Dependent Tasks Scheduling Algorithm for Multi-Agent System and Network based Workshop

Huang Jian-she[1],Tao Jian-wen[2]
([12]Computer Application Research Institute of Zhejiang Business Technology Institute, Zhejiang Ningbo 315012, China)
1.e-mail: hjs@zjbti.net.cn, 2. E-mail:tjw@zjbti.net.cn

Abstract: This paper researches the dependent tasks parallel scheduling problem of multi-agent and network based workshop jobs system. An optimal dependent tasks scheduling algorithm for multi-agent and network based workshop jobs system, namely multi-agent and network based workshop dependent tasks balanced and compressed scheduling algorithm, is presented in the paper. Using multi-job's balanced boost attains the target of using agents efficiently with regarding scheduling efficiency as scheduling criterion. Simultaneously using static compressed algorithm compresses scheduling length much more to improve utilization ratio of agents. The experimental result shows that the algorithm has less scheduling length and higher utilization ratio of agents than before.

Key words: Multi-Agent system; scheduling algorithm; network based workshop; Task allocation

基于多 Agent 的网络化车间相关任务调度算法

黄建设[1], 陶剑文[2]
（[12]浙江工商职业技术学院计算机应用研究所，浙江 宁波 315012）
1.e-mail: hjs@zjbti.net.cn, 2. E-mail:tjw@zjbti.net.cn

【摘要】研究在基于多Agent的网络化车间作业系统中相关任务的并行调度问题，提出了一种优化的多Agent网络化车间相关任务并行调度算法——基于多Agent网络化车间相关任务均衡-压缩调度算法。以调度效率作为标准，通过追求多组作业的均衡推进来达到有效利用Agent时间的目的，同时利用静态压缩算法，来进一步压缩调度长度，提高了各Agent的总利用率。实验显示该算法具有较短的调度长度和较高的Agent利用率。

【关键词】多 Agent 系统;调度算法;网络化车间;任务分配

1 引言

车间调度问题对制造过程具有重要的实际意义，已经被证明是一个复杂的 NP-hard 问题。一般的研究方法包括数学规划、规则调度、启发式调度、仿真方法、遗传算法、神经网络算法及模拟退火算法等[8]。随着计算机技术和网络通讯技术的发展，车间内相关的制造设备可以通过局域网相互连接起来，为实现车间制造系统的网络化提供了最基本的硬件平台。在此基础上，运用面向对象技术构造车间应用软件系统，将原有的应用系统封装起来，经过 Agent 化处理，形成多 Agent 系统（MAS）。

在网络技术的支撑下，运用多 Agent 技术对车间调度系统建模，将车间中的各类对象转化为 Agent，形成开放的、一致的多 Agent 系统框架，为有效实现网络化车间任务动态调度提供了新的途径。本文提出的基于多 Agent 的调度系统模型如图 1 所示。

基金项目：浙江教育厅科研基金项目（Y200803936）
Foundation Item: Research of Education Office of Zhejiang Province(Y200803936)

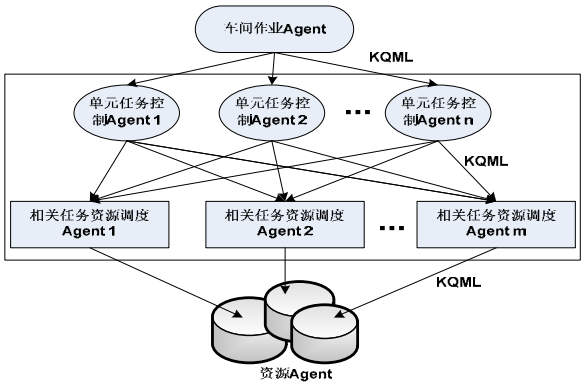

图 1 基于 MAS 的网络化车间相关作业调度模型

该模型分为 3 个层次：上层是车间作业 Agent，负责从上层或外界接受任务信息，进行加工能力、加工成本及利润的判断，负责车间级零件任务的排队，并通过网络协调与合作伙伴的关系；中间层由动态单元任务控制 Agent 和相关资源调度 Agent 构成，其根据动态到来的零件作业即时生成，并负责某一零件作业的资源组织和协调；下层是代表多种制造设备的资源 Agent。

