文章编号: 1001-0920 (2015) 11-1930-07 DOI: 10.13195/j.kzyjc.2014.1308

基于改进Q学习的知识化制造自适应动态调度策略

王玉芳[1,2], 严洪森[1]

(1. 东南大学 a. 复杂工程系统测量与控制教育部重点实验室，b. 自动化学院，
南京 210096；2. 南京信息工程大学 自动化系，南京 210044)

摘 要: 针对知识化制造系统生产环境的不确定性, 构建一个基于多 Agent 的知识化动态调度仿真系统. 为了保证设备 Agent 能够根据当前的系统状态选择合适的中标作业, 提出一种基于聚类-动态搜索的改进 Q 学习算法, 以指导不确定生产环境下动态调度策略的自适应选择, 并给出算法的复杂性分析. 所提出的动态调度策略采用顺序聚类以降低系统状态维数, 根据状态差异度和动态贪婪搜索策略进行学习. 通过仿真实验验证了所提出动态调度策略的适应性和有效性.

关键词: 知识化制造；自适应；动态调度；基于聚类-动态搜索的改进 Q 学习算法；多 Agent

中图分类号: TH165 **文献标志码:** A

Adaptive dynamic scheduling strategy in knowledgeable manufacturing based on improved Q-learning

WANG Yu-fang[1,2], *YAN Hong-sen*[1]

(1a. MOE Key Laboratory of Measurement & Control of Complex Systems of Engineering，b. School of Automation, Southeast University，Nanjing 210096，China；2. Department of Automation，Nanjing University of Information Science & Technology，Nanjing 210044，China．Correspondent：WANG Yu-fang，E-mail：qing_0325@163.com)

Abstract: Aiming at the uncertainty of the production environment in knowledgeable manufacturing system, a dynamic scheduling simulation system based on the multi-agent is built. To ensure that the machine agent can select the appropriate bid job based on the current system status, the improved Q-learning based on clustering-dynamic search(CDQ) algorithm is presented, which is used to guide the adaptive selection of dynamic scheduling strategy in the uncertain production environment, and the complexity analysis of the algorithm is given. The dynamic scheduling strategy adopts the method of the sequence clustering to reduce the dimension of system state and learns according to status different degree and the dynamic greed search strategy. Simulation experiments verify the adaptability and effectiveness of the dynamic scheduling strategy.

Keywords: knowledgeable manufacturing；self-adaptive；dynamic scheduling；CDQ algorithm；multi-Agent

0 引 言

知识化制造 (KMS) 是本世纪初提出的一种智能制造理念[1]. 先进制造模式作为一种先进制造知识, 以知识网进行表征, 通过建立知识网与 Agent 网之间的同构映射关系, 将以知识网表示的先进制造模式归入 KMS 中, 以满足企业的多样需求[2]. 自适应是 KMS 高智能特征中的一个主要方面, 触及制造系统的多个应用领域. 其中, 高效生产的优化调度和不确定或复杂生产环境下的自适应调度即是一个重要的问题. 因此, 对于 KMS, 面对动态生产环境实现自适应的动态调度具有重要意义.

不确定生产环境下的自适应生产调度研究正逐渐成为一个活跃的研究领域. 赵宁等[3]针对动态调度约束复杂、多变问题, 建立了一种约束联动调度模型和算法以实现快速的人机交互动态调度; Blackstone 等[4]认为特定调度规则仅在一定生产环境和状态下最优, 据此提出了一种根据系统当前状态动态选择最适应规则的随机自适应调度策略; 李琳等[5]基于动态事件对调度的影响分析, 采用改进启发式算法局部调整调度中受影响的操作, 以实现事件驱动的自适

收稿日期: 2014-08-24；**修回日期:** 2014-11-10.

基金项目: 国家自然科学基金重点项目(60934008)；中央高校基本科研业务费专项资金项目(2242014K10031).

作者简介: 王玉芳(1979–), 女, 博士生, 从事知识化制造系统的研究；严洪森(1957–), 男, 教授, 博士生导师, 从事知识化制造、生产计划与调度等研究.

应调度; 在单机调度系统中, Xanthopoulos 等[6]提出了基于整合强化学习和模糊逻辑多目标优化的调度方法; Lee[7]提出了一种依据动态制造环境产生自适应调度模糊规则的方法, 该方法可以根据调度环境的当前状态动态选择并运用最适合的调度策略; 文献 [8] 和文献 [9] 分别建立了一种动态调度系统模型, 采用改进的 Q 学习算法确定自适应调度策略. 上述文献中基于 Q 学习的动态调度策略能够根据生产环境的变化动态选择恰当的调度规则, 以满足动态调度的自适应要求. 但在学习过程中, 其动态调度算法的动作搜索采用固定参数值的贪婪策略, 其贪婪参数取值具有一定的主观性和盲目性, 忽略了学习过程中学习经验的动态累积. 基于此, 本文在动态调度算法的学习过程中采用改进的动态搜索策略来选择最优的状态-动作对. 随着学习次数的增加和经验的累积, 动态贪婪搜索策略合理控制知识“利用”和“探索”的概率, 避免盲目搜索, 提高搜索效率. 同时, 本文对文献 [9] 中加权求和的 Q 值迭代策略进行了改进, 根据系统状态与聚类状态的差异度、将来回报和最大模数收益加权均值综合确定迭代策略, 进一步契合算法学习与目标函数寻优的一致性.

