基于 MPICH2 和并行 Q 学习的模具制造项目群随机调度 *

张沙清[1,2], 陈新度[1], 陈庆新[1], 陈 新[1]
(1. 广东工业大学 机电工程学院, 广州 510006; 2. 广东工业大学 管理学院, 广州 510520)

摘 要: 通过分析模具制造项目工期、费用与报酬的不确定性以及项目返修频繁发生的特点,建立了基于离散时间马尔可夫链的模具制造项目群随机演化模型,提出了基于 MPICH2 和并行 Q 学习的模型求解算法,在一定程度上克服了维数灾难问题。最后以 Visual C ++6 为工具,在多核环境下实现了该算法,并结合示例说明了算法的可行性与有效性。结果表明该算法在模具制造项目群随机调度中具有一定的应用价值。

关键词: 模具制造项目群; MPICH2; 多核; 并行 Q 学习; Metropolis 准则
中图分类号: TP391 **文献标志码**: A **文章编号**: 1001-3695(2010)09-3242-05
doi:10.3969/j.issn.1001-3695.2010.09.010

Stochastic scheduling for multiple mould and die manufacturing projects based on MPICH2 and parallel Q-learning

ZHANG Sha-qing[1,2], CHEN Xin-du[1], CHEN Qing-xin[1], CHEN Xin[1]
(1. *School of Electromechanical Engineering, Guangdong University of Technology, Guangzhou* 510006, *China*; 2. *School of Management, Guangdong University of Technology, Guangzhou* 510520, *China*)

Abstract: Through the analysis of uncertainties of the durations, costs and rewards as well as the characteristic of frequent repairing in the mould and die manufacturing project, this paper proposed a stochastic evolution model of multiple mould and die manufacturing projects, which was on the basis of a discrete time Markov chain. With aim to overcome the curse of dimensionality, an algorithm based on MPICH2 and parallel Q-learning was put for ward to solve the above stochastic dynamic programming model. Finally, the algorithm was realized in a multi-core environment by using Visual C ++ 6 and was explained with a sample example. The results show that the model is applicable and the algorithm is reliable and effective as well. And the results show that this algorithm can effectively solve stochastic scheduling problems for multiple mould and die manufacturing projects.

Key words: multiple mould and die manufacturing projects; MPICH2; multi-core; parallel Q-learning; Metropolis criterion

0 引言

模具制造的特征是面向资源的单件订货型生产、多项目并行、多企业协作,在管理上存在多企业、多项目共享关键资源的协同管理典型需求。在模具项目执行过程中存在任务工期、费用定额不准、项目报酬随完工时间变化等不确定因素,而且常会出现模具返修、工程变更等突发事件。因多个项目共享着企业的关键资源,一个项目中的不确定因素和突发事件也势必严重影响到其他项目的执行进度,结果导致项目在执行过程中无法遵循开始制订的计划;而不合理地调度,不仅严重影响到本企业其他项目的进度,还会严重干扰协作企业的生产调度,使得生产混乱的现象从本企业扩散到合作伙伴,影响了一大批项目的进度而导致拖期。只有采取科学的项目协同监控与动态调度手段,才能有效应对各种不确定因素与突发事件,确保项目按期完成。

专门研究风险性和不确定环境下项目调度理论的文献较少。从调度角度可分为两大类:a)将项目网络图的结构假设为确定性,而将项目中的其他参数,如任务工期等作为不确定性变量,对调度方案的制定开展研究。文献[1]认为主要存在反应性、随机性、模糊和前摄鲁棒性调度方法;文献[2,3]对相关研究领域进行了总结与展望。b)将项目网络图的结构也假设为随机性,研究项目调度计划与控制方法。文献[4,5]介绍了单机、同等并联多机、带有紧前约束的 Job-shop 和 Flow-shop 以及 GERT 网络的调度研究。

就模具制造项目调度而言,廖仁[6]对确定条件下的多个模具项目的工期费用粗规划作了初步研究,苏志龙等人[7,8]在单个模具项目的工期与费用的不确定性规划方面作了大量的

收稿日期: 2010-02-21; **修回日期**: 2010-04-01 **基金项目**: 国家"863"计划资助项目(2006AA04Z132); 国家自然科学基金资助项目(50875051); 广东工业大学青年基金资助项目(20062014)

作者简介: 张沙清(1973-),男,湖北天门人,博士研究生,主要研究方向为生产计划与控制、网络化制造(zhangshaqing@126.com); 陈新度(1967-),男,湖南望城人,教授,博士,主要研究方向为网络化制造、信息系统、智能制造; 陈庆新(1963-),男,陕西西安人,教授,博导,博士,主要研究方向为网络化制造、信息系统、智能制造; 陈新(1960-),男,湖南常德人,教授,博导,博士,主要研究方向为网络化制造、信息系统、微电子制造装备及自动化.

