并行计算

——结构•算法•编程

主讲教师：谢磊

第三章 并行计算性能评测

* 3.1 并行机的一些基本性能指标
* 3.2 加速比性能定律
  * 3.2.1 Amdahl定律
  * 3.2.2 Gustafson定律
  * 3.2.3 Sun和Ni定律
* 3.3 可扩放性评测标准
  * 3.3.1 并行计算的可扩放性
  * 3.3.2 等效率度量标准
  * 3.3.3 等速度度量标准
  * 3.3.4 平均延迟度量标准

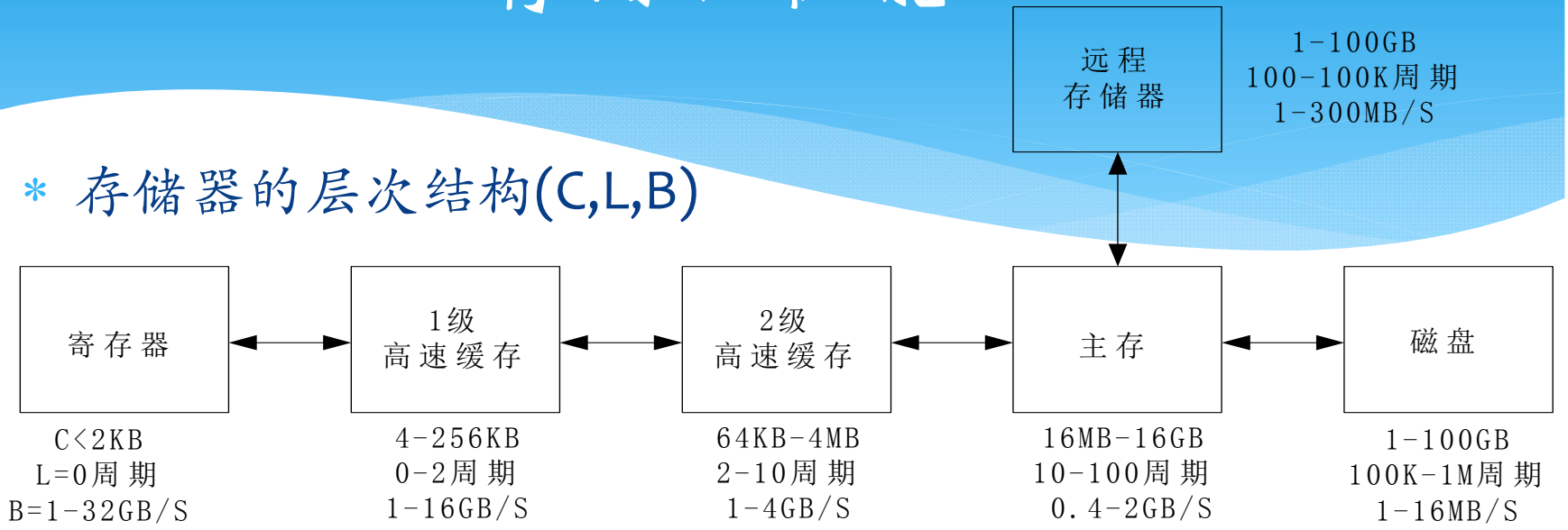
CPU的某些基本性能指标

* 工作负载
  * 执行时间
  * 浮点运算数
  * 指令数目

* 并行执行时间 T_{comput} 为计算时间，T_{paro} 为并行开销时间，T_{comm} 为相互通信时间

  $T_n = T_{comput} + T_{paro} + T_{comm}$

存储器性能



* 存储器的层次结构(C,L,B)

* 估计存储器的带宽
  * RISC add r1,r2,r3
  * 字长 8bytes 100MHz
  * B = 3*8*100*10^6 B/s= 2.4GB/s

并行与通信开销

* 并行和通信开销：相对于计算很大。
* 开销的测量：
  * 乒--乓方法（Ping-Pong Scheme）：节点0发送m个字节给节点1；节点1从节点0接收m个字节后，立即将消息发回节点0。总的时间除以2，即可得到点到点通信时间，也就是执行单一发送或接收操作的时间。
  * 可一般化为热土豆法（Hot-Potato），也称为救火队法（Fire-Brigade）：　0——1 —— 2 —— … —— -n-1 —— 0

Ping-Pong Scheme

```
For i=0 To Runs-1 do
  if （my _node _id =0） then /*发送者*/
      start _time =second（ ）
      send an m-byte message to node 1
      receive an m-byte message from node 1
      end_time = second（ ）
      total_time = end_time – start_time
      communication_time[i] = total_time/2
  else if （my_node_id = 1） then /*接收者*/
            receive an m-byte message from node 0
            send an m-byte message to node 0
      endif
  endif
endfor
```

