负载可分应用在两种主从计算平台上的调度

赵明宇, 张田文
(哈尔滨工业大学计算科学与技术学院, 黑龙江哈尔滨 150001)

摘 要: 很多现实中的负载可分应用通常要求划分是有重叠的或者需要额外的附加信息. 文章通过引入上述因素而对经典的 DLS-3 模型进行了扩展, 在有/无通信协处理器两类主从平台上分别得到了平均划分、LIFO 和 FIFO 三种调度方案的解析解, 并对它们的调度性能进行了严格比较. 分析结果表明, 与经典的 DLS-3 模型不同, 在这个新的约束下 FIFO 总是上述三种调度策略中最优的, 而与系统的规模和类型无关.

关键词: 负载可分应用的调度; 数据并行; 调度
中图分类号: TP311 文献标识码: A 文章编号: 0372-2112 (2007) 08-1582-06

Divisible Load Scheduling on Two Types of Master-Worker Platforms

ZHAO Ming-yu, ZHANG Tian-wen
(*School of Computer Science and Technology, Harbin Institute of Technology, Harbin, Heilongjiang 150001, China*)

Abstract: For the real divisible load applications, additional information and overlapped partitions are always needed. This paper introduces these factors into the divisible load scheduling model. We revised three efficient-proved schemes: equal allocation, LIFO and FIFO for the extended model, on two platforms — the master with or without communication coprocessor. Closed-form solutions for the response time of these schemes are derived. Based on these expressions, we rigorously compared the performance of the three schemes, and proved that FIFO is always the best not as without additional information, independent of the scale or type of the systems.

Key words: divisible load scheduling (DLS); data parallel; scheduling

1 引言

负载可分应用的调度(DLS)是并行调度领域的重要课题和新的研究热点[1~3]. 负载可分应用是一类数据并行应用, 它们具有可以对所处理的负载进行任意分割, 并能独立处理每个片段的性质. 按照负载处理过程的不同, 可以将 DLS 的研究分为两类. 在早期研究中, 将每个节点上的处理过程分为负载接收和运算两个阶段[3], 本文称之为二阶段负载可分应用的调度模型(DLS-2). 在更一般的情形下, 处理结束之后还需要将结果收集在一起, 即需要考虑处理结果返回的时间[4], 相应地称之为三阶段负载可分应用的调度模型(DLS-3). 求解 DLS-3 的最优调度的复杂度是一个尚未解决的问题, 目前的研究主要基于三种策略: (1) 平均划分策略使每个处理器分担相同规模的负载, 它是计算机视觉和图像处理领域的经典调度方法[5]; (2) LIFO 策略, 后得到负载的处理器先传输计算结果; (3) FIFO 策略, 是让先得到负载的处理器先传输计算结果. Kwangil 等人[6]基于 K 元完全树采用平均划分策略来解决 DLS-3 问题. Barlas[4]的研究表明 LIFO 策略可以利用任意多的计算资源而达到最短处理时间, 但若系统中的处理器数量在某个范围内 FIFO 策略能获得更好的调度质量. 文献[7~9]通过一些具体的应用对 LIFO 和 FIFO 策略进行了实验验证.

经典的 DLS-3 模型假设划分是完全的且每个计算节点只需要得到其所承担的负载. 但很多实际应用通常也需要传输一些附加信息, 例如要求划分是相互重叠的或者需要某些额外的参数. 图像处理中的卷积操作[7]和科学计算中的矩阵向量积[10]分别是这两种情况的典型应用. 本文通过引入附加信息对 DLS-3 模型进行了泛化, 在 Master 有、无通信协处理器两种类型的平台上分别得到了平均划分、LIFO 和 FIFO 三种调度策略的解析形式的调度方案. 附加信息的引入也带来了一个处理器数目的权衡问题, 本文基于三种策略的解析解分别给出了求解有效的和最优的处理器数目算法. 最后, 分析了附加信息对调度策略的影响, 并在此基础上对它们进行了严格的比较.

2 模型与符号

目标平台是由一个 Master(记做 p_0)和 m 个 Worker(记做 $p_m, p_{m-1}, \cdots, p_1$)构成的主从计算平台. Master 分为有、无前端(也称为通信协处理器)两种类型, 分别记

收稿日期: 2006-06-06; 修回日期: 2007-05-15
基金项目: 国家自然科学基金(No. 60475011)

为 MWFE(Master with front end) 和 MNFE(Master no front end). 它们的相同点在于, Master 只有一个端口, 即在同一时刻只能与一个 Worker 通信; 区别在于 Master 能否在处理负载的同时传输数据. Worker p_i 只有在接收到全部发送给它的数据后才能开始计算.

不失一般性, 令一个处理器处理整个应用的时间为单位时间, σ 是在链路上传输该应用的全部负载的时间, 分配给处理器 p_i 的负载百分比记做 α_i, 每个处理器所需要的附加信息与总负载量之比为 δ, 处理结果与输入规模之比为 τ. 处理器 p_i 的任务分成三个阶段: p_i 接收信息的时间等于 $\sigma\cdot(\alpha_i+\delta)$, p_i 的计算时间等于 α_i, p_i 向 Master 返回计算结果所需要的时间等于 $\sigma\tau\cdot\alpha_i$. 系统参数 σ, τ 和 δ 满足式(1)所定义的关系:

$$\sigma+\sigma\tau+2\sigma\delta<1 \tag{1}$$

这是当 $m=1$ 时整个应用的处理时间小于串行处理($m=0$)的充要条件, 即仅考虑最少能使用两个处理器的应用. 为了便于讨论设 $\tau\leqslant 1$, 但本文的结论也适合相反的情况. 这是因为对于本文所讨论的三种算法, 当 $\tau>1$ 时, 交换输入输出过程, 与 $\tau\leqslant 1$ 的情况是对称的.

