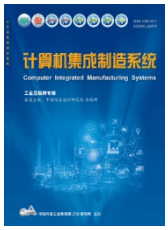

计算机集成制造系统
Computer Integrated Manufacturing Systems
ISSN 1006-5911,CN 11-5946/TP

《计算机集成制造系统》网络首发论文

题目：基于 Q-学习的超启发式模型及算法求解多模式资源约束项目调度问题
作者：崔建双，吕玥，徐子涵
收稿日期：2020-07-02
网络首发日期：2021-01-05
引用格式：崔建双，吕玥，徐子涵．基于 Q-学习的超启发式模型及算法求解多模式资源约束项目调度问题．计算机集成制造系统.
https://kns.cnki.net/kcms/detail/11.5946.TP.20210105.1415.021.html

网络首发时间：2021-01-05 17:00:21
网络首发地址：https://kns.cnki.net/kcms/detail/11.5946.TP.20210105.1415.021.html

基于 Q-学习的超启发式模型及算法求解多模式资源约束项目调度问题

崔建双，吕 玥， 徐子涵
(北京科技大学 经济管理学院，北京 100083)

摘 要：提出了一种基于 Q-学习的超启发式模型并基于该模型设计实现了一种超启发式算法求解多模式资源约束项目调度问题。模型架构分为高低两层，低层由具有多种异构机制和不同参数的元启发式算子组成，高层则依据 Q-学习策略自动选择低层算子。模型把多种优秀的元启发式算法与反馈-学习强化机制有机地整合在一起，具备灵活的可扩展性。为了检验算法效果，从多模式资源约束项目调度问题标杆算例库中选取了上千个规模不等的算例，设计了等价比较实验环节，并与最新公开文献提供的结果做出比较。结果表明基于 Q-学习的超启发式算法在目标值、通用性、鲁棒性等多项性能指标上均表现优异，可以借鉴应用到其他各种组合优化问题。值得一提的是，针对 J30 算例的计算结果有多达 41 个算例获得了比当前公开文献报告的已知最优解更好的结果。

关键词：超启发式；强化学习；Q-学习；多模式资源约束项目调度问题
中图分类号：TP301 **文献标识码**：A

Q-learning based hyper-heuristic algorithm for solving multi-mode resource-constrained project scheduling problem

CUI Jianshuang，LYU Yue ，XU Zihan
(Dolinks School of Economics and Management，University of Science and Technology Beijing，Beijing 100083，China)

Abstract：A Reinforcement Learning based Hyper-heuristic Model and algorithm (RLHM) is presented. The model based algorithm is applied for solving the Multi-mode Resource-Constrained Project Scheduling Problem (MRCPSP). The model architecture is divided into two layers, the Lower Layer Heuristic (LLH) is composed of meta-heuristic operators with multiple heterogeneous mechanisms, and the upper layer automatically selects the LLH operators according to the reinforcement learning strategy. The model integrates a variety of excellent meta-heuristic implementation methods and feedback-learning reinforcement mechanisms, which is scalable. In order to test the effect of the model, thousands of instances of different sizes are selected from the well-known benchmark library of Project Scheduling Problem (PSPLIB), and verification links under different conditions are designed and compared with the results

收稿日期：2020-07-02；修订日期：2020-11-23。Received 02 July 2020; accepted 23 Nov.2020.
基金项目：国家自然科学基金资助项目(71871017)；北京市教委社科基金项目(SM201910037004)。
Foundation items: Project supported by the National Natural Science Foundation, China((No. 71871017)，and the Social Science Foundation of Beijing Municipal Education Commission, China (No. SM201910037004)。

provided by the latest public literature. The results show that the RLHM and its algorithm perform well in a number of performance indicators such as target value, calculation time, and robustness, and are worthy of popularization and application to various combinations and optimization problems in other phases. It is worth mentioning that for the calculation results of the J30, as many as 41 instances have obtained better results than the known optimal solutions reported in the current public literature.
Key words： Hyper-heuristic; Reinforcement learning; Q-learning; Multi-mode resource-constrained project scheduling problem

0 引言

在人工智能、生物信息科学以及智能决策为代表的众多学科领域都存在着大量的以大规模、多模态、非连续性为特征的组合优化问题。针对这类问题，传统的运筹优化方法难以奏效，因而多采用启发式或元启发式算法加以解决。这类方法大多基于直观经验或模拟自然现象，通过嵌入随机性因子，利用进化、群集、仿生等启发式技术，结合广域探查和局域搜索策略，在可接受的时空条件下获得问题的近优解。多年来先后涌现出许多优秀的元启发式算法，例如遗传、进化、模拟退火、禁忌搜索、蚁群、粒子群、人工蜂群、人工免疫、混合蛙跳、人工鱼群等等[1]。