2 相关研究分析

关于一组相关任务的调度问题，文献[1]改进了E．C．Horvath 提出的将任务按偏序关系规定位级数的方法(改进的位级数抢先调度算法)。文献[2]从时间角度对多机相关任务提出均衡调度算法。文献[3]则从时间和空间两方面考虑对相关任务提出相关矩阵调度算法。关于多组作业的并行调度问题，文献[4]针对批量执行同一相关任务的多机系统，提出了一种优化策略和组织方法。文献[5]对在有限个(s>0)同构处理机上执行 (>s)个并行任务的调度问题，给出了非抢先调度下的一个优化算法。

本文认为，在基于 MAS 网络化车间作业处理系统中，存在多个功能相同或不同的 Agent，而所处理的作业中有的含有多个资源相关的任务（Task）。本文提出一种优化的基于多 Agent 的网络化车间相关任务调度算法——多 Agent 网络化车间相关任务均衡-压缩调度算法 [2][3]，该算法缩短了作业的执行周期，进一步提高了 Agent 的总利用率。

3 相关概念

在一个基于MAS的网络化车间作业系统中，MAS表示为(A，P)[4]，A={A1，A2，…，An}是一作业集，A中任意作业$A_i=\{J_{i1},J_{i2},J_{iqi}\}$是一个资源相关的任务集，用前趋图($A_i$，<)表示；P={P1，P2，…，Pm }是一个MAS系统，由m个解题能力各异的Agent构成。前趋图Ts(T，<)是描述作业T的有向图[2]。图中每个结点对应作业中的一个任务（task），结点权量是任务的运行长度，表示任务需用的单位执行时间UET(unit executing time)，图中的有向线表示任务间的偏序关系<：令s,t∈T，若存在一条从s到t有向线，则表示s是t的直接前驱，t是s的直接后继，记作s<t。

定义1 对作业(A_i，<)，从任务J_{ij}的有效源点到任务J_{ij}的最长路径的长度(这里所说的路径长度是指路径上各任务运行长度之和)叫做任务J_{ij}的最早开始时间，记为ET_{ij} 。若任务J_{ij}的运行长度为T_{ij} ，则称$ET_{ij}+T_{ij}$ 为任务J_{ij}的最早结束时间，记作TS_{ij} 。显然存在如下关系：

$$TS_{ij} := T_{ij} + \max\{TS_{ik} \mid J_{ik} < J_{ij} \ ,\ 1\le k\le Q_i\ ,Q_i \in Z^+\} \quad (1)$$

定义2 对作业(A_i，<)，在任务J_{ij}的所有直接前驱中，最迟结束的任务叫做任务J_{ij}的关键前驱，记作KeyPre(J_{ij})。

定义3 在作业(A_i，<)中，最迟结束任务的最早结束时间叫做作业A_i的最早完成时间，记作$T_i:=\max\{TS_{ij}\}$ 2)

定义4 对作业(A_i，<)，任务J_{ij}的最早结束时间与作业A_i的最早完成时间的比值叫做作业A_i调度到第j道任务时的调度效率，记作$r(i,j):=TS_{ij}/T_i$ (3)

定义5 对J_{is}，$J_{it}\in A_i$，若$r(i,s)\le r(i,t)$，则称任务J_{is}，J_{it} 具有关系“≤”，记作$J_{is}\le J_{it}$ 。

定理1 关系“≤”是任务集A_i上的一个全序。

证明 根据调度效率的定义，对任意J_{is}，$J_{it}\in A_i$必有实数r(i，s)，r(i，t)与其对应。因为r(i，s)，r(i，t)是实数，所以r(i，s)≤r(i，t)或者r(i，s)≥r(i，t)。又由定义1知$J_{is}\le J_{it}$，或者$J_{it}\le J_{is}$，所以关系“≤”是任务集A_i的一个全序。证毕。