$m+1$ 元组 $S=(\alpha_0,\alpha_1,\cdots,\alpha_m)$ 给出了一个负载分布, 只有对 $\forall i$ 都满足 $\alpha_i\geqslant 0$, 且满足式(2)给出的关系, 以及通信是没有重叠的, S 才是可行的.

$$\alpha_0+\alpha_1+\cdots+\alpha_m=1 \tag{2}$$

使一个调度策略可行的最大处理器数目称为它的有效处理器数目, 记做 N'. 调度的目标是求使整个应用的响应时间 T 尽可能小的可行负载分布 S, T 是 p_0 得到全部处理结果的时间. 使响应时间函数最小的处理器数目记做 N^*.

3　调度算法

表 1 给出了平均划分、LIFO 和 FIFO 三种调度算法的解析解.

表 1　三种调度算法的解析解

	MNFE	MWFE
EQS	$\alpha_0^E=\alpha_1^E=\cdots=\alpha_m^E=\frac{1}{m+1}$ (3)	$\alpha_0^E=\alpha_1^E=\cdots=\alpha_m^E=\frac{1}{m+1}$ (4)
	$T_E(m,\sigma,\tau,\delta)=f-1+(2-f)\cdot\alpha_0^E+m\cdot\sigma\delta$ (5)	$T_E(m,\sigma,\tau,\delta)=(m\sigma+\sigma\tau+1)\cdot\alpha_0^E+m\sigma\delta$ (6)
	$N_E^*=\sqrt{\frac{1-\sigma-\sigma\tau}{\sigma\delta}}-1$ (7)	$N_E^*=\sqrt{\frac{1-\sigma+\sigma\tau}{\sigma\delta}}-1$ (8)
	$N_E'=+\infty$ (9)	$N_E'=\frac{\sqrt{\sigma^2+4\sigma\delta+4\sigma^2\delta+4\sigma^2\delta^2}-\sigma}{2\sigma\delta}$ (10)
LIFO	$\alpha_0^L(m,\sigma,\tau,\delta)=B_0^L(m,\sigma,\tau)-\delta\cdot X_0^L(m,\sigma,\tau)$ where $\begin{cases}B_0^L(m,\sigma,\tau)=\frac{1}{F(m)+1}\\ X_0^L(m,\sigma,\tau)=\frac{F(m)-m}{(1+\tau)(F(m)+1)}\end{cases}$ (11)	$\alpha_0^L(m,\sigma,\tau,\delta)=B_0^L(m,\sigma,\tau)+\delta\cdot X_0^L(m,\sigma,\tau)$ where $\begin{cases}B_0^L(m,\sigma,\tau)=\frac{F(m)\cdot(f-1)+1}{F(m,\sigma,\tau)\cdot f+1}\\ X_0^L(m,\sigma,\tau)=\frac{m-F(m)\cdot(1-m\cdot(f-1))}{(1+\tau)\cdot(F(m)\cdot f+1)}\end{cases}$ (12)
	$T_L(m,\sigma,\tau,\delta)=f-1+(2-f)\cdot\alpha_0^L+m\cdot\sigma\delta$ (13)	$T_L(m,\sigma,\tau,\delta)=\alpha_0^L(m,\sigma,\tau,\delta)$ (14)
	$gf^{N_L^*}=(2-f)(\delta\ln f\cdot N_L^*+\delta+(1+\tau+\delta)\ln f)$ (15)	$\delta\cdot f^{N_L^*+1}=\delta\ln f\cdot N_L^*+(1+\tau+\delta)\cdot\ln f+\delta$ (16)
	$gf^{N_L'}=\sigma\delta(1+\tau)\cdot N_L'+\sigma(1+\tau)^2+\delta$ (17)	$\delta\cdot f^{N_L'+1}=\sigma\delta(1+\tau)\cdot N_L'+gf+\sigma(1+\tau)^2$ (18)
FIFO	$\alpha_0^F(m,\sigma,\tau,\delta)=B_0^F(m,\sigma,\tau)-\delta\cdot X_0^F(m,\sigma,\tau)$ where $\tau<$ $\begin{cases}B_0^F(m,\sigma,\tau)=\frac{1+\sigma\tau-\sigma\tau\cdot G(m)}{(1-\sigma\tau)\cdot G(m)+1+\sigma\tau},\\ X_0^F(m,\sigma,\tau)=\frac{(1+\sigma\tau)\cdot(G(m)-m)}{(1-\tau)((1-\sigma\tau)\cdot G(m)+1+\sigma\tau)}\end{cases}$ $\tau=$ $\begin{cases}B_0^F(m,\sigma,\tau)=\frac{1+\sigma-\sigma m}{(1-\sigma)m+1+\sigma},\\ X_0^F(m,\sigma,\tau)=\frac{1}{(1-\sigma)m+1+\sigma}\cdot\frac{m(m-1)}{2}\end{cases}$ (19)	$\alpha_0^F(m,\sigma,\tau,\delta)=B_0^F(m,\sigma,\tau)+\delta\cdot X_0^F(m,\sigma,\tau)$ where $\tau<$ $\begin{cases}B_0^F(m,\sigma,\tau)=\frac{1+\sigma\tau+\sigma\cdot G(m)}{(1+\sigma)\cdot G(m)+1+\sigma\tau},\\ X_0^F(m,\sigma,\tau)=\frac{m\sigma(1-\tau)\cdot G(m)-(1+\sigma\tau)(G(m)-m)}{(1-\tau)((1+\sigma)\cdot G(m)+1+\sigma\tau)}\end{cases}$ $\tau=$ $\begin{cases}B_0^F(m,\sigma,\tau)=\frac{1+\sigma\cdot(m+1)}{(1+\sigma)(m+1)},\\ X_0^F(m,\sigma,\tau)=\frac{m\sigma}{2(1+\sigma)}\end{cases}$ (20)
	$T_F(m,\sigma,\tau,\delta)=f-1+(2-f)\cdot\alpha_0^F+m\cdot\sigma\delta$ (21)	$T_F(m,\sigma,\tau,\delta)=\alpha_0^F(m,\sigma,\tau,\delta)$ (22)
	$(2-f)\cdot\frac{\partial}{\partial m}\alpha_0^F(N_F^*,\sigma,\tau,\delta)+\sigma\delta=0$ (23)	$\frac{\partial}{\partial m}\alpha_0^F(N_F^*,\sigma,\tau,\delta)=0$ (24)
	$(\sigma\tau(1-\tau)+\delta(1+\sigma\tau))\cdot g^{N_F'}=\sigma\delta(1-\tau)\cdot N_F'+\sigma(1-\tau)+\delta(1+\sigma\tau)$ (25)	$\alpha_1^F(N_F')=\delta\cdot\frac{\sigma\tau(G(N_F')-N_F')}{2+\tau-\sigma\tau\cdot(G(N_F')-1)}$ (26)

