第34卷 第24期 计 算 机 工 程 2008年12月
Vol.34 No.24 Computer Engineering December 2008
·软件技术与数据库· 文章编号：1000—3428(2008)24—0052—03 文献标识码：A 中图分类号：TP393

基于扩展的随机DAG的*EST*估算与任务调度

胡 凯，姜 燕，杨志斌，张新宇
(北京航空航天大学计算机学院，北京 100083)

摘 要：针对DAG调度算法中采取多次执行后的平均值估算任务的*EST*值问题，通过对DAG调度中常用的调度算法ETF算法进行分析提出基于扩展的随机DAG的调度方法SETF，给出扩展的随机DAG中节点的*EST*计算方法，以标准方差和平均值之和的数学期望表示，并以ETF算法为例进行实验模拟。实验结果表明，SETF算法相对于ETF算法，减少并行任务执行时间，并能更精确地预测任务调度的平均执行时间。

关键词：扩展的随机DAG；*EST*时间；ETF算法；SETF算法

EST Estimation and Tasks Scheduling Based on Expanded Stochastic DAG

HU Kai, JIANG Yan, YANG Zhi-bing, ZHANG Xin-yu
(School of Computer, Beijing University of Aeronautics and Astronautics, Beijing 100083)

【Abstract】Considering the fact that the *EST* value in scheduling DAG is composed of the means of computation and communication time, some researches on parallel tasks scheduling algorithms are done while one typical algorithm ETF is analyzed, and SETF algorithms for the expanded stochastic DAG is presented correspondingly. Then a method to compute the nodes' *EST* is provided. And experiments are done to simulate it. Experimental results indicate that a significant improvement in the average parallel execution times of expanded stochastic DAG can be achieved by the proposed approaches and it is able to more accurately predict the actual performance than the algorithms ETF.

【Key words】expanded stochastic DAG; *EST*; ETF algorithm; SETF algorithm

1 概述

自20世纪60年代以来，国内外的学者就开始研究怎样调度DAG任务图，并且诞生了很多调度算法。一般情况下，DAG调度是NP完全问题。目前的大部分DAG调度算法均基于启发式调度，包括表调度算法[1]、聚簇类算法[2]、基于任务复制的调度算法[3]以及基于遗传算法和随机搜索技术的调度算法[4]。

在上述DAG调度算法中，一般采用多次执行后的平均值估算任务的计算量和通信量，忽略每次执行时，计算环境的异构性或者并行算法带来的计算时间和通信时间偏移量，从后面的实验仿真结果可以看出这种算法得到的调度结果并不理想。目前，也有一些调度算法考虑了计算环境异构性带来的计算时间和通信时间偏移量，但只针对独立任务的情况作了研究，即只考虑任务间没有任何依赖关系的情况，而没有考虑任务之间的约束依赖关系。文献[5]给出了在分布式计算系统中的基于随机DAG模型的静态调度方法，考虑了计算环境异构性带来的计算时间的偏移量，意在最小化随机DAG的总的执行时间。

2 DAG调度

2.1 DAG模型术语及符号描述

DAG模型术语及符号描述如下：

(1) P_j：处理器j；

(2) n_i：随机DAG中的节点i；

(3) t_i：节点n_i的计算时间，由平均值m_i和偏移量σ_i组成；

(4) t_{ij}：从节点n_i到节点n_j的通信时间，由平均值m_{ij}和σ_{ij}组成；

(5)*EST*：任务节点被分配到某个可用节点上执行的最早时间；

(6)*SL*：基于扩展的随机DAG的调度长度；

(7)*PT*：通过DAG调度算法预测的并行执行时间。

2.2 DAG调度算法

任务图DAG中任一节点的执行只有在全部优先它的节点完成后才能开始，而节点的*EST*值总是直接或间接地用于决定各个节点的优先级。因此，针对DAG调度，一个关键的问题是精确地预估一些时间变量，如*EST*等。本文通过ETF调度算法对基于扩展随机DAG的并行任务调度算法进行了研究。

ETF算法首先计算准备就绪的节点的*EST*，选取其中*EST*值最小的任务节点在处理器上执行。算法描述如下：

(1)初始化就绪节点队列，只包括入口节点；

(2)计算就绪队列中各个节点的*EST*；

(3)选取最早可用的处理器并分配任务；

(4)更新就绪节点队列，添加完成步骤(3)以后变成就绪的节点；

基金项目：航空科学基金资助项目(20060151003, 2007ZC51032)
作者简介：胡 凯(1963－)，男，副教授、博士，主研方向：分布式并行计算；姜 燕，硕士研究生；杨志斌，博士研究生；张新宇，硕士研究生
收稿日期：2008-03-18 **E-mail：**ginger@cse.buaa.edu.cn