由定理1知，在任务集A_i上，可由关系“≤”构造一全序。

定理2 在任务集A_i上，由关系“≤”构造的全序序列$\{J_i^{(1)},J_i^{(2)},\ldots J_i^{(Qi)}\}$ $(J_i^{(j)}\le J_i^{(j+1)}$,j=1，2，…，Q_i-1)是前趋图(A_i，<)的一个拓扑有序序列。

证明 在前趋图(A_i，<)中，对J_{is}，$J_{it}\in A_i$，若$J_{is}<J_{it}$，则$TS_{is}\in\{TS_{ik} \mid J_{ik}<J_{it},\ 1\le k\le Q_i,\ Q_i\in Z^+\}$。又由式(1)知：

$$TS_{it}:=T_{it}+\max\{TS_{ik} \mid <J_{ik}<J_{it},\ 1\le k\le Q_i\ ,\ Q_i\in Z^+\}.$$

所以$TS_{is}<TS_{it}$，进而有$TS_{is}/T_i<TS_{it}/T_i$ ，即r(i，s)<r(i，t)；又根据定义1，有$J_{is}\le J_{it}$。

这就说明在前趋图(A_i，<)中，凡是具有偏序关系“<”的序偶，都具有全序关系“≤”。所以，由关系“≤”构造的全序序列$\{J_i^{(1)},J_i^{(2)},\cdots J_i^{(Qi)}\}(J_i^{(j)}\le J_i^{(j+1)}$,j=1，2，…，Q_i-1)是前趋图(A ，<)的一个拓扑有序序列。证毕。

由定理2知，任务集A_i上的全序$\{J_i^{(1)},J_i^{(2)},\ldots J_i^{(Qi)}\}$能够体现各任务间的偏序关系。用$\{P_{ij}\}$表示具有求解任务$J_i^{(j)}$能力的Agent集合，$\{P_{ij}\}\subseteq\{P_1，P_2，\cdots，Pm\}$。

为衡量各Agent的使用情况，我们给出Agent的占用量和利用率的定义。

定义6 在选出$J_i^{(j)}$要调度时，第k个Agent上最后一个执行任务的实际结束时间叫做第k个Agent此刻的占用量，记作$L_k(i，j)$。

为叙述方便，用$L_{max}(i，j)$表示此时任务$J_i^{(j)}$可用Agent集合$\{P_{ij}\}$中占用量的最大值，显然$L_{max}(i，j)=\max\{L_k(i，j)\}$。另外，用$FT_{ij}$表示任务$J_i^{(j)}$的实际结束时间。如果当系统选出$J_i^{(j)}$要调度时，第k个Agent上最后一个执行任务为$J_h^{(l)}$，则有$L_k(i，j)=FT_{hl}$。

定义7 $L_k(i，j)$与$L_{max}(i，j)$的比值叫做系统选出$J_i^{(j)}$要调度时，Agent P_k的利用率，记作$e_k(i，j)$:

$$e_k(i，j):=L_k(i，j)/L_{max}(i，j) \quad (4)$$

用W_{ij}表示Agent P_k在完成任务$J_u^{(v)}$后，等待下一任务$J_i^{(j)}$到来的闲置时间。

4 基于多 Agent 网络化车间相关任务调度算法

4.1 目标函数

由定义4知实数r(i，j)体现了作业J_i运行到第j道任务时已完成的相对运行时间，而1-r(i，j)即为此时的相对剩余运行时间。为使各作业的推进速度均衡且运行周期最短，对于某一时刻，我们应优先安排相对剩余运行时间最长(即已落后)的作业的相应任务先调度，因此优先调度满足max{1-r(i，j)}的作业。若使最长的相对剩余运行时间最短(使落后的作业赶上)，则可保证各作业的推进速度均衡，所以选用下式作为目标函数:

$$\text{Min}\ \max\{1-r(i，j)\} \quad (5)$$

978-1-935068-01-3 © 2009 SciRes.