并行开销的表达式：点到点通信

* 通信开销 $t(m) = t_0 + m/r_\infty$
* 通信启动时间 t_0
* 渐近带宽r_∞：传送无限长的消息时的通信速率
* 半峰值长度$m_{1/2}$：达到一半渐近带宽所要的消息长度
* 特定性能π_0：表示短消息带宽

$$t_0 = m_{1/2}/r_\infty = 1/\pi_0$$

并行开销的表达式：整体通信

* 典型的整体通信有：
  * 广播（Broadcasting）：处理器0发送m个字节给所有的n个处理器
  * 收集（Gather）：处理0接收所有n个处理器发来在消息，所以处理器0最终接收了m n个字节；
  * 散射（Scatter）：处理器0发送了m个字节的不同消息给所有n个处理器，因此处理器0最终发送了m n个字节；
  * 全交换（Total Exchange）：每个处理器均彼此相互发送m个字节的不同消息给对方，所以总通信量为mn^2个字节；
  * 循环移位（Circular-shift）：处理器i发送m个字节给处理器i+1，处理器n-1发送m个字节给处理器0，所以通信量为m n个字节。

机器的成本、价格与性/价比

* 机器的成本与价格
* 机器的性能/价格比 Performance/Cost Ratio： 系指用单位代价（通常以百万美元表示）所获取的性能（通常以MIPS或MFLOPS表示）
* 利用率（Utilization）： 可达到的速度与峰值速度之比

算法级性能评测

* 加速比性能定律
  * 并行系统的加速比是指对于一个给定的应用，并行算法（或并行程序）的执行速度相对于串行算法（或串行程序）的执行速度加快了多少倍。
  * Amdahl 定律（适用于固定计算负载）
  * Gustafson定律（适用于可扩放问题）
  * Sun Ni定律（受限于存储器）
* 可扩放性评测标准
  * 等效率度量标准
  * 等速度度量标准
  * 平均延迟度量标准

Amdahl 定律

* P：处理器数；
* W：问题规模（计算负载、工作负载，给定问题的总计算量）；
  * W_s：应用程序中的串行分量，f是串行分量比例（$f = W_s/W$，$W_s=W_1$）；
  * W_P：应用程序中可并行化部分，1-f为并行分量比例；
  * $W_s + W_p = W$；
* $T_s=T_1$：串行执行时间，T_p：并行执行时间；
* S：加速比，E：效率；
* 出发点：
  * 固定不变的计算负载；
  * 固定的计算负载分布在多个处理器上的，
  * 增加处理器加快执行速度，从而达到了加速的目的。

Amdahl定律（cont'd)

* 固定负载的加速公式：$S = \frac{W_s + W_p}{W_s + W_P / p}$
* $W_s + W_p$可相应地表示为f+（1-f)

$$S = \frac{f + (1 - f)}{f + \frac{1 - f}{p}} = \frac{p}{1 + f(p - 1)}$$

* p→∞时，上式极限为：　S= 1 / f
* 这意味着随着处理器数目的无限增大，并行系统所能达到的加速比上限为= 1 / f，这是一个很悲观的结论。

Amdahl定律（cont'd)