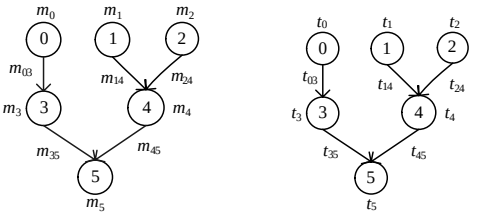
(5)重复步骤(2)~步骤(4)，直到所有节点被调度。

2.3 DAG 中 *EST* 的计算

以图 1(a)的 DAG 图为例，计算任务节点 *EST* 的算法如下：

(1)如果节点 n_i 是入口节点，则 n_i 的 *EST* 为 0，如图 1 中的节点 n_0 , n_1 , n_2 的 *EST* 即为 0；

(2)否则比较 n_i 的每个父节点的*EST*以及父节点和 n_i 之间的通信值之和，最大的值即为 n_i 的*EST*值，图 1 中节点 n_3 的*EST*表示为$E[m_0+m_3]$，节点n_4的*EST*为$E[\max(m_1+m_{14},m_2+m_{24})]$，节点$n_5$的*EST*值可以表示为$E[\max(m_3+m_{35},m_4+m_{45})]$，其中$E[]$表示求数学期望。



(a)DAG 模型　　(b)扩展的随机 DAG 模型

图 1　*EST* 时间计算

3　基于扩展的随机 DAG 的 ETF 算法

本文基于扩展的随机 DAG，结合第 2 节描述的 ETF 算法，给出了改进的调度算法，即 SETF 算法。

3.1　扩展的随机 DAG 中 *EST* 计算

扩展的随机DAG模型可以由图 1(b)[5]来表示。基于此图对扩展的随机DAG中节点的*EST*值的计算方法进行分析。

假设扩展随机DAG中 2 类随机性产生因素(计算环境的异构性及并行程序结构)可由向量$\boldsymbol{X}=(x_1,x_2)$表示，则节点 n_i 的计算时间 t_i 可以表示为向量$\boldsymbol{X}$的函数，即 $t_i(\boldsymbol{X})=m_i+\varepsilon$ ，其中，m_i为t_i的平均值 $m_i=\frac{1}{n}\sum_{k=1}^{n}t_ik$ ；ε 为向量$\boldsymbol{X}$产生的偏移量。

一般情况下，ε 的变化服从一个随机概率分布，很难对 ε 进行预测。Chebyshev 不等式表明，随机变量的标准方差 σ_i 和随机变量本身满足如下概率关系：

$$p[|t_i|\geq c\sqrt{\sigma_i}]\leq\frac{1}{c^2}$$

其中，$\sigma_i=\sqrt{\frac{1}{n}\sum_{k=1}^{n}(t_{ik}-m_i)^2}$ 。

比如，当取c值为 3，则 $p[|t_i|\geq 3\sigma_i]\leq 0.11$。该不等式显示了$t_i$的最大变化界限。在大部分情况下，$t_i$的变化界限会更小。如当$t_i$满足高斯随机分布时，$p[|t_i|\geq 3\sigma_i]\leq 0.998$。因此，在扩展的随机DAG模型中[6]，节点$n_i$的计算时间$t_i$可以表示为

$$t_i(\boldsymbol{X})=m_i+\sigma_i,(\sigma_i=\sqrt{\frac{1}{n}\sum_{k=1}^{n}(t_{ik}-m_i)^2})$$

由此，扩展的随机 DAG 中节点的 *EST* 值计算方法如下：

(1)当节点为入口节点时，其*EST*值仍为 0；否则比较n_i的每个父节点的*EST*以及父节点和n_i之间的通信值之和，最大的值即为n_i的*EST*值，此时任务的计算量和通信量为随机变量，例如节点n_4的*EST*为$E[\max(t_1+t_{14},t_2+t_{24})]$，节点$n_5$的*EST*值可以表示为$E[\max(t_3+t_{35},t_4+t_{45})]$。