为使每个Agent的负载均衡，对于某一时刻已选出的待调度任务$J_i^{(j)}$，我们应在其可用的Agent集合中选择占用量最小(即选用利用率最低)的Agent来执行$J_i^{(j)}$。由利用率的定义知1-$e_k(i，j)$为剩余利用率，显然我们应优先选取剩余利用率最大(即满足max{1- $e_k(i，j)$})的Agent。若使最大的剩余利用率最短，则可保证各Agent的负载均衡。因此选用下式作为目标函数:

$$\text{Min}\{\max\{1- e_k(i，j)\}\} \quad (6)$$

设n_k为第k个Agent上执行任务的个数，E_{knk}为第k个Agent上最后一个任务的结束时间，max{E_{knk}}为最长执行路径，若使最长执行路径最短，则可保证证执行周期最短，因此选用式(7)作为目标函数。

$$\text{Min}\{\max\{ E_{knk} \}\} \quad (7)$$

综上所述，选用下式作为系统调度的目标函数：

$$\min\{\alpha\max\{1-e_k(I,j)\}|_{(i,j)} = \text{Min}\max\{1-r(i，j)\}+(1-\alpha)\max\{ E_{knk} \}\} \quad (8)$$

这里α=0或1。当α=1为均衡度算法，此时(8)式变为：

$$\text{Min}\max\{ \max\{1- e_k(i，j)\}|_{(i,j)= \text{Min}\max\{1-r(i，j)\}}\} \quad (9)$$

其中式(i,j)= Min max{1-r(i，j)}表示i，j值由映射f，ϕ确定：i=f(max{1-r(i，j)})，j=ϕ(max{1-r(i，j)})(即选取相对剩余运行时间最长的作业i的第j道待调任务$J_i^{(j)}$先调度，以使最长的剩余运行时间最短，从而使各作业的调度进度均衡)，则式(9)表示在保证各作业均衡调度的前提下(即确定了i,j值的情况下)，从$J_i^{(j)}$的可用Agent集合{P_k}中选择占用量最小(即剩余利用率最大max{1- $e_k(i，j)$})的Agent来执行$J_i^{(j)}$，以使最大的剩余利用率最短，从而使各Agent的负载均衡。

当α=0时为静态压缩算法，此时式(8)变为式(7)，从而保证执行周期最短。

4.2 算法设计

4.2.1 算法1 前趋图的拓扑排序算法

1)对作业A_i的前趋图进行深度优先遍历，同时根据式(1)计算所有任务的最早结束时间TS_{ij}并求得它们的关键前趋KeyPre(J_{ij})，其中j=1，2，…，Q_i；

2)根据式(2)计算作业A_i的最早完成时间T_i；

3)根据式(3)计算所有任务的调度效率r(i，j)，其中j=1，2，…，Q_i；

4)按r(i，j)的递增顺序对作业A_i的所有任务进行排序。

4.2.2 算法2 系统初始化算法

1)对所有作业都执行一遍算法1，将得到的拓扑有序序列链成作业的待命队列$QueueO_i$(i=1，2，…，n)，队尾置空；

2)为每个作业设一个移动头指针MoveHead(i)，初值为该作业的待命队列$QueueO_i$的队头，每调度完一道任务后MoveHead(i)便指向下一待调任务，可见MoveHead(i)总指向该作业下一步要调度的任务。因此对于某一时刻，满足max{1-r(I，j)}的任务一定在MoveHead(i)所指的任务中；

3)将每道任务的可用Agent集合{P_k}中的Agent链成队列QueueP(I，j)(i=l,2,…,n;j=1,2,…,Q_i)，队尾置空；

4)由于初始状态各 Agent 均未被分配任务，因此置所有 Agent 的负载均 0，$L_k(0)$=0(k =1, 2,…, m)，且其任务队列均为空，$QueueTA_k$ = NULL(k =1, 2, …, m)。

4.2.3 算法3 并行均衡调度算法(α=1)

1)按目标函数式(5)的要求，在MoveHead(i)(i=1，2，…，n)所指的待调任务中选取满足max{1-r(i,j)}的任务$J_i^{(j)}$，进行调度，并移动MoveHead(i)指针，使其指向下一待调任务$J_i^{(j+1)}$。