多 Agent 系统因其分布式结构特征, 实际的调度执行可通过多个 Agent 协商完成, 具有良好的灵活性, 非常适合处理动态调度问题[10]. 本文采用改进合同网协商机制, 在多 Agent 技术基础上, 建立动态调度仿真模型. 标准 Q 学习算法的状态空间维数庞大, 降低了算法的收敛速度, 因此本文通过顺序聚类法对系统状态聚类, 降低系统维数. 为了减少聚类状态与系统状态之间的误差, 采用状态差异度定量地度量状态间的距离, 并作为权系数参与对聚类状态 Q 值的加权迭代. 同时, 为了提高搜索精度和速度, 在 Q 值更新迭代中加入最大模糊收益加权均值, 并采用动态贪婪策略搜索具有最大 Q 值的动作. 综上, 本文提出基于聚类-动态搜索的改进 Q 学习 (CDQ) 算法, 用于指导设备 Agent 在动态环境下调度策略选择.

1　问题描述

调度过程中的符号定义如下: 车间作业集表示为 $J=\{J_1,J_2,\cdots,J_N\}$; 加工设备集表示为 $M=\{M_1,M_2,\cdots,M_M\}$; 每个作业由多道工序组成, O_{ij} 表示作业 J_i 的第 j 道工序的加工时间, 同一作业的相邻工序不能在同一台设备上加工, 且某个时间段内一台设备只能加工一道工序; 作业相互独立且无优先级, 作业 J_i 实际完工时间为 C_i, 到达时间为 AT_i, 交货期为 D_i[9].

$$D_i=\mathrm{AT}_i+f_i\sum_{j=1}^{k_i}O_{ij}. \tag{1}$$

其中: f_i 为交货因子; k_i 为作业 J_i 的工序总数. 在已有的研究中[7-9,11], 调度目标主要集中为最小化拖期, 而未考虑作业提前完成对库存压力的影响和相应的成本提高. 因此, 本文借鉴精良制造理念, 充分考虑作业拖期和提前对企业产生的影响, 将调度目标确定为最小化提前拖期惩罚:

$$\mathrm{OBJ}=\min\sum_{i=1}^{N}(\mathrm{EP}_i\cdot\max\{D_i-C_i,0\}+\mathrm{TP}_i\cdot\max\{C_i-D_i,0\}). \tag{2}$$

其中: EP_i 为单位提前惩罚系数; TP_i 为单位拖期惩罚系数.

2　基于多 Agent 的动态调度仿真系统

2.1　动态调度系统仿真模型

针对生产过程的复杂性和动态生产环境的不确定性, 本文建立了基于多 Agent 的动态调度系统仿真模型, 如图 1 所示.

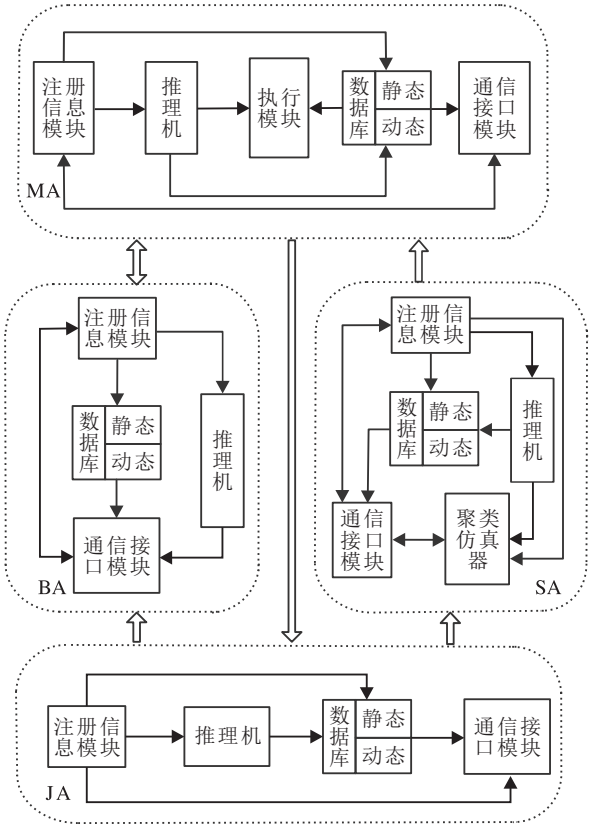

图 1　基于多 Agent 的动态调度仿真模型

模型主要由以下几种 Agent 组成.

作业 Agent (JA). 与人机交互接口对接, 包含作业的数量、工序及加工时间等信息. 内部封装数据库、注册信息模块、推理机和通讯接口模块.

状态 Agent (SA). 代理调度系统的状态, 内部封装数据库、注册信息模块、推理机、执行模块 (聚类仿真器) 和通讯接口模块.

设备 Agent (MA). 代理调度系统的生产设备, 通

过自身的可用时段进行招标的方式进行动态调度. 其内部封装数据库、注册信息模块、推理机、执行模块和通讯接口模块.

缓冲 Agent (BA). 表示生产设备的缓冲区, 代理缓冲区中的待加工工序与 MA 进行协商, 内部封装数据库、注册信息模块、推理机和通讯接口模块.

