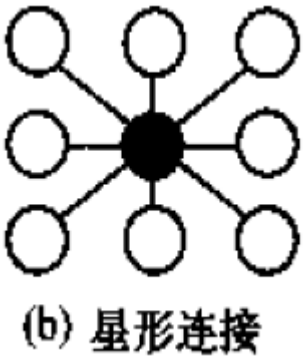


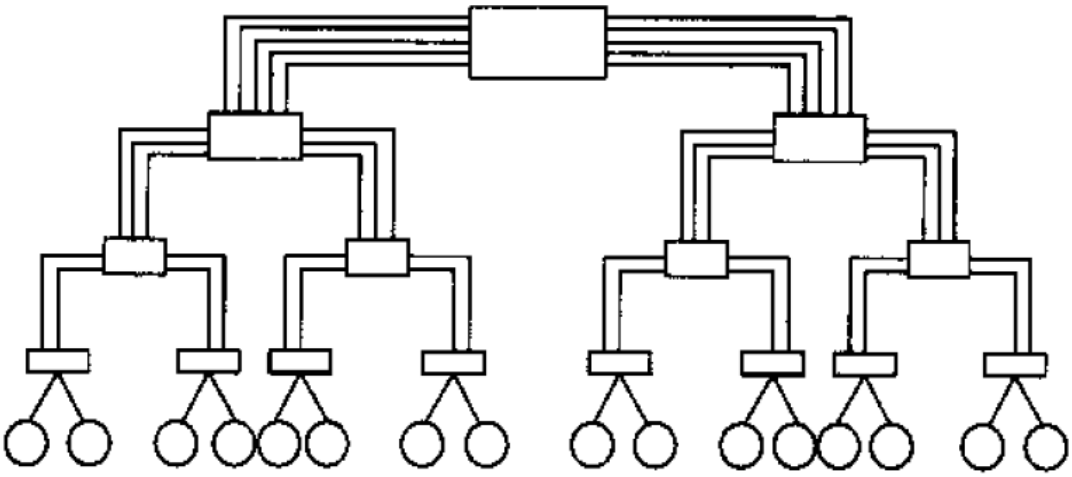
  - 只与左右近邻节点相连
  - N 个节点用 N-1 条边
  - 节点度为 2
  - 网络直径为 N-1
  - 对剖度为 1
  - 首尾连接时构成环（单向或双向）