$f=1+\sigma(1+\tau)$　　$F(m)=1+f+\cdots+f^{m-1}=\frac{f^m-1}{f-1}$　　$g=\frac{1+\sigma}{1+\sigma\tau}$　　$G(m)=1+g+g^2+\cdots+g^{m-1}=\frac{g^m-1}{g-1}$

平均划分(EQS)是最简单和最经典的调度算法,包括 Master 节点每个处理器都分担同等规模的负载,式(3)和(4). 图 1 给出了平均划分的调度时间图. 对于 MNFE 平台,处理器 p_i 在计算结束之后需要等待的时间记为 x_i. $x_1=\frac{m-1}{m+1}\cdot\sigma\tau$,对于 $\forall i>1$, $x_i-x_{i-1}=\frac{1}{m+1}\cdot\sigma(1-\tau)+\sigma\delta$,因此 $x_i=\frac{\sigma((m-i)\tau+i-1)}{m+1}+(i-1)\cdot\sigma\delta$. 在 MWFE 平台上 Master 最早完成计算,Worker 不需要推迟其结果的发送. 不过,p_0 在接收 p_i 和 p_{i-1} 的计算结果之间有空闲时间 $\sigma\cdot\left(\frac{1-\tau}{m+1}+\delta\right)$.

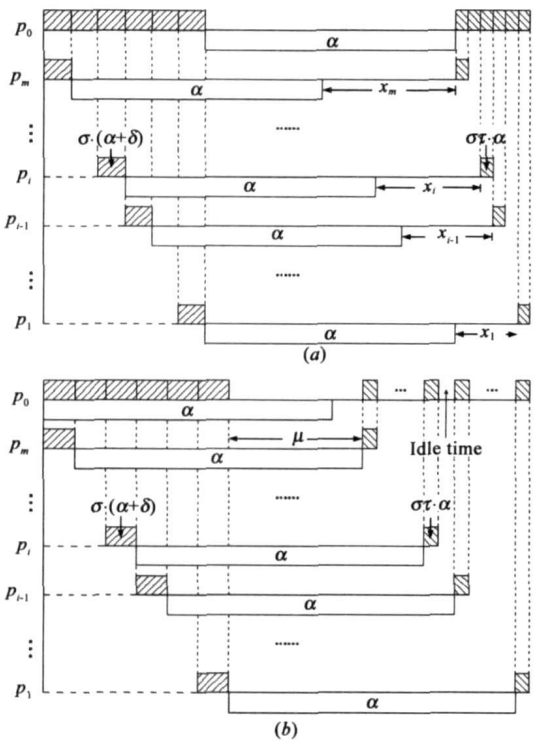


图 1 平均划分的时间图

在 LIFO 中,Master 按照编号从大到小的顺序连续地向 Worker 分发负载及附加信息,最后按照与分发相反的顺序连续地从 Worker 收集处理结果,如图 2. Worker 在接收到分配给它的数据后立即进行处理,处理结束后立即将结果返回. 这等价于 p_i 在进行计算时 p_{i-1} 完成全部的接收、计算和发送三个阶段. 因此对于 $\forall i(2\leqslant i\leqslant m)$ 都有 $\alpha_i=f\cdot\alpha_{i-1}+\sigma\delta$,其中 $f=1+\sigma(1+\tau)$. 对于 MNFE 平台,$\alpha_0=\alpha_1$. 而对于 MWFE 平台,$\alpha_0^L=f\cdot\alpha_m^L+\sigma\delta$. 这些方程和条件(2)分别构成了一个有 $m+1$ 个等式和未知数的方程组,于是可以得到它们的解析解(式(11),(12)).