2.2 调度协商过程

多 Agent 系统的问题求解是通过 Agent 之间的协商和优化得到结果. 合同网协议是多 Agent 之间的基本作业分配技术, 通过作业的招标、投标和制定合同进行合作与分配. 招标 Agent 以广播的形式向其他所有 Agent 发布邀标信息. 显然, 这将产生庞大的通信量, 并降低通信效率. 为了避免这一问题, 在本文的动态调度系统仿真模型中引入缓冲 Agent, 即把可在设备 Agent 加工的工序添加到缓冲 Agent 中. 当设备 Agent 在可用的空闲时段发出邀标信息时, 只需向相应的缓冲 Agent 发出通知即可, 从而降低通信量并大幅提升协商通信效率. 基于设备空闲的改进合同网招投标协商过程如下:

1) 设备 Agent 在可用的空闲时段发出邀标请求, 通知缓冲 Agent 中的工序进行投标;

2) 缓冲 Agent 中的推理机根据数据库中的信息及参数生成标书, 并向设备 Agent 提交标书;

3) 设备 Agent 根据调度策略评价所有收集到的标书, 确定中标标书;

4) 设备 Agent 通知中标工序并等待中标工序返回确认信息, 若未返回, 则重新进行招投标, 否则双方签订合同;

5) 更新设备 Agent 的可用空闲时段, 从缓冲 Agent 中移除中标工序, 通知作业 Agent 发放新的加工工序;

6) 所有作业分配完成, 调度过程结束.

3 基于 CDQ 算法的自适应调度策略

在动态的调度环境中选择合适的调度规则以获得满意的调度结果是自适应调度需要解决的一个关键问题. 本文提出一种能够自适应环境变化的动态调度策略, 即基于改进 Q 学习的 CDQ 算法, 用以调度规则的选择和决策.

3.1 CDQ 算法概述

CDQ 学习算法的研究思路为: 利用顺序聚类方法[12]对复杂的系统状态进行聚类, 以降低状态空间的复杂性. 利用状态差异度度量聚类状态与瞬时系统状态之间的距离, 将聚类状态-动作对值代替系统状态-动作对值, 以聚类状态与系统状态的差异度作为权系数进行 Q 值函数的迭代更新. 引入动态贪婪搜索策略, 以提高算法的速度和精度. 在本文建立的动态调度仿真系统中, 各 Agent 通过通信接口与动态生产环境交互, 以获得最优的 Q 函数值, 从而选择最合适的中标标书.

3.2 状态空间划分

本文将 CDQ 算法应用在设备 Agent 的标书选择过程中. 在动态调度系统中, 系统状态是连续且不断变化的, 因此需要数十个状态特征来描述, 必将导致 Q 学习陷入“维数灾难”. 为了合理选择调度规则 (即动作), 借鉴已有研究[8-9], 并经过多次仿真实验, 归纳出对调度规则影响较大的 4 个状态特征: 设备利用率 U_M、相对机器负载 L_R、平均交货因子 F_A 和平均提前拖期损失 P_A, 则 t 时刻系统状态可表示为 $S_t = (U_M, L_R, F_A, P_A)$. 下面给出 4 种状态特征的具体定义.

设备利用率 $U_M = T_o/T_e$, 表示 t 时刻之前设备 Agent 的有效使用时间 T_o 与设备 Agent 总的空闲可利用时间 T_e 的比值. 机器相对负载 $L_R = \omega_m/\bar{\omega}$, 表示 t 时刻设备缓冲区中最大剩余加工时间 ω_m 与所有设备的平均剩余加工时间 $\bar{\omega}$ 的比值. 平均交货因子 $F_A = \Big(\sum\limits_{i=1}^{N} f_i\Big)\Big/N$, 表示 N 项作业交货因子 f_i 的平均值. 平均提前拖期损失

$$P_A = \Big(\sum_{i=1}^{n_b}(\mathrm{EP}_i \cdot \max\{D_i - C_i, 0\} + \mathrm{TP}_i \cdot \max\{C_i - D_i, 0\})\Big)\Big/n_b,$$

表示 t 时刻作业的损失成本. 其中: n_b 为提前完工和拖期作业的数量之和, EP_i 和 TP_i 分别为单位提前及拖期惩罚系数. 此外, 为了消除上述 4 种状态特征在聚类时单位和数量级差别所产生的影响, 且保持各状态特征的原有语义, 本文采用比例因子法处理上述状态特征值.

设 $C_x = (C_{x1}, C_{x2}, \cdots, C_{xq})$ 为状态聚类后得到的 K 个聚类中第 x 个聚类状态 (系统状态中心). 其中: q 为特征维数; $x = 1, 2, \cdots, K$. 与传统的 Q 学习算法对明确的系统状态进行判断不同, 本文定义状态差异度为度量系统状态与各聚类状态 (系统状态中心) 的距离.