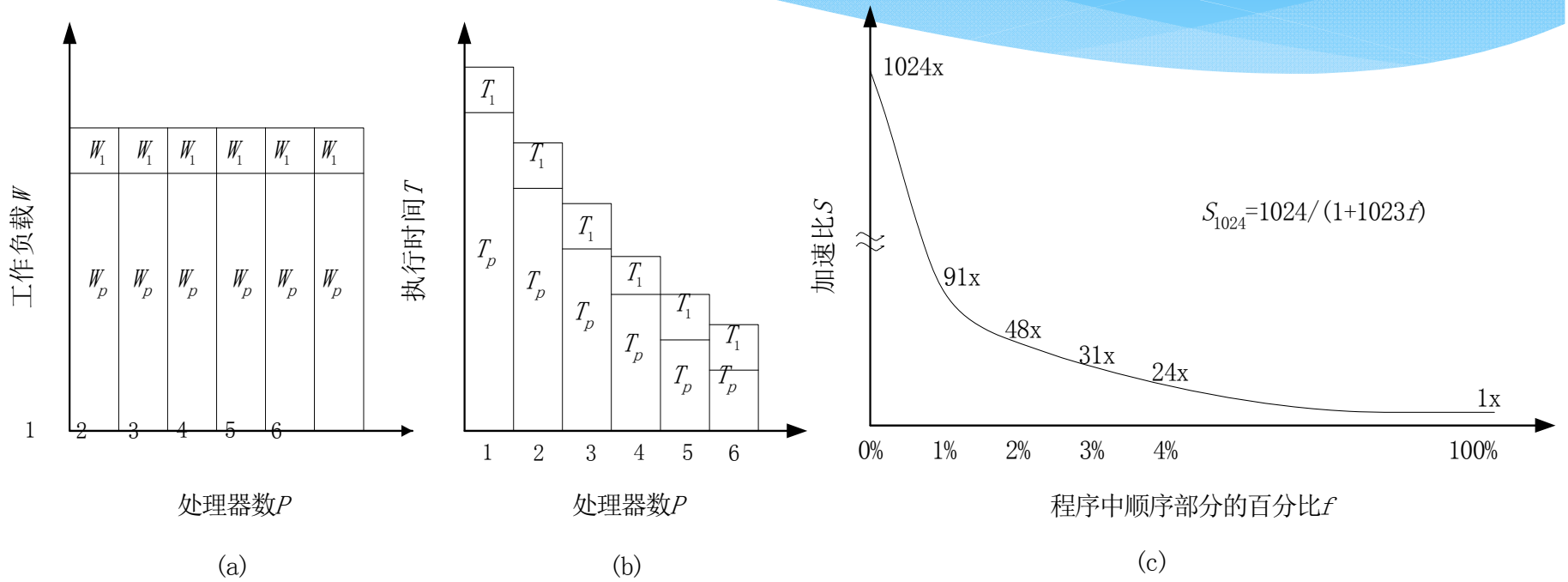
* W_O为额外开销

$$S = \frac{W_S + W_P}{W_S + \frac{W_P}{p} + W_O} = \frac{W}{fW + \frac{W(1-f)}{p} + W_O} = \frac{p}{1 + f(p-1) + W_O p / W}$$

* p→∞时，上式极限为：$$S = \frac{1}{f + W_O / W}$$

* 可见，串行分量越大和并行额外开销越大，则加速越小。

Amdahl's law (cont'd)



Gustafson定律

* 出发点：
  * 对于很多大型计算，精度要求很高，即在此类应用中精度是个关键因素，而计算时间是固定不变的。此时为了提高精度，必须加大计算量，相应地亦必须增多处理器数才能维持时间不变；
  * 除非学术研究，在实际应用中没有必要固定工作负载而计算程序运行在不同数目的处理器上，增多处理器必须相应地增大问题规模才有实际意义。

Gustafson定律（cont'd)

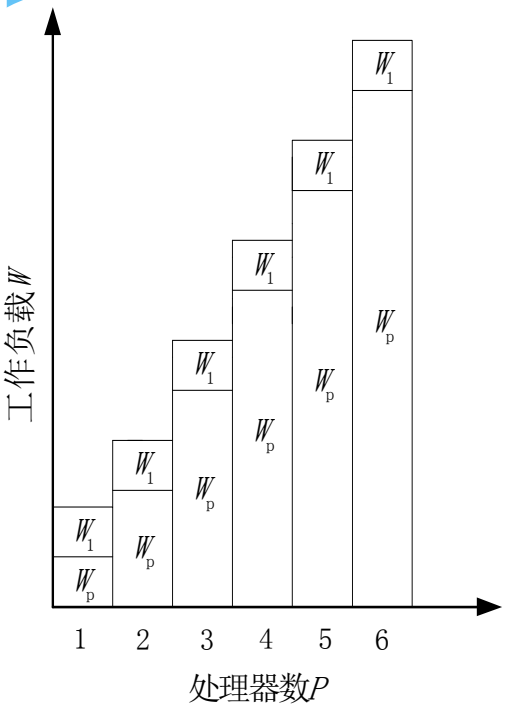
* Gustafson加速定律：

$$S' = \frac{W_S + pW_p}{W_S + p \cdot W_p / p} = \frac{W_S + pW_p}{W_S + W_P}$$

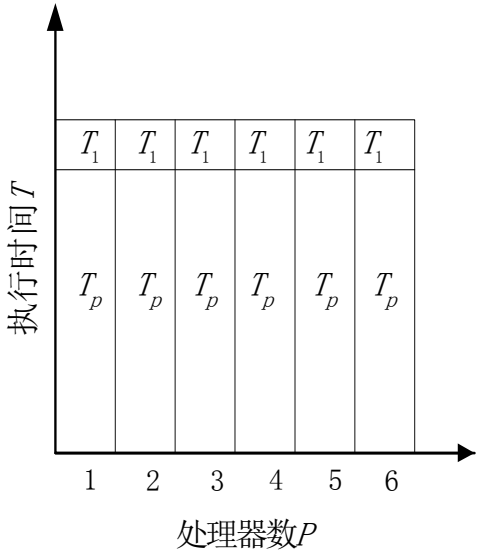
$$S' = f + p(1-f) = p + f(1-p) = p - f(p-1)$$

