软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2013,24(2):378−390 [doi: 10.3724/SP.J.1001.2013.04224]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

单层树型网格下独立任务的周期性调度*

王振宇[1], 李照瑜[2]

[1](华南理工大学 软件学院,广东 广州 510006)

[2](华南理工大学 计算机科学与工程学院,广东 广州 510006)

通讯作者: 王振宇, E-mail: wangzy@scut.edu.cn, http://www.scut.edu.cn

摘 要: 提出单层树型网格下单位独立任务的周期性调度方法,单位独立任务是大小相等的独立任务.首先,为单层树型网格下的单位独立任务调度建立线性规划模型,通过分析整数线性规划求解过程,发现一个单层树型网格平台在节点构成不同时,分别具有非饱和态、临界态或冗余态特征;并且,随着网格节点上任务数的增多,线性规划最优解呈线性增长,任务调度具有周期性特性.据此给出非饱和态、临界态或冗余态网格的定义、性质和判定方法,推导出单位独立任务调度的周期长度.最后,分析了周期性调度的时间复杂性,提出一种周期性调度算法 Periodic-Sched.实验结果表明,周期性调度是有效的.单位独立任务的周期性调度将大规模的任务调度问题简化为一个周期内的任务调度,降低了调度问题的复杂度.该调度方法适用于对 Hadoop 平台的 Map 任务进行调度.

关键词: 树型网格;独立任务;周期性调度;整数线性规划;Map-Reduce

中图法分类号: TP316 **文献标识码**: A

中文引用格式: 王振宇,李照瑜.单层树型网格下独立任务的周期性调度.软件学报,2013,24(2):378−390. http://www.jos.org.cn/1000-9825/4224.htm

英文引用格式: Wang ZY, Li ZY. Scheduling periodic independent tasks on single-level tree grid. Ruanjian Xuebao/Journal of Software, 2013,24(2):378−390 (in Chinese). http://www.jos.org.cn/1000-9825/4224.htm

Scheduling Periodic Independent Tasks on Single-Level Tree Grid

WANG Zhen-Yu[1], LI Zhao-Yu[2]

[1](School of Software Engineering, South China University of Technology, Guangzhou 510006, China)

[2](School of Computer Science and Engineering, South China University of Technology, Guangzhou 510006, China)

Corresponding author: WANG Zhen-Yu, E-mail: wangzy@scut.edu.cn, http://www.scut.edu.cn

Abstract: This paper suggests that the periodic scheduling of identical independent tasks on a single-level tree grid and identical independent tasks are independent tasks in the same size in computation. First, an integer linear programming model for scheduling identical independent tasks is presented. By analyzing the procedure of solving the linear programming model, and obtaining the optimal number of tasks assigned to each computing node, the study finds that a single-level tree grid composed of different nodes will be in one of three states, respectively: unsaturated, critical, and redundant states. Furthermore, the optimal number of tasks assigned to each node increases linearly as the number of tasks arriving to the master grows, and the periodic feature appears in the task scheduling. Next, some features and theories that determine the states of grid systems among unsaturated, critical, and redundant states are obtained, periodic scheduling of identical independent tasks on single-level grid is proposed, the length of period is derived. Their computational complexity is also analyzed, and a periodic scheduling algorithm, Periodic-Sched, is given. Both the theory and experiments have proved that the periodic scheduling is feasible. In this way, the problem of large-scale independent task scheduling is simplified and resolved in one period, which reduces the computational complexity dramatically. The periodic scheduling of independent tasks is suitable for Map tasks scheduling on Hadoop platform.

* 基金项目: “核高基”国家科技重大专项(2012ZX01039-004-03-2); 广东省教育部产学研合作专项(2012B091100420)
收稿时间: 2011-10-18; 定稿时间: 2012-04-01

Key words: tree-based grid; independent task; periodic scheduling; integer linear programming; Map-Reduce

网格下一般任务调度问题是 NP 完全问题[1],独立任务调度问题一直是网格调度的研究热点之一[2].独立任务是指相互之间没有依赖关系,且不可再划分的任务,其调度问题的研究有重要应用价值.例如,Hadoop 平台采用 Map-Reduce 分布式计算模型将应用分解成小的任务,任务之间相对独立且大小相同.Hadoop 平台运行众多典型应用,如文本分类、数据挖掘等,研究独立任务调度对提升此类应用的性能有重要意义.

在同构网格下调度同构任务(计算量相同),最优解可通过轮转法得到[3].异构任务(计算量不同)调度问题属于 NP 难问题.当问题规模较小时,常采用穷举、分支限界、动态规划等方法;当问题规模较大时,多采用启发式或智能算法求近似解.树型异构平台下的独立同构任务调度同样是 NP 难问题[4],已提出的多数调度算法属于启发式算法,如 Min-Min 算法[2]、Max-Min 算法[2]以及复杂的遗传算法[5,6]、量子算法[7]、A^*算法等;也有部分算法研究精解,使用动态规划[8]、分支限界[9]等方法,但算法复杂,计算开销大,调度性能与简单算法相比提升有限.

周期性思想被用来解决可分任务的多趟(multi-installment)调度,通过多个周期间的合理衔接来最小化节点的计算空闲.文献[10]提出一种异构环境下的较优调度算法,任务完成时间划分为多个相同的周期,各周期内任务分发顺序和数量相同,通过数学方法找到最小化完成时间的周期长度,确定合适的任务划分.文献[11]采用的多趟调度算法属于周期性调度,其周期大小可变,一趟调度内,处理机获得一个相等大小的任务并计算,各趟间任务大小不相同.可分任务多趟调度中,确定周期长度是关键.将周期性调度引入独立任务调度的研究较少.

本文首先描述问题的调度模型,给出同构单层树型网格中单位独立任务的周期性调度示例;研究异构单层树型网格下单位独立任务的周期性调度的特性;为最优任务分配数求解建立整数线性规划模型,分析求解过程我们发现,在节点构成不同时,网格呈现出不同的状态特性.在单层树型网格中,线性规划最优解具有稳定性,网格平台最大完成任务数与时间增长呈线性关系;推导出周期长度,提出一种周期调度算法 Periodic-Sched.最后,讨论周期性调度对 Hadoop 平台的 Map 任务调度的应用价值.

1 调度模型

多层树型网格结构如图 1 所示.文中研究的网格限于单层树型结构[12,13],如图 2 所示,是基于树结构的网络资源集成模型,适用于多种编程范型,如主/从(master-slave)、RPC(remote procedure call)、分治等[13].本文着重考虑主/从模型.网格由 k 个计算节点组成,节点计算单位任务所需时间分别为 $w_0,w_1,\ldots,w_{k-1}$,到达任务最初位于根节点(master)n_0 上,由 n_0 负责传输单位任务给各个子节点 n_i,所需时间分别为 $c_1,\ldots,c_{k-1}$(n_0 自身的传输时间记为 c_0,令 c_0=0),通信服从单口模式(single-port model)[14].使用 $n_i(w_i,c_i)$表示单个节点.节点的通信计算比小于 1.调度对象为计算量大小相同的独立任务,一个任务只能在 1 个节点上完成计算,且一个节点只能同时执行 1 个任务.节点具有缓存、通信和计算可同时进行,一个任务必须等待传输完成后才开始运行.算法以最短完成时间(makespan)作为性能指标,最短完成时间是一个给定应用或任务集中最后一个任务的结束时间.