- 拓扑结构 2 （四近邻连接）



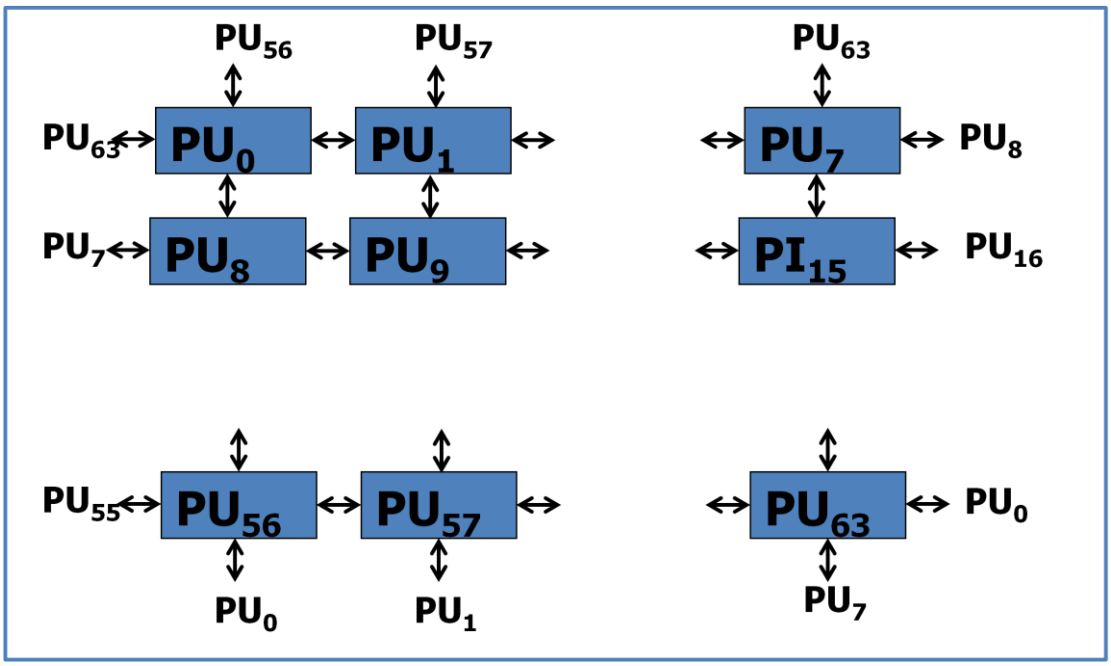
- 拓扑结构 3（树形连接）





(c) 二叉胖树

- 实例 64 个处理单元的 Illiac IU 型处理器



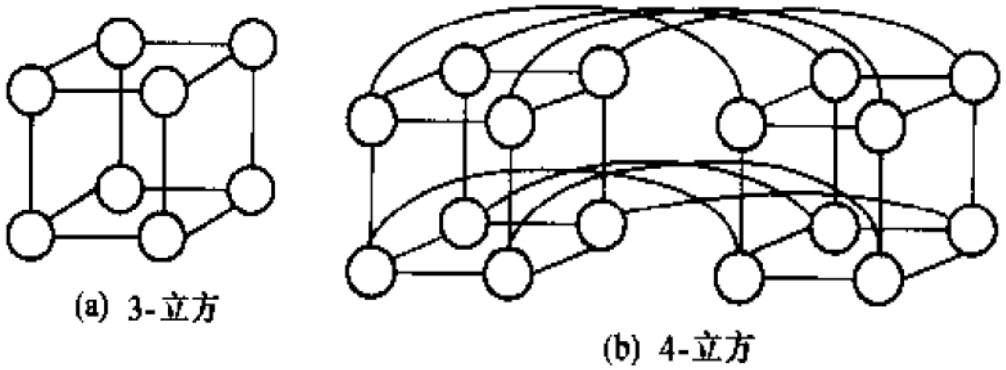
- 说明
  - $\sqrt{N} * \sqrt{N}$ =8 * 8
  - 节点度 4
  - 网络直径 $\sqrt{N}$-1=7
  - PU63->PU10：PU63->PU7->PU8->PU9->PU10

    或 PU63->PU10->PU8->PU9->PU10

- PU9->PU45：PU9->PU1->PU57->PU56->PU48->PU47->PU46->PU45

  或：PU9->PU1->PU57->PU49->PU48->PU47->PU46->PU45
- 对剖宽度2根下N=16（横向连接8根， 蛇形连接8根）

拓扑结构 4（超立方连接）



(a) 3-立方　　(b) 4-立方

静态互联网络特性一览表

表 1.3　静态互连网络特性一览表

网络名称	网络规模	节点度	网络直径	对剖宽度	对称	链路数
线性阵列	N 个节点	2	$N-1$	1	非	$N-1$
环形	N 个节点	2	$\lfloor N/2 \rfloor$(双向)	2	是	N
2－D 网孔	$(\sqrt{N}\times\sqrt{N})$个节点	4	$2(\sqrt{N}-1)$	$\sqrt{N}$	非	$2(N-\sqrt{N})$
Illiac 网孔	$(\sqrt{N}\times\sqrt{N})$个节点	4	$\sqrt{N}-1$	$2\sqrt{N}$	非	$2N$
2－D 环绕	$(\sqrt{N}\times\sqrt{N})$个节点	4	$2\lfloor \sqrt{N}/2 \rfloor$	$2\sqrt{N}$	是	$2N$
二叉树	N 个节点	3	$2(\lceil \log N \rceil - 1)$	1	非	$N-1$
星形	N 个节点	$N-1$	2	$\lfloor N/2 \rfloor$	非	$N-1$
超立方	$N=2^n$ 个节点	n	n	$N/2$	是	$nN/2$
立方环	$N=k\cdot 2^k$ 个节点	3	$2k-1+\lfloor k/2 \rfloor$	$N/(2k)$	是	$3N/2$