FIFO 与平均划分有相同的负载分配和结果接收顺序. 不同的是,FIFO 使 Master 连续地从 Worker 接收处理结果且每个处理器都没有空闲阶段,如图 3. 对于 $\forall i(2\leqslant i\leqslant m)$,都有 $\alpha_i+\sigma\tau\cdot\alpha_i=\sigma(\alpha_{i-1}+\delta)+\alpha_{i-1}$. 对于 MNFE 平台,$p_0$ 与 p_m 同时完成计算,即 $\sigma\cdot\sum_{i=1}^{m}(\alpha_i^L+\delta)+\alpha_0^L=\sigma\cdot(\alpha_m^L+\delta)+\alpha_m^L$. 对于 MWFE 平台,Master 的运算时间等于接收到全部处理结果的时间. 这样可以分别为 MNFE 和 MWFE 两类平台各构造一个方程组,其解

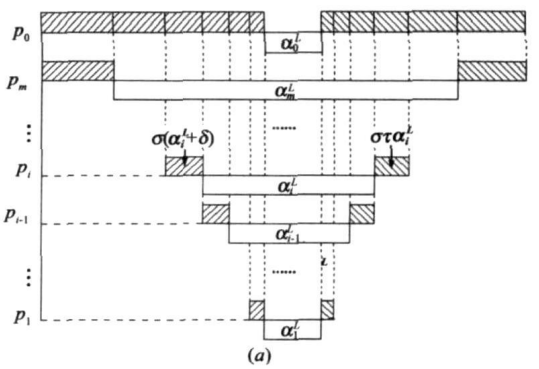


图 2 LIFO 算法的时间图

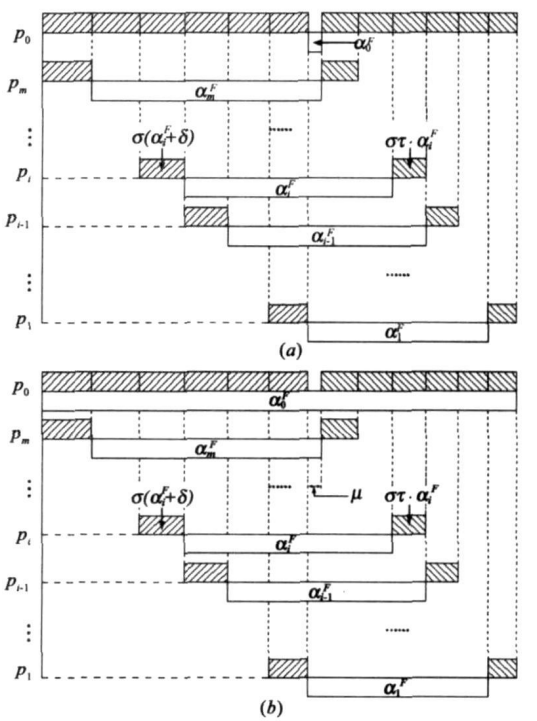


图 3 FIFO 算法的时间图

析解分别是式(19)和(20).

在 EQS 和 FIFO 算法中, 应用的响应时间是获得 p_1 的处理结果的时间, 式(5), (6)和式(21), (22)分别给出了它们的表达式. LIFO 算法下的响应时间 $T_L(\boldsymbol{T_L})$ 等于 Master 接收到 p_m 返回的处理结果的时间, 式(13)和(14).

引理 **1**　对于 MNFE 平台, α_0^L 和 α_0^F 相对于处理器数目 m 是单调递减的.

证明: $\dfrac{\partial}{\partial m}\alpha_0^L=\dfrac{\sigma}{(f^m+f-2)^2}\cdot h(m,\sigma,\tau,\delta)$, 其中 $h(m,\sigma,\tau,\delta)=\beta^m-(1+\tau+(m+1)\delta)\cdot\ln f\cdot f^m-(2-f)\cdot\delta$. 第一项显然是大于 0 的, $\dfrac{\partial h}{\partial m}=-f^m(\ln f)^2(1+\tau+(m+1)\delta)<0$ 说明 h 是递减的, 即 $h(m,\sigma,\tau,\delta)<h(1,\sigma,\tau,\delta)=2(f-1)\delta-f\ln f\cdot(1+\tau+2\delta)\cdot\dfrac{\partial}{\partial\delta}h(1,\sigma,\tau,\delta)=2(f-1-f\ln f)$, 易证 $\dfrac{\partial}{\partial\delta}h(1,\sigma,\tau,\delta)<0$. 因此 $h(m,\sigma,\tau,\delta)<h(1,\sigma,\tau,\delta)<h(1,\sigma,\tau,0)=-(1+\tau)\cdot f\ln f<0$. 类似可证, α_0^F 也是单调递减的.

MNFE 平台上的 EQS 算法总有 $\alpha_i^E>0$, 即它可以利用任意数量的处理器(式(9)). 在 MWFE 平台上, 调度可行的充要条件是 p_m 完成计算的时间大于等于全部数据的传输时间, 因此可以得到有效处理器数目 $\boldsymbol{N_E'}$(式(10)); 对于 LIFO 算法, 容易验证 p_1 总是承担最少的负载且由引理 1, 使 $\alpha_1^L=0$($\alpha_1^L=0$)的 $N_L'(\boldsymbol{N_L'})$ 是其所能利用的最大处理器数目(方程(17), (18)); 在 MNFE 平台上的 FIFO 算法中, p_0 所承担的负载总是最少的且由引理 1, 可以得到 N_F' 的方程(25). 对于 MWFE 平台, 容易证明调度可行的充要条件是 Master 开始接收结果的时间大于完成数据传输的时间, 从而可以得到 $\boldsymbol{N_F'}$ 的方程(26).