(2)一个节点即使它的全部父节点已全部执行完，也有可能由于没有可用的处理器而不能被立即调度。因此，实际的*EST*计算还应该考虑处理器的最早可用时间，用ATP_i表示处理器i的最早可用时间，则处理器个数为n时节点n_4在处理器i的*EST*为

$$E[\max(t_1+t_{14},t_2+t_{24}),\min(ATP_i)],\ i=1,2,\cdots,n$$

3.2　算法分析

为了清楚地说明 ETF 算法以及 SETF 算法的区别，以图 2 的 DAG 模型为例进行研究分析。为了简化分析，任务间的通信时间忽略。

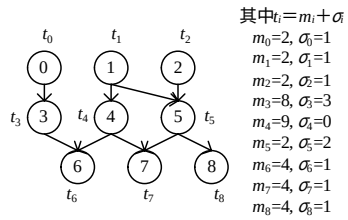
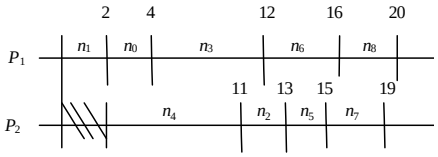
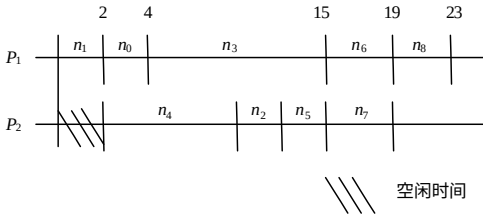


图 2　扩展的随机 DAG 模型及其计算权值与偏移量

对图 2 中 DAG 模型分别采用 ETF 算法和 SETF 算法进行调度，得到的调度结果(处理器个数取 2)如图 3 和图 4 所示。

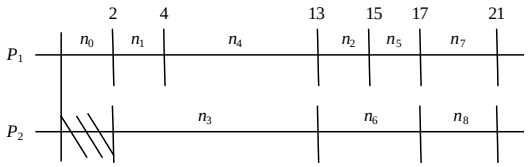


(a)由 ETF 算法得到的调度结果

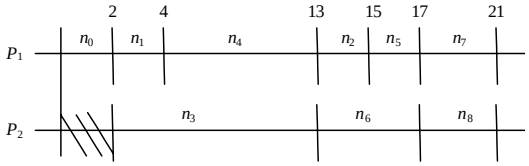


(b)实际分配在处理器上的执行结果

图 3　ETF 算法调度及执行结果



(a)由 SETF 算法得到的调度结果



(b)实际分配在处理器上的执行结果

图 4　SETF 算法调度及执行结果

在ETF调度算法中，任务节点的计算时间采用平均值表示，*EST*的计算也只和平均值相关。例如，节点n_3的*EST*为 4，其表达式为$E[\max(m_0,m_1,\min(ATP_i))]$，其中$i$=1,2。节点$n_6$的*EST*为$\max(m_0+m_3,m_1+m_4+\min(ATP_j))$，计算得 12，调度结果如图 3(a) 所示，*SL*=20。在实际 DAG 执行过程中，由于$E[\max(t_0+t_3,t_1+t_4+\min(ATP_j))]>\max(m_0+m_3,m_1+m_4+\min(ATP_j))$，节点$n_6$的*EST*将会大于 12。根据图 2 给出的$\sigma_i$ 值，$E[\max(t_0+t_3,t_1+t_4+\min(ATP_j))]=15$，此时，*PT*=23。而在SETF

算法中，计算节点n_6的EST值为$E[\max(t_0+t_3,t_1+t_4+\min(ATP_j))]$，如图 4 所示，由于节点$n_3$的计算时间 σ_3 最大，为 3，致使节点n_6的EST后延，因此在调用节点n_4之前先调用n_3，最后得到$SL=PT=21$。与ETF算法相比，SETF减少了并行任务执行时间。

4 实验仿真

4.1 仿真过程描述

为了更清楚地分析对比改进的调度方法(SETF)和原调度算法(ETF)，本文做了一组仿真实验，选取基于并行程序调度算法的一个重要衡量指标并行执行的时间PT，将SETF算法与ETF算法进行对比分析。仿真过程如下[5]：

(1)给 DAG 中节点和边赋予平均值和偏移量区间，表示其计算时间和通信时间的随机值。

(2)分别选取 ETF, SETF 调度算法对 DAG 中任务进行调度。

(3)计算由步骤(2)得到的调度结果的调度长度 SL。

(4)实际利用上述 2 种调度算法进行调度分配到处理器上执行得到其实际执行时间。

(5)重复步骤(4)得到并行程序的平均并行执行实际 PT。

(6)改变处理器个数，重复步骤(2)~步骤(5)。

在第(2)步中节点n_i计算时间和通信时间的值区间分别表示为$[t_{i\min},t_{i\max}]$及$[t_{ij\min}, t_{ij\max}]$。因此，节点n_i的计算时间和通信时间的平均值则可分别表示为 $\frac{t_{i\min}+t_{i\max}}{2},\frac{t_{ij\min}+t_{ij\max}}{2}$，相应的偏移量为 $\frac{t_{i\min}+t_{i\max}}{2\sqrt{3}}$ 及 $\frac{t_{ij\min}+t_{ij\max}}{2\sqrt{3}}$。