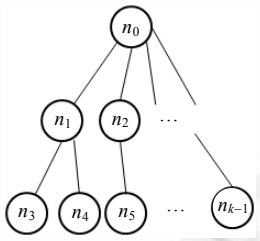


Fig.1 Multi-Level tree grid computing platform model

图 1 多层树型网格计算平台模型

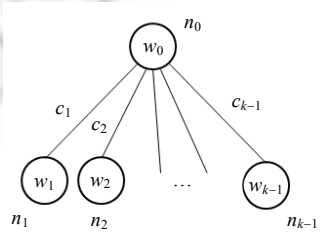


Fig.2 Single-Level tree grid computing platform model

图 2 单层树型网格计算平台模型

2 单层树型网格下的独立任务周期性调度

2.1 同构平台下调度的周期性特征

对于同构的单层树型网格,轮转法可得到最优解[12]:根节点开始传输任务,系统首先进入准备期,子节点顺序等待首个任务的到来,部分子节点通信和计算空闲;然后,子节点依次收到首个任务并开始运行,系统稳定地完成计算任务,进入稳定期.系统在稳定期显现出周期性特征:存在一个时间间隔θ,在θ内,系统完成任务数的数量和调度顺序一致,θ可被视为系统的调度周期.图 3 为 $n_0,\ldots,n_3$ 这 4 个节点组成的同构网格的调度,节点通信和计算速度分别为 3 和 10.图中每个节点对应两条时间水平线,分别表示任务的计算时间与通信时间.系统在 0~12 时间区间处于准备期.该平台可容纳的最多的计算节点数为 4,调度周期为 12.一个周期最多可完成 4 个任务,平台的最大计算速度为 1/3.

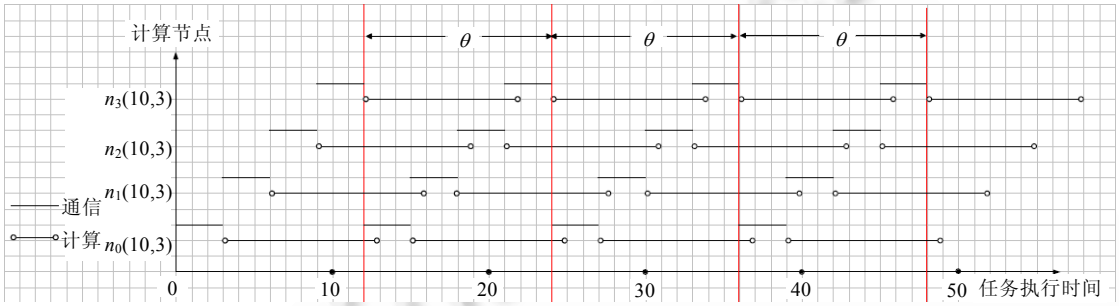


Fig.3 A sample of periodic scheduling on homogeneous platform

图 3 同构平台下周期性调度示例

2.2 异构平台下调度的周期性特征

线性规划模型可以解决异构平台下最优任务数求解问题[13,15,16],给出的启发信息可以改善调度效果[13,16],但是这类方法难以适应任务数庞大且动态增长的情况.若能把大规模的独立任务调度在时间上分割为多个周期,每个周期内任务调度顺序相同,则只需求解出单个周期内任务的最优调度,按周期重复,可得到整个问题的较优任务调度.

2.2.1 使用线性规划求解调度问题

求解给定 M 个任务的最短完成时间问题[13,16]可转化为求解时间 T 内最大的任务完成数问题.该问题的整数线性规划方程见公式(1):

(a) 节点 n_i 完成的任务数记为 x_i,所有节点周期内执行的任务数之和等于 M;

(b) 单个节点完成任务数小于总完成任务数;

(c) 根节点 n_0 执行任务的时间小于等于总任务完成时间;

(d) 除根节点外的每个节点执行任务的总时间小于等于时间 T 减去该节点首个任务的通信等待时间;

(e) 所有任务的通信之和小于时间 T.

方程组中,T,c_i,w_i,k 为已知量,x_i 为变量,Maximize M 为目标函数.

$$\begin{cases}(a)\ \sum_{i=0}^{k-1}x_i=M\\(b)\ 0\leqslant x_i\leqslant M\ \text{for}\ 0\leqslant i\leqslant k-1\\(c)\ w_0\times x_0\leqslant T\\(d)\ w_i\times x_i\leqslant T-c_i\ \text{for}\ 1\leqslant i\leqslant k-1\\(e)\ \sum_{j=1}^{k-1}c_j\times x_j\leqslant T\\(f)\ M,T,c_i,w_i,k,x_i\text{均为正整数}\end{cases}\tag{1}$$

2.2.2 线性规划求解分析

公式(2)是公式(1)的标准型[17],有 $2k+1$ 个变量,决策变量$\{x_0,\ldots,x_{k-1}\}$是各节点完成的任务数,$\{x_k,\ldots,x_{2k}\}$为松弛变量,目标函数是 Minimize $(-M)$.最优解的基由 $k+1$ 个基向量组成,基向量共有 C_{2k+1}^{k+1} 种组合.由于约束(c)、约束(d)相似,变量划分为决策变量$\{x_0,\ldots,x_{k-1}\}$,约束(c)、约束(d)对应的松弛变量$\{x_k,\ldots,x_{2k-1}\}$和约束(e)对应的松弛变量 x_{2k}.下面考察基这 3 组变量对应的基向量所占数量,分析线性方程组最优解的特性.

$$\begin{cases}(a)\ -\sum_{i=0}^{k-1}x_i=-M\\(b)\ 0\leqslant x_i\leqslant M\ \text{for}\ 0\leqslant i\leqslant k-1\\(c)\ x_0+\frac{x_k}{w_0}=\frac{T}{w_0}\\(d)\ x_i+\frac{x_{k+i}}{w_i}=\frac{T-c_i}{w_i}\ \text{for}\ 1\leqslant i\leqslant k-1\\(e)\ \sum_{j=1}^{k-1}c_j\times x_j+x_{2k}=T\\(f)\ M,T,c_i,w_i,k,x_i\text{均为正整数}\end{cases}\tag{2}$$

公式(3)是从公式(2)化简的标准型线性规划问题(LP)[17].

$$\left.\begin{array}{l}\min \boldsymbol{Cx}\\ \text{s.t. } \boldsymbol{Ax}=\boldsymbol{b}\end{array}\right\}\tag{3}$$

- $\boldsymbol{A}$ 为系数矩阵,$\boldsymbol{A}_{(k+1)\times(2k+1)}=(\boldsymbol{a}^1,\ldots,\boldsymbol{a}^{2k+1})=(a_{ij})_{m\times n}$,$\boldsymbol{x}=(x_0,\ldots,x_{2k})$,$\boldsymbol{C}=(c_0,\ldots,c_{2k})^T$,$\boldsymbol{b}=(b_0,\ldots,b_k)^T$;
- 假设 $\boldsymbol{x}^{(0)}$为 LP 的一个基解,I 为非基变量的上标集合,$I=\{i_0,\ldots,i_{k-1}\}$,$\boldsymbol{N}=\{\boldsymbol{a}^i:i\in I\}$;
- J 为基变量的上标集合,$J=\{j_0,\ldots,j_k\}$,$\boldsymbol{B}=\{\boldsymbol{a}^j:j\in J\}$,$\boldsymbol{C}_B$ 和 $\boldsymbol{x}_B$ 分别为 $\boldsymbol{C}$ 和 $\boldsymbol{x}$ 中相应于 $\boldsymbol{B}$ 的分量.