引理 **2**　无论 Master 是否有前端, 若 $\delta>0$ 则三种调度算法的应用响应时间都是有一个极小值的凹函数. 并且, 对于 LIFO 算法, $N_L'>N_L^*$ 和 $\boldsymbol{N_L'}>\boldsymbol{N_L^*}$.

证明: 以 MNFE 平台上的 LIFO 算法为例证明. $\dfrac{\partial}{\partial m}T_L=\dfrac{\beta^m}{(f^m+f-2)^2}\cdot h(m)$, 其中 $h(m)=\beta^m-(2-f)\cdot(\delta\ln f\cdot m+\delta+(1+\tau+\delta)\cdot\ln f)\cdot\dfrac{\partial}{\partial m}T_L$ 的第一项总是大于 0 的. 易证 $h(m)$ 是递增的, 设方程 $h(m)=0$ 的解为 N_L^*. 若 $m<N_L^*$ 则 $h(m)<0$, 于是 $\dfrac{\partial}{\partial m}T_L(m)<0$, 即 $T_L(m)$ 是递减的. 反之 $T_L(m)$ 是递增的. N_L' 满足方程(17), 将等式(17)的右端代替 $h(N_L')$ 中的 $\beta^{N_L'}$ 有 $h(N_L')=(f-1-(2-f)\ln f)\cdot(\delta N_L'+1+\tau+\delta)$. 易证 $h(N_L')$ 的第一项相对于 f 是递增的且 $f>1$, 因此 $h(N_L')>0$. 这表明一定有 $N_L'>N_L^*$. 类似地可以为其他情况进行证明.

引理 2 表明, N^* 是使响应时间相对于 m 的偏导数为零的值. 因此, 可以得到 N_E^* 和 $\boldsymbol{N_E^*}$ 的解析式(7)和(8), N_L^* 和 $\boldsymbol{N_L^*}$ 分别满足方程(15)和(16), N_F^* 和 $\boldsymbol{N_F^*}$ 分别满足方程(23)和(24).

4　调度性能的比较

本节对三种调度算法的性能进行比较. 首先证明了若 Master 没有前端, LIFO 的响应时间总是小于 EQS 的响应时间(定理 1). 然后, 分析了 α_0^L 和 α_0^F 中 δ 因子的性质(引理 3). 基于这个性质证明了无论 Master 是否有前端, 对于相同的处理器数目 LIFO 和 EQS 的响应时间总是大于 FIFO 的响应时间(定理 2 和 3). 最后, 假设系统中的处理器资源是足够多的, 得到了 FIFO 仍能给出最小响应时间的结论(定理 4). 容易验证, 当 $m=1$ 时三种策略所产生的调度是相同的, 因此只需考虑 $m\geqslant 2$ 的情况.

定理 **1**　若 $m\geqslant 2$, 则 $T_L(m)<T_E(m)$.

证明: 由于 $\alpha_0^L=\alpha_1^L<\alpha_2^L<\cdots<\alpha_m^L$ 和公式(2)以及 $m\geqslant 2$, 因此有 $\alpha_0^L<\dfrac{1}{m+1}=\alpha_0^E$. 由式(5)和(13), $T_E(m)-T_L(m)=(2-f)\cdot(\alpha_0^E-\alpha_0^L)>0$.

引理 **3**　对于任意的 m, 且 $0<\sigma\leqslant 1$ 和 $0\leqslant\tau<1$, 公式(11)中的 $X_0^L(m,\sigma,\tau)$ 相对于 τ 是单调递减的, 而公式(19)中的 $X_0^F(m,\sigma,\tau)$ 相对于 τ 是单调递增的.

证明: 令 $r=\sigma(1+\tau)$, 则 $f=1+r$. $\dfrac{\partial}{\partial\tau}X_0^L(m,\sigma,\tau)=-((1+r)(1+\tau)^2((1+r)^m+r-1))^{-2}\cdot N(m,r)$, 其中 $N(m,r)=[f^{m+1}-(m^2+m-2)\cdot r^2-2]\cdot f^m-r^2(1+r)\cdot m+(1-2r)(1+r)$. $\dfrac{\partial^2}{\partial m^2}N(m,r)=K(m,r)\cdot f^m$, 其中 $K(m,r)=4(\ln f)^2\cdot f^{m+1}-(r\ln f)^2m^2-(4+\ln f)r^2\ln f\cdot m+2(\ln f)^2\cdot(r^2-1)-2r^2(\ln f+1)$. $\dfrac{\partial^3}{\partial m^3}K(m,r)=4(\ln f)^5\cdot f^{m+1}>0$ 说明 $\dfrac{\partial^2}{\partial m^2}K(m,r)$ 相对于 m 是递增的, 类似地 $\dfrac{\partial}{\partial m}K(m,r)$ 也是递增的, 即 $\dfrac{\partial}{\partial m}K(m,r)\geqslant\dfrac{\partial}{\partial m}K(1,r)=4(\ln f)^3\cdot f^2-2(r\ln f)^2-(4+\ln f)r^2\ln f$. 易证 $\dfrac{\partial}{\partial m}K(1,r)$ 是增函数, 因此有 $\dfrac{\partial}{\partial m}K(1,r)>\dfrac{\partial}{\partial m}K(1,0)=0$, 这说明 $\dfrac{\partial^2}{\partial m^2}N(m,\sigma,\tau)>0$. 类似可证 $N(m,r)$ 是增函数, 即 $N(m,r)>N(1,r)=0$. 并且 $\dfrac{\partial}{\partial\tau}X_0^L$ 的第一项总是大于 0 的, 因此有 $\dfrac{\partial}{\partial\tau}X_0^L<0$. 同理, 可以证明 $X_0^F(m,\sigma,\tau)$

相对于 τ 是单调递增的.