* 当p充分大时，S'与p几乎成线性关系，其斜率为1-f。这意味着随着处理器数目的增加，加速几乎与处理器数成比例的线性增加，串行比例f不再是程序的瓶颈，这对并行计算的发展是个非常乐观的结论。
* 考虑并行开销W_O：$S' = \frac{W_S + pW_P}{W_S + W_P + W_O} = \frac{f + p(1-f)}{1 + W_O / W}$

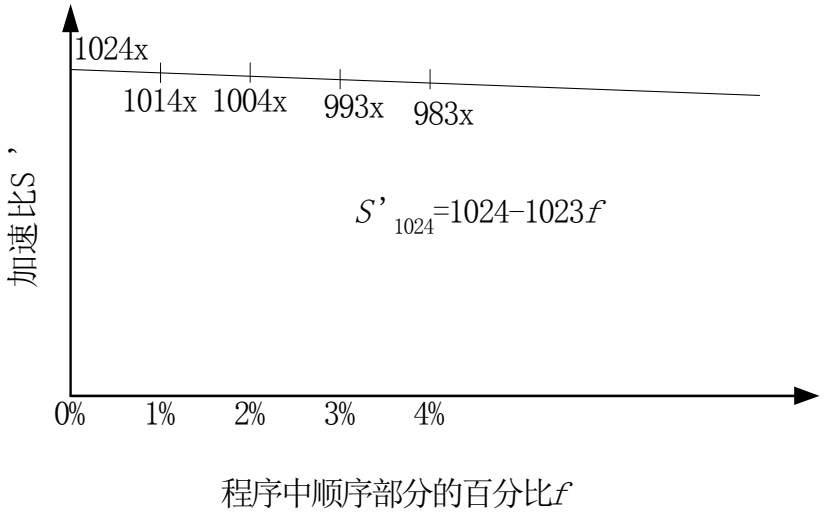
Gustafson定律（cont'd)



(a)



(b)



(c)

Sun 和 Ni定律

* 基本思想：
  * 只要存储空间许可，应尽量增大问题规模以产生更好和更精确的解（此时可能使执行时间略有增加）。
  * 假定在单节点上使用了全部存储容量M并在相应于W的时间内求解之，此时工作负载W= fW + （1-f）W。
  * 在p 个节点的并行系统上，能够求解较大规模的问题是因为存储容量可增加到pM。令因子G（p）反应存储容量增加到p倍时并行工作负载的增加量，所以扩大后的工作负载W = fW + （1-f）G（p）W。

Sun 和 Ni定律(cont'd)

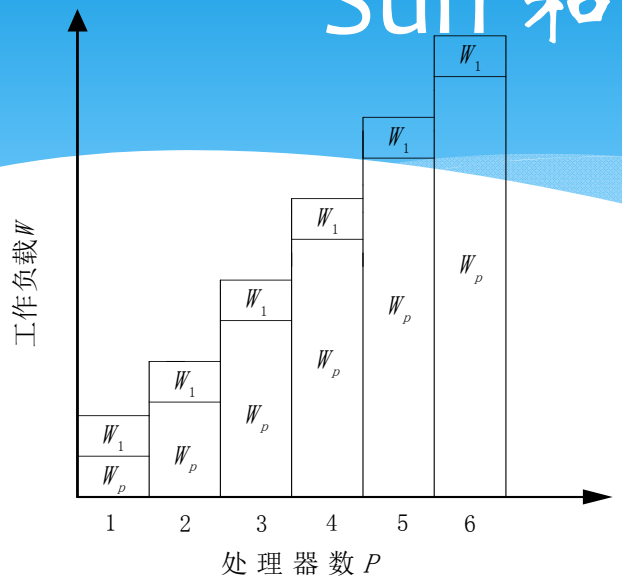
* 存储受限的加速公式：

$$S'' = \frac{fW + (1-f)G(p)W}{fW + (1-f)G(p)W/p} = \frac{f + (1-f)G(p)}{f + (1-f)G(p)/p}$$