研究,但针对不确定因素和突发事件影响下多个模具项目的随机调度研究,国内外文献并不多见。

1　问题描述

考虑项目关键路径上需要企业关键资源的任务,可将一个模具制造项目的状态描述为图 1 所示,其中虚线表示可能的处理过程(某些情况下需要进行热处理,项目完工后可能需要返修)。由于每个任务的实际完成时间、费用、结果(质量的优劣、是否返修)是不确定的,会受其前一个任务的影响,同时也会影响其后一个任务,每个项目的实际工期和报酬是不确定的。如果项目的实际工期超过了订单交货时间,那么承担该项目的模具企业将会受到高额的罚款。

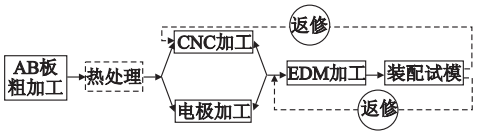

图1　关键路径上的模具制造项目结构示意图

由于第 i 个项目的第 n 个任务存在 C_{in} 种可能的完成方式,每种实现方式由完成该任务所需的时间、费用以及任务完成后的结果三者来描述。例如,图 2 描述的是某个模具制造项目的六个任务(无须热处理,并且将返修当做一个任务)的执行情况。第一个任务(AB 板粗加工)有两种可能的实现方式,分别是(D_{11},C_{11},E)、(D_{12},C_{12},G)。前者表示执行时间、费用、执行结果,分别是 D_{11}、C_{11}、E(表示优良),后者中的 G 表示合格。项目的相邻两个任务之间的联系由转移概率矩阵来描述,如 CNC 加工有三种实现方式,而前一任务 AB 板粗加工有两种实现方式,因此转移概率矩阵 2 的大小为 3×2,表示 AB 板粗加工以某种方式完成的前提下,项目转向 CNC 加工的某种执行结果的概率。图 2 中矩阵 2A、3A 分别表示省略了 AB 板粗加工转向电极加工的转移概率矩阵以及电极加工转向 EDM 加工的转移概率矩阵。

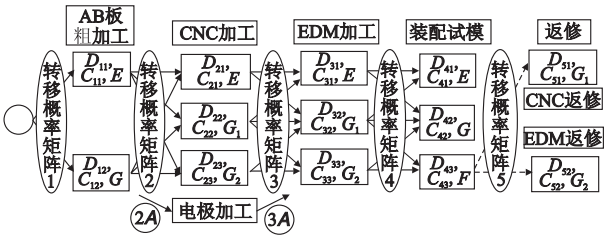

图2　某模具制造项目的不确定参数模型

项目状态的转移是由事件触发而非定时触发的,定义事件为一个任务的开始或结束,这样可以反映不同的项目因自身的“节奏”(项目的长短和任务的紧迫程度)不同所导致的项目群间歇性特征。在这里,第 $K+1$ 个事件发生时的状态,将根据第 K 个事件发生时的状态和第 K 个事件发生时的决策共同导出,但是它们之间的关联是隐性的而非显性的泛函形式。多个项目的执行状态构成了整个项目群的状态。在项目群执行的每个可能的状态,应该采取合适的行动,充分合理地利用有限的企业关键资源,从而实现在保证工期和质量的前提下最大化项目群的利润。

2　基于离散时间 Markov 链的系统建模

上述问题的本质是随机资源受限调度问题,即对资源的多阶段随机优化问题,可以采用离散时间马尔可夫链来进行系统建模。在此之前,先作以下假设:a)所有项目中的任务都是不可分离的,即一旦项目的某个任务开始之后,就必须进行到该任务结束,中途不能中断;b)每个项目的任务执行顺序(工序)是确定的,而且一个任务在同一时刻只占用一类资源;c)忽略任务等待加工的时间和准备时间;d)只考虑部件加工的直接费用的不确定性,将其他直接费用与间接费用视为确定量(两者之和为项目计划报酬的 65%);e)若项目出现返修的情况,则返修一次后即合格。

2.1　系统状态定义

考虑某企业同时承担 M 个模具制造项目,共需要 N 类关键资源,每类资源的个数为 G_k($k=1,2,\cdots N$)。定义系统状态如下:

$$X=[s_1,s_2,\cdots,s_M,q_1,q_2,\cdots,q_M,R_1,R_2,\cdots,R_N,t]^{\mathrm{T}} \quad (1)$$