在各种优化算法的应用实践中人们不难观察到如下一些现象：（1）同一种算法对于类型相近的问题或类型相同但数据不同的算例，在效率和效果上差异很大。为了达到理想的优化目标，人们不得不进行算法定制。基于个人经验和灵动、尝试不同的参数、拓扑结构和搜索策略，缺乏理论层次的指导，导致算法应用成本居高不下。（2）虽然不同算法的寻优策略各有千秋，但许多算法展现出相同或相似的实现机制。例如，受自然现象启发、利用群集智能、包含随机成分、不使用梯度信息、有若干可调参数等。这些现象无疑为开发通用型算法、实现算法软件重用、转换即用型算法等需求提供了契机。人们有理由提出并尝试各种算法融合技术，研发一类适应性更强且结果令人可接受的“超启发式”算法。目前，在优化算法研究领域出现的诸如自适应技术[2]、通用算法软件框架[3]、混合元启发式[4]、超启发式[5]、优化算法推荐[6]、算法合成 [7]等方法和技术无不以此为目标，寄希望于通过算法的自动动态匹配或混合技术，降低定制成本，改善应用效果。其中，超启发式（hyper-heuristics）方法与技术已成为当前一大研究热点。超启发式算法通过自动选择或生成一组启发式过程来解决各种优化问题，除了提升算法解决问题的效率之外，更重要的一点是追求算法的通用性和自适应性[8]。

本文提出了一种基于强化学习技术的超启发式模型(Reinforcement Learning based

Hyper-Heuristic Model, RLHM)并在此基础上实现了一种基于 Q-学习的超启发式算法。强化学习是机器学习的一个重要分支，通过与环境交互获得经验，量化为奖惩值并根据奖惩值来决定进一步的执行动作。在 RLHM 中，设计了高层启发式组件对低层启发式算子（Low Level Heuristic, LLH）的选择和移动接受策略。其中，LLH 初步选择了经典的禁忌搜索（Tabu Search, TS），粒子群（Particle Swarm Optimization，PSO），人工蜂群（Artificial Bee Colony，ABC）和蚁群系统（Ant Colony System，ACS）四种具有异构机制的元启发式算子并预留了灵活方便的扩展接口，包括同类算法不同参数的扩展和不同算法算子的扩展。高层策略使用 Q 学习通过奖惩机制来对 LLH 和状态组合进行选择，在本文中，不同的状态对应不同的接受准则，LLH 为下一步执行的动作，Q 学习作为高层策略选择的不单是 LLH，而是状态-动作组合，通过对状态-动作组合的选择，使算法趋向于针对不同的算例选择适合的动作，提高算法应用的效果。

本文设计的超启发式算法在 LLH 选择上采用元启发式算法，而非简单的交叉变异算子，因此 LLH 具有相对独立性，同时具备不依赖于特定问题的通用性。首先，其不同寻优机制的区别有益于实现大范围多样化搜索，充分利用 TS 大规模邻域搜索能力、大范围调节的 PSO 粒子飞行速度和位置、ABC 良好的个体淘汰机制、ACS 构建性的概率选择特长等；其次，不同组合优化问题的编码作为低层算子的基本组件可以预先确定，转换问题仅需要变换不同的编码组件；再其次，LLH 的扩充简单易行，例如增加进化算子、模拟退火算子、异参算子等。算法利用 Q-学习机制智能化地从低层多种元启发式算子中择优使用，充分发挥算子的异构机制的多样性特征，实现了超启发式的概念。

为了检验 RLHM 的应用效果，从资源约束的项目调度问题（MRCPSP）标杆算例库中选取了 1608 个不同规模的问题算例，与公开文献计算结果做出比较。实验结果充分表明了 RLHM 的竞争力和推广价值。

1 超启发式算法与 Q-学习机制

1.1 超启发式算法文献综述

超启发式算法的提出源于各类启发式和元启发式算法存在的不足。正如引言中所指出的那样，不同算法各有优势和劣势，同时每一个具体问题都存在着算法“偏好”。超启发式算法的动机之一就是开发更普遍适用的算法，通过自动化设计和调整启发式算子更高效地解决搜索计算问题[5]。与手动算法定制不同，超启发式算法可被视为根据问题自动化地定制算法[9]。