定义 1 t 时刻系统状态 S_t 与聚类状态 C_x 之间的 Manhattan 距离为

$$d_{tx} = \sum_{i=1}^{q} |S_{ti} - C_{xi}|, \tag{3}$$

则系统状态 S_t 与聚类状态 C_x 的差异度为

$$\mu_{C_x}(S_t) = \frac{d_{tx} - \min\limits_{1\leqslant z\leqslant K}(d_{tz})}{\max\limits_{1\leqslant z\leqslant K}(d_{tz}) - \min\limits_{1\leqslant z\leqslant K}(d_{tz})}. \tag{4}$$

显然, $0 \leqslant \mu_{C_x}(S_t) \leqslant 1$, 且当且仅当系统状态 S_t 与聚类状态 C_x 距离最小时, $\mu_{C_x}(S_t) = 0$; 当且仅当

系统状态 S_t 与聚类状态 C_x 距离最大时, $\mu_{C_x}(S_t)=1$. 所有聚类的状态差异度向量为 $\mu_C(S_t)=(\mu_{C_1}(S_t), \mu_{C_2}(S_t),\cdots,\mu_{C_x}(S_t),\cdots,\mu_{C_K}(S_t))$.

定义 2　若满足 $\forall S_t^C\in C_x$, $S_t^C=\arg\min\limits_{1\leqslant z\leqslant K}\mu_{C_x}(S_t)$, 则称 S_t^C 为当前系统状态 S_t 对应的聚类状态; 同理, S_{t+1}^C 为状态 S_{t+1} 对应的聚类状态.

3.3　Q 值更新策略

若系统状态 S_t 对每个聚类状态的差异度为 $\mu_{C_x}(S_t)$, 经动作 a_t 后达到系统状态 S_{t+1} 时的差异度为 $\mu_{C_x}(S_{t+1})$, 则 $\forall a\in A$, A 为系统动作 (调度规则) 集, 各聚类状态-动作值为 $Q(C_x,a)$. 为了反映将来时刻最大收益的平均水平, 取系统状态 S_{t+1} 下所有聚类状态的最大收益加权平均和作为最大模糊收益加权均值 $\bar{Q}^{S_{t+1}}$, 并按下式计算:

$$\bar{Q}^{S_{t+1}}=\frac{\sum\limits_{x=1}^{K}(1-\mu_{C_x}(S_{t+1}))\cdot\max(Q(C_x,a))}{K}. \tag{5}$$

文献 [9] 的 Q 值迭代策略同时考虑了将来回报和最大模糊收益, 但此两项采用加权求和方式. 当系统状态 S_{t+1} 与当前聚类状态 S_{t+1}^C 的相似度较高时, 迭代策略中将来回报的权系数较大, 使得将来回报与权系数的乘积成为主导因素, 最大模糊收益加权系数则非常小, 导致最大模糊收益与权系数的乘积对迭代策略产生的影响微弱; 反之, 将来回报与权系数乘积转为弱项. 然而, 实际应用中, 若系统状态与聚类状态距离较近, 则将来回报和最大模糊收益与各自权系数的乘积均应较大; 反之, 乘积均应较小, 表示与聚类状态距离较大系统状态的贡献小一些. 因此, 本文对文献 [9] 中的 Q 值更新策略进行改进, 给出基于系统瞬时状态对聚类状态之差异度权系数的 Q 值更新迭代公式如下:

$$\begin{aligned}&Q_n(S_t^C,a_t)=\\&(1-\alpha_n(S_t^C,a_t)\cdot(1-\mu_{S_{t+1}^C}(S_{t+1})))Q_{n-1}(S_t^C,a_t)+\\&\alpha_n(S_t^C,a_t)\cdot(1-\mu_{S_{t+1}^C}(S_{t+1}))\times\\&\{r_{t+1}+\gamma\max_{b\in A}[Q_{n-1}(S_{t+1}^C,b)+\bar{Q}_{n-1}^{S_{t+1}}]\}.\end{aligned} \tag{6}$$

其中: $Q_n(S_t^C,a_t)$ 为当前聚类状态 S_t^C 第 n 次循环生成的 Q 值; $\alpha_n(S_t^C,a_t)$ 为步长参数; $\mu_{S_{t+1}^C}(S_{t+1})$ 为系统状态 S_{t+1} 与聚类状态 S_{t+1}^C 的差异度; $Q_{n-1}(S_t^C,a_t)$ 为第 $n-1$ 次循环生成的 Q 值; r_{t+1} 为即时回报因子, 采用启发式立即回报设计; γ 为延迟回报的折扣因子; $Q_{n-1}(S_{t+1}^C,b)$ 为将来回报; $\bar{Q}_{n-1}^{S_{t+1}}$ 表示第 $n-1$ 次循环时状态的最大模糊收益加权均值. 以一定的速率减小步长参数 $\alpha_n(S_t^C,a_t)$, 则可以使式 (6) 收敛于最优 Q 值, 步长参数 $\alpha_n(S_t^C,a_t)$ 可由下式获得:

$$\alpha_n(S_t^C,a_t)=W_\alpha/(1+\rho\cdot\mathrm{VST}_{S_n(S_t^C,a_t)}). \tag{7}$$

其中: W_α 为 α_n 的权系数变量, 非负; ρ 为非负的收缩因子, 控制 α_n 的收缩速率; $\mathrm{VST}_{S_n(S_t^C,a_t)}$ 为第 n 次循环中状态-动作对 (S_t^C,a_t) 被访问的总次数, 如果 $\mathrm{VST}_{S_n(S_t^C,a_t)}$ 增加, 则步长参数 α_n 随之减小.

