一种多 Agent 系统相关任务的并行调度算法①

A Novel Dependent Tasks Parallel Scheduling Algorithm for Multi – Agent System

黄崇本　陶剑文　张振宇　（浙江工商职业技术学院计算机应用研究所 浙江宁波 315012）
王凤儒　（哈尔滨理工大学计算机与控制学院 黑龙江哈尔滨 150080）

摘要：本文研究多 Agent 系统(MAS)相关任务调度问题，从时间和空间两方面考虑，提出了一种新颖的多 Agent 相关任务的并行调度算法——多 Agent 相关任务关联矩阵调度算法(Multi – Agent Dependent Tasks Correlation Matrix Scheduling Algorithm, MADTCMSA)，利用可变的关联矩阵表示任务的时间需求与 Agent 的局部存储空间的关系及任务分配的状态。实验显示该算法具有最短或较短的调度长度，并且具有较好的时间均衡性和空间协调性。

关键词：多智能代理 调度算法 目标函数 任务分配

1 引言

Agent 是一个封装了某些状态和知识并使用消息进行通信，在一定的环境中能独立自主地完成特定功能的智能主体[1]。多 Agent 系统(Multi – Agent System, MAS)是指由多个智能 Agent 通过相互协作完成某些任务或达到某些目的的分布式智能计算系统。一个多 Agent 系统主要包括任务请求 Agent、任务调度 Agent、任务服务 Agent 等部件，各 Agent 间通过 KQML (Knowledge Query and Manipulation Language)协议进行通信。随着分布式人工智能(DAI)技术的发展，目前有许多分布式并行处理系统可由 MAS 完成。

在基于 MAS 的分布式并行处理系统中，存在多个功能相同或不同的 Agent，而 MAS 所处理的作业(job)中有的作业含有多个数据相关的任务(Task)，并行调度主要解决的问题是，如何根据各 Agent 的解题能力及诸任务解题需求，为作业中的每一任务安排一个 Agent，并在兼顾任务间的偏序关系及 Agent 的负载情况的同时，为诸任务确定开始时间且使其调度长度最短、任务分配均衡[2]。本文提出一种新颖的多 Agent 相关任务的并行调度算法——多 Agent 相关任务关联矩阵调度算法(multi – agent dependent tasks correlation matrix scheduling algorithm, MADTCMSA)[4]，该算法能保证 MAS 相关任务的并行调度时间最短或较短，且具有较好的时间均衡性和空间协调性。

2 相关概念

2.1 智能主体 Agent

一个智能主体 Agent 可以定义为一个七元向量[4]，即 A = (F,I,T,L,D,C,O)。其中各变元的含义为：

F：Agent 标志，是 Agent 相互区分的特征信息及内部状态信息，如名称、地址等；

I：用户接口模块，主要用于用户或管理人员交互；

T：任务求解模块，主要表达 Agent 的行为能力，包括执行与分析功能；

L：学习模块，随着环境变化，从中学习经验，为知识库增加知识或规则；

D：数据模块，实现 Agent 运行所需数据的存取及保存 Agent 学习所得的知识与规则；

C：约束规则集，主要用于对 Agent 执行性能控制；

O：通信模块，用于 Agent 间的交互，实现消息的收发与任务的传递等。

2.2 任务调度概念

① 基金项目：浙江省教育厅科研项目资助(20050568)

一个任务系统 Ts 表示为一个二元组，即 Ts = (T, <)，T = {$T_1, T_2, \cdots, T_m$} 是一个数据相关的任务集，T 中任意任务 $T_j = (F_j, S_j)$ 由一组功能需求 F_j 和空间需求 S_j 组成，< 表示任务的偏序关系；任务 T 的执行时间集合 $\tau = \{\tau_1, \tau_2, \cdots, \tau_m\}$，$\tau_j$ 为任务 T_j 的执行时间($j = 1,2,\cdots,m$)；一个多 Agent 系统 A = {$A_1, A_2, \cdots, A_n$} 由 n 个 Agent 构成，A 中任意 Agent $A_k = (F_k, S_k)$ 由一组局部处理功能 F_k 和局部存储空间 S_k 组成($k = 1,2,\cdots,n$)。对任务系统 Ts 的有效调度取决于对时间和空间资源的有效利用[4]。