其中:s_i($i=1,2,\cdots,M$)为正整数,表示项目 i 的当前状态,用来描述项目 i 已经完成的任务和正在执行的任务情况。例如,一个有五个任务的模具项目若正常完成,则最多有 15 种可能的状态,如图 3 所示。各状态定义如表 1 所示。若该项目最后一个任务(装配试模)的执行结果为返修,则其状态数为 17。q_i($i=1,2,\cdots,M$)为正整数(初始状态除外),表示项目 i 中离当前时刻最近的已经完成了的那个任务的完成方式的编号。若该任务为项目 i 的第 n 个任务,且 C_{in} 表示第 i 个项目的第 n 个任务的可能完成方式的数目,则 $q_i\in[1,C_{in}]$。一旦项目的某个任务完成,则 q_i 就被更新。$R_j\in[0,G_j]$($j=1,2,\cdots,N$)为整数,表示当前状态空闲的第 j 类资源的数目。若 $R_j=0$,则表示该类资源全部被占用;若 $R_j=G_j$,则表示该类资源全部空闲。t 为当前状态时刻,用来描述项目报酬随项目完成时间而变化。若某项目计划工期为 125 天,报酬为 150 万,若实际完成时间超过 125 天,但小于 150 天(约定的小额罚款期限),则每超期一天罚款为报酬的 0.2%;若大于 150 天,则每超期一天罚款为报酬的 0.5%。

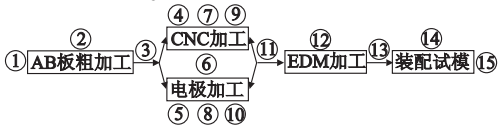

图3　具有五个任务(不返修)的模具制造项目的可能状态

表 1　某模具制造项目状态定义

状态 S_i	项目状态定义
1	AB 板粗加工尚未开始
2	AB 板粗加工正在进行
3	AB 板粗加工完成,CNC 加工和电极加工尚未开始
4	CNC 加工正在进行,电极加工尚未开始
5	电极加工正在进行,CNC 加工尚未开始
6	CNC 加工和电极加工均正在进行
7	CNC 加工正在进行,电极加工完成
8	电极加工正在进行,CNC 加工完成
9	CNC 加工完成,电极加工尚未开始
ú	ú
15	装配试模完成

2.2　行动

定义行动 $U=[\zeta_1,\zeta_2,\cdots,\zeta_M]^{\mathrm{T}}$。其中:$\zeta_i=\rho$ 或 0($i=1,2,\cdots,M$;ρ 为较小的整数,且 $\rho\neq0$)。$\zeta_i=\rho$ 表示执行第 i 个项目的某个任务,其值等于第 i 个项目后一个状态的值减去前一个状态的值。例如表 1 中,s_i 由 3 变成 5,则表示执行电极加工,

但不执行 CNC 加工,此时 $\zeta_i=2$;若 s_i 由 9 变成 8,则 $\zeta_i=-1$。只有当执行任务所需要的资源至少有一个空闲的时候才能采取行动。$\zeta_i=0$ 表示不执行第 i 个项目的任务。

2.3 状态转移规则

系统初始状态为

$$X(0)=[\underbrace{1,\cdots,1}_{M\text{个项目的状态}},\overbrace{0,\cdots,0}^{M\text{个任务完成结果}},\underbrace{R_1,\cdots,R_N}_{N\text{种资源}},\overbrace{0}^{\text{时间}}]^{\mathrm{T}}$$

当开始执行一个任务,或者某个任务完成时,系统状态发生转移。设当前状态为 $X(k)$,采取行动 $U(k)$ 后,则转移到一个临时状态 $X'(k+1)$。

$$X'(k+1)=f(X(k),U(k))=$$
$$[s'_1,s'_2,\cdots,s'_M,q_1,q_2,\cdots,q_M,R'_1,R'_2,\cdots,R'_N,t]^{\mathrm{T}}$$

其中:

$$X(k)=[s_1,s_2,\cdots,s_M,q_1,q_2,\cdots,q_M,R_1,R_2,\cdots,R_N,t]^{\mathrm{T}}$$
$$U(k)=[\zeta_1,\zeta_2,\cdots,\zeta_M]^{\mathrm{T}},s'_i=s_i+\zeta_i$$

若在 $X'(k+1)$ 状态,正在执行的多个项目中第 p 个项目的某个任务(假设占用第 n 类资源,由第 m 种方式实现)比其他任务先完成,则状态 $X'(k+1)$ 转移到状态 $X(k+1)$。

$$X(k+1)=[s'_1,\cdots,s''_p,s'_M,q_1,\cdots,q'_p,\cdots,q_M,$$
$$R'_1,\cdots,R''_n,\cdots,R'_N,t']^{\mathrm{T}}$$

其中:$s''_p=s'_p+\Psi_p$,$q'_p=m$,$R''_n=R'_n+\sigma$,$\sigma=1$ 或 2,Ψ_p 的取值与2.2节中的 ζ_i 相同;t'是任务结束的时刻,如果同一时刻还有其他任务结束,其状态转移也按上述方式相应更新。若某项目的最后一个任务的执行结果不是返修,则该项目正常结束;否则该项目进入返修状态。

2.4 目标函数

企业的目标为最大化所有项目完成后的利润,该利润函数(目标函数)用 Q 值来表示,具体在 3.2 节中描述。

3 基于 *MPICH2* 的并行 Q 学习算法

3.1 算法框架描述

Q 学习是求解上述马尔可夫决策问题的有效方法之一,但往往面临维数灾难和计算量大等问题,无法获得全局最优解。为此,本文采用基于多 agent 的并行 Q 学习算法,通过多 agent 的有限次探索、学习和结果融合之后,在得到的受限状态行动对空间中寻求最优解。算法框架如图 4 所示。