Case 1:$I=\{0,\ldots,k-1,2k\}$,即所有决策变量$(x_0,\ldots,x_{k-1})$与松弛变量 x_{2k} 为基变量.

基向量$\{\boldsymbol{a}^0,\ldots,\boldsymbol{a}^{k-1},\boldsymbol{a}^{2k}\}$构成满秩的$(k+1)\times(k+1)$矩阵,把 LP 化为对应于基 $\boldsymbol{B}$ 的典式[17]后得到增广矩阵(4).

$$\begin{array}{c}\begin{array}{cccccccccccc}x_0 & x_1 & \ldots & \ldots & x_{k-1} & x_k & x_{k+1} & \ldots & \ldots & x_{2k-1} & x_{2k} & b\end{array}\\ \left[\begin{array}{cccccccccccc}1 & 0 & \ldots & \ldots & 0 & \frac{1}{w_0} & 0 & \ldots & \ldots & \ldots & 0 & \frac{T-c_0}{w_0}\\ 0 & 1 & 0 & & \vdots & 0 & \frac{1}{w_1} & & & & \vdots & \frac{T-c_1}{w_1}\\ \vdots & & & \ddots & \vdots & \vdots & & \ddots & & & \vdots & \vdots\\ \vdots & & & & 1 & \vdots & & & \frac{1}{w_{k-1}} & & \vdots & \frac{T-c_{k-1}}{w_{k-1}}\\ 0 & 0 & \ldots & \ldots & 0 & -\frac{c_0}{w_0} & -\frac{c_1}{w_1} & \ldots & \ldots & -\frac{c_{k-1}}{w_{k-1}} & 1 & T-\sum_{i=0}^{k-1}\frac{(T-c_i)c_i}{w_i}\end{array}\right]\end{array}\tag{4}$$

在其最右列中,令 $T-\sum_{i=0}^{k-1}\frac{(T-c_i)c_i}{w_i}=\alpha,\frac{T-c_i}{w_i}=\beta_i(i=0,\ldots,k-1)$.

方程组(2)的单纯型表见表 1.

对于基 $\boldsymbol{B}$,若 $\boldsymbol{B}^{-1}\boldsymbol{b}\geqslant 0$,且判别数 $\boldsymbol{\lambda}_N=\boldsymbol{C}_B\boldsymbol{B}^{-1}\boldsymbol{N}-\boldsymbol{C}_N\leqslant\boldsymbol{0}$,则对应于 $\boldsymbol{B}$ 的基解 $\boldsymbol{x}^{(0)}$是 LP 的最优解[17].

由表中最后一行可知,判别数 $\boldsymbol{\lambda}$ 皆小于 0,满足 $\boldsymbol{\lambda}_N\leqslant\boldsymbol{0}$.

讨论条件 $\boldsymbol{B}^{-1}\boldsymbol{b}\geqslant 0$.因 $\boldsymbol{B}$ 为单位阵,$\boldsymbol{b}$ 是基变量的解,因此 $\boldsymbol{B}^{-1}\boldsymbol{b}\geqslant 0$ 表示所有基变量的解大于等于 0.

查看表 1,当 $T\geqslant\max\{c_i\}$时,满足 $\beta_i\geqslant 0(i=0,\ldots,k-1)$,下文可默认 $\beta_i\geqslant 0(i=0,\ldots,k-1)$.

- 若 $\alpha\geqslant 0$,即 $\boldsymbol{B}^{-1}\boldsymbol{b}\geqslant 0$,$\boldsymbol{B}$ 成为最优基,方程组对应最优解的分量 $x_{2k}=\alpha$;
- 若 $\alpha<0$,说明 x_{2k} 为非基变量,最优解的分量 $x_{2k}=0$.

Table 1 Simplex tabuleau of Case 1

表 1 Case 1 的单纯型表

c_B	基	$\boldsymbol{b}$	基变量					非基变量			
			x_0	x_1	…	x_{k-1}	x_{2k}	x_k	x_{k+1}	…	x_{2k-1}
−1	x_0	β_0	1	0	…	0	0	$\frac{1}{w_0}$	0	…	0
−1	x_1	β_1	0	1	…	0	0	0	$\frac{1}{w_1}$	…	0
…	…	…	…	…	…	…	…	…	…	…	…
−1	x_{k-1}	β_{k-1}	0	0	…	1	0	0	0	…	$\frac{1}{w_{k-1}}$
0	x_{2k}	α	0	0	…	0	1	$-\frac{c_0}{w_0}$	$-\frac{c_1}{w_1}$	…	$-\frac{c_{k-1}}{w_{k-1}}$
	$\boldsymbol{\lambda}$		0	0	…	0	0	$-\frac{1}{w_0}$	$-\frac{1}{w_1}$	…	$-\frac{1}{w_{k-1}}$

Case 2:$I=\{0,\ldots,k-1,k+m\}$,即所有决策变量$(x_0,\ldots,x_{k-1})$与除松弛变量 x_{2k} 外的一个松弛变量构成基变量.该松弛变量记为 $x_{k+m}(m<k)$,x_{k+m} 所在行对应的决策变量记为 x_m.

基向量$\{\boldsymbol{a}^0,\ldots,\boldsymbol{a}^{k-1},\boldsymbol{a}^{k+m}\}$也构成满秩的$(k+1)\times(k+1)$矩阵,LP 化为对应于基 $\boldsymbol{B}$ 的典式.由 Case 1,当 x_{2k} 为非基变量时,$\alpha<0$, $\boldsymbol{B}^{-1}\boldsymbol{b}=\left[\beta_0,\ldots,\beta_m+\frac{\alpha}{c_m},\ldots,\beta_{k-1},-\frac{w_m\alpha}{c_m}\right]^{-1}$. 其中, $\beta_m+\frac{\alpha}{c_m}$ 等于 x_m.因 Case 1 中约定$\beta_i(i=0,\ldots,k-1)$非负,故除 x_m 外,$\boldsymbol{B}^{-1}\boldsymbol{b}$ 中所有分量非负.若基 $\boldsymbol{B}$ 为最优基,则有 $x_m\geqslant 0$ 且 $\frac{c_i}{w_i c_m}-\frac{1}{w_i}\leqslant 0(i=0,\ldots,k-1)$ 成立,即要满足 $c_m\geqslant c_i$,可推出 $c_m=\max\{c_i\}$.此时已确定 c_m 的值,可计算得到 x_m 的取值.若 $x_m\geqslant 0$,则 $\boldsymbol{B}$ 为最优基.

Case 3:$I=\{0,\ldots,n-1,n+1,\ldots,k-1,k+m,k+l\}$,即 2 个以上松弛变量为基变量,1 个以上决策变量为非基变量.此处重点讨论 2 个松弛变量 x_{k+m} 与 x_{k+l} 为基变量,1 个决策变量为非基变量的情形,其余类似.