为便于研究，设任务系统中所有任务执行时间之和 $t = \sum \tau_j$；Agent 平均承担任务的执行时间 $\tau^* = t/n$；关键路径长度 L^* = 关键路径上的所有任务的执行时间之和。

(1) 定义 1。若对于任务系统 Ts 中任意任务 T_j($j = 1,2,\cdots,m$)，至少存在一个 Agent $A_k \in A(1 \leq k \leq n)$，使得 A_k 有能力承担 T_j，则称任务系统 Ts 是可分配的；否则称 Ts 是不可分配的。

(2) 定义 2。若对于任务 $T_j \in T(1 \leq j \leq m)$，至少存在一个 Agent $A_k \in A(1 \leq k \leq n)$，使得 A_k 有能力承担 T_j，则称任务 T_j 是可执行的；否则称 T_j 是不可执行的。

(3) 定义 3。MAS A 中的任意 Agent A_k($k = 1,2,\cdots,n$)承担的任务集合 $C_k = \{T_e\}$ 称为 Agent A_k 的任务集(其中 T_e 是 A_k 承担的任务)。

(4) 定义 4 对任务集 T 中的任务按偏序关系进行分类，同一层次序的任务构成的集合称为同层任务集，记为 $B_i(i = 1,2,\cdots,H)$。

(5) 定义 5。任务集 $C_k(k = 1,2,\cdots,n)$ 中任务的执行时间之和 $t_k = \sum \tau_e$ 称为任务集 C_k 的执行时间(其中 τ_e 是 C_k 中任务 T_e 的执行时间)。

(6) 定义 6。任务集 $C_k(k = 1,2,\cdots,m)$ 的执行时间与等待时间之和 $L_k = t_k + \Sigma d_k(e)$ 称为 Agent 的调度长度(其中 $d_k(e)$ 为在保证任务偏序关系的条件下任务 Te($1 \leq e \leq m$)在 Agent 上执行所需要的等待时间)。

(7) 定义 7。Ts 中所有的任务全部分配完成后，Agent A_k 的调度长度的最大值 $L = \max\{L_k\}$ ($k = 1,2,\cdots,n$)称为 Ts 在 MAS A 上的调度长度。

(8) 定义 8。Ts 中所有的任务全部分配完成后，若 $t_k/t \approx 1/n$($k = 1,2,\cdots,n$)，则称任务系统 Ts 在 MAS A 上的任务分配是均衡的。

(9) 定义 9。任务系统 Ts 中所有的任务全部分配完成后，若存在 $\varepsilon \geq 0$，使 $\max\{|t_k/t - 1/n|\} \leq \varepsilon$ ($k = 1,2,\cdots,n$)，则称 ε 为任务系统 Ts 在 MAS A 上任务分配的均衡度。

2.3 相关定理

(1) 定理 1。对于 MAS A 和任务系统 Ts，若存在一任务 $T_h \in T(1 \leq h \leq m)$ 是不可执行的，则任务系统 Ts 是不可分配的。

(2) 定理 2。已知 MAS A 和任务系统 Ts，若对于一个分配，n 个 Agent $A_1, A_2, \cdots, A_n$ 的调度长度分别为 $L_1, L_2, \cdots, L_n$，则 $L \geq L^*$；当 $L = L^*$ 时，任务系统 Ts 在 MAS A 上的调度长度最短。