因此，一个重要的目标是其通用性，基于一组易于实现的低级启发式方法生成质量可接受的解决方案[10]。

“超启发式”一词最早由 Denzinger 等[11]提出，后由 Cowling 等[12]给出实质性定义。事实上，上世纪 60 年代超启发式思想已初露端倪，涉及到运筹学、计算机科学和人工智能等研究领域。代表性的研究成果表现在自动启发式排序[13]、自动规划系统[14, 15]、进化算法中的自动参数控制[16]和自动学习启发式方法[17]等。早期阶段（2000 年之前）的超启发式偏重于启发式自动设计，强调若干启发式规则或方法的组合优于仅使用单个独立的规则或方法。2000 年之后，人们对超启发式算法的认识渐趋完善。陆续出现一些关于超启发式的综述性文献， Burke 等[8,9,10]归纳了超启发式算法的分类及研究现状（见图 1）。超启发式本质上具有“学习”能力，其“学习”的含义在于算法能够从当前运行结果获得经验并向着有利于解决问题的方向调整。根据学习过程中反馈信息的来源，超启发式可以分为“在线”和“离线”学习。前者依据当前即时状态提供的信息决定下一步的搜索走向，后者则依据以往经验决定下一步的搜索走向。

从目前公开发表的文献来看，大多数研究属于在线扰动（或称移动）的选择启发式，其模型由两个层次组成，如图 2 所示。低层包含问题的表示、评估函数和一组特定于问题的 LLH，通过启发式扰动修改当前解；高层则控制 LLH 选择并依据既定规则判断是否接受所做的扰动选择[18,19]。可用的 LLH 选择方法包括简单随机、选择函数、禁忌搜索和强化学习等，而移动接受策略则包括仅改进、任何移动、Metropolis 条件、模拟退火、延迟和 Naive 等[20-23]。在现实应用方面，超启发式算法已经取得了令人鼓舞的成果。文献[24]提出基于大洪水（greatdeluge）策略的超启发式算法解决考试时间表问题。文献[25]用于解决城市公交路线问题（UTRP）。文献[26]提出了一种基于随机自动机网络的超启发式方法，该网络具有学习功能，可控制一组元启发式方法展开搜索。

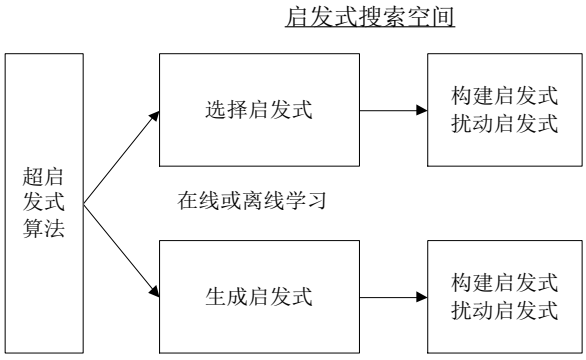

图 1 超启发式算法的分类

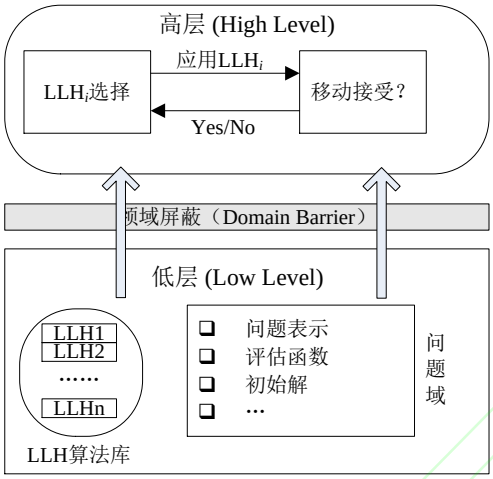

图 2 选择启发式的一般模型

1.2 强化学习与 Q-学习

强化学习主要解决序贯决策问题。Q-学习是强化学习算法之一，专注于从交互中进行以目标为导向的学习。Q-学习过程主要包含学习体的三个联动元素：状态（state）、动作（action）和奖励（reward），以获得最多累计奖励为目标。在没有任何先验信息的情况下首先尝试做出一个动作得到反馈结果，根据反馈结果来调整下一步的动作，在此过程中选择特定情境下得到最大回报的动作。