定理 2 若 $m \geqslant 2$, 则 $T_L(m) > T_F(m)$ 和 $\boldsymbol{T}_L(m) > \boldsymbol{T}_F(m)$.

证明: 对于 MNFE 平台, 由式(13)和(21)有 $T_L - T_F = (2-f)\cdot(\alpha_0^L - \alpha_0^F)$. 因此只需证明 $\alpha_0^L > \alpha_0^F$.

(1) 设 $\tau < 1$, Barlas[4] 证明了若 $\delta = 0$, 则 $T_L(m) > T_F(m)$, 即 $B_0^F(m, \sigma, \tau) < B_0^L(m, \sigma, \tau)$. 由于 $X_F(m, \sigma, 0) = X_L(m, \sigma, 0)$及引理 3, $X_0^F(m, \sigma, \tau) \geqslant X_0^F(m, \sigma, 0) = X_0^L(m, \sigma, 0) \geqslant X_0^L(m, \sigma, \tau)$.

(2) 设 $\tau = 1$,

$$\alpha_0^L - \alpha_0^F = \frac{(2+(m+1)\delta)}{2((1+2\sigma)^m - (1+2\sigma))((1-\sigma)m+1+\sigma)} \cdot (1+\sigma+\sigma m+((m-1)\sigma-1)(1+2\sigma)^m).$$

上式中的第一项显然是大于 0 的, 而第二项相对于 m 是递增的且 $m = 2$ 时有最小值 $4\sigma^3 > 0$.

对于 MWFE 平台, 式(14)和(22)表明仍只需比较 α_0^L 和 α_0^F. 首先按照 MNFE 的规则来分配负载. 不失一般性, 设总负载量为 1 个单位. 处理器 p_i 在 LIFO 和 FIFO 算法下所承担的负载量分别记为 $\chi_i^L = 1\cdot\alpha_i^L$ 和 $\chi_i^F = 1\cdot\alpha_i^F$ 个单位. $\chi_0^L > \chi_0^F$, 因为 $\alpha_0^L > \alpha_0^F$, 如图 4. LIFO 和 FIFO 算法中 Master 用于通信的时间分别等于 $\mu_L = \sigma(1+\tau)\cdot(1-\alpha_0^L) + m\cdot\sigma\delta$ 和 $\mu_F = \sigma(1+\tau)\cdot(1-\alpha_0^F) + m\cdot\sigma\delta$. 在 MWFE 平台上 Master 可以在通信的同时进行计算, 且在时间 μ_L 和 μ_F 内它所能完成的负载量分别是 $\xi_L = 1\cdot\mu_L$ 和 $\xi_F = 1\cdot\mu_F$, FIFO 和 LIFO 所能完成的总负载量分别是 $1+\xi_F$ 和 $1+\xi_L$. 而 $\alpha_0^F = \frac{\chi_0^F + \xi_F}{1+\xi_F}$ 和 $\alpha_0^L = \frac{\chi_0^L + \xi_L}{1+\xi_L}$, 因此 $\alpha_0^L - \alpha_0^F = \frac{(\chi_0^L - \chi_0^F)\cdot(1+m\sigma\delta)}{(1+\xi_L)(1+\xi_F)} > 0$.

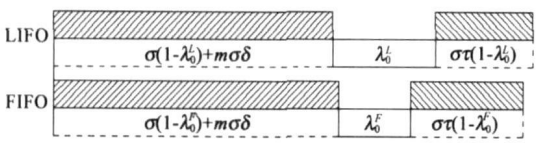


图 4 Master完成的负载量

定理 3 若 $m \geqslant 2$, 则 $T_E(m) > T_F(m)$ 和 $\boldsymbol{T}_E(m) > \boldsymbol{T}_F(m)$.

证明: 对于 MNFE 平台, 由定理 1 和定理 2 可以直接得到 $T_E(m) > T_F(m)$; 对于 MWFE 平台, 设 $\tau < 1$, $\boldsymbol{T}_E(m) - \boldsymbol{T}_F(m) = \frac{\sigma(m+\tau)(1+\sigma\tau+\sigma G)+(G-m)(1+\sigma\tau)}{(m+1)(1+\sigma\tau+G+\sigma G)} + \delta\cdot\frac{m\sigma(1-\tau)(1+\sigma\tau+\sigma G)+(G-m)(1+\sigma\tau)}{(1-\tau)(1+\sigma\tau+G+\sigma G)}$. 由于 $G > m$, 因此总有 $\boldsymbol{T}_E(m) > \boldsymbol{T}_F(m)$. 若 $\tau = 1$, $\boldsymbol{T}_E(m) - \boldsymbol{T}_F(m) = \frac{\sigma(2(m\sigma+\tau+\sigma\tau)+m(m+1)(1+2\sigma))}{2(m+1)(1+\sigma)} > 0$.

定理 4 如果网络中有足够多的处理器资源, FIFO 的最优响应时间是三种调度算法中最小的. 最优响应时间是指在有效处理器数目范围内, 响应时间的最小值, 记做 T.

证明: 分别证明 FIFO 的最优响应时间小于 LIFO 和 EQS 的最优响应时间.