若 Case 2 中解得 $x_m<0$,对应的基 $\boldsymbol{B}$ 非最优基,即决策变量 $x_0,\ldots,x_{k-1}$ 中有 1 个变量成为离基变量[16],而有相应个数的松弛变量成为进基变量.被换出的决策变量记为 x_l,若除 x_l 外的所有决策变量与某个松弛变量构成最优基,则需满足 $\boldsymbol{\lambda}_N\leqslant\boldsymbol{0}$,即公式(5),可得 $c_l\geqslant c_m\geqslant c_i$,$c_m=\max\{c_i\}$.当由 $I=\{0,\ldots,n-1,n+1,\ldots,k-1,k+m,k+l\}$对应的基向量组成最优基时,在最优解中非基变量 x_l 为 0.把一个决策变量变成离基变量的情形推广到多个决策变量发现:随着节点数的增加,决策变量按照通信速率的排序,从大到小依次成为离基变量.

$$\begin{cases}\dfrac{c_i}{w_i c_m}-\dfrac{1}{w_i}\leqslant 0\ (i=0,\ldots,k-1,i\neq l,m)\\ 1-\dfrac{c_l}{c_m}\leqslant 0\end{cases}\tag{5}$$

Case 4:$I=\{0,\ldots,m-1,m+1,\ldots,k-1,k+m,2k\}$,对应 x_{2k} 与其余任一松弛变量成为基变量的情形.

因 $x_m=\beta_m+\frac{\alpha}{c_m}, x_{m+k}=-\frac{w_m}{c_m}\alpha$, 若基 $\boldsymbol{B}$ 为最优基,则 $x_{k+m}>0$,表示 α 小于 0.该情形与 Case 1 得出的结论矛盾,

故 Case 4 不会出现.

2.2.3 最优解的线性增长特性

当公式(2)中的 T 变化时,问题演变为参数线性规划问题[17],该参数线性规划方程如公式(6):

$$\left.\begin{array}{l}\min \boldsymbol{Cx}\\ \text{s.t. } \boldsymbol{Ax}=\boldsymbol{b}(T)\end{array}\right\} \tag{6}$$

设 $\boldsymbol{b}(T)=\boldsymbol{b}+\boldsymbol{F}\cdot T$,$\boldsymbol{F}$ 为 T 的系数矩阵,且令 $Z^{(J)}(T)$为相应于 $\boldsymbol{B}_J$的最优值函数.由于最优值函数在相应的最优基的临界区域上是线性的[17],且有

$$Z^{(J)}(T)=\boldsymbol{C}_B\boldsymbol{B}_J^{-1}\boldsymbol{b}+\boldsymbol{C}_B\boldsymbol{B}_J^{-1}\boldsymbol{F}\cdot T \tag{7}$$

假如使 T 的增量θ固定,即有 $Z^{(J)}(T+\theta)-Z^{(J)}(T)=\boldsymbol{C}_B\boldsymbol{B}_J^{-1}\boldsymbol{F}\cdot\theta$, 则当 LP 最优基不变时,最大完成任务数与给定的时间 T 呈线性关系:当 T 的增量θ固定时,最优值的增量为常量.在最优调度下,当网格进入稳定期后,每经过时间θ,系统可稳定地完成 $\boldsymbol{C}_B\boldsymbol{B}_J^{-1}\boldsymbol{F}\cdot\theta$ 个任务,也即网格的最大计算速度为 $\boldsymbol{C}_B\boldsymbol{B}_J^{-1}\boldsymbol{F}$, 记为 $v_{\max}$.最优解的线性增长特性表明,网格在一个固定时间间隔θ内的任务调度具有周期性.

2.2.4 网格平台状态分析

网格进入稳定期后,计算速度达到最大,网格显现出稳定的状态特征.下面定义网格的不同资源利用状态,分析网格的状态特性与网络构成的内在关系,提出网格状态的判别方法.

定义 1(最小有效节点集). 从单层树型网格 G 中选择一组子节点,与根节点构成集合 V,$V=\{n_i,i\in\{0,\ldots,k-1\}\}$.当集合 V 构成的子网格与原网格的最大计算速度相同,且 V 中元素个数最少时,称为最小有效节点集.

定义 2(有效节点/无效节点). 最小有效节点集中的节点称为有效节点,反之称为无效节点.

线性规划最优解中非 0 的决策变量对应有效节点,为 0 的决策变量对应无效节点.在线性规划存在多个最优解时,最小有效节点集不唯一,这时选定某个有效节点集作为网格 G 的最小有效节点集,其中的节点是有效节点,其他节点为无效节点.在单层树型网格中,新增的子节点对网格最大计算速度的提升作用受制于根节点的通信能力.

定义 3(非饱和态/饱和态). 对于单层树型网格 G,若任意加入一个子节点,G 的最大计算速度增加,称 G 处于非饱和态;反之称 G 处于饱和态.

处于非饱和态的单层树型网格扩展子节点的能力强于饱和态的单层树型网格.

定理 1. 一个单层树型网格 G,若$\alpha>0$ 成立,则 G 处于非饱和态;反之则处于饱和态.

证明:若节点 n_i 在首个任务到达后开始运行,且不再出现空闲等待时间,则称 n_i 满负荷运行.

$\alpha=T-\sum_{i=0}^{k-1}\frac{(T-c_i)c_i}{w_i}$ 中,$\frac{(T-c_i)}{w_i}$ 表示节点 n_i 满负荷运算时可处理的任务数,α表示根节点的通信空闲时间.若在运行时间 T 内α恒为正,即根节点满足各子节点满负荷运行的通信需求后仍存在通信空闲时间,那么向网格中新增的任意子节点,都可获得任务分配,从而提升网格的最大计算速度,故网格处于非饱和态.若运行时间 T 内,存在时间点 t_0 使$\alpha\leqslant 0$,即根节点不能满足已有子节点满负荷运算的通信需求,则不存在通信空闲时间.若网格中新增的子节点的计算性能弱于网格中已有子节点,则为其分配任务将导致其余计算性能较强的子节点获分配的任务数减少,从而使网格的最大计算速度下降,故网格处于饱和态.证毕. □

当系统长时间运行时,可假定 T 趋于无穷,得出推论 1:

推论 1. 一个单层树型网格 G,令 $\chi=\left(1-\sum_{i=0}^{k-1}\frac{c_i}{w_i}\right)$,若$\chi>0$,则 G 处于非饱和态;反之则处于饱和态.

当$\alpha<0$ 时,平台处于饱和态,可能出现的情形有:系统中节点数等于或大于系统有效节点个数.下面讨论等于的情形(大于的情形在冗余态讨论),与第 2.2.2 节的 Case 2 对应,x_m 表达式变换后得公式(8):

$$x_m=\frac{1}{c_m}\left[T-\sum_{i=0,i\neq m}^{k-1}\frac{(T-c_i)c_i}{w_i}\right] \tag{8}$$

$\sum_{i=0,i\neq m}^{k-1}\frac{(T-c_i)c_i}{w_i}$ 对应根节点与除通信时间最长的节点 n_m 之外,所有节点的通信时间总和.

若 $x_m\geqslant 0$,则 $T-\sum_{i=0,i\neq m}^{k-1}\frac{(T-c_i)c_i}{w_i}\geqslant 0$, 表明根节点的通信满足除节点 n_m 外所有节点满负荷运算的通信需求.

在 Case 2 中,由于满足 x_{2k} 不为基变量,有 $\alpha<0$,得公式(9),可知系统已不满足所有节点满负荷运转的通信需求,可容纳的有效节点数达到临界态.从 $x_m+\frac{x_{m+k}}{w_m}=\frac{T-c_m}{w_m}$ 且 $x_{k+m}\geqslant 0$,推出 $x_m\leqslant\frac{T-c_m}{w_m}$, 即通信速度最慢的子节点属于有效节点,但并非满负荷运行,这样的节点称为饥饿节点.饱和态网格进一步细分为临界态和冗余态.

$$\begin{cases}T-\sum_{i=0,i\neq m}^{k-1}\frac{(T-c_i)c_i}{w_i}\geqslant 0\\ T-\sum_{i=0}^{k-1}\frac{(T-c_i)c_i}{w_i}<0\end{cases}\tag{9}$$

定义 4(临界态). 一个单层树型网格 G,若处于饱和态且所有子节点皆为有效节点,则称 G 处于临界态.