假设 $S = [s_1, s_2, ..., s_n]$表示学习体的 n 种可能的状态；$A = [a_1, a_2, ..., a_m]$表示 m 个可能的动作。学习体在时刻 t 从状态 s_t 执行动作 a_t 之后进入新状态 s_t+1。r_t+1 表示即时强化信号，即采取动作 a_t 后获得的奖励值（可正可负）；令$\alpha \in [0,1]$表示用于权衡旧状态影响程度的学习率，该值越大越重视以往学习的效果；$\gamma \in [0,1]$表示折扣因子，用于权衡奖励值对于新状态的影响程度，该值越大表明越重视当前学习的效果。$Q(s_t, a_t)$ 表示时刻 t 的 Q 值。每个状态-动作对被给予总累积奖励 Q 值，记录在 Q 表中，通过如下 Q 函数式计算获得：

$$Q_{t+1}(s_t,a_t)=(1-\alpha)Q(s_t,a_t)+\alpha\left[r_{t+1}+\gamma \max_a Q(s_{t+1},a)\right]。$$

Q-学习已被广泛用于各种能够从反馈中获得信息的应用场合。例如，目标转移 Q 学习（Target Transfer Q-learning, TTQL）[27]、超启发式算法自动设计[28]、机器人导航[29]、智能民居能源优化管理[30]、智能游戏控制[31]、动态跟踪控制[32]等等。

1.3 超启发式与 Q-学习

把 Q-学习的奖惩机制与超启发式思想结合，通过评价低层算子的表现来决定下一步的算子选择，就可以实现基于 Q-学习的超启发式算法。不少超启发式算法文献用到了 Q-学习策略，但并未明确提及 Q-学习。

Sim 等[33]提出基于强化学习和禁忌搜索的模拟退火超启发式。Özcan 等[23]提出一种超启发式模型，利用所谓的“大洪水”（Great Deluge）策略作为移动接受方法。Zamli 等[22]提出了一种混合 t-路测试生成策略，采用禁忌搜索作为其高级元启发式，并利用四种低级元启发自适应地选择最合适的算法。Ferreira 等[35]提出了一种“多臂强盗”选择机制策略（multi-arm bandit, MAB），使用 CHeSC 2011[3]挑战赛改编的方法与其他二十种超启发式方法进行了比较，其结果可以与挑战赛中最佳超启发式方法获得的结果相媲美。Di Gaspero 等[36]也提出了一种遵循 Q-学习标准的超启发式模型，研究了立即强化方案的一些变体以及选择策略和学习函数的影响，提供了一类独特的状态和动作表示。Mosadegh 等[37]开发了一种超模拟退火算法，该算法使用 Q 学习策略来选择启发式。张景玲等[34]设计了一种基于强化学习的超启发算法求解有容量车辆路径问题，算法使用强化学习中的深度 Q 神经网络算法构造选择策略，总体求解效果优于对比算法。

严格地看，Q-学习机制满足两个重要特征，即通过试错（trial-and-error）和延迟奖励（delayed reward）来反复探索来实现自动化的与问题无关的搜索。相对于算法定制方法，基于 Q-学习机制的超启发式算法的效率不一定更好，但其效果往往更佳，最主要的优点是摒弃了算法定制，提高了算法通用性水平。

2 RLHM 模型及算法的实现

RLHM 模型及其算法的实现基于如下两个目标：一是算法具备不依赖于特定问题的通用性；二是其寻优机制确保能够实现大范围多样化的全局搜索和小范围的精细搜索。这两个目标都能够通过低层算子加以保证，因为这些算子都不是基于特定问题的启发式算法，而是通用性很强的元启发式算法。其搜索机理可以简单地表述为：利用 Q-学习机制智能化地从低层元启发式算子群中择优使用，充分发挥群算子异构机制的多样化，实现超启发式算法的概念。多种优秀的元启发式算法与反馈-学习强化机制有机地整合在一起，具备灵活的可扩展性。

2.1 RLHM 框架

在图 2 模型基础上本文引入高层 Q-学习策略之后得到图 3 所示 RLHM 框架。其中，低层预留了可扩展的算子接口，预设了多种组合优化问题编码和评估函数。高层针对 Q 表设计了可扩展的多种接受策略。为了增加多样性，接受策略采用随机选择方式获得，一旦选中了一种接受策略，就会根据低层评估函数提供的计算结果和全域最大 Q 值更新 Q 表并进入下一轮动作（算子）选择。