由式 (6) 可知, 将来回报和最大模糊收益加权均值的系数均为 $1-\mu_{S_{t+1}^C}(S_{t+1})$. 当系统状态 S_{t+1} 与聚类状态 S_{t+1}^C 较近时, 差异度 $\mu_{S_{t+1}^C}(S_{t+1})$ 较小, 而 $1-\mu_{S_{t+1}^C}(S_{t+1})$ 较大. 这使得将来回报 $Q_{n-1}(S_{t+1}^C,b)$ 和最大模糊收益加权均值 $\bar{Q}_{n-1}^{S_{t+1}}$ 与系数的乘积 $(1-\mu_{S_{t+1}^C}(S_{t+1}))Q_{n-1}(S_{t+1}^C,b)$ 和 $(1-\mu_{S_{t+1}^C}(S_{t+1}))\bar{Q}_{n-1}^{S_{t+1}}$ 均较大, 从而保证迭代更新中与聚类状态距离较近的系统状态更易获得最大的 Q 值, 更容易满足迭代策略需求.

3.4　奖惩函数设计

奖惩函数设计应与系统的调度目标相对应[13]. 本文目标函数为式 (2) 的最小化提前拖期惩罚, 而 CDQ 学习算法又是收敛于最大值. 为了使最小化目标函数和最大化 Q 值函数的优化方向一致, 本文采用启发式立即回报函数设计思想, 通过算法的学习, 系统将授予启发式的立即回报, 引导学习算法更快地收敛到最优策略. 因此, 算法中的立即回报函数设计为

$$r=\begin{cases}-\left(\sum\nolimits_E+\sum\nolimits_P\right), \text{ 作业提前或拖期;}\\ 1, \text{ 其他.}\end{cases} \tag{8}$$

其中

$$\sum\nolimits_E=\sum_{j=1}^{l_E}\mathrm{EP}_j\cdot(D_j-C_j),$$

$$\sum\nolimits_P=\sum_{l=1}^{l_T}\mathrm{TP}_l\cdot(C_l-D_l).$$

l_E 为提前完工作业数量, l_T 为缓冲区内拖期作业数量, TP_l 为拖期作业 J_l 的单位拖期惩罚系数, EP_j 为提前完工作业 J_j 的单位提前惩罚系数, D_j 和 C_j 分别为作业 J_j 的交货期和实际完工时间. 式 (8) 将目标函数的最小化问题转变为回报函数的最大化问题. 具体地, 在每次学习迭代中, 若有工件提前或拖期, 则目标函数 $\sum\nolimits_E+\sum\nolimits_P>0$, 立即回报 $r=-\left(\sum\nolimits_E+\sum\nolimits_P\right)<0$. 每次迭代学习中, 目标函数越小, 获得的立即回报就越大. 若无工件提前或拖期, 则目标函数最小为 0. 根据式 (8), 系统获得最大的立即回报为 1. 因此, 每次迭代的目标函数累积获得总的目标函数最小, 则意味着立即回报的累积最大. 在调度系统的运行状态下, 式 (8) 定义的启发式立即回报函数能够较精确地评价动作的优劣, 为 CDQ 学习算法直接、及时地提供回报信息, 进而引导 CDQ 算法更快地收敛到最优控制策略.

3.5　搜索策略

Q 学习算法中, 动作搜索往往采用贪婪策略 (ε-

greedy). ε 表示知识搜索和利用的概率, 它表示状态 S_t 下, 选择最大状态-动作对评估函数值的动作 (即“利用”) 的概率为 $1-\varepsilon$, 以概率 ε 随机选择其他动作 (即“探索”). ε 的大小影响动作的搜索效果. 根据 ε 值对算法的影响, 开始学习时应主要进行“探索”. 随着学习及经验的积累, 知识“利用”的成分逐渐增加. 在此过程中, ε 值应逐渐减小. 鉴于上述分析, 经过实验验证, 本文提出以下基于学习次数 n 的动态贪婪策略:

$$\varepsilon(n)=\max\left(0,\frac{n+\xi_0}{G}\right)\cdot\eta. \tag{9}$$

其中: n 为当前学习次数; G 为总学习次数; η 为搜索幅值, 且满足 $0.95\leqslant\eta\leqslant 1$; ξ_0 为限幅调节系数, 避免取无意义的边界值, $\xi_0\in(0,(1-\eta)G)$. 在学习之初, $\varepsilon\approx 1$ 表示学习过程中几乎只“探索”不“利用”; 随着学习次数的递增, “利用”成分增加, “探索”成分减少; 当 n 接近 G 时, $\varepsilon\approx 0$, 表示学习过程中几乎只“利用”不“探索”. $\varepsilon(n)$ 随着 n 渐变的过程就是搜索过程由“探索”向“利用”经验知识的过渡过程. 与传统的固定 ε 贪婪策略相比, 动态贪婪策略更具智能化, 可使学习过程动态调整, 同时也避免了搜索的盲目性, 提高了搜索效率.

3.6 算法步骤

结合 Agent 技术和 CDQ 算法, 基于 CDQ 算法的自适应动态调度的具体实现概括如下.

Step 1: 设置最大聚类数 K, 状态 Agent 利用顺序聚类法对系统状态进行聚类, 得到 K 个聚类状态 C_x, $x=1,2,\cdots,K$, 并将聚类结果存储到状态 Agent 的数据库中.