第二章 并行计算性能评价

2.1 加速比性能定律

2.1.1 Amdahl 定律

- 在实时性要求很高的应用类型中，计算负载**W**固定不变，随着处理器的数目的增加，计算时间将同时缩短。
- 这里的计算负载**W**包含可并行化部分，即并行分量**Wp**和串行分量**Ws**。即 **W=Wp+Ws**
- 加速比 $S = \frac{\text{最快的串行算法最坏的运行时间}}{\text{并行算法最快的运行时间}}$

- 假设：
  - 串行比例因子
    - $f = W_S/W$
  - 并行比例因子
    - **1-f**

- 所以加速比**S**为：
  - $S=\frac{Ws+Wp}{\frac{Wp}{p}+Ws}$

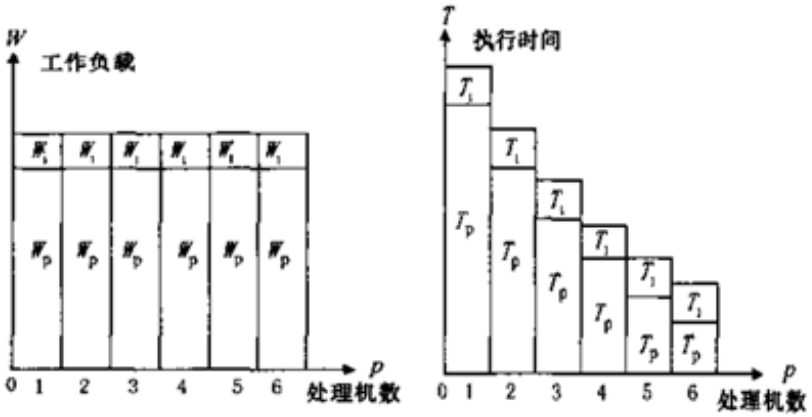
    = (f+(1-f))/(f+(1-f)/p)

    =p/(1+f(p-1))

    =1/f (p->∞)

当处理机数无限增加时，加速比仅为**1/f**

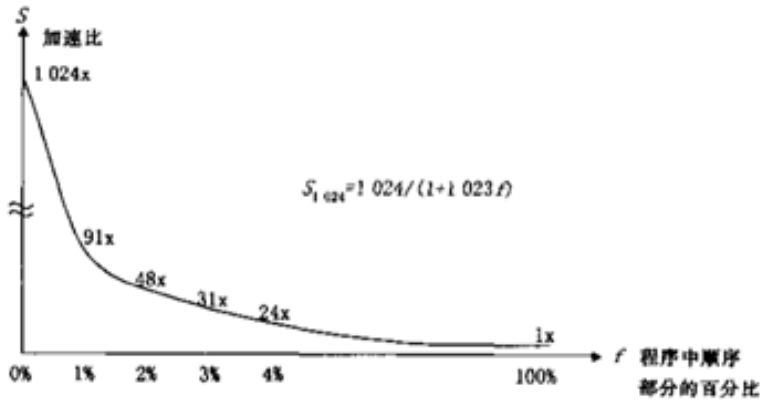
- **Amdahl**几何意义



计算负载固定不变

处理机数增加，执行时间缩短

- **Amdahl几何意义**



- 影响加速比的因素：
  - 求解问题中的串行分量
  - 并行处理所引起的额外开销（通信，等待，冗余操作等）
  - 处理机数目的增加超过了算法中的并发程度
- 增加问题的规模有利于提高加速比的因素：
  - 加大问题的规模可提供较高的并发程度
  - 额外开销的增加可能慢于有效计算的增加
  - 算法中的串行分量的比例不是固定不变的