证明：因对 Ts 实施调度时，需保证 Ts 中任务的偏序关系。因此实施调度后的关键路径上的任务的完成时间大于或等于 L^*。设关键路径上最后一个任务由 Agent Ah($1 \leq h \leq m$)承担，则 $L_h \geq L^*$。因为 $\max\{L_k\} \geq L_k$($k = 1,2,\cdots,n$)，所以 $\max\{L_k\} \geq L^*$。由定义 7 知，$L = \max\{L_k\}$ ($k = 1,2,\cdots,n$)，所以 $L \geq L^*$。显然当 $L = L^*$ 时，Ts 在 MAS A 上的调度长度 L 最短。

由定理 2 知，$\min \max\{L_k\}$ 能保证任务系统在 MAS 上的调度长度 L 最短或较短。

(3) 定理 3。已知 MAS A 和任务系统 Ts，若对于一个分配，任意任务集 C_k($k = 1,2,\cdots,n$)的执行时间 t_k 均不满足 $t_k >> \tau^*$ 且 $t_k << \tau^*$，则任务分配是均衡的。

证明(反证法)：假设任务分配是不均衡的，由定义 8 知，任务分配不均衡的充分必要条件是存在 k。($1 \leq k_0 \leq n$)，使 $t_{k_0}/t >> 1/n$ 或 $t_{k_0}/t << 1/n$。令 $t_{h1}/t = \max\{t_k/t\}$，$t_{h2}/t = \min\{t_k/t\}$ ($1 \leq h1, h2 \leq n$)，则 $t_{h1} >> \tau^*$ 或 $t_{h2} << \tau^*$。而这与已知任意任务集 C_k($k = 1,2,\cdots,n$)的执行时间 t_k 均不满足 $t_k >> \tau^*$ 且 $t_k << \tau^*$ 矛盾，所以任务分配是均衡的。

由定理 3 知，$\min \max\{t_k/t\}$ 或 $\max \min\{t_k/t\}$ 能保证任务分配是均衡的。也就是说，$\min \max\{|t_k/t - 1/n|\}$ ($k = 1,2,\cdots,n$)能保证任务分配是均衡的。

2.4 目标函数选用

由定理 2 和定理 3 可知，在任务系统 Ts 和 MAS A 上，为首先使调度长度 L 最短或较短，然后使之任务分配均衡，我们选用目标函数：

$\min\{\alpha_k \max\{L_k\} + (1 - \alpha_k)\{\max\{|t_k/t - 1/n$

$|\}|\min\{L_k\}\}\}(k=1,2,\cdots,n)$ (1)

其中 $\alpha_k=0$ 或 1($k=1,2,\cdots,n$)，设 $L'=\min\{L_k\}$，当只有一个 Agent 对应的 $Lk=L'$ 时，$\alpha_k=1$；否则 $\alpha_k=0$。$\max\{|t_k/t-1/n|\}|\min\{L_k\}$ 表示在保证 $\min\{L_k\}$ 的条件下 $\max\{|t_k/t-1/n|\}$。

对目标函数作如下说明：

(1) $L'=\min\{L_k\}$，当只有一个 Agent 对应的 $L_k=L'$ 时，$\alpha_k=1$，此时目标函数为：

$$\min \max\{L_k\} \quad (2)$$

为保证调度长度 L 最短或较短，把任务 T_i 分配给 $\min\{L_k\}$ 所对应的 Agent。

(2) $L'=\min\{L_k\}$，当有两个以上(包括两个)Agent 对应的 $L_k=L'$ 时，$\alpha_k=0$，此时目标函数为：

$$\min \max\{|t_k/t-1/n|\}|\min\{L_k\}\} \quad (3)$$

为保证在调度长度 L 最短或较短的情况下任务分配均衡，把任务 T_i 分配给 $\min\{L_k\}$ 所对应的多个 Agent 中 $\max\{|t_k/t-1/n|\}$ 所对应的 Agent。