Step 2: 初始化所有聚类状态-动作对的 Q 值, 并存储于设备 Agent 的知识库中.

Step 3: 对于 $\tau_t=\tau_{t_0}$ 时刻, 置学习次数 $n=1$, 协商调度开始.

Step 4: 如果 τ_t 刻有设备 Agent 空闲, 则随机选择其中之一作为 MA_k, 然后对其空闲时段发布招标信息, 并邀请相应 BA_k 中的工序参与投标, 转入 Step 5, 否则, 转入 Step 14.

Step 5: 若 MA_k 未收到 BA_k 反馈的标书, 则表示缓冲区 BA_k 中无待加工工序, 转入 Step 12, 否则, 转入 Step 6.

Step 6: 根据式 (4) 计算当前系统状态 S_t 与聚类状态 C_x $(x=1,2,\cdots,K)$ 的差异度.

Step 7: 若 MA_k 收到 h 个标书, 则接收 SA 中的状态差异度, 根据定义 2 求出当前状态 S_t 所对应的聚类状态 S_t^C. 根据式 (9) 的动态贪婪策略从数据库的动作 (规则) 集中选择具有最大回报值的动作, 根据该规则从 h 个标书中选择中标工序, 并发出工序中标通知.

Step 8: BA_k 中的中标工序接收到中标消息后, 向 MA_k 发出确认信息, 双方签订合同.

Step 9: MA_k 通过式 (8) 计算立即回报值; SA 观测到下一时刻系统状态 S_{t+1}, 并计算 S_{t+1} 与各聚类状态的差异度.

Step 10: MA_k 根据定义 2 求取 S_{t+1} 对应的聚类状态 S_{t+1}^C, 推理机通过搜索数据库获得聚类状态 S_{t+1}^C 下的最大将来回报 $\max\limits_{b\in A}Q_{n-1}(S_{t+1}^C,b)$, 根据式 (5) 计算最大模糊收益加权均值 $\bar{Q}_{n-1}^{S_{t+1}}$, 根据式 (6) 迭代更新状态-动作对 Q 值, 并将其存储于数据库中, 置 $n=n+1$.

Step 11: 将已签约工序从 BA 中移除.

Step 12: 如有其他设备 Agent 空闲, 则转入 Step 4, 否则, 转入 Step 13.

Step 13: 如果所有空闲设备 Agent 对应的缓冲 Agent 中均无待加工工序, 则转入 Step 14, 否则, 转入 Step 15.

Step 14: BA 接收 JA 分配的新工序.

Step 15: 置 $t=t+1$, 更新 τ_t, 转入 Step 4.

Step 16: 重复 Step 4 ~ Step 15, 当学习到所有状态-动作对 Q 值的最优值时, 算法结束.

3.7 算法复杂性分析

在本文建立的动态调度仿真系统中, Agent 之间的通信是影响系统性能的重要方面, 而系统中主要通信发生在招投标阶段. 因此, 影响算法效率的 Agent 通信主要有以下几个方面.

1) MA 向 BA 发送招标信息. 因为系统中 MA 和 BA 分别为 M 个, 招标次数表示为 U, 所以此阶段的通信量为 $O(MU)$;

2) BA 提交标书至 MA. 因为 BA 中工序数一定小于等于系统中的作业总数 N, 所以此阶段的最大通信量为 $O(MNU)$;

3) JA 向 BA 发布新工序信息. 系统中唯一的 JA 向 M 个 BA 发布的新工序数量不大于系统的作业总数 N, 故此阶段最大通信量为 $O(MN)$;

4) JA 向 SA 发布系统作业. 因为调度模型中仅有一个 SA, 所以通信量为 $O(N)$;

5) SA 与 MA 的信息通信. 系统中唯一的 SA 向 M 个 MA 提供状态差异度信息, 通信量为 $O(M)$.

由上述分析可知, 本文算法的最大通信量为

$$O(MU)+O(MNU)+O(MN)+O(N)+O(M)=O((M+MN)U+MN+N+M),$$

在已知的动态调度系统中, 机器数 M 和作业数 N 均为确定的常数, 因此最大通信量近似为 $O((M+MN)U)$, 为计算机可接受.

4　仿真实验

本文模仿生产过程中不确定生产环境下的调度环境, 设计了一个动态调度仿真模型, 以验证上述调度模型和策略的有效性. 将作业到达及工序完工定义为系统事件, 仿真以事件触发方式进行. 系统由M台设备Agent组成, 作业总数为N且随机进入系统, 到达系统的时间间隔服从负指数分布, 平均到达率为λ. 作业J_i的交货因子f_i服从均匀分布$[u_{f1}, u_{f2}]$, 其包含的工序数是介于$[n_{k1}, n_{k2}]$之间的随机整数, 每道工序的加工时间O_{ij}服从均匀分布$[u_{p1}, u_{p2}]$, 拖期惩罚系数TP_i和提前惩罚系数EP_i分别服从均匀分布$[u_{t1}, u_{t2}]$和$[u_{t3}, u_{t4}]$. 设备Agent知识库中封装的调度规则为最短加工时间优先SPT、最早交货期优先EDD和最小松弛时间优先MST三种常用规则. 当进入调度系统的作业数达到N后仿真停止.