4.2 仿真结果及分析

仿真时选取的 DAG 拓扑结构图以及 DAG 中节点的计算时间数据见图 5 和表 1。

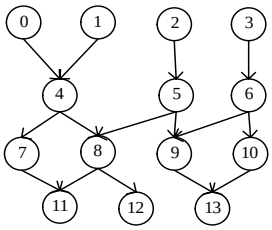


图 5 扩展的随机 DAG 模型

表 1 DAG 节点计算时间数据区间值

节点号	计算时间区间	m_i	σ_i
0	2~4	3.0	0.577
1	2~5	3.5	0.866
2	3~5	4.0	0.577
3	2~4	3.0	0.577
4	4~8	6.0	1.155
5	4~9	6.5	1.443
6	3~8	5.5	1.443
7	2~6	4.0	1.155
8	3~7	5.0	1.155
9	5~10	7.5	1.443
10	4~8	6.0	1.155
11	3~9	6.0	1.732
12	3~9	6.0	1.732
13	4~9	6.5	1.443

仿真结果如图 6 和图 7 所示。从图 6 和图 7 可以看出，SETF 算法的 PT 值和 SL 值的差值比 ETF 算法的 PT 值和 SL 值的差值小。这是由于在 SETF 中考虑了计算时间的随机性，利用一个偏移量 σ_i 加上平均值 m_i 来表示，而 ETF 算法的计算时间只考虑平均值 m_i，忽略了其随机性，会出现过早调度，对 PL 值的计算也就相对缩短，因此差值就较大。这也证明了扩展的随机 DAG 模型可以更精确地预估任务节点的 EST 值，在此基础上进行任务调度，能够更准确地预测调度结果，即 PT 值。

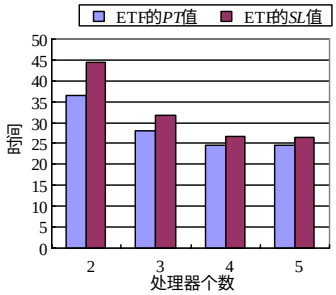


图 6 ETF 算法的 *PT* 值和 *SL* 值

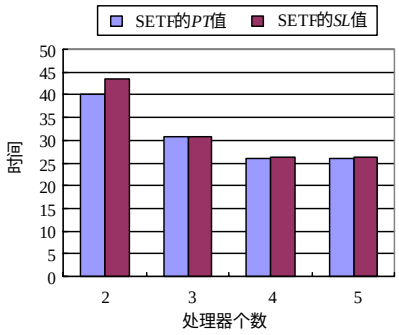


图 7 SETF 算法的 *PT* 值和 *SL* 值

5 结束语

针对实际运行时并行程序结构和并行环境的异构性 2 种因素带来的计算时间的偏移量，提出了基于扩展的随机 DAG 的调度方法 SETF，给出了关键的 EST 时间计算方法，并以 ETF 算法为例进行实验模拟。研究及实验结果表明，SETF 算法相对于 ETF 算法，可有效减少并行任务执行时间，并能更精确地预测任务调度的平均执行时间。

参考文献

[1] Grajcar M. Genetic List Scheduling Algorithm for Scheduling and Allocation on a Loosely Coupled Heterogeneous Multiprocessor System[C]//Proc. of the 36th Design Automation Conference. New Orleans, USA: [s. n.], 1999: 280-285.

[2] Chan W Y, Li C K. Heterogeneous Dominant Sequence Cluster(HDSC): A Low Complexity Heterogeneous Scheduling Algorithm[C]//Proc. of IEEE Pacific Rim Conference on Communications, Computers and Signal Processing. Victoria, Canada: IEEE Press, 1997: 956-959.

[3] Kwok Y. Parallel Program Execution on a Heterogeneous PC Cluster Using Task Duplication[C]//Proc. of the 9th Heterogeneous Computing Workshop. Cancun, Mexico: [s. n.], 2000: 364-374.

[4] Woo S H, Yang S B, Kim S D, et al. Task Scheduling in Distributed Computing Systems with a Genetic Algorithm[C]//Proc. of High Performance Computing Workshop on the Information Superhighway. Seoul, Korea: [s. n.], 1997: 301-305.

[5] Kamthe A, Lee S Y. A Stochastic Approach to Estimating Earliest Start Times of Nodes for Scheduling DAGs on Heterogeneous Distributed Computing Systems[C]//Proc. of the 19th IEEE International Parallel and Distributed Processing Symposium. Los Alamitos, USA: IEEE Computer Society Press, 2005: 121-122.

[6] 胡 凯, 姜 燕, 陈诗然, 等. 一种扩展的随机 DAG 模型[J]. 北航学报, 2008, 34(4): 400-404.