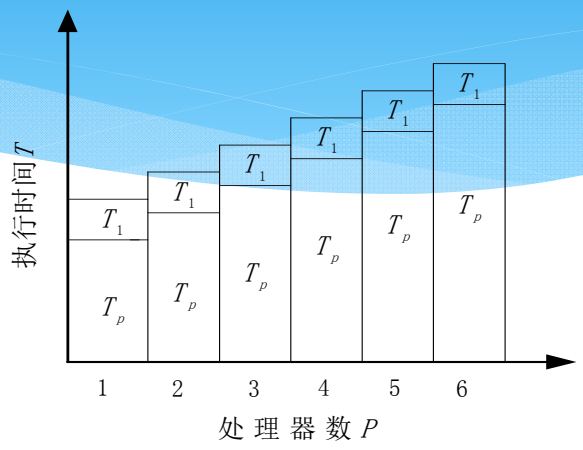
* 并行开销W_o:

$$S' = \frac{fW + (1-f)WG(p)}{fW + (1-f)G(p)W/p + W_O} = \frac{f + (1-f)G(p)}{f + (1-f)G(p)/p + W_O/W}$$

Sun 和 Ni定律(cont'd)



(a)



(b)

* G（p）=1时就是Amdahl加速定律；
* G（p）=p 变为 f + p（1-f），就是Gustafson加速定律
* G（p）>p时，相应于计算机负载比存储要求增加得快，此时 Sun和 N i 加速均比 Amdahl 加速和 Gustafson 加速为高。

加速比讨论

* 参考的加速经验公式： $p/\log p \le S \le p$
* 线性加速比：很少通信开销的矩阵相加、内积运算等
* $p/\log p$的加速 比：分治类的应用问题
* 通信密集类的应用问题：S = 1 / C （p）
* 超线性加速
* 绝对加速：并行算法与最佳串行算法
* 相对加速：同一算法在单机和并行机的运行时间

可扩放性评测标准

* 并行计算的可扩放性（Scalability）也是主要性能指标
  * 可扩放性最简朴的含意是在确定的应用背景下，计算机系统（或算法或程序等）性能随处理器数的增加而按比例提高的能力
* 影响加速比的因素：处理器数与问题规模
  * 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、竞争、冗余操作和同步等）
  * 加大的处理器数超过了算法中的并发程度

可扩放性评测标准 (cont.d)

* 增加问题的规模有利于提高加速的因素:
  * 较大的问题规模可提供较高的并发度;
  * 额外开销的增加可能慢于有效计算的增加;
  * 算法中的串行分量比例不是固定不变的（串行部分所占的比例随着问题规模的增大而缩小）。
* 增加处理器数会增大额外开销和降低处理器利用率，所以对于一个特定的并行系统（算法或程序），它们能否有效利用不断增加的处理器的能力应是受限的，而度量这种能力就是可扩放性这一指标。

可扩放性评测标准(cont.d)

* 可扩放性:调整什么和按什么比例调整
  * 并行计算要调整的是处理数p和问题规模W，
  * 两者可按不同比例进行调整，此比例关系（可能是线性的，多项式的或指数的等）就反映了可扩放的程度。
* 并行算法和体系结构

可扩放性评测标准(cont.d)

* 可扩放性研究的主要目的：
  * 确定解决某类问题用何种并行算法与何种并行体系结构的组合，可以有效地利用大量的处理器；
  * 对于运行于某种体系结构的并行机上的某种算法当移植到大规模处理机上后运行的性能；
  * 对固定的问题规模，确定在某类并行机上最优的处理器数与可获得的最大的加速比；
  * 用于指导改进并行算法和并行机体系结构，以使并行算法尽可能地充分利用可扩充的大量处理器
* 目前无一个公认的、标准的和被普遍接受的严格定义和评判它的标准