3 MADTCMSA 实现

3.1 调度策略

为将任务系统 Ts 中的所有任务合理地分配到 MAS A 上，采用如下调度策略：

(1) 对于任务系统 Ts，若存在一个任务 $T_i\in T(1\leqslant j\leqslant m)$ 不可执行，由定理 1 知，Ts 是不可分配的。

(2) 对于任务 $T_i\in T(1\leqslant j\leqslant m)$，若只有唯一的一个 Agent $A_h\in A(1\leqslant h\leqslant m)$ 能承担 T_i，则将 T_i 分配给 A_h。

(3) 对于任务 $T_i\in T(1\leqslant j\leqslant m)$，若有 $q(2\leqslant q\leqslant n)$ 个 Agent 能承担 T_i，并且其中只有一个 Agent $A_{q0}(1\leqslant q0\leqslant m)$ 对应的 L_{q0} 最小，此时 $\alpha_k=1$，根据目标函数，将 T_i 分配给 A_{q0}。

(4) 对于任务 $T_i\in T(1\leqslant j\leqslant m)$，若有 $r(2\leqslant r\leqslant n)$ 个 Agent 能承担 T_i，其中有 $r1(2\leqslant r1\leqslant r)$ 个 Agent 对应的 L_k 最小，此时 $\alpha_k=0$，若这 r1 个 Agent 中只有一个 Agent $A_{r0}(1\leqslant r0\leqslant n)$ 对应的 $|t_k/t-1/n|$ 最大，根据目标函数，将 T_i 分配给 A_{r0}。

(5) 对于任务 $T_i\in T(1\leqslant j\leqslant m)$，若有 $s(2\leqslant s\leqslant n)$ 个 Agent 能承担 T_i，其中有 $s1(2\leqslant s1\leqslant s)$ 个 Agent 对应的 L_k 最小，此时 $\alpha_k=0$，若这 s1 个 Agent 中有 $s2(2\leqslant s2\leqslant s1)$ 个 Agent 对应的 $|t_k/t-1/n|$ 最大，根据目标函数，将 T_i 分配给这 s2 个 Agent 中的某一个 Agent。若这 s2 个 Agent 中只有一个 Agent A_{s0}。($1\leqslant s_0\leqslant n$) 的局部存储空间最小，则将 Tj 分配给 A_{s0}。

(6) 对于任务 $T_i\in T(1\leqslant j\leqslant m)$，若有 $g(2\leqslant g\leqslant n)$ 个 Agent 能承担 T_i，其中有 $g1(2\leqslant g1\leqslant g)$ 个 Agent 对应的 L_k 最小，此时 $\alpha_k=0$，若这 g1 个 Agent 中有 $g2(2\leqslant g2\leqslant g1)$ 个 Agent 对应的 $|t_k/t-1/n|$ 最大，根据目标函数，将 T_i 分配给这 g2 个 Agent 中的某一个 Agent。若这 g2 个 Agent 中有 $g3(2\leqslant g3\leqslant g2)$ 个 Agent 的局部存储空间相同且最小，则将 T_i 分配给这 g3 个 Agent 中的任意一个 Agent。

3.2 关联矩阵

对于任务系统 Ts，用矩阵 $T[m\times v]$[3][4] 表示 Ts 中 m 个数据相关任务的功能需求，其中 T_{jk} 表示任务 T_i 的第 k 项功能需求；对于 MAS A，用矩阵 $A[n\times v]$[3] 表示 A 中 n 个 Agent 的解题能力，其中 A_{jk} 表示 Agent A_i 的第 k 项功能($1\leqslant j\leqslant n$；$1\leqslant k\leqslant v$)。

引入关联矩阵 $M_u=(z_{jk})_{m\times n}(u=0,1,\cdots,m)$，其元素 $z_{jk}=x_j+iy_k$ 是复数，$i^2=-1(j=1,2,\cdots,m;k=1,2,\cdots,n)$，$M_u$ 表示 Ts 中各任务的空间需求和 MAS A 中各 Agent 的局部存储空间的关系及任务分配的状态。对 M_u 的说明：