2.2 可扩放性

2.2.1 概念

- 什么是可扩放性？

一个计算机系统（硬件、软件、算法、程序等）被称为可扩放的，是指其性能随处理机数目的增加而按比例提高。例如，工作负载能力和加速比都可随处理机的数目的增加而增加。

- 可扩放性包括哪些方面?
  - 机器规模的可扩放性
    - 系统性能是如何随着处理机数目的增加而改善的
  - 问题规模的可扩放性
    - 系统的性能是如何随着数据规模和负载规模的增加而改善
  - 技术的可扩放性
    - 系统的性能上如何随着技术的改变而改善
- 可扩放性研究的目的是什么？
  - 确定解决某类问题时何种并行算法与何种并行体系结构的组合，可以有效的利用大量的处理器；
  - 对于运用于某种并行机上的某种算法，根据在小规模处理机的运行性能预测移植到大规模处理机上的运行性能
  - 对固定问题规模，确定最优处理机数和可获得的最大的加速比
  - 指导和改进并行算法和并行处理机结构，充分利用处理机可扩放性的度量标准
    - ISO-efficiency 等效度量标准
    - ISO-speed 等速度量标准
    - Aerage Lantency 平均延迟度量标准

2.2.2 等效率度量标准（ISO-efficiency）

- 基本概念
  - 等效度量标准是研究如何维持并行系统的等效性
- 推导

- 设 T1 是一个给定问题在一台机器上串行执行的时间（例如 W），Tp 是在 p 台处理机上并行执行的时间，T0 是额外开销

- 结论
  - 如果问题规模 W 不变，那么随着处理机数 P 的增加，额外开销 T0 也会增加，从而引起效率下降
  - 为了保证 E 不变，就要保证 $\frac{T0}{T1}$ 不变，这就要求增加处理机数 p 的同时，要同时增加问题规模 W，即 T1
  - 依此定义的函数称为等效率函数
- 优点
  - 等效率函数是一种用分析方法处理工作负载增长率与处理机增长率之间关系的有用的工具，可用简单的、可定量计算的、少量的参数就能计算出等效率函数，并由其复杂性可指出算法的可扩放程度
    - 如果 W 与 p 呈线性关系，则系统是可扩放的
    - 如果 W 与 p 呈指数关系，则系统是不可扩放的
- 缺点
  - 对共享存储器结构的机器难以计算等效率函数值

2.2.3 等速度度量标准（ISO-speed）

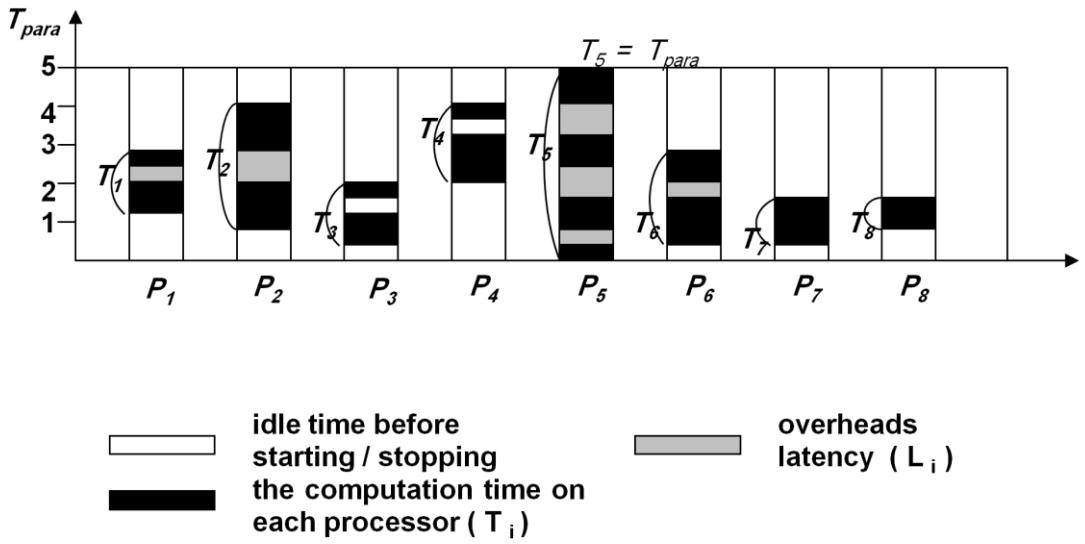
- 基本概念
  - 等速度标准时在机器规模由 p 增加到 p’，问题规模由 W 增加到 W’时，维持平均速度不变
- 推导
  - 设平均速度 $\overline{V} = \frac{W}{p * Tp}$，又设，W 是使用 p 个处理机时算法的工作量，W’表示当处理机数从 p 增加到 p’时，为了保持整个系统的平均速度不变需执行的工作量