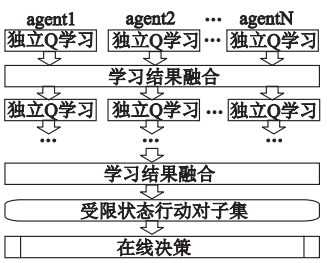

图4 并行Q学习模型

3.2 基于 Metropolis 准则的单 agent 独立学习

在各学习周期,每个 agent 在各自的环境中独立地执行 Q 学习算法。在学习过程中,agent 为了探索未知的环境,往往要采取在当前策略看来并非最优的动作,但是当前策略接近最优时,过多的探索将变得多余。因此,agent 面临探索新知识,还是遵循当前策略的矛盾。ε-greedy 策略($0<\varepsilon<1$)是解决该矛盾的主要策略之一,它以概率 $1-\varepsilon$ 选择当前策略看来最优的动作,而以概率 ε 随机选择动作。相比贪心策略,ε-greedy 增加了随机选择的概率,有助于 agent 突破经验的束缚,探索新知识,但是,采用固定的 ε 值也有一定的局限性。当 agent 学习一段时间后,当前策略已经接近最优策略了,仍然以概率 ε 随机选择动作会降低系统性能,所以应考虑采用可变值 ε 的方法。在学习过程的初期,ε 取较大的值,随着学习时间的增加,逐渐减少 ε 值。模拟退火中的 Metropolis 准则正是这样一种技术,它通过降低温度,逐渐降低接受恶化解的概率[9]。因此,本文中单个 agent 的学习将采用基于 Metropolis 准则的 Q 学习算法。其流程如下:

a)初始化。仿真次数为 M(学习周期长度),$N=0$。

b)初始化退火温度 T,$k=1$。

c)对当前状态 $x(k)$,由贪心策略得到动作 $u(k)^*$,随机选择动作 $u(k)^r$,随机产生[0,1]的随机数 λ。若

$$\lambda<\exp\{\frac{Q(x(k),u(k)^r)-Q(x(k),u(k)^*)}{T}\}$$

则 $u(k)=u(k)^r$,否则 $u(k)=u(k)^*$。

d)执行动作 $u(k)$,观测回报 $R(x(k),u(k))$ 和下一个状态 $x(k+1)$。

e)按照下式更新 $Q(x(k),u(k))$:

$$Q(x(k),u(k))=Q(x(k),u(k))+\alpha(R(x(k),u(k))+$$
$$\gamma\max_{u(k+1)\in U(x(k+1))}Q(x(k+1),u(k+1))-Q(x(k),u(k)))$$

其中:$R(x(k),u(k))$ 为在状态 $x(k)$ 采取行动 $u(k)$ 获得的利润,其值等于所有项目完成后回溯得到的在状态 $x(k)$ 的项目群报酬之和减去采取行动 $u(k)$ 的费用;$Q(x(k),u(k))$ 为不断累加的每个阶段的利润 $R(x(k),u(k))$;$\alpha\in(0,1]$ 为学习率;$\gamma\in[0,1]$ 为折扣系数。

f)$k=k+1$。

g)若 $x(k)$ 为结束状态,则本次仿真结束,$N=N+1$;否则降低退火温度:$T=\beta T(\beta\in(0,1))$,返回 c)。

h)若 $N<M$,返回 b),否则算法结束。

3.3 学习结果融合

设系统中共有 N 个 agent,当所有 agent 都完成了一个周期的学习之后,对它们的学习结果进行融合。由于系统状态空间巨大,无法用 lookup 表穷举所有系统状态行动对,本文利用多个单链表分别保存每个 agent 各自的学习结果以及融合后的学习结果。对所有的 $x(k)\in X$,$u(k)\in U$,设 $Q^m_{\mathrm{avg}}(x(k),u(k))$ 表示在第 m 个学习周期开始时所有 agent 的 $Q(x(k),u(k))$ 值的平均值,$Q_i^{m+1}(x(k),u(k))$ 表示第 m 个学习周期之后第 i 个 agent 对应的单链表中 $Q(x(k),u(k))$ 的值,则在第 m 个周期学习之后,多个单链表的融合过程[10]如下:

a)计算 $Q^{m+1}_{\mathrm{avg}}(x(k),u(k))=\frac{1}{N}\sum_{i=1}^{N}Q_i^{m+1}(x(k),u(k))$;

b)计算 $\Delta=Q^{m+1}_{\mathrm{avg}}(x(k),u(k))-Q^m_{\mathrm{avg}}(x(k),u(k))$;

c)计算 $Q^{m+1}(x(k),u(k))$:

$$Q^{m+1}(x(k),u(k))=\begin{cases}\max\limits_{1\leqslant i\leqslant N}\{Q_i^{m+1}(x(k),u(k))\},\Delta>0\\ Q^{m+1}_{\mathrm{avg}}(x(k),u(k)),\Delta=0\\ \min\limits_{1\leqslant i\leqslant N}\{Q_i^{m+1}(x(k),u(k))\},\Delta<0\end{cases}$$

d)对所有的 $i \in \{1,2,\cdots N\}$

$$Q_i^{m+1}(x(k),u(k)) = Q^{m+1}(x(k),u(k))$$

上述融合过程表明,当 $Q(x(k),u(k))$ 呈上升趋势(平均值增大),即在状态 $x(k)$ 采取行动 $u(k)$ 可能带来较高的报酬时,其他的 agent 都向 Q 值最高的 agent 学习这一经验;反之,当 $Q(x(k),u(k))$ 呈下降趋势(平均值减小),即在状态 $x(k)$ 采取行动 $u(k)$ 可能带来较低的报酬时,其他的 agent 都向 Q 值最低的 agent 学习这一教训。因此,多个 agent 之间学习经验和教训的过程提高了各自的学习速度与效果,从而提高了整个系统的探索和学习能力。

3.4　基于 MPICH2 的并行 Q 学习算法

MPI(message passing interface)是目前应用最为广泛的支持多核并行程序设计的编程接口标准之一,该标准是由消息传递接口论坛(message passing interface forum)发起讨论并进行规范的。MPI 是一种消息传递编程模型,定义了一个实现消息传递模型标准的程序库。MPICH 是一种最重要的 MPI 实现,其版本开发与 MPI 规范制定同步。MPICH 是 MPI-2 的完全实现,本文采用现行 MPI-2 的最新版本 MPICH2-1.0.8。MPICH2 除包含 MPI 函数库之外,还包含一套程序设计与运行环境(包括并行性能可视化分析工具和性能测试工具等),支持 UNIX、Linux 和 Windows 平台,支持 C/C ++ 、Fortran 77 和 Fortran90 的绑定。其主要特点是:a)开放源码;b)高效率、可移植性好、功能强大;c)支持多核(multi-core)、多程序多数据(multiple program multiple data, MPMD)编程、对称多处理系统(symmetrical multi-processing, SMP)、异构集群系统和大规模并行计算系统[11]。

基于前面的叙述并参考文献[10],本文提出基于 MPICH2 的主从模式的并行 Q 学习算法。其流程如下:

a)设置算法参数。Agent 个数为 N, MPICH2 进程个数为 $N+1$。其中 1 个为主进程,其余 N 个为从进程。每个 agent 对应一个从进程,学习周期长度为 M,算法终止周期数为 L。

b)初始化,$k=0$。

c)$k=k+1$,在第 k 个学习周期,每个从进程(agent)独立执行 Q 学习算法(仿真 M 次),并将学习结果发送给主进程。

d)主进程接收各从进程传来的学习结果并进行融合,然后将融合结果传送给各从进程,各从进程接收主进程传来的融合结果,作为下个学习周期的基础。

e)主进程判断学习终止条件,若 $k>L$ 则终止学习;否则转向 c)。

学习结束后,就可以进行在线决策,即

$$u^{\mathrm{opt}}(k) = \arg \max_{u(k) \in U^{x(k)}} Q(x(k),u(k)) \qquad (2)$$

4　算例与分析

某模具企业同时承担 6 个模具制造项目,项目 1 ~4 不需要热处理,项目 5、6 需要热处理。6 个项目的计划报酬分别是 45、52、70、68、160、200 万元,项目计划工期分别是 60、70、90、86、132、160 天。项目实际报酬随项目完成时间而变化,约定的小额罚款(0.2%)期限分别是 65、78、100、95、148、180 天,若超过该期限,则每超期 1 天的罚款分别为项目报酬的 0.5%。

该企业现有的可用于该项目群的资源数量如下:粗加工设备 6 台,热处理设备 1 台,CNC 加工中心 6 台,电极加工设备 3 台,电火花机 4 台,装配单元 3 个。考虑一个模具任务的加工是 A、B 板同时进行的,所以除热处理、电极加工和装配试模以外,每个任务都相应地占用两个同类资源。各项目的各任务执行方式以及任务之间的转移概率分别如表 2、3 所示。在表 3 中,项目 4、5 所对应的转移矩阵 3A、4A 分别表示热处理转向电极加工的转移概率矩阵以及电极加工转向 EDM 加工的转移概率矩阵。