等效率度量标准

* 令t_{ie} 和t_{io} 分别是并行系统上第i个处理器的有用计算时间和额外开销时间（包括通信、同步和空闲等待时间等）

$$T_e = \sum_{i=0}^{p-1} t_e^i = Ts \qquad T_o = \sum_{i=0}^{p-1} t_o^i$$

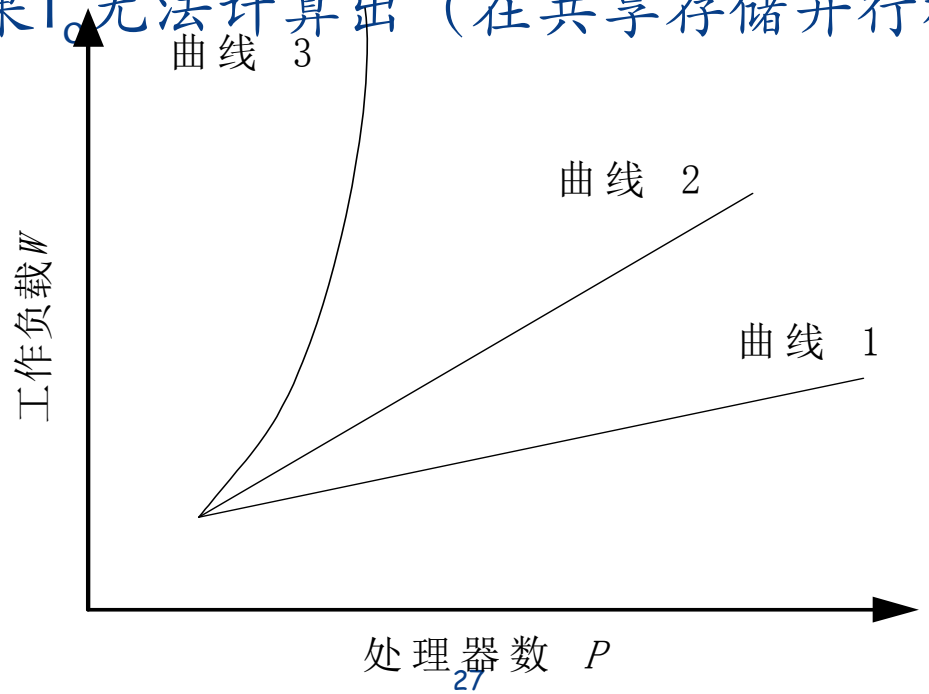
* T_p 是p个处理器系统上并行算法的运行时间，对于任意i显然有$T_p = t_{ie} + t_{io}$，且$T_e + T_o = pT_p$
* 问题的规模W为最佳串行算法所完成的计算量$W=T_e$

$$S = \frac{T_e}{T_p} = \frac{T_e}{\frac{T_e + T_o}{p}} = \frac{p}{1 + \frac{T_o}{T_e}} = \frac{p}{1 + \frac{T_o}{W}} \qquad E = \frac{S}{P} = \frac{1}{1 + \frac{T_o}{T_e}} = \frac{1}{1 + \frac{T_o}{W}}$$

* 如果问题规模W 保持不变，处理器数p增加，开销T_o增大，效率E下降。为了维持一定的效率（介于0与1之间），当处理数p增大时，需要相应地增大问题规模W的值。由此定义函数f_E（p）为问题规模W随处理器数p变化的函数，为等效率函数（ISO-efficiency Function）（Kumar1987）

等效率度量标准（cont'd)

* 曲线1表示算法具有很好的扩放性；曲线2表示算法是可扩放的；曲线3表示算法是不可扩放的。
* 优点：简单可定量计算的、少量的参数计算等效率函数
* 缺点：如果T_0无法计算出（在共享存储并行机中）



等速度度量标准

* p表示处理器个数，W表示要求解问题的工作量或称问题规模（在此可指浮点操作个数），T为并行执行时间，定义并行计算的速度V为工作量W除以并行时间T
* p个处理器的并行系统的平均速度定义为并行速度V除以处理器个数p：

$$\overline{V} = \frac{V}{p} = \frac{W}{pT}$$

* 等平均速度可扩放性度量标准如下：对于运行在并行机上的某个算法，当处理器数目增加时，若增大一定的工作量能维持整个并行系统的平均速度不变，则称该计算是可扩放的。

等速度度量标准

* W是使用p个处理器时算法的工作量，令W'表示当处理数从p增大到p'时，为了保持整个系统的平均速度不变所需执行的工作量，则可得到处理器数从 p到p'时平均速度可扩放度量标准公式

$$\Psi(p,p')=\frac{W/p}{W'/p'}=\frac{p'W}{pW'}$$

* 上述 公式计算出的值介于0与1之间，值越大表示可扩放性越好。当平均速度严格保持不变时，

$$\frac{W}{pT}=\frac{W'}{p'T'}\rightarrow\frac{p'W}{pW'}=\frac{T}{T'}\qquad\Psi(p,p')=\frac{T}{T'}$$