- 结论
  - 如果速度能与处理机的数目的增加而线性增加，即意味着平均速度不变，则说明此系统具有很好的扩放性
- 优点
  - 使用机器性能速度指标这一明确的物理量来度量可扩放性是比较直观的（速度常被用来测量浮点运算）
    - 速度是由工作负载 W 和执行时间 T 决定的，而 W 反映了应用程序的性质，T 反映了结构和程序效率的影响
    - 速度在各种结构的机器之间具有可比性
    - 执行时间包含了计算和延迟这两个主要的时间量
    - .速度是比较容易测量的。（如何使用浮点操作数量）
- 缺点
  - 某些非浮点运算可造成性能的变化
  - 延迟虽包含在执行时间中，但它明确地定义为 W 的函数

2.2.4 平均延迟度量标准（Aerage Lantency）

- 基本概念
  - 效率不变前提下，用平均延迟来标志处理机数 p 和工作量 W 之间的增量关系。平均延迟时间定义为一个处理机完成分配给他的任务所需要的平均时间开销。包括运行时的延迟 Li，启动时间及停止时间。

    因此第 i 个处理器 Pi 的总的延迟时间为： Li ＋ 启动时间 +停止时间



- 优点
  - 平均延迟是一个实验性标准，可精确地评估低档系统
- 缺点
  - 需要特殊的硬件和系统软件进行实验测试

2.2.5 小结

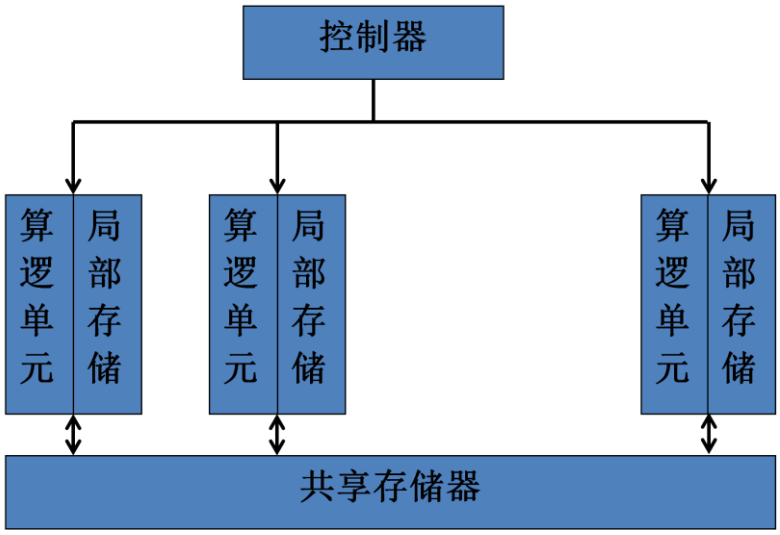
- 等效标准与等速标准是等价的
- 平均延迟标准可以源自等速标准
- 三者均是等价的