(1) FIFO 与 LIFO 的比较

(a) MNFE. 由引理 2 一定有 $T_L = T_L(N_L^*)$. 若 $T_L = T_F(N_F^*)$, 由定理 2 有 $T_F = T_F(N_F^*) < T_F(N_L^*) \leqslant T_L(N_L^*) = T_L$. 若 $T_F = T_F(N_F')$, 则有下面两种可能: 设 $N_L^* < N_F'$, 由定理 2 有, $T_F(N_F') < T_F(N_L^*) \leqslant T_L(N_L^*)$; 设 $N_L^* \geqslant N_F'$, $T_L(N_L^*) = \sigma + \sigma\tau + (2-f)\cdot\alpha_0^L(N_L^*) + N_L^*\cdot\sigma\delta$ 和 $T_F(N_F') = \sigma + \sigma\tau + N_F'\cdot\sigma\delta$. 因此, $T_L(N_L^*) - T_F(N_F') = (2-f)\cdot\alpha_0^L(N_L^*) + (N_L^* - N_F')\cdot\sigma\delta > 0$.

(b) MWFE. 当 $\boldsymbol{T}_F = \boldsymbol{T}_F(N_F^*)$ 或 $\boldsymbol{T}_F = \boldsymbol{T}_F(N_F')$ 且 $N_L^* < N_F'$ 时, 类似 MNFE 的情况可证 $\boldsymbol{T}_F < \boldsymbol{T}_L$. 设 $\boldsymbol{T}_F = \boldsymbol{T}_F(N_F')$ 且 $N_L^* \geqslant N_F'$, 则 $\alpha_0^L(N_L^*) = \sigma(1+\tau)\cdot(1-\alpha_0^L(N_L^*)) + N_L^*\cdot\sigma\delta + \alpha_1^L(N_L^*)$ 和 $\alpha_0^F(N_F') = \sigma(1+\tau)\cdot(1-\alpha_0^F(N_F')) + N_F'\cdot\sigma\delta$, 因此 $\alpha_0^L(N_L') - \alpha_0^F(N_F') = \frac{(N_L^* - N_F')\cdot\sigma\delta + \alpha_1^L(N_L^*)}{1+\sigma+\sigma\tau}$. 由引理 2 有 $N_L^* \leqslant N_L'$, 因此 $\alpha_1^L(N_L^*) > 0$. 于是 $\alpha_0^L(N_L^*) \geqslant \alpha_0^F(N_F')$.

(2) FIFO 与 EQS 的比较

(a) MNFE. 当 $\boldsymbol{T}_F = \boldsymbol{T}_F(N_F^*)$ 或者 $\boldsymbol{T}_F = \boldsymbol{T}_F(N_F')$ 且 $N_E^* \leqslant N_F'$, 类似于 LIFO 的情况, 并由定理 3 总有 $\boldsymbol{T}_F < \boldsymbol{T}_E$; 若 $\boldsymbol{T}_F = \boldsymbol{T}_F(N_F')$ 且 $N_E^* > N_F'$, $T_E = T_E(N_E^*) = \sigma + \sigma\tau + (2-f)\cdot\alpha_0^E(N_E^*) + N_E^*\cdot\sigma\delta + I(N_E^*)$, I 表示在结果接收阶段信道的空闲时间. 于是 $T_E - T_F = (2-f)\cdot\alpha_0^E(N_E^*) + (N_E^* - N_F')\cdot\sigma\delta + I(N_E') > 0$.

(b) MWFE. 当 $\boldsymbol{T}_F = \boldsymbol{T}_F(N_F^*)$ 或 $\boldsymbol{T}_F = \boldsymbol{T}_F(N_F')$ 且 $N_E' < N_F'$ 时, 类似 MNFE 的情况可证 $\boldsymbol{T}_F < \boldsymbol{T}_E$. 若 $\boldsymbol{T}_F = \boldsymbol{T}_F(N_F')$ 且 $N_E' \geqslant N_F'$, 设 $\boldsymbol{T}_E = \boldsymbol{T}_E(N_E')$, γ 表示 $\boldsymbol{T}_E$ 与 Master 完成计算的时间的差值, 则有 $\alpha_0^E(N_E') + \gamma(N_E') = \sigma(1+\tau)(1-\alpha_0^E(N_E')) + N_E'\cdot\sigma\delta + I(N_E') + \mu(N_E')$. 因此可以将 $\boldsymbol{T}_E(N_E')$ 表示为 $\boldsymbol{T}_E(N_E') = \alpha_0^E(N_E') + \gamma(N_E') = \frac{\sigma+\sigma\tau+N_E'\cdot\sigma\delta+I(N_E')+\mu(N_E')+\gamma(N_E')\cdot(\sigma+\sigma\tau)}{1+\sigma+\sigma\tau}$.