从第 2.2.2 节的 Case 2 可知,最优解中所有决策变量取值非 0,最小有效节点集包括全体子节点.另外,饱和态网格中的每个子节点,除饥饿节点外每个节点最优计算任务数为 $\frac{(T-c_i)}{w_i}$, 计算能力已达到上限.

非饱和态及临界态对应 Case 1 与 Case 2.在最优解中,每个决策变量取值非 0,最小有效节点集由全体子节点组成.

性质 1. 处于非饱和及临界态的单层树型网格 G,最小有效节点集为 $\{n_0,\ldots,n_{k-1}\}$,即 G 中所有节点.

性质 2. 处于饱和态的单层树型网格 G,通信时间最长的节点是饥饿节点.在系统计算速度达到最大值时,除饥饿节点外,其余节点皆满负荷运转.

定理 2. 一个单层树型网格 G,令 $\delta=1-\sum_{i=0,i\neq m}^{k-1}\frac{c_i}{w_i}$,若满足 $\chi<0$ 且 $\delta\geqslant 0$,则 G 处于临界态.

证明:当 $\chi<0$,T 趋于无限时有 $\alpha\leqslant 0$,系统处于饱和态.把第 2.2.2 节 Case 2 中的 x_m 公式(8)变换可得

$$x_m=\frac{1}{c_m}\left(\delta T+\sum_{i=0,i\neq m}^{k-1}\frac{c_i^2}{w_i}\right).$$

除 δ 外,其余项皆非负,因此若 $\delta\geqslant 0$,则必有 $x_m\geqslant 0$.由 $\alpha<0$,$x_m\geqslant 0$,从 Case 2 可知,此时所有决策变量 $(x_0,\ldots,x_{k-1})$ 皆为基变量,在最优解中值非 0,故 G 处于临界态. □

若网格 G 中有效节点数达到上限后继续扩展子节点,则会出现无效节点.此时,网格中的节点数大于有效节点的个数.

定义 5(冗余态). 一个单层树型网格 G,若处于饱和态且有无效节点,则称 G 处于冗余态.

由 Case 3 可知,在最优解中,所有被换出成为非基变量的决策变量取值为 0,系统中通信时间最长的节点成为无效节点.

性质 3. 处于冗余态的单层树型网格 G,节点按照通信速度递增进行排序,$p_0,\ldots,p_{k-1}$.若最小有效节点集的元素个数为 q,那么最小有效节点集为 $\{p_0,\ldots,p_{q-1}\}$.

定理 3. 一个单层树型网格 G,若 $\delta<0$,则 G 处于冗余态.

证明:若 $\delta<0$,则当 T 趋于无穷时,有 $\alpha<0$ 且 $x_m<0$.由 $\alpha<0$ 可知 G 处于饱和态;参见 Case 3,当 $x_m<0$ 时,x_m 会被换出,成为非基变量,其在最优解中值为 0,所以决策变量 x_m 对应的节点为无效节点.证毕. □

网格状态与第 3.2.2 节线性规划最优解的各情形相对应.随着子节点数的增加,网格状态将依次从非饱和态转变为临界态,再到冗余态.

2.3 异构平台下的周期性调度

2.3.1 网格拓扑的剪枝

在冗余态网格中,通过删减网格中的无效节点,网格将从冗余态转变成临界态,网格结构得到优化,这个过程称为剪枝.由性质 3 可知,冗余态网格中无效节点通信速度最慢,结合定理 2 和定理 3,逐个删除冗余态网格中通信速度最慢的子节点,同时监视χ和δ的取值变化.当满足定理 2 的条件时,表明网格从冗余态转化为临界态,在不改变平台最大计算速度的前提下减少了拓扑结构的冗余.

2.3.2 周期的选取

下面研究非饱和态与临界态的调度周期选择.利用最大处理任务数的线性增长特性,在时间增量为θ时,通过公式(7)分别求得非饱和态和临界态下对应完成任务数ρ的表达式为公式(10)和公式(11):

$$\sum_{i=0}^{k-1}\frac{1}{w_i}\cdot\theta \tag{10}$$

$$\left[\frac{1}{c_m}+\sum_{i=0}^{k-1}\frac{1}{w_i}\left(1-\frac{c_i}{c_m}\right)\right]\cdot\theta \tag{11}$$

网格 G 处于非饱和态或临界态,在构成节点无变化时,LP 的解在时间 T 趋于无限的情况下依然保持为正,故最优基不随 T 的增加而改变.ρ与θ相关,并可通过合适的θ来确定ρ的大小.从网格任务调度的误差考虑θ的取值,不考虑机器由于负载增加而导致的节点通信和计算时间变化,调度误差主要由两部分构成:(1) 最优调度与能够实际采用的近似最优调度之间的误差.按照 LP 的解,在最优情形下,网格能在时间增量θ内完成ρ个任务,若按已知的近似最优调度次序完成ρ个任务,所需的最短时间为$\theta+\eta$,η表示最优调度与近似最优调度之间的误差;(2) 使用整数线性规划求解网格调度问题,其整数最优解与实际最优解之间存在误差,时间上的误差记为μ.调度误差在对调度方案做周期性扩展时有扩大的趋势:每扩展一个周期,调度误差增加$\eta+\mu$,当时间 T 增大时,总的调度误差值可观.对于η,可通过寻找更优的调度次序来缩小实际任务完成时间,使其更接近θ;对于μ,可通过选择θ的值来减小.观察公式(10)和公式(11),若令公式(10)中θ的取值为 $w_i(i=0,\ldots,k-1)$的最小公倍数,公式(11)的θ为 c_m 与 $w_i(i=0,\ldots,k-1)$最小公倍数的乘积,则实际最优解为整数,此时η等于 0,因此使用最小公倍数为时间增量,将使周期调度带来的误差最小化.同时,每个节点的最优任务分配数为整数,使得每个周期结束时,子节点当前无通信或计算任务,确保周期间相互独立和顺利衔接.θ与ρ可通过预先计算获得.

2.3.3 周期性调度的时间复杂性分析

线性规划模型求解的经典算法[18]的时间复杂度为 $O(n^{3.5}L)$,n 为计算节点个数,L 为线性规划方程组输入的规模.但根据定理 1~定理 3,可在常数时间内判别出网格的状态,并通过 x_i 解表达式及公式(10)、公式(11)求得各节点的最优任务分配数及网格的周期长度,求解复杂度下降为 $O(n)$.得到调度周期后,周期内可完成的最大任务数ρ可确定.此时,周期内任务的调度顺序有待确定:需寻找一个任务调度序列,使网格能够在周期θ内完成ρ个任务的计算.由于平台通信串行进行,任务的调度顺序等同于根节点分发任务给 $k-1$ 个子节点的顺序.根节点上运行的任务数是 x_0,需分发任务数是$\rho-x_0$,这时,寻找周期内任务调度顺序的时间复杂度为 $O((k-1)^{\rho-x_0})$. 若以节点最优任务分配数为启发式信息,则搜索空间缩小至 $\prod_{i=0}^{k-2}C^{x_{i+1}}_{\rho-\sum_{0}^{i}x_j}=\frac{(\rho-x_0)!}{x_1!x_2!\ldots x_{k-1}!}$. 此时求解复杂度仍很高,仍需寻找智能算法,获取周期内的较优调度.当通过预计算得到周期内调度顺序后,运行时的调度时间复杂度为 $O(1)$.