表 2　各项目任务实现方式

	AB 板粗加工	热处理	CNC 加工	电极加工	EDM 加工	装配试模	返修
项目 1	5 1250 E 4 1000 G	—	16 6200 E 17 6500 $G1$ 17 6800 $G2$	10 4200 E 8 3600 $G1$ 9 4000 $G2$	22 8900 E 23 9200 $G1$ 24 9500 $G2$	5 920 E 4 850 G 6 960 F	3 1150 $G1$ 4 1400 $G2$
项目 2	6 1300 E 5 1250 $G1$ 4 1000 $G2$	—	20 7900 E 19 7600 $G1$ 21 8300 $G2$	12 5000 E 11 4600 $G1$ 13 5250 $G2$	25 10200 E 24 9700 $G1$ 26 10500 $G2$	6 1300 E 8 1400 G 7 1350 F	4 1500 $G1$ 3 1250 $G2$
项目 3	6 1350 E 6 1300 $G1$ 7 1450 $G2$	—	23 9400 E 24 9500 $G1$ 25 9900 $G2$	14 5600 E 15 5900 $G1$ 16 6200 $G2$	31 13000 E 31 12500 $G1$ 32 13500 $G2$	9 1800 E 8 1600 G 9 1900 F 9 2000 F	4 1450 $G1$ 4 1550 $G2$
项目 4	5 1100 E 4 1050 $G1$ 6 1250 $G2$	—	21 8600 E 22 8900 $G1$ 23 9100 $G2$	12 5200 E 13 5300 $G1$ 14 5500 $G2$	28 12500 E 27 12000 $G1$ 29 12800 $G2$	7 1400 E 6 1300 $G1$ 7 1300 F 8 1450 F	3 1300 $G1$ 4 1650 $G2$
项目 5	8 1900 E 9 2000 G	6 1000 E 5 900 $G1$ 7 1200 $G2$	42 17000 E 41 16500 $G1$ 42 17300 $G2$	18 7200 E 17 6800 $G1$ 19 7500 $G2$	44 18000 E 45 18200 $G1$ 46 18500 $G2$	12 2400 E 11 2250 $G1$ 11 2150 F	6 2300 $G1$ 5 2000 $G2$
项目 6	9 2000 E 10 2100 G	9 1450 E 8 1300 G	47 18500 E 46 18000 $G1$ 48 19000 $G2$	22 8800 E 21 8500 $G1$ 23 9300 $G2$	52 20800 E 53 20600 $G1$ 54 21600 $G2$	13 2700 E 13 2650 G 12 2600 F	6 2300 $G1$ 7 2700 $G2$

表 3　各项目任务之间的转移概率矩阵

	矩阵 1	矩阵 2	矩阵 2A	矩阵 3	矩阵 3A	矩阵 4	矩阵 5
项目 1	0.28 0.72	0.43 0.10 0.32 0.32 0.25 0.58	0.49 0.12 0.31 0.45 0.20 0.43	0.70 0.16 0 0.30 0.52 0.45 0 0.32 0.55	0.63 0.23 0.11 0.24 0.50 0.60 0.13 0.27 0.29	0.68 0.15 0 0.32 0.70 0.72 0 0.15 0.28	0 0 0.56 0 0 0.44
项目 2	0.23 0.40 0.37	0.49 0.16 0.08 0.32 0.43 0.40 0.19 0.41 0.52	0.53 0.16 0.02 0.28 0.47 0.43 0.19 0.37 0.55	0.62 0.15 0 0.33 0.59 0.39 0.05 0.26 0.61	0.63 0.21 0 0.32 0.54 0.37 0.05 0.25 0.63	0.50 0.14 0 0.48 0.70 0.81 0.02 0.16 0.19	0 0 0.47 0 0 0.53
项目 3	0.23 0.42 0.35	0.69 0.20 0.06 0.20 0.45 0.43 0.11 0.35 0.51	0.68 0.17 0.05 0.20 0.48 0.46 0.12 0.35 0.49	0.63 0.10 0 0.32 0.54 0.35 0.05 0.36 0.65	0.62 0.11 0 0.32 0.52 0.44 0.06 0.37 0.56	0.57 0.22 0 0.39 0.59 0.70 0.03 0.10 0.18 0.01 0.09 0.12	0 0 0.35 0.39 0 0 0.65 0.61
项目 4	0.19 0.46 0.35	0.67 0.18 0.09 0.20 0.44 0.40 0.13 0.38 0.51	0.75 0.12 0.07 0.15 0.52 0.50 0.10 0.36 0.43	0.65 0.10 0 0.31 0.55 0.44 0.04 0.35 0.56	0.72 0.19 0 0.21 0.42 0.54 0.07 0.39 0.46	0.66 0.23 0 0.30 0.58 0.70 0.03 0.11 0.11 0.01 0.08 0.19	0 0 0.45 0.40 0 0 0.55 0.60
	矩阵 1	矩阵 2	矩阵 3	矩阵 3A	矩阵 4	矩阵 4A	矩阵 5	矩阵 6
项目 5	0.30 0.70	0.70 0.22 0.18 0.54 0.12 0.24	0.63 0.23 0.11 0.24 0.50 0.58 0.13 0.27 0.31	0.60 0.28 0.13 0.26 0.51 0.57 0.14 0.21 0.30	0.73 0.21 0 0.14 0.64 0.68 0.13 0.15 0.32	0.67 0.25 0 0.21 0.58 0.68 0.12 0.17 0.32	0.65 0.23 0 0.32 0.67 0.77 0.03 0.10 0.23	0 0 0.49 0 0 0.51
项目 6	0.33 0.67	0.69 0.33 0.31 0.67	0.57 0.19 0.30 0.58 0.13 0.23	0.65 0.15 0.20 0.49 0.15 0.36	0.63 0.16 0 0.26 0.64 0.61 0.11 0.20 0.39	0.70 0.32 0 0.21 0.53 0.59 0.09 0.15 0.41	0.72 0.18 0 0.25 0.69 0.74 0.03 0.13 0.26	0 0 0.38 0 0 0.62