(1) 初始关联矩阵 M_0 中元素 $z_{jk}=x_j+iy_k(j=1,2,\cdots,m;k=1,2,\cdots,n)$ 的 x_j 表示任务的空间需求；当 Agent A_k 不能承担任务 T_i 时，$y_k=0$，当 Agent A_k 能承担任务 T_i 时，y_k 为 Agent A_k 的局部存储空间。

(2) 关联矩阵 M_u 是可变的。初始关联矩阵为 M_0，分配完第一个任务 $T_{j1}(1\leqslant j1\leqslant n)$ 后，使 M_0 中元素 $z_{j1k}=x_{j1}+iy_k$ 的 $x_{j1}=0,y_k=0(k=1,2,\cdots,n)$，得到新的矩阵 M_1，依此类推，可得到矩阵 $M_2,M_3\cdots M_m$。显然，当 $M_m=0$ 时，Ts 中的所有任务分配完毕。

3.3 算法步骤

(1) 输入任务系统 Ts 及其功能需求 $T[m\times v]$ 和 MAS A 及其解题能力 $A[n\times v]$；

(2) 按偏序关系对任务集 T 中的任务进行分类，得同层任务集 $B_1,B_2,\cdots,B_H$；

(3) 构造初始关联矩阵 M_0；

(4) 扫描 M_0 中各行元素，若存在一个不可执行任务 T_i，即存在 j，对任意 k 使 z_{jk} 的 $y_k=0(j=1,2,\cdots,m;k=1,2,\cdots,n)$，则任务系统 Ts 是不可分配的，结束。否则执行第(5)步；

(5) 初始化:$t=\sum\tau_l$,$\tau^*=t/n$;$L^*=$关键路径上的所有任务的执行时间之和,对任意 $k(k=1,2,\cdots,n)$,使 $C_k=\Phi$,$L_k=0$,$t_k=0$;

(6) 依次对同层任务集 $B_1,B_2,\cdots,B_H$ 中的所有任务 $T_{lu}(u=1,2,\cdots,m;1\leq|_u\leq m)$ 逐一采用调度策略,把任务 T_{lu} 分配给 Agent $A_k(1\leq k\leq n)$,C_k,L_k,t_k,$|t_k/t-1/n|$,M_u 的变化过程如下:

① $C_k\leftarrow C_k\cup T_{lu}$;

② $L_k\leftarrow L_k+\tau_{lu}+d_k(|_u)$;

③ $t_k\leftarrow t_k+\tau_{lu}$;

④ $|t_k/t-1/n|\leftarrow|(t_k+\tau_{lu})/t-1/n|$;

⑤ $M_u\leftarrow M_{u-1}$;

⑥ 直至 $M_m=0$ 时,分配完毕;

(7) 根据分配结果及任务的偏序关系、任务的运行时间对 Ts 实施调度;

(8) 结束。

4 仿真实验

设任务系统 Ts 由六个任务组成,$T=\{T_1,T_2,\cdots,T_6\}$,其偏序关系及解题需求分别如图 1 和表 1 所示;MAS 由三个 Agent 组成,$A=\{A_1,A_2,A_3\}$,其解题能力如表 2 所示[3]。

表 1　Ts 任务的执行时间和解题需求

任务	执行时间	解题需求		
		Windows	VC	RAM
1	5	1		200k
2	4	1	1	200k
3	4	1	1	320k
4	2	1		160k
5	2	1	1	160k
6	4	1		400k

根据图 1、表 1、表 2 等内容可知:

任务系统 Ts 中的任务总数 m=6;

MAS A 中 Agent 总数 n=3;

Ts 中的所有任务执行时间之和 $t=\sum\tau_l=21$;

Agent 平均承担任务的执行时间 $\tau^*=t/n=7$;