2.3.4 周期性调度算法 Periodic-Sched

由上一节可知,根节点分发$\rho-x_0$ 个任务的次序即为周期内调度顺序,下面给出一种周期性调度算法 Periodic-Sched.首先,求解一个周期内的调度顺序;在其余周期内复制该调度顺序,得到全局调度.对于冗余态网格,在运行调度算法之前,需先对网格拓扑做剪枝,去除无效节点,使网格转化为临界态.

寻找周期内的调度采用树遍历的方法.对一棵高度为$\rho-x_0$ 的 $k-1$ 叉搜索树,树中每个非叶子节点有 $k-1$ 个子节点,编号为 $1,\ldots,k-1$.非叶子节点与其第 i 号子节点 p_i 相连的边权值为根节点 n_0 与节点 n_i 的任务通信时间

c_i.目标是寻找一条由根节点至叶子节点的长度不大于周期长度θ的路径 Path,Path 中记录的节点顺序是根节点n_0与子节点的通信顺序,即周期内的任务调度顺序.遍历时,使用以下启发式信息:

(a) 非饥饿节点的计算不停顿;

(b) 非饥饿节点必须在结束当前计算任务之前完成下一个待处理任务的通信;

(c) 在开始计算前,每个节点需预先完成首个任务的通信;

(d) 各节点的最优任务分配数;

(e) 路径 Path 长度不大于θ.

非饥饿节点在一个周期内完成的任务个数为θ/w_i,即节点无空闲,得到信息(a);节点有缓存,非饥饿节点须使前一任务的计算与下一任务通信完全并行,得到信息(b);为解决周期间衔接问题,采用预通信,即在首个周期前,每个节点需预先完成第 1 个任务的通信,在后续周期中,各节点的首个任务已经在前一个周期传输完毕,周期开始时即可开始计算,得到信息(c);由线性规划最优解,得到信息(d);周期长度为θ,得到信息(e).

以根节点为当前节点,按以下步骤遍历,Path 初始值为空:

(1) 若当前节点没有未遍历的子节点,则回溯至其父节点,若父节点为空,则搜索失败,退出.

(2) 选取当前节点的一个未遍历的子节点 $p_i(i=1,\ldots,k-1)$,并把其状态设置为已遍历.

(3) 判断当前 Path 上编号为 i 的节点出现次数是否等于节点 n_i 的最优任务分配数,若等于,则返回步骤(1).

(4) 判断与 n_i 通信是否符合启发信息(a)、信息(b),若不符合,则返回步骤(1).

(5) 判断若把 p_i 加入 Path,Path 长度是否大于θ,若大于,则返回步骤(1).

(6) 把 p_i 加入 Path,判断 Path 中的节点个数是否等于$\rho-x_0$:若不等于,则返回步骤(1);否则,当前 Path 即为所求路径.

2.3.5 多层树型网格下的周期性调度

上述结果可推广到多层树型网格.多层树结构可看成是由一层层单层树结构构成.通过迭代,树型结构可自下而上层层“上推”,收缩成单层树结构;单层树结构沿着“上推”的逆过程“下拉”,恢复成原来的树型结构,称为 Push-Pull 方法[16,19].Push-Pull 方法中,每个两层子树用一个相应的等价节点 Enode(虚拟子节点)代替,Enode 与父节点的通信和计算能力近似等价于该两层子树作为一个整体与父节点的通信和计算能力.Enode 的通信和计算能力可由 BUFF-O-R-SLTN 算法求得[19].在多层树型网格中,应用上推方法把多层树转化成一棵单层树结构,求解其周期长度和调度顺序;下拉过程把虚拟子节点向下层层展开,可迭代求得原多层树中各节点的最优任务分配数,实现周期性调度.

3 调度实例及实验

3.1 调度实例

在图 4 中,$\chi=0.275$,为非饱和态网格;调度周期为 $w_i(i=0,\ldots,k-1)$的最小公倍数,即 40;一个周期内最大完成任务数为 17.根据一些启发信息,可得出符合线性规划模型最优解的调度次序,图 6 是图 4 的任务调度顺序,包括两个调度周期.在图 4 中添加节点 n_3,得到图 5,$\chi=-0.225<0$,$\delta=0.375>0$,处于临界态.调度周期为 c_m 与 $w_i(i=0,\ldots,k-1)$的最小公倍数的乘积,即 120,周期内最大完成任务数为 72.可找到一个周期内的调度,如图 7 所示.

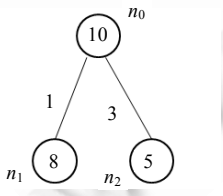


Fig.4 A single-level tree grid under un-saturated state

图 4 非饱和态单层树型网格

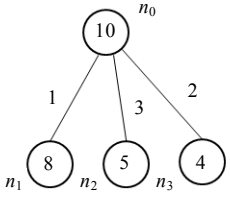


Fig.5 A single-level tree grid under critical state

图 5 临界态单层树型网格

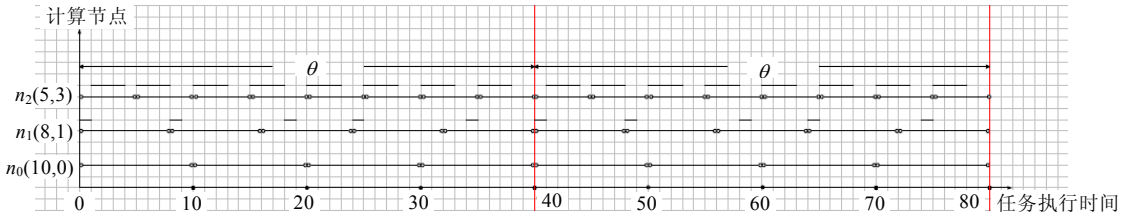


Fig.6 A sample of periodic scheduling on single-level tree grid under un-saturated state

图 6 非饱和态单层树型网格下周期性调度实例

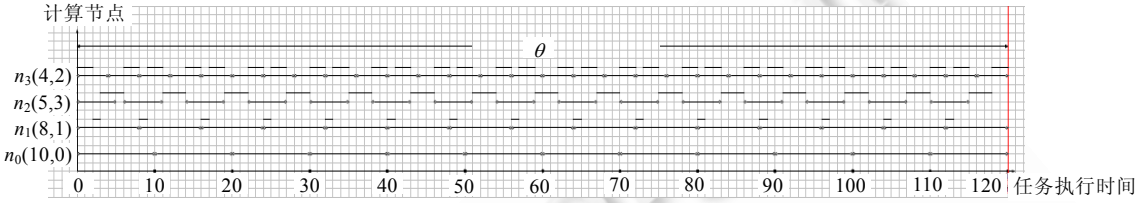
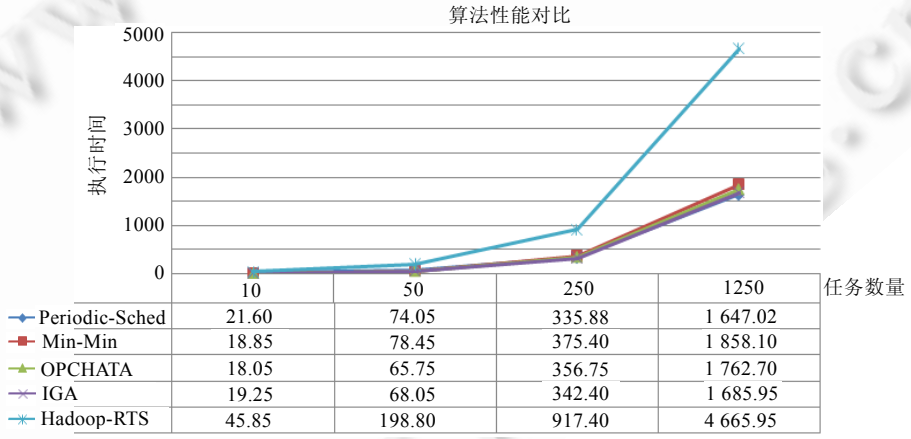