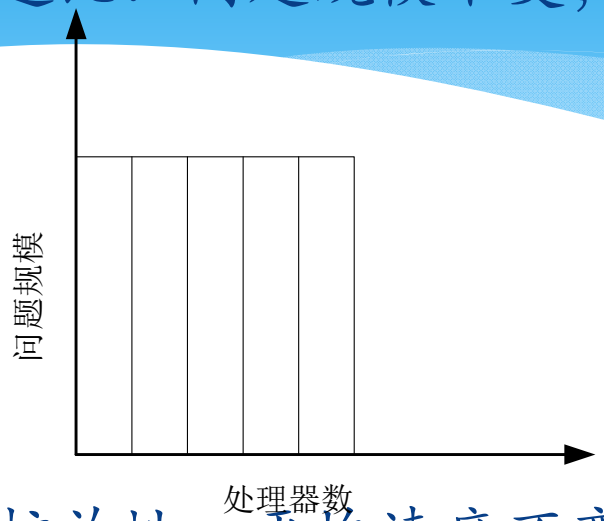
* 当p=1时，T=T1；当处理器个数为p'时，记T'=Tp'，于是有

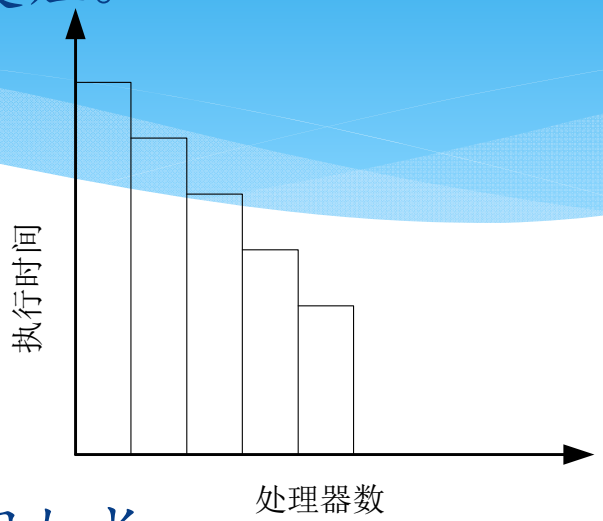
$$\Psi(1,p')=\frac{W}{W'/p'}$$

即为解决工作量为W的问题所需的串行时间/解决工作量为W'的问题所需的并行时间。

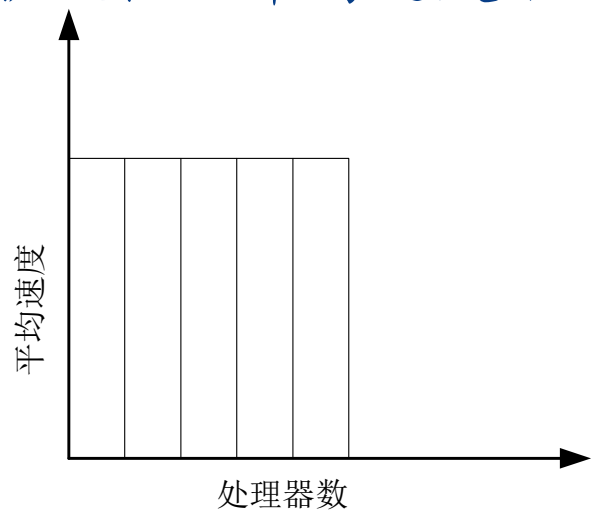
等速度度量标准（cont'd）

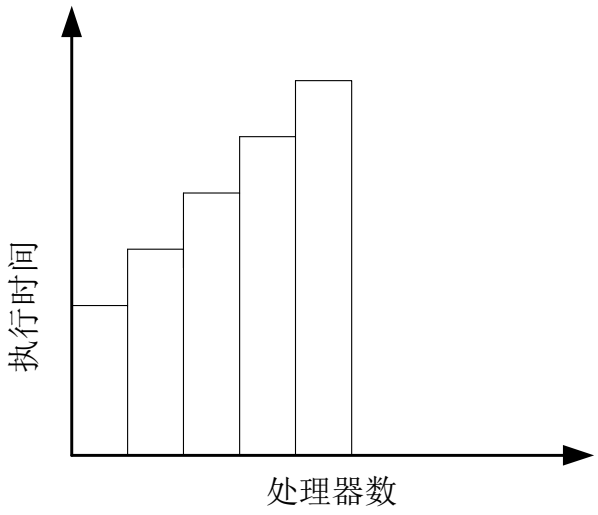
* 加速比：问题规模不变，时间变短。





* 可扩放性：平均速度不变，时间加长。





平均延迟度量标准

* T_i为P_i的执行时间，包括包括延迟L_i，Pi的总延迟时间为“L i+启动时间+停止时间”。定义系统平均延迟时间为

$$\overline{L}(W,p)=\sum_{i=1}^{p}\left(T_{para}-T_i+L_i\right)\Big/p$$

* 因为$pT_{para}=T_o+T_s$，$T_o=p\overline{L}(W,p)$，所以