2.2 状态-动作组合对

状态和动作是强化学习的两个要素，通过执行状态和动作的组合获得奖励并帮助算法趋向选择回报最大的动作。把执行 LLH 看成是动作 Action，把执行 LLH 之后的改进与否看成是状态 State。“仅改进”接受和“Naive 接受”是两种接受策略，前者要求计算结果有所改进才接受，拒绝未改进结果；后者则除了接受改进结果之外，以 50%的概率接受未改进结果。表 1 给出 RLHM 算法中的接受策略。

表 1 RLHM 的接受策略

接受策略	描述
仅改进	返回解若优于原解，则无条件接受；否则，重新生成一组新解。
Naive 接受	返回解若优于原解，则无条件接受；否则，以概率 50%接受，以概率 50%重新生成一组新解。

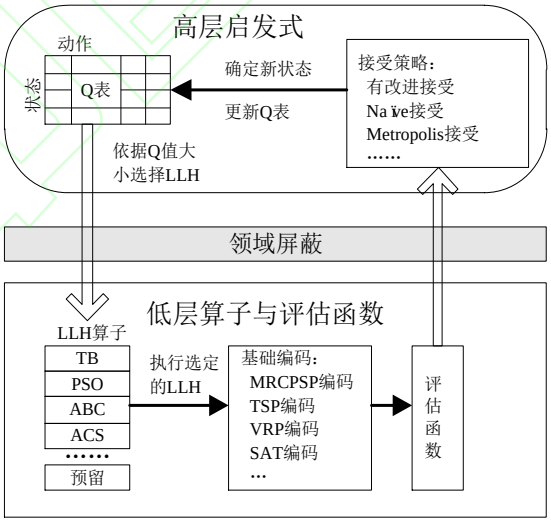

图 3 RLHM 框架

2.3RLHM 算法流程

RLHM 算法流程参见算法 1。为了使算法不陷入局优并增加全局搜索能力，在根据 Q

值大小选择状态-动作组合对时，采用了有保留的贪婪机制，即选择 max $Q(S, A)$状态下对应的动作，但若 max $Q(S, A)<\varepsilon$，则随机选择该状态下的一个动作，其中 ε 为一较小的固定值（0.3）。状态-动作对确定后按照选择的动作执行优化，并及时更新最优值。下一步状态的确定根据执行当前状态-动作对后得到的解是否有所改进作为判断依据。Q 值的更新按照前面给出的 Q 函数式执行。若优化后的种群得到改进，则给予奖励值 r=10；否则，令 r-2→ r。重复以上迭代过程直到可行解数量达到规定值（实验中设定为 5000 次）为止。

算法 1：RLHM 流程

1: Initialization()

Q 表值=0,参数 γ=0.8, α=1-$\frac{0.9*itr}{MaxItNum}$

GlobalValue=min($fitness_{\text{initial}}$)

initial-state()和 initial-action()

%随机指定一个初始状态和动作

2: **While !**Termination-Criteria **Do**

3: Select() %根据当前状态下 Q 值大小选择动作

4: Execution() %执行状态-动作组合

5: Update-Global-Value() %更新全局目标最优解

6: Determine-Next-State() %确定下一步状态

7: Update Q-value() %根据 Q 函数式更新 Q 表

8: **End While**

2.4 可行解数量的确定

对应算法 1 中的 Execution()是执行 LLH 的过程。由于不同 LLH 的实现机制的不同，完整地执行一次 LLH 所需时间不同。为了增加算法的多样性，使算法不至于过早收敛或陷入局优，每个 LLH 可设为运行有限的时间或者迭代次数。本算法将其设置为执行每个 LLH 时记录生成可行解的数量，达到规定数量后无条件跳出执行。

为便于与其他文献结果做出比较，本文通过实验确定无论执行哪一种 LLH，每次可行解数量≤100，控制每个算例累计总可行解数量不超过 5000 次。

2.5 LLH 的设计

传统超启发式算法的 LLH 采用简单启发式序列，多依赖于问题，从而影响了算法的广泛适用性。RLHM 的低层 LLH 算子均采用相对独立模块化的元启发式算法，并可根据问题需要随时扩充新的算法模块。例如，可以根据需要随时给定 TB 算法的不同参数，一组新的参数可以看成是一种新的算法。也可以增加新的元启发式算子，每一个新算法都是一个新的动作，Q 表的规模也会随之扩大。针对相同的问题采用相同的编码格式，可大大提升算法的