Fig.7 A sample of periodic scheduling on single-level tree grid under critical state

图 7 临界态单层树型网格下周期性调度实例

3.2 实验结果与分析

选取常用的评测基准[2,20]算法 Min-Min、与本文相近的 OPCHATA 算法[12]、遗传算法 IGA[2,6]、Hadoop 任务调度算法 Hadoop-RTS(hadoop random task scheduler)[21]与 Periodic-Sched 做比较.随机生成 20 组临界态网格,使用上述 4 种算法实现 10 个、50 个、250 个、1 250 个单位独立任务的调度,其中,节点计算时间是[1,20]之间,通信时间是[0,15]之间.各算法的平均任务完成时间如图 8 所示.



	10	50	250	1250
Periodic-Sched	21.60	74.05	335.88	1 647.02
Min-Min	18.85	78.45	375.40	1 858.10
OPCHATA	18.05	65.75	356.75	1 762.70
IGA	19.25	68.05	342.40	1 685.95
Hadoop-RTS	45.85	198.80	917.40	4 665.95

Fig.8 Comparison of experimental results of task scheduling on single-level tree grid under critical state

图 8 临界态单层树型网格下任务调度算法的实验结果对比

由实验结果得出:

1) 当任务数量较少时(小于 50),周期性调度算法的平均任务完成时间多于其余算法;
2) 当任务数量较多时(大于 250),周期性调度算法的平均完成时间少于其余 3 种算法;
3) 当任务数量较多时,任务数越多,周期性调度算法的性能优势越明显;
4) Hadoop-RTS 算法性能最差,运行时间约为其余算法的 2.5 倍.

周期性调度算法以周期为单位,通过不断重复ρ个任务的调度顺序实现全局调度:

- 当任务数量较少且不等于ρ的倍数时,采用周期性调度算法获得的未必是较优调度序列,得出结论 1).
- 周期性调度算法通过扩展局部较优调度序列得到全局较优的调度序列,Min-Min 与 OPCHATA 算法属于带启发式信息的贪心算法,任务调度时只考虑当前最优的结果.当任务数量较多时,考虑全局因素的周期性调度算法性能更优,得出结论 2).
- 随着周期数的增加,周期性调度算法的性能优势逐渐积累,待处理任务的规模越大,周期性调度算法比其余算法的优势越明显,得出结论 3).
- Hadoop 平台中,Hadoop-RTS 算法随机选择数据块文件的存储节点,任务的传输与执行分开进行(在第 3.3.2 节深入分析),得出结论 4).

实验结果表明了周期性调度算法的有效性,揭示出其在处理大规模任务调度时的性能优势.

3.3 算法应用

3.3.1 Hadoop 平台结构特征

Hadoop 平台支持 Map-Reduce 分布式计算模型[21],适用于对海量数据的并行处理,例如气象数据挖掘、倒排索引、词频统计(WordCount,统计文本文件中单词的出现次数)等.Hadoop 的作业被分成多个独立任务,主节点 JobTracker 负责作业调度和任务分配,从节点 TaskTracker 通过“心跳”向 JobTracker 请求任务并执行[21],Hadoop HDFS(hadoop distributed file system)实现数据文件的分布存储.HDFS 是 Master/Slave 架构,包括一个主节点(namenode)和多个从节点(datanode),数据文件被分割成一系列大小相等的数据块,基于容错和性能考虑,每个数据块可有多个副本(缺省为 3),副本分布存储在 Datanode 上,Namenode 负责调度数据块的存放.

Hadoop 平台一般由多个机架构成,同一机架存放多个 Datanode,机架间形成相对独立的分组.客户端每当写入一个数据块时,首先从 Namenode 得到该数据块及相应副本存放位置,然后创建数据块文件及副本.Namenode 的数据块分配策略是:在与客户端相同的节点上放置第 1 个副本,第 2 个副本放置在随机选择的、不同于第 1 个副本的机架上,第 3 个副本存放在与第 2 个副本相同的机架上,但放在不同的节点.若有更多副本,则被随机放置在集群中的节点上.

运行 Map-Reduce 作业时,每个 map 任务处理一个数据块文件.JobTracker 负责任务调度,TaskTracker 运行 Map 任务.当一个 TaskTracker n'发出请求时,JobTracker 调度 n'上待处理数据块文件对应的 map 任务,在 n'运行.Datanode 节点又作为 TaskTracker 运行.当完成 Map 任务数达到某个阈值后,JobTracker 可调度 Reduce 任务在某个 TaskTracker 上运行.在第 3.2 节关于 Hadoop-RTS 算法的实验中,设定数据块副本数为 1,且不考虑机架内 Map 任务的迁移.这时,Hadoop 拓扑结构符合单层树型结构,Hadoop 中,Client,JobTracker,Namenode 对应单层树型结构的根节点,Datanode 及 TaskTracker 对应单层树的儿子节点.

3.3.2 Hadoop 的任务调度

Hadoop Map-Reduce 分布式并行编程模型如下:

map: $(K_1,V_1)\rightarrow list(K_2,V_2)$;

reduce: $(K_2,list(V_2))\rightarrow list(K_3,V_3)$;

以 WordCount 为例:

map:(文本行在文件中的字节偏移量,文本行内容)$\rightarrow$*list*(单词,单词个数);

reduce:(单词, *list*(单词数))$\rightarrow$*list*(单词,单词的总出现次数).

一个待处理的大型文本文件被 Hadoop 分割成大小相等的数据块文件存储在 HDFS 中.TaskTracker 开始执行一个 Map 任务之后,将数据块文件预处理成(K_1,V_1),即(文本行在文件中的字节偏移量,文本行内容)列表,交给 map 函数处理,对每个键值对执行一次 map 函数;map 函数对文本行做分词,得到$\langle K_2,V_2\rangle$列表,如〈“Hello”,1〉.例如,两个数据块文件 file01 和 file02:

- file01: Hello World Bye World\n Hello;
- file02: Hello Hadoop Goodbye Hadoop.

file01 的 map 任务把 file01 预处理成〈0,“Hello World Bye World”〉,〈23,“Hello”〉;交给 map 函数处理,得到 $list(K_2,V_2)$:〈“Hello”,1〉,〈“World”,1〉,〈“Bye”,1〉,〈“World”,1〉,〈“Hello”,1〉;将生成的 $list(K_2,V_2)$按 K_2 排序分组,得到:〈“Bye”,1〉,〈“Hello”,2〉,〈“World”,2〉.file02 的 map 任务的处理过程类似.Map-Reduce 框架将两个 map 任务产生的 $list(K_2,V_2)$传递给执行 reduce 任务的节点,reduce 任务汇总所有 map 任务的输出结果,得到$(K_2,list(V_2))$:〈“Bye”,1〉,〈“Goodbye”,1〉,〈“Hadoop”,2〉,〈“Hello”,〈2,1〉〉,〈“World”,2〉,reduce 函数对同一个单词的数目求和,得到 $list(K_3,V_3)$:〈“Bye”,1〉,〈“Goodbye”,1〉,〈“Hadoop”,2〉,〈“Hello”,3〉,〈“World”,2〉.