2)根据式(4)计算集合{P_k}中各Agent的利用率$e_k(i，j)$。

3)遍历队列QueueP(i，j)，按目标函数式(6)的要求，从中选取满足max{1- $e_k(i，j)$}的Agent P_k 。

4)将任务$J_i^{(j)}$插入到Agent P_k的任务队列$QueueTA_k$的队尾。

5)计算$J_i^{(j)}$的实际开始时间ST_{ij}，实际结束时间FT_{ij}和P_k的闲置时间W_{ij}。

设Agent P_k上任务$J_i^{(j)}$前一执行任务为$J_u^{(v)}$，另设$J_i^{(j)}$的关键前驱KeyPre(J_{ij})的实际结束时间为FT_{ij}^{KP}，则置ST_{ij}=max{FT_{uv},FT_{ij}^{KP}}，置L_k=ST_{ij}+T_{ij},W_{ij}=ST_{ij}-FT_{uv}。

6)判断是否存在MoveHead(i)≠NULL(i=1，2，…，n)的作业，若是则转1)，否则算法结束。

4.2.4 算法4 静态压缩算法(α=0)

在静态压缩算法中，我们首先寻找最长路径，然后按目标函数式(7)的要求压缩最长路径。

1)计算各Agent的占用量L_k (k=1，2，…，m)：设第k台Agent的任务队列$QueueTA_k$的队尾任务为$J_o^{(p)}$，则L_k=FT_{op}。

2)寻找最长路径：求满足max{L_k}的Agent(显然max{E_{knk}}=max{ L_k })(=1， 2， …， m), 不妨设L_k=max{L_k}，这里i存在但不唯一。

3)对其中某一i，遍历其任务队列$QueueTA_i$，若所有结点的W_{ij}均为0，则调度结果最优，不可调解，算法结束。

4)求最小但不为O的W_{ij}(i =1，2，…，n，1≤ k ≤Q_i，$Q_i \in Z^+$)，不妨设W_{hc}，则其所对应的任务为$J_h^{(c)}$。若$J_h^{(c)}$为队尾(即是本Agent的最后一道加工任务)，则转8）。

5)设$J_h^{(c)}$在P_i上的下一任务为$J_s^{(t)}$，若$J_s^{(t)}$的关键前驱的实际结束时间小于$J_h^{(c)}$的实际开始时间(即FT_{st}^{KP} < ST_{hc})，则交换$J_s^{(t)}$与$J_h^{(c)}$在P_i上的执行顺序，否则转8）。

6)重新计算任务$J_s^{(t)}$所有后继任务的关键前驱：令实际结束时间最大的前驱为关键前驱。

7)重新计算队列$QueueTA_i$中$J_s^{(t)}$及其以后结点的关键前驱，实际开始时间，实际结束时间，和P_i的闲置时间W_{ij}，并更新L_i，若L_i<max{L_k}转2），否则算法结束。

8)置W_{hc}=0，转2）。

5 调度实例分析

设某基于MAS的网络化车间作业系统中，有单元任务控制Agent三个：P={P_1,P_2,P_3}，某时刻车间作业Agent提交了两个作业系统：（A,<）和（B,<）,每个作业分别包括六个任务系统，其间的偏序关系分别如图2、3所示。

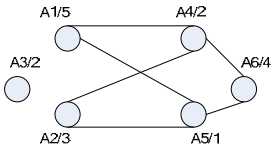

图2 （A,<）任务偏序图

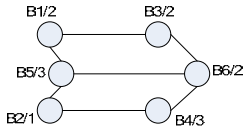

图3 （B,<）任务偏序图

现将上述两个作业系统分发到P上，分发情况如表1、表2所示。

表 1 （A，<）分发到 P 上情况

任务	A1	A2	A3	A4	A5	A6
可用Agent	P1,P3	P1,P3	P2	P1,P3	P2	P1,P3

表 2 （B，<）分发到 P 上情况

任务	B1	B2	B3	B4	B5	B6
可用Agent	P2	P1,P3	P1,P3	P1,P3	P2	P2

表 3 调度效率值 r(i,j)

R(i,j)	1	2	3	4	5	6
作业A	0.5000	0.3000	0.2000	0.7000	0.6000	1.0000
作业B	0.2857	0.1428	0.5714	0.5714	0.7142	1.0000