给出4个基于上述模型的仿真案例, 4种案例中作业总数皆为$N = 3\,000$, 惩罚系数均取$u_{t1} = 2$, $u_{t2} = 3$, $u_{t3} = 1$及$u_{t4} = 2$, 其他参数设置如表1所示. 其中: 案例1和案例2分别表示6台设备运行时, 市场需求平稳、产品结构较简单和较复杂的生产情况; 案例3和案例4表示8台设备运行时与案例1和案例2所对应的生产情况.

表 1　案例参数设置

案例	M	λ	n_{k1}	n_{k2}	u_{p1}	u_{p2}	u_{f1}	u_{f2}
1	6	1/5.5	1	5	2	8	1	6
2	6	1/5.5	1	7	2	13	1	6
3	8	1/5.5	1	5	2	8	1	6
4	8	1/5.5	1	7	2	13	1	6

在Petium-1. 8G个人计算机上, 采用Matlab 7.0编程语言对上述案例进行仿真运行. CDQ算法中, 取延迟回报的折扣因子为$\gamma = 0.7$; 动作搜索过程中, 采用式(9)的动态贪婪系数ε. 仿真系统处理完3 000个作业后仿真结束. 为了减少随机因素的影响, 对每个案例进行300次仿真, 计算其提前拖期惩罚的均值, 并与文献[8-9]提出的B-Q和WSQ算法以及文献[11]提出的CMSQ算法进行比较. 上述文献研究的目标函数为最小化平均拖期, 为了便于比较分析, 将文献[8-9]和文献[11]的目标函数修正为最小化提前拖期惩罚, 目标函数的结果比较如表2所示.

表 2　不同策略下的作业提前拖期惩罚比较

案例	B-Q	CMSQ	WSQ	CDQ	结果比较/%
1	2 178.9	2 265.8	2 177.6	2 100.0	3.56
2	520 516	499 107	482 441	441 457	8.49
3	152 649	153 103	152 379	152 134	0.43
4	219 127	217 334	221 119	212 665	2.15

为了分析调度策略的求解效率, 在4种案例情况下, 对不同调度策略的平均运行时间(单位为s)进行比较, 结果如表3所示. 可以看出, 本文提出的基于聚类-动态搜索的CDQ算法性能优于已有文献中的3种改进Q学习算法. 在不同的调度环境中, CDQ算法的调度结果比B-Q、CMSQ和WSQ算法中的最优结果均有所提升, 且缩短了算法的运行时间, 提高了算法的求解效率.

表 3　不同策略下的求解效率比较

案例	B-Q	CMSQ	WSQ	CDQ	结果比较/%
1	31.83	30.94	31.92	30.50	1.42
2	37.84	34.85	39.41	32.28	7.37
3	31.83	31.11	32.08	30.12	3.48
4	41.94	42.35	42.57	41.33	1.45

以市场需求平稳、产品结构复杂的案例4为例, 进一步验证本文所提出的自适应动态调度策略性能. 令交货因子$u_{f1} = 1$, $u_{f2} = 6, 6.5, \cdots, 9$分别对案例4进行300次仿真, 得到4种算法的提前拖期惩罚如图2所示. 同时, 为了分析算法的求解效率, 在相同的仿真环境下, 对4种算法在不同交货因子情况下的平均运行时间进行比较, 如图3所示. 可以看出, 对于不同交货因子的调度情况, 基于CDQ算法的调度策略求解的提前拖期惩罚值均小于其他3种算法对应的调度策略, 同时提高了求解效率.

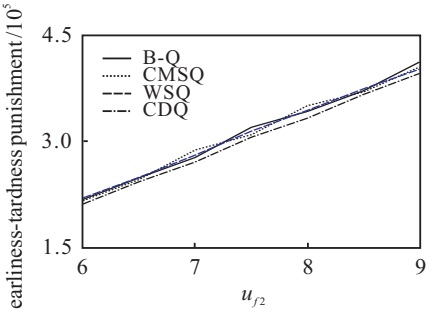

图 2　不同交货因子的提前拖期惩罚比较

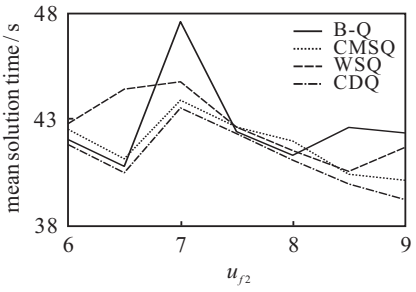

图 3　不同交货因子的求解效率比较

在案例4的基础上, 将作业到达率分别取$\lambda = 1/4.5, 1/5, \cdots, 1/6.5$, 以分析市场需求变化对调度性能的影响. 同样经过300次仿真, 得到提前拖期惩罚和求解效率结果分别见图4和图5.

可以看出, 随着市场需求的变化, 本文算法的提前拖期惩罚和平均求解时间均小于已有文献算法的提前拖期惩罚和求解时间, 这说明本文算法对动态环境变化的适应性更强.