从以上分析得出,Hadoop 中 Map 任务是典型的单位独立任务,单层树型网格下单位独立任务周期性调度适用于 Hadoop 平台中的 Map 任务调度.使用 Periodic-Sched 算法可明显改善 Hadoop 的 Map 任务调度能力.

4 结束语

在为单层树型网格平台下的单位独立任务调度建立线性规划模型的基础上,本文通过分析整数线性规划求解过程得出,一个网格平台在节点构成不同时,分别具有非饱和态、临界态或冗余态特征,给出各状态的定义及资源利用特征.分析线性规划最优解的线性增长特性,给出单层树型网格最大计算速度的度量方法.提出在异构平台下周期性调度方法,并推导出周期的长度,提出了一种周期调度算法 Periodic-Sched.周期性调度的提出,使得大规模任务的调度简化为周期内任务调度.文中提出的调度方法具有可预计算、可适应任务数动态增长等优点,实用性较强.实验结果显示,独立任务的周期性调度是可行的、较优的.研究结果可较容易地推广到普通树型网格,方法适用于 Hadoop 平台的 Map 任务调度.

下一步将研究周期内任务调度算法以及基于资源状态特征的拓扑结构优化问题.首先,需要改进 Periodic-Sched 算法,研究计算强度可接受的在周期内寻找最优或较优调度顺序的智能算法;其次,网格在冗余态时存在无效节点,为了提高网格的资源利用率,可利用饱和态和冗余态的特性,对单层树型网格的结构进行优化;最后,进一步研究周期性调度算法对 Hadoop 机群拓扑结构的适应性,改进 Hadoop 平台的 Map 任务调度算法.

References:

[1] Abraham A, Buyya R, Nath B. Nature’s heuristics for scheduling jobs on computational grids. In: Proc. of the 8th Int’l Conf. on Advanced Computing and Communications (ADCOM 2000). New Delhi: Tata McGraw-Hill Publishing, 2000. 45−52.

[2] Braun TD, Siegel HJ, Beck N. A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing systems. Journal of Parallel and Distributed Computing, 2001,61(6):810−837. [doi: 10.1006/jpdc.2000.1714]

[3] Beanumont O, Boudet V, Petitet A, Rastello F, Robert Y. A proposal for a heterogeneous cluster ScaLAPACK (dense linear solvers). IEEE Trans. on Computers, 2001,50(10):105−1070. [doi: 10.1109/12.956091]

[4] Dutot P. Complexity of master-slave tasking on heterogeneous trees. European Journal on Operational Research, 2005,164(3):690−695. [doi: 10.1016/j.ejor.2003.10.051]

[5] Vincenzo DM, Marco M. Sub optimal scheduling in a grid using genetic algorithms. Parallel Computing, 2004,30(5-6):553−565. [doi: 10.1016/j.parco.2003.12.004]

[6] Lin JN, Wu HZ. Scheduling in grid computing environment based on genetic algorithm. Journal of Computer Research and Development, 2004,41(12):2195−2199 (in Chinese with English abstract).

[7] Mo Z, Wu GW, He YH, Liu HW. Quantum genetic algorithm for scheduling jobs on computational grids. In: Proc. of the Int’l Conf. on Measuring Technology and Mechatronics Automation. 2010. 964−967. [doi: 10.1109/ICMTMA.2010.505]

[8] Karger D, Stein C, Wein J. Scheduling algorithms. 2010. http://theory.lcs.mit.edu/~karger/Papers/schdeuling.ps.gz

[9] Dell’ Amico M, Martello S. Optimal scheduling of tasks on identical parallel processors. ORSA Journal on Computing, 1995,7(2):686−689. [doi: 10.1287/ijoc.7.2.191]

[10] Beaumont O, Legrand A, Robert Y. Scheduling divisible workloads on heterogeneous platforms. Parallel Computing, 2003,29(9):1121−1152. [doi: 10.1016/S0167-8191(03)00095-4]

[11] Yang Y, Krijin R, Henri C. Multi-Round algorithms of scheduling divisible loads. IEEE Trans. on Parallel and Distributed Systems, 2005,16(11):1092−1101. [doi: 10.1109/TPDS.2005.139]

[12] Baumont O, Casanova H, Legrand A, Robert Y, Yang Y. Scheduling divisible loads on star and tree networks: Results and open problems. IEEE Trans. on Parallel and Distributed Systems, 2005,16(3):207−218. [doi: 10.1109/TPDS.2005.35]

[13] Lin WW, Qi DY. Li YJ, Wang ZY, Zhang ZL. Independent tasks scheduling on tree-based grid computing platforms. Ruanjian Xuebao/Journal of Software, 2006,17(11):2352−2361 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/17/2352.htm [doi: 10.1360/jos172352]

[14] Fujita S. A fault-tolerant broadcast scheme in the star graph under the single-port, half-duplex communication model. IEEE Trans. on Computers, 1999,48(10):1123−1126. [doi: 10.1109/12.805160]

[15] Moges MA, Yu D, Robertazzi TG. Grid scheduling divisible loads form two sources. Computers and Mathematics with Applications, 2009,58(6):1081−1092. [doi: 10.1016/j.camwa.2009.07.046]

[16] Wang ZY, Luo XS. Practical scheduling algorithm of independent tasks on tree-based grid. Journal of South China University of Technology, 2008,36(4):56−62 (in Chinese with English abstract).

[17] Jiang JS, He CX, Pan SH. Computational Methods for Optimization. Guangzhou: South China University of Technology Press, 2004. 43−73 (in Chinese).

[18] Karmarkar N. A new polynomial-time algorithm for linear programming. Combinatorica, 1984,4(4):373−395. [doi: 10.1145/800057.808695]

[19] Veeravalli B, Yao J, Divisible load scheduling strategies on distributed multi-level tree networks with communication delays and buffer constraints. Computer Communications, 2004,27(1):93−110. [doi: 10.1016/S0140-3664(03)00181-6]

[20] Ibarra OH, Kim CE. Heuristic algorithms for scheduling independent tasks on nonidentical processors. Journal of the ACM, 1977, 24(2):280−289. [doi: 10.1145/322003.322011]

[21] White T. Hadoop: The Definitive Guide, 2nd ed., Sebastopol: O'Reilly Press, 2011. 41−73.

附中文参考文献:

[6] 林剑柠,吴慧中.基于遗传算法的网格资源调度算法.计算机研究与发展,2004,41(12):2195−2199.

[13] 林伟伟,齐德昱,李拥军,王振宇,张志立.树型网格计算环境下的独立任务调度.软件学报,2006,17(11):2352−2361. http://www.jos.org.cn/1000-9825/17/2352.htm [doi: 10.1360/jos172352]

[16] 王振宇,罗晓生.树型网格下独立任务的使用调度算法.华南理工大学学报,2008,36(4):56−62.

[17] 蒋金山,何春雄,潘少华.最优化计算方法.广州:华南理工大学出版社,2007.43−73.

王振宇(1966－),男,河南许昌人,博士,教授,博士生导师,主要研究领域为分布式计算,操作系统与资源虚拟化,Web 文本挖掘.

E-mail: wangzy@scut.edu.cn

李照瑜(1986－),男,博士生,主要研究领域为网格计算,分布式计算.

E-mail: lizhaoyu.ctit@gmail.com