关键路径长度 $L^*=11$。

表 2　MAS A 的解题能力

MAS A	Windows	VC	RAM
1	1	1	320k
2	1	1	320k
3	1		400k

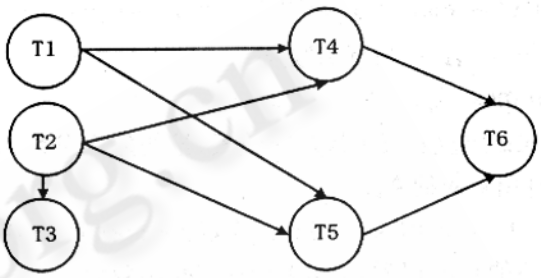

图 1 由 6 个任务构成的任务系统与偏序关系

依据调度算法可得关联矩阵 $M_0,M_1,M_2,M_3,M_4,M_5,M_6$(其中 M_0 如图 2 所示,M_1 如图 3 所示),各任务集分别为:$C_1=\{T_1,T_5\}$,$C2=\{T_2,T_3\}$,$C_3=\{T_4,T_6\}$,它们的执行时间分别为 $t_1=7$,$t_2=8$,$t_3=6$;各 Agent 的调度长度分别为 $L_1=7$,$L_2=8$,$L_3=11$;Ts 在 MAS A 上的调度长度 $L=\max\{L_k\}=11$;$L=L^*$;任务分配的均衡度 $\varepsilon=1/21$。Ts 在 MAS A 上的调度,如表 3 所示。

$$\begin{bmatrix} 200+i320 & 200+i320 & 200+i400 \\ 200+i320 & 200+i320 & 200+i0 \\ 320+i320 & 320+i320 & 320+i0 \\ 160+i320 & 160+i320 & 160+i400 \\ 160+i320 & 160+i320 & 160+i0 \\ 400+i0 & 400+i0 & 400+i400 \end{bmatrix}$$

图 2　初始关联矩阵 M_0

$$\begin{bmatrix} 0+i0 & 0+i0 & 0+i0 \\ 200+i320 & 200+i320 & 200+i0 \\ 320+i320 & 320+i320 & 320+i0 \\ 160+i320 & 160+i320 & 160+i400 \\ 160+i320 & 160+i320 & 160+i0 \\ 400+i0 & 400+i0 & 400+i400 \end{bmatrix}$$

图 3　关联矩阵 M_1

(下转第 33 页)

(上接第 30 页)

表 3　任务调度序列

MAS A	1	2	3	4	5	6	7	8	9	10	11
A_1	1	1	1	1	1	5	5				
A_2	2	2	2	2	3	3	3	3			
A_3						4	4	6	6	6	6

上述实验数据显示:利用多 Agent 相关任务关联矩阵调度算法(MADTCMSA)处理 MAS A 相关任务的调度问题具有最短或较短的调度长度及很好的时间均衡性和空间协调性。

5　结束语

本文提出了一种多 Agent 系统相关任务的并行调度算法,在兼顾到任务间的偏序关系及 MAS 时间和空间的同时,为任务安排最早的开工时间,以免因安排不紧凑拖延后续任务的执行。该调度算法实现于 1 个由 3 个 Agent 组成的 MAS 中,经仿真测试,MADTCMSA 在处理 MAS 相关任务的调度问题上具有最短或较短的调度长度及很好的时间均衡性和空间协调性,同时也是一种提高 Agent 利用率的主要途径。

参考文献

1　张云勇、刘锦德,移动 Agent 技术[M],清华大学出版社,2003,9.

2　Imtiaz Ahmad. Task assignment in distributed computing system[C]. In:1995 IEEE Fourteenth Annual International, Phoenix Conference On Computers and Communication. Scottsdale, AZ, USA,1995.

3　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.

4　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).

5　陶剑文,基于多 Agent 的协作式网络学习系统模型研究[J],计算机时代,2006,7.