第三章 并行算法的设计基础

3.1 并行计算模型

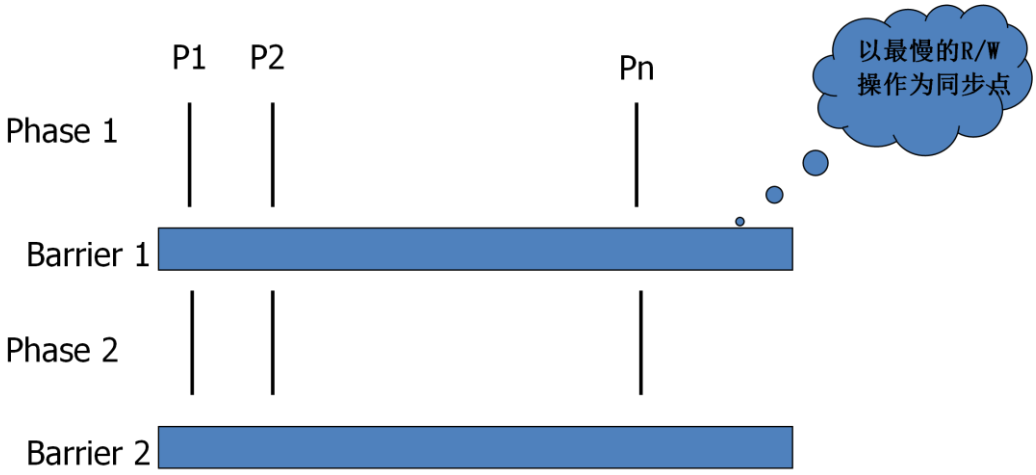
3.1.1 PRAM：SIMD-SM

- 基本概念：
  - PRAM（并行随机访问机器）模型也称为 SIMD-SM 模型，用于细粒度并行计算；
  - 采用集中式共享存储器模式，单一的编程访问空间；
  - 隐式同步机制
- 优点：
  - 适于表示和分析并行计算的复杂性；
  - 隐匿了并行计算机的大部底层细节（如通信、同步），从而易于使用。
- 缺点：
  - 不适于 MIMD 计算机，存在存储器竞争和通信延迟问题。
- 模型图示

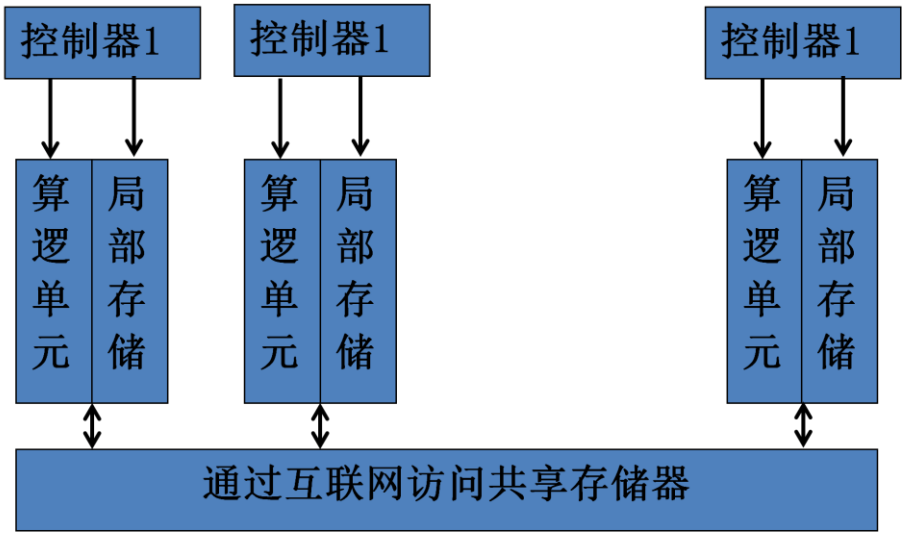


3.1.2 APRAM：MIMD-SM

- 基本概念
  - APRAM 模型也称为分段（phase）PRAM 模型；
  - 用于中粒度的并行计算；
  - 采用集中式共享存储器，单一的访问地址空间；
  - （进程间）异步操作，但读/写共享变量操作采用显示同步方式。
- 计算模式
  - 计算由若干个用同步点（barrier）划分的段组成；
  - 每一段异步运行局部程序；
  - 读/写操作在同步点进行同步。
- 优点：
  - 保存了 PRAM 的简单性；
  - 可编程性和可调试性（correctness）好；
  - 易于进行程序复杂性分析。
- 缺点：
  - 不适于具有分布式存储器的 MIMD 计算机。
- 计算模式图示