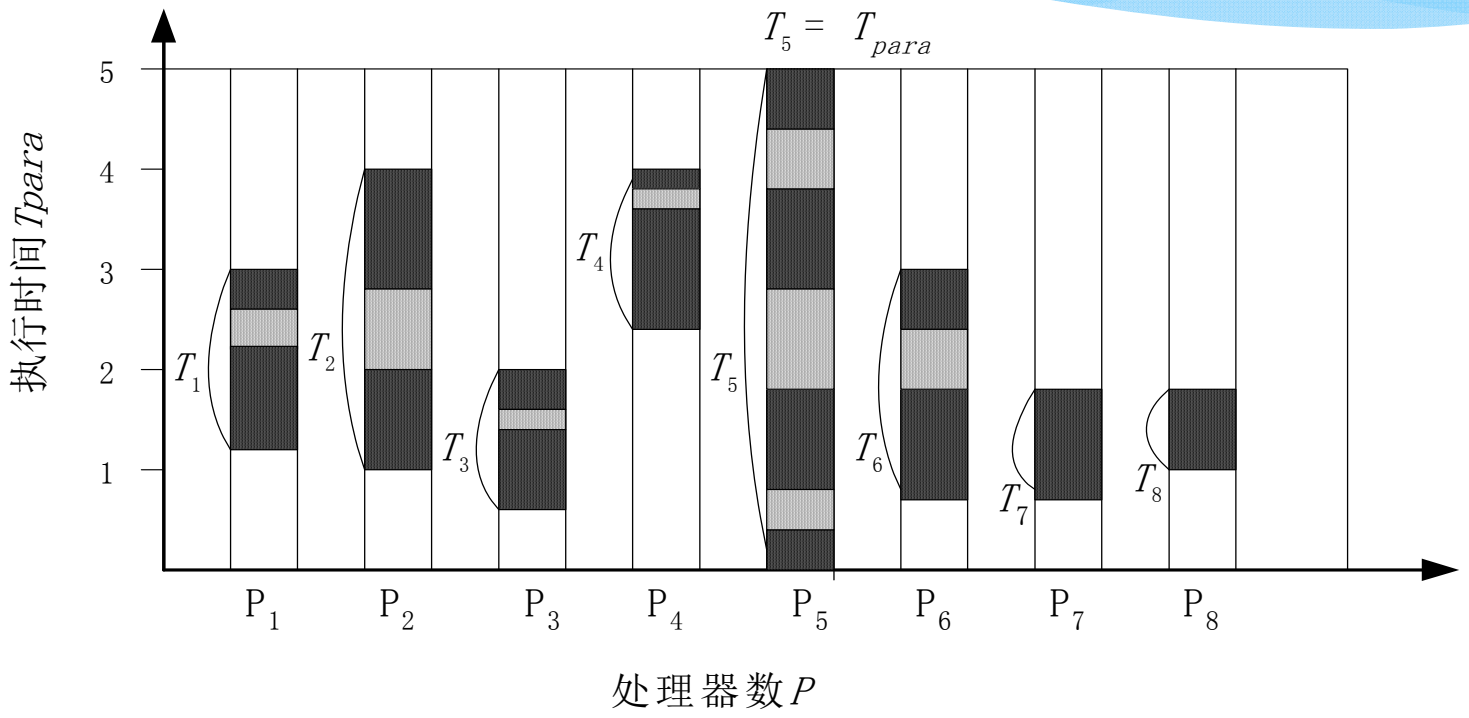
$$\overline{L}(W,p)=T_{para}-T_{seq}/p$$

* 在p个处理器上求解工作量为W问题的平均延迟 $\overline{L}(W,p)$
* 在p’个处理器上求解工作量为W’问题的平均延迟$\overline{L}(W',p')$
* 当处理器数由p变到p’，而推持并行执行效率不变，则定义平均延迟可扩放性度量标准为

$$\Phi(E,p,p')=\frac{\overline{L}(W,p)}{\overline{L}(W',p')}$$

平均延迟度量标准（Cont'd)

* 优点：平均延迟能在更低层次上衡量机器的性能
* 缺点：需要特定的软硬件才能获得平均延迟



启动前与结束后空闲时间　　开销延迟 L_i　　运行时间 T_i

有关可扩放性评测标准的讨论

* 等效率度量标准
  * 在保持效率E不变的前提下，研究问题规模W如何随处理器个数p而变化。
* 等速度度量标准
  * 在保持平均速度不变的前提下，研究处理器个数p增多时应该相应的增加多少工作量W。
* 平均延迟度量标准
  * 在保持效率E不变的前提下，用平均延迟的比值来标志随处理器个数p增加需要增加的工作量W。
* 三种度量可扩放性的标准是彼此等效的。