利用前面得到的结果, 有如下表达式: $\boldsymbol{T}_E(N_E') - \boldsymbol{T}_F(N_F') = \frac{(N_E' - N_F')\cdot\sigma\delta + I(N_E^*) + \mu(N_E^*) + \gamma(N_E^*)\cdot(\sigma+\sigma\tau)}{1+\sigma+\sigma\tau} > 0$;

设 $\boldsymbol{T}_E = \boldsymbol{T}_E(N_E^*)$, 若 $N_E^* < N_E^*$ 显然有 $\boldsymbol{T}_F < \boldsymbol{T}_E$. 否则, 类似上面的情况, 在 N_E^* 点可以得到下面的表达式:

$$\boldsymbol{T}_E(N_E^*) - \boldsymbol{T}_F(N_F') = \frac{(N_E^* - N_F')\cdot\sigma\delta + I(N_E^*) + \mu(N_E^*) + (\sigma+\sigma\tau)\cdot\gamma(N_E^*)}{1+\sigma+\sigma\tau} > 0.$$

5 数值实例

本节通过两个例子对前面的结论进行说明. 图 5 (a)给出了对于 σ= 30%, τ= 50%和 δ= 5%的应用, 在 MNFE 平台上三种调度算法的响应时间曲线. 可以看到尽管 LIFO 取得最小响应时间的处理器数目大于 FIFO 取得最小响应时间的处理器数目, 但最终 FIFO 所给出的响应时间更小; 在 MWFE 平台上, 对于 σ= 5%, τ= 100%和 δ= 1%的应用, 当处理器数目较大时, EQS 的性能优于 LIFO 算法. τ= 100%和 δ= 1%意味着结果接收阶段的空闲时间足够小, 而 σ 也较小使得 Master 完成计算的时间也接近于应用的响应时间. 这时 EQS 非常接近于 FIFO 算法, 但仍比 FIFO 策略的性能差. 如图 5 (b). 这个例子表明, 对于一些特殊情况 EQS 算法有可能比 LIFO 算法有更好的性能.

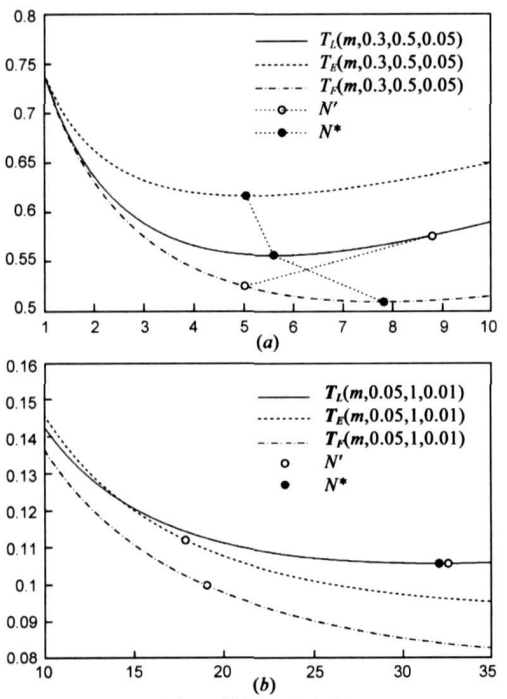


图 5　数值比较实例

6 结论

本文通过引入附加信息泛化了 DLS-3 模型, 并得到了平均划分、LIFO 和 FIFO 三种调度算法在 Master 有、无前端两类平台上的解析解. 分别给出了求解它们所能利用的处理器数目和使响应时间最小的处理器数目的算法. 本文的分析结果表明, 附加信息所带来的传输开销, 使得 FIFO 策略所能获得的调度性能总是优于其他两种调度策略, 与系统的类型及处理器资源的数量无关.

参考文献:

[1] V Bharadwaj, D Ghose, T G Robertazzi. Divisible load theory: a new paradigm for load scheduling in distributed systems[J]. Cluster Computing, 2003, 6(1): 7- 17.

[2] T G Robertazzi. Ten reasons to use divisible load theory[J]. IEEE Computer, 2003, 36(5): 63- 68.

[3] O Beaumont, et al. Scheduling divisible loads on star and tree networks: results and open problems[J]. IEEE Transactions on Parallel and Distributed Systems, 2005, 16(3): 207- 218.

[4] G Barlas. Collection-aware optimum sequencing of operations and closed-form solutions for the distribution of a divisible load on arbitrary processor trees[J]. IEEE Transactions on Parallel and Distributed Systems, 1998, 9(5): 429- 441.

[5] H J Siegel, J B Armstrong, D W Watson. Mapping computer-vision-related tasks onto reconfigurable parallel-processing systems[J]. IEEE Computer, 1992, 25(2): 54- 63.

[6] K Ko, T G Robertazzi. Equal allocation scheduling for data intensive applications[J]. IEEE Transactions on Aerospace and Electronic Systems, 2004, 40(2): 695- 704.

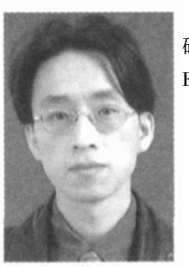
[7] X Li, V Bharadwaj, C Ko. Distributed image processing on a network of workstations[J]. Int J Computers and Applications, 2003, 25(2): 1- 10.

[8] D Altilar, Y Paker. An optimal scheduling algorithm for parallel video processing[A]. IEEE International Conference on Multimedia Computing and Systems[C]. Austin, TX, USA: IEEE Computer Society Press, 1998. 245- 248.

[9] M Drozdowski, P Wolniewicz. Experiments with scheduling divisible tasks in clusters of workstations[A]. Proceedings of the 6th Euro-Par Conference[C]. Munich, Germany: Springer, 2000. 311- 319.

[10] S Chan, V Bharadwaj, D Ghose. Large matrix-vector products on distributed bus networks with communication delays using the divisible load paradigm: performance and simulation[J]. Mathematics and Computers in Simulation, 2001, 58(1): 71- 92.

作者简介:



赵明宇　男, 1971 年生于哈尔滨, 博士生. 研究方向为分布式并行计算.
E-mail: zhmy@21cn.com.

张田文　男, 1940 年生于大连, 教授, 博士生导师. 主要研究领域为人工智能、图像处理及压缩、面向网络的大型多媒体信息分类及检索系统等.