表 4 并行调度轨迹

	0	1	2	3	4	5	6	7	8	9	10	11
P1	A2	A2	A2	B4	B4	B4	A4	A4				
P2	A3	A3	B1	B1			A5	B5	B5	B5	B6	B6
P3	B2	A1	A1	A1	A1	A1	B3	B3	A6	A6	A6	

表 5 静态压缩算法生成的调度轨迹

	0	1	2	3	4	5	6	7	8	9	10
P1	A2	A2	A2	B4	B4	B4	A4	A4			
P2	A3	A3	B1	B1	B5	B5	B5	A5	B6	B6	
P3	B2	A1	A1	A1	A1	A1	B3	B3	A6	A6	A6

本文作业系统中单元任务控制Agent数m=3，包含两个同构Agent集合{P1,P3}、{P2}（n=2）。调度效率值r(i,j)如表3所示。

根据优先调度满足max{1-r(i，j)}的任务原则得到调度序列：

B2->A3->B1->A2->A1->B4->B3->A5->A4->B6->A6

经并行均衡调度算法，调度轨迹如表4；经静态压缩算法生成的调度轨迹如表5所示。

在表4所示的并行调度轨迹中，在执行A5前Agent P2有闲置，静态压缩算法的思想是考虑将A5与A6互换，得到表5所示调度轨迹。由此可见，两个车间作业并行调度的总长度为12，Agent的总利用率达81.23%；静态压缩后，总长度为11，Agent总利用率达88.31%。

6 结束语

在基于MAS的网络化车间作业系统中，多个相关任务并行调度是提高Agent总利用率的主要途径，本文提出的优化算法在兼顾到任务间的偏序关系及Agent负载情况的同时，以多组作业的均衡推进为目标，为任务安排最早的开始时间，以免因安排不紧凑拖延后继任务的运行；静态压缩算法以压缩最长的执行路径为目标，来进一步缩短总的调度长度，从而进一步提高各Agent总利用率。

978-1-935068-01-3 © 2009 SciRes.

References (参考文献)

[1] 尹祚明．带后继位级跟踪的抢先位级调度[J]．计算机学报，1989，12(1)：10-16．
Yin Zuoming. Preemptive level schedule with successors level tracking[J].*Chinese Journal of Computers*．1989，12(1)：10-16．

[2] 许日滨．多机相关任务的均衡调度算法[J]．计算机学报，1996，19(1)：77-80．
Xu Ribin．The balanced scheduling algorithm of dependent tasks in multiprocessors[J] .*Chinese Journal of Computers*．1996，19(1)：77-80．

[3] 王凤儒，张淑丽．多机相关任务的相关矩阵调度算法[J]．计算机学报，1998，21(10)：933-938．
Wang Fengru，Zhang Shuli．The relation matrix scheduling algorithm of dependent tasks in multiprocessors[J] .*Chinese Journal of Computers*．1998，21(10)：933-938．

[4] 杨羽，邬伶俊．多机相关任务调度优化策略与组织方法[J]．计算机学报，1993，16(9)：661-669．
Yang Yu，Wu Lingjun．Optimization strategy for the scheduling of dependent tasks in multiprocessors[J] .*Chinese Journal of Computers*．1993，16(9)：661-669．

[5] JANSEN K，PORKOLAB L．Linear-time approximation schemes for scheduling malleable parallel tasks[A]．Proceedings of the tenth annuaIACM—SlAM symposium on Discrete lgorithms[C]．Baltimore，Maryland，United States，1999．490-498．

[6] AHMAD I．Task assignment in distributed computing system[A]．1995 IEEE Fourteenth Annual International Phoenix Conference on Computers and Communication[C]．Scottsdale，AZ，USA，1995．49 -53．

[7] AMONRA AK，BAMPIS E，KENYON C，et川f．Scheduling Independent Muhiprocessor asks[A]．Proceedings of the 5th Annual European Symposium on Algorithms[C]．London SW1997，JZ UK，1997．

[8] 高锣，王治森．基于多 Agent 的网络化车间制造系统调度问题研究[J]．中国机械工程．2003，14(12)：1033-1036．
Gao E，Wang Zhisen．Research on scheduling problem in multi-agent based networked shop Floor [J] .*China Mechanical Engineering*．2003，14(12)：1033-1036．