该企业现有的可用于该项目群的资源数量如下:粗加工设备 6 台,热处理设备 1 台,CNC 加工中心 6 台,电极加工设备 3 台,电火花机 4 台,装配单元 3 个。考虑一个模具任务的加工是 A、B 板同时进行的,所以除热处理、电极加工和装配试模以外,每个任务都相应地占用两个同类资源。各项目的各任务执行方式以及任务之间的转移概率分别如表 2、3 所示。在表 3 中,项目 4、5 所对应的转移矩阵 3A、4A 分别表示热处理转向电极加工的转移概率矩阵以及电极加工转向 EDM 加工的转移概率矩阵。

采用 Visual C ++ 6.0 编写基于 MPICH2 的多核并行程序,在配置为 Intel(R) Xeon(R) CPU E5335 @2.00 GHz(8 核),3 GB 内存的服务器上执行图 4 所示算法。参数设置如下:学习率 $\alpha=0.85$,折扣因子 $\gamma=0.90$,初始退火温度 $T=100$,降温系数 $\beta=0.95$。共利用了 8 核处理器的其中 4 核,其中 1 核执行主进程,其余 3 核执行从进程(即共设置 3 个 agent),每个agent 仿真五次后进行一次学习结果融合,共融合 1 000 次。整个算法耗时约 52.6 h。算法的执行时间主要取决于融合的次数、agent 的个数以及每两次融合之间各 agent 独立仿真的次数。上述三者越大,则算法执行的时间越长。

图 5 描述了融合多个 agent 学习结果的情况,横坐标表示融合的次数,纵坐标表示学习经验和教训的次数。可以看出,随着融合次数的增大,学习经验和教训的次数(分别用 count1 和 count2)逐步增加,表明多个 agent 共享学习结果越来越充分。当融合次数达到一定数量后,count1 总是大于 count2,表明随着融合的不断深入,多个 agent 学习的经验相对越来越多,教训相对越来越少,即越来越多的教训转换为经验。

从并行算法的性能方面看,其中一个重要的评价指标是加速比。根据 Gustafson 定律,加速比 S 由下式给出[11]:

$$S = (W_S + N \times W_P)/(W_S + W_P)$$

其中：W_S 为串行计算的时间；W_P 为并行计算时间；N 为 CPU 内核个数。图 6 描述了加速比与融合次数和 agent 个数（$N-1$）的关系。由图 6 可见，当融合次数较小时，3 个 agent 的加速比较大；当融合次数较大时，4 个 agent 的加速比较大。当 agent 个数相同时，S 先是取得最大值，接着随着融合次数的增加而迅速减小，然后逐步缓慢增加。主要原因是：由于每个 agent 的探索结果用单链表保存，融合结果也用单链表保存，刚开始每个 agent 探索得到的结果链表较短，融合需要的时间也较短，W_P/W_S 相对较大（大于 1），此时 S 取得最大值（图中融合次数为 100 时）。随着探索和融合的不断深入，各个 agent 的学习结果链表不断增长，而融合过程需要循环遍历多个单链表，使得 W_P/W_S 不断减小（远小于 1），因此 S 迅速减小；当融合到达一定程度后，并行探索学习和串行融合的时间比相对增大，即 W_P/W_S 逐步增大，因此 S 也逐步增加。

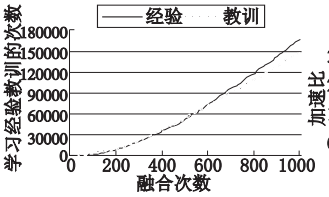

图5　学习结果融合情况

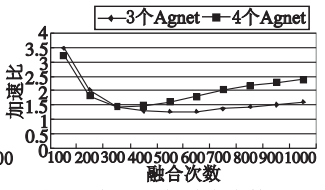

图6　加速比与融合次数和 Agent个数的关系

学习结束后，在得到的有限状态行动对中，就可以利用式（2）进行在线决策。仿真 10 000 次后，得到的平均利润值较随机决策提高约 4%。

5　结束语

本文针对模具制造项目的工期、费用与报酬的不确定性以及项目返修频繁发生的典型特征，构建了基于离散时间马尔可夫链的模具项目群随机演化模型，在此基础上提出了基于 MPICH2 和并行 Q 学习的模型求解算法，结合示例说明了算法的可行性与有效性，并分析了算法的性能。但是，本文只是在仿真得到的有限状态空间中进行决策，并未考虑有限状态空间之外的状态，因此具有一定的局限性。利用神经网络对整个系统状态空间进行函数近似和泛化，将是今后研究的主要问题。

参考文献：

[1] HERROELEN W, LEUS R. Project scheduling under uncertainty: survey and research potentials[J]. European Journal of Operation Research, 2005, 165(2): 289-306.