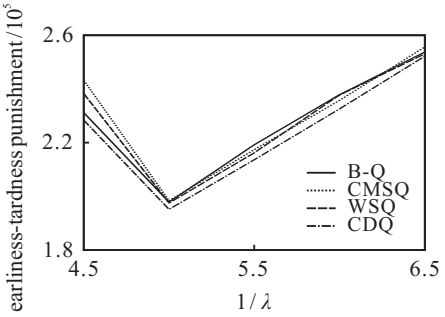

图 4 不同到达率的提前拖期惩罚比较

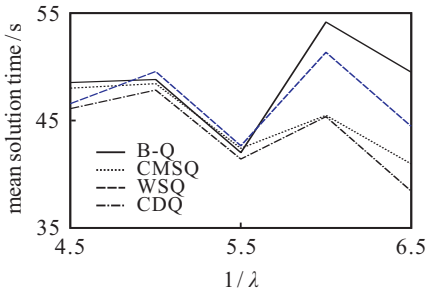

图 5 不同到达率的求解效率比较

5 结 论

本文提出了一种基于多Agent和改进合同网的动态调度仿真系统, 以满足知识化制造系统在不确定生产环境下的自适应调度要求. 在设备Agent的可用时段, 待加工作业向设备Agent进行竞标. 考虑到设备Agent在标书选择, 即调度知识获取方面存在困难, 提出基于聚类-动态搜索的CDQ学习算法用于设备Agent在标书评价中的自适应选择. 该算法采用顺序聚类降低系统状态维数, 定义状态差异度度量状态间的距离, 在Q值迭代中增加最大模糊收益加权均值, 并采用动态贪婪策略搜索迭代中的最优状态-动作对, 以提高算法的求解速度和精度. 不同调度案例下的仿真实验结果表明, 本文所提出的动态调度策略优于文献[8-9]和文献[11]提出的策略, 且对动态调度环境具有更强的适应性, 在调度效益和求解效率上均具有一定的优越性.

参考文献(References)

[1] 严洪森, 刘飞. 知识化制造系统—新一代先进制造系统[J]. 计算机集成制造系统, 2001, 7(8): 7-11.
(Yan H S, Liu F. Knowledgeable manufacturing system—A new kind of advanced manufacturing system[J]. Computer Integrated Manufacturing Systems, 2001, 7(8): 7-11.)

[2] Yan H S. A new complicated knowledge representation approach based on knowledge meshes[J]. IEEE Trans on Knowledge and Data Engineering, 2006, 18(1): 47-62.

[3] 赵宁, 丁文英, 董绍华, 等. 基于约束联动的炼钢-连铸动态调度[J]. 系统工程理论与实践, 2012, 31(11): 2177-2184.
(Zhao N, Ding W Y, Dong S H, et al. Dynamic schedule of steel making-continuous casting based on group adjustment orienting restrict[J]. Systems Engineering-Theory & Practice, 2012, 31(11): 2177-2184.)

[4] Blackstone J H, Phillips D T, Hogg G L. A state-of-the-art survey of dispatching rules for manufacturing job shop operations[J]. Int J of Production Research, 1982, 20(1): 27-45.

[5] 李琳, 江志斌. 虚拟生产系统的自适应动态调度机理及算法[J]. 计算机集成制造系统, 2006, 12(9): 1444-1452.
(Li L, Jiang Z B. Self-adaptive dynamic scheduling mechanisms & algorithm of virtual production systems[J]. Computer Integrated Manufacturing Systems, 2006, 12(9): 1444-1452.)

[6] Xanthopoulos A S, Koulouriotis D E, Tourassis V D, et al. Intelligent controllers for bi-objective dynamic scheduling on a single machine with sequence-dependent setups[J]. Applied Soft Computing, 2013, 13(12): 4704-4717.

[7] Lee K K. Fuzzy rule generation for adaptive scheduling in a dynamic manufacturing environment[J]. Applied Soft Computing, 2008, 28(8): 1295-1304.

[8] 杨宏兵, 严洪森. 知识化制造系统中动态调度的自适应策略研究[J]. 控制与决策, 2007, 22(12): 1335-1340.
(Yang H B, Yan H S. Adaptive strategy of dynamic scheduling in knowledgeable manufacturing system[J]. Control and Decision, 2007, 22(12): 1335-1340.)

[9] 汪浩祥, 严洪森. 基于多Agent可互操作知识化制造动态自适应调度策略[J]. 控制与决策, 2013, 28(2): 161-168.
(Wang H X, Yan H S. Interoperable dynamic adaptive scheduling strategy in knowledgeable manufacturing based on multi-agent[J]. Control and Decision, 2013, 28(2): 161-168.)

[10] Dix J, Seghrouchni A E F. Multi-Agent programming[M]. New York: Springer Science + Business Media, Incorporated, 2005: 53-59.

[11] 王国磊, 林琳, 钟诗胜. 基于聚类状态隶属度的动态调度Q-学习[J]. 高技术通讯, 2009, 19(4): 428-433.
(Wang G L, Lin L, Zhong S S. Clustering state membership-based Q-learning for dynamic scheduling[J]. Chinese High Technology Letters, 2009, 19(4): 428-433.)

[12] Theodoridis S, Kout roumbas K. Pattern recognition[M]. 2nd ed. San Diego: Academic Press, 2003: 634-635.

[13] 魏英姿, 谷侃锋. 基于性能预测的遗传强化学习动态调度方法[J]. 系统仿真学报, 2010, 22(12): 2809-2820.
(Wei Y Z, Gu K F. Genetic reinforcement learning approach to dynamic scheduling based on performance prediction[J]. J of System Simulation, 2010, 22 (12): 2809-2820.)

(责任编辑: 齐 霁)