- 模型图示

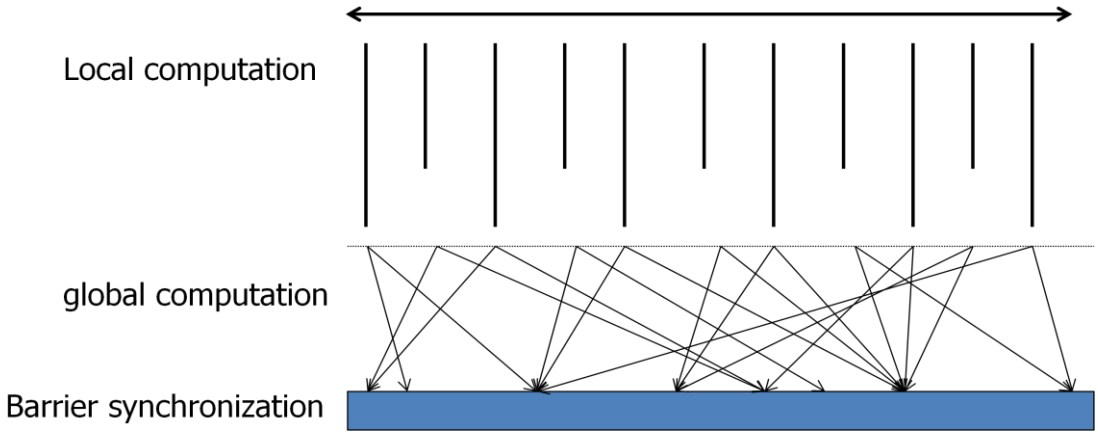


3.1.3 BSP：MIMD-DM

- 基本概念：
  - BSP 模型是一种分布式存储器的多处理机模型，又称大同步模型；
  - 用于中大粒度并行计算；
  - 进程间）异步操作；
  - 采用报文发送和接收的通信方式进行显示同步
- 参数和计算模式

- BSP 把并行计算机抽象为 3 个参数：P（处理机),g(宽带因子）和 l（同步间隔）；
- 计算由同步点（barrier）划分为若干个 Supersteps；
- 每个 Superstep 中实现异步的局部计算；
- 在同步点通过发送和接收 h-message 进行同步。

- 优点:
  - 把计算和通信分割开来；
  - 使用 hashing 自动进行存储器和通信管理；
  - 提供了一个编程环境。
- 缺点：
  - 显式的同步机制限制并行计算机数据的增加；
  - 在一个 Superstep 中最多只能传递 h 各报文。
- BSP 编程模式



3.1.4 LogP：MIMD-DM

- 基本概念：
  - LogP 模型是技术趋势，编程经验和现行理论的综合产物；
  - 用于大粒度并行计算；

- 使用分布式存储器的单一的单一的和多重访问地址空间；
- 进程间异步操作；
- 采用报文通信方式隐式实现同步操作，即子集同步。

- 参数和计算
  - LogP 模式把通信网络抽象为 3 个参数：L（网络延时），O（通信开销），g（网络带宽）；
  - 计算过程有若干 superstep 组成；
  - 在每个 superstep 中异步地实现局部计算并通过发送/接收 L/g 报文进行同步。
- 优点：
  - 可捕捉并行计算机的（同步）通信瓶颈（通过发送或接收 L/g 个报文）；
  - 可隐匿拓扑结构，路由算法和网络协议的细节；
  - 可用于共享变量，报文传递和数据并行处理等方案。
- 缺点：
  - 受限于网络的通信能力（当进行处理机数量扩充时）；
  - 难以计算同步开销和进行算法描述和设计。
    - 注：优点中的“捕捉同步通信开销”是指当处理机数一定的情况下，通过发送/接收关于 L/g 参数的报文来获取网络通信量情况，并避免拥塞；
    - 缺点中“受限于网络通信能力”是指当处理机数量增加时，获取 L/g 参数的通信开销也要增加，反过来要现在处理机数量的（无限制）增加。
- BSP 与 LogP 的比较：
  - LogP 是具有子集同步方式的 BSP（bulk superstep-subset）；
  - BSP=LogP+Barrier-Overhead。
  - BSP 可用模拟 LogP，呈线性下降关系；

- LogP 可用模拟 BSP，呈对数下降关系。
- BSP 为算法设计和编程提供了方便使用的抽象，易于编程；
- LogP 提供更好的机器资源的控制，但算法和程序的正确性分析较复杂。