[2] AYTUG H, LAWLEY M, McKAY K, *et al*. Executing production schedules in the face of uncertainties: a review and some future directions[J]. European Journal of Operation Research, 2005, 161(3): 86-110.

[3] DEBELS D, VANHOUCKE M. Future research avenues for resource constrained project scheduling: search space restriction or neighborhood search extension[R]. Belgium: Ghent University, 2006.

[4] WEGLARZ J. Project scheduling: recent models, algorithms and applications[M]. Boston: Kluwer Academic Publishers, 1999: 309-332.

[5] NEUMANN K, STEINHARDT U. GERT networks and the time-oriented evaluation of projects[R]. New York: Springer-Verlag, 1979.

[6] 廖仁. 模具虚拟企业项目调度研究[D]. 广州：广东工业大学, 2003.

[7] 苏志龙，陈庆新，陈新，等. 基于协商的模具虚拟企业生产项目规划[J]. 中国机械工程, 2002, 13(22): 1931-1937.

[8] 李英杰，陈庆新，陈新度，等. 多属性的虚拟企业并行协商项目规划算法[J]. 机械工程学报, 2005, 41(2): 215-222.

[9] 陈圣磊，吴慧中，肖亮，等. 基于 Metropolis 准则的多步 Q 学习算法与性能仿真[J]. 系统仿真学报, 2007, 19(6): 1284-1287.

[10] 周浦城，洪炳镕，韩学东，等. 基于多 agent 的并行 Q 学习算法[J]. 小型微型计算机系统, 2006, 27(9): 1704-1707.

[11] 多核系列教材编写组. 多核程序设计[M]. 北京：清华大学出版社, 2007.

（上接第 3238 页）

3　结束语

在对目前主要的主题词提取方法进行分析总结的基础之上，本文提出了一种基于增量词集频率的文本主题词提取算法。实验结果表明，本文算法取得了较理想的效果，对复旦大学上海（国际）数据库研究中心 NLP 小组提供的文本集中 1 400篇政治经济类论文提取的主题词总体满意度达到了 82.36%。

但本文算法也存在一些需要改进的地方：a）本文算法依赖于分词系统，分词的准确性直接影响到主题词的提取准确率，本文算法采用了中国科学院的 ICTCLAS30，分词的准确性得到了一定的保证，但对于一些新词、组合词，则无法标志，这也是后续需要改进的地方；b）候选主题词集的生成，需要采用更好的生成算法；c）需要解决提取的主题词中有同义或语义相近的词，如一篇文章提取到的主题词为“技术，高新技术，产业，企业，科技，经济，地区，发展，知识”。其中的技术与高新技术、产业与企业就比较相近。如何消除这种现象也是本文算法后续需要重点解决的问题。

参考文献：

[1] 施聪莺，徐朝军，杨晓江. TFIDF 算法研究综述[J]. 计算机应用，2009，29(z1)：167-170，180.

[2] 刘菲，董萱菁，吴立德. 利用关联规则挖掘文本主题词的方法[J]. 计算机工程，2008，34(7)：81-83.

[3] 赵鹏，蔡庆生，王清毅，等. 一种基于复杂网络特征的中文文档关键词抽取算法[J]. 模式识别与人工智能，2007，20(6)：817-831.

[4] 耿焕同，蔡庆生，于琨，等. 一种基于词共现图的文档主题词自动抽取方法[J]. 南京大学学报：自然科学版，2006，42(2)：156-162.

[5] 唐培丽，王树明，胡明. 基于语义的汉语文献主题词提取算法研究[J]. 吉林大学学报：信息科学版，2005，23(5)：535-540.

[6] TURNEY P D. Coherent Keyphrase extraction via Web mining[C]// Proc of International Joint Conference on Artificial Intelligence. Acapulco, Mexico: [s. n.], 2002: 434-439.

[7] TURNEY P D. Learning algorithms for keyphrase extraction[J]. Information Retrieval, 2000, 2(4): 303-336.

[8] CUI Hang, KAN Min-yen, CHUA T S. Unsupervised learning of soft patterns for generating definitions from online news[C]//Proc of the 13th World Wide Web Conference. 2004: 90-99.

[9] WITTEN I H, PAYNTER G W, FRANK E, *et al*. KEA: practical automatic keyphrase extraction[C]//Proc of the 4th ACM Conference on Digital Libraries. 1999: 254-255.