3.1.5 C^3 模型

- 基本概念：
  - C^3 (Computetion, Communication, Congestion)模型也是一种分布式存储器的多计算机模型；
  - 用于粗粒度并行计算；
  - 具有多重访问空间；
  - 可实现异步操作；
  - 使用报文传递方式通信；
  - 强调由网络链接和处理机链接引起的拥塞。
- 参数和计算：
  - C^3 模型把网络操作抽象为 3 个参数：l（报文长度），S（启动时间），h（通信跳数），借助这 3 个参数可计算 C_l（链接拥塞）和 C_g（处理机拥塞）；
  - 使用 Barrier 把计算分为 Supersteps；在每个 superstep 中实现局部的异步计算和点一点的报文传递。
- 优点：
  - 考虑了一对一和一对多的通信方案细节；
  - 反应了受拥塞影响的计算性能。
- 缺点：
  - 模型的参数较复杂；

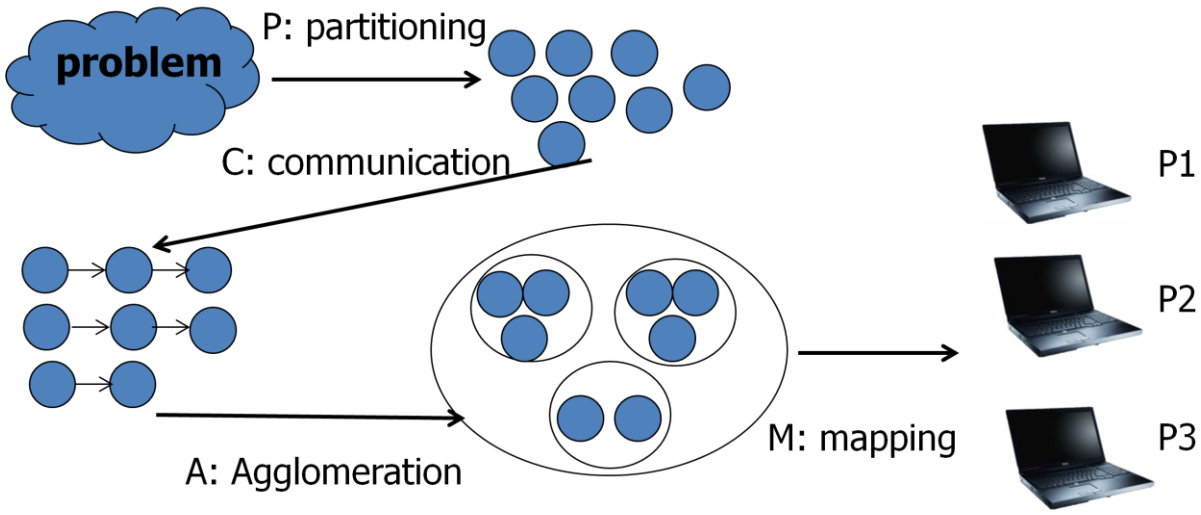
- 算法的设计与分析和计算机的结构状况有关。

3.1.6 小结

- PRAM、APRAM 模型均为 SM 存储模式，无需考虑通信开销
- BSP、logP、C^3 均为 DM 存储模式，必须考虑通信开销

第四章 并行计算的基本设计技术

4.1 PA 的基本设计过程



4.1.1 划分（P）

- 目的
  - 开发并行性的可行性
- 方法
  - 数据分解+功能分解

- 规划
  - 常用的数据，通信频率的进程分为一组
- 判据（Check list 的设计问题）

4.1.2 通信（C）

- 目的
  - 根据任务执行的需要交换数据后；协调任务的执行
- 通信要求
  - 在域分解中的确定通信要求
  - 在功能分解时，容易确定通信需求
- 通信模式
  - 局部通信　　结构化　　静态　同步
  - 全局通信　　非结构化　　动态　异步
- 判据（测试表的设计问题）

4.1.3 组合（A）

- 目的
  - 按性能要求和时间的代价来考察前两阶段的结果对小的任务进行必要的组合以减少通信开销和提交性能
- 需回答 8 个方面的问题
- 判据（测试表的设计问题）

4.1.4 匹配（M）

- 目的
  - 将每个任务分配到一个处理机上，降低通信开销和执行时间，提高处理机利用率
- 判据（涉及策略，方法和测试表设计等问题）