通用性。

RLHM 算法初始集成了 TB、PSO、ABC 和 ACS 四种元启发式算法模块，各 LLH 算法参数设计如下：

(1)TB：TB 的基础是邻域搜索算法。禁忌对象 2-opt 或 3-opt 邻域交换；限定邻域解最大数量、破禁策略、禁忌表长等参数。

(2)PSO：使用标准粒子群算法公式，参数学习因子 c_1、c_2，惯性权重 ω。本文粒子的速度和位置的更新方式采用 Jarboui[38]提出的方法。

(3)ABC：设计蜜蜂角色变换上限值 Limit 参数是关键，超过上限值予以淘汰。下一代蜂群的选择采用轮盘赌方式。

(4)ACS：本文在黄少荣[39]提出的蚁群算法基础上进行了改进。参数 ρ、α、β和 Q 可调节，残留信息素更新采用蚁周模型。

3 实验结果及分析

3.1 MRCPSP 的定义

一个项目由 n 个非抢占实活动所组成，通常用网络图 $G(V,E)$ 来描述[40]。V={1,2,⋯,n+1,n+2}代表所有结点的集合，结点 $i\in V$ 表示活动的开始和结束时间点，其中活动 1 和 n+2 是虚拟活动，仅代表项目的开始和结束时点。E=(i, j)代表所有的有向边的集合，活动 i, j 之间遵从完成-开始先后顺序约束。一个活动所需资源和时长表示其一种执行模式，活动 $i\in V$ 可以取模式 $m_i \in M_i = \{i=1,\cdots,|M_i|\}$ 之一来执行，且每个模式的活动时长和各种资源的用量是已知的。所需资源分为可再生和不可再生两类。可再生资源的集合为 R^{ρ}，不可再生资源的集合为 R^{ν}。$r^{\rho}_{i,m_i,k}$ 代表活动 i 取模式 m_i 时第 k 种可再生资源的使用量，$r^{\nu}_{i,m_i,l}$ 代表活动 i 取模式 m_i 时第 l 种不可再生资源使用量。第 k 种可再生资源在项目期内的总量为 R^{ρ}_k，第 l 种不可再生资源最大拥有量是 R^{ν}_l。任意时刻 t 对第 k 种可再生资源的使用总量需要满足 $\sum_i r^{\rho}_{i,m_i,k} \le R^{\rho}_k, t \ge 0$。项目期内第 l 种不可再生资源的消耗总量需要满足 $\sum_i r^{\nu}_{i,m_i,l} \le R^{\nu}_l$。项目调度方案 $S=\{s_1,\cdots,s_{n+2}\}$ 是对各个活动开始时间 $s_i(i=1,\cdots,\ n+2)$ 的一个指定，要求在满足各项活动先后顺序约束和各类资源用量约束的前提下求得项目工期（*makespan*）最小化。MRCPSP 的数学模型描述如下：

$$\min s_{n+2} \quad (1)$$

$$\text{s.t.} \quad s_i + d_{i,m_i} \le s_j \quad \forall (i,j) \in E \quad (2)$$

$$\sum_{\forall t} r^{\rho}_{i,m_i,k} \le R^{\rho}_{l} \quad \forall k \in R^{\rho}, \forall m_i \in M_i, t \ge 0 \quad (3)$$

$$\sum_{t \le S_{n+2}} r^{v}_{i,m_i,l} \le R^{v}_{l} \quad \forall l \in R^{v}, \forall m_i \in M_i, t \ge 0 \quad (4)$$

$$m_i \in M_i = \{i = 1, \cdots, |M_i|\}, \forall i \in n \quad (5)$$

$$s_o = 0 \quad (6)$$

$$s_i = \text{int}^{+} \quad \forall i \in n \quad (7)$$

式（1）是最小化项目工期。式（2）表明活动之间遵从完成-开始时间约束关系，$d_{i,m}$ 是活动 i 取模式 m_i 时的执行时间。式（3）（4）分别是可再生和不可再生资源约束；式（5）确保每一活动仅取一种模式；式（6）要求项目开始时间为 0；式（7）假定所有活动的开始时间均为非负整数。

过去多年来，针对该 NP-难问题已经提出了许多求解方法[41]。Spreche 和 Drexl [42]曾使用以分支定界为代表的精确算法求解该问题，但受搜索空间的制约难以在合理的时间内解决规模较大的问题（迄今为止部分活动数量超过 30 的问题仍处于开放状态）。为此，业界大多求助于启发式[43-45]或元启发式算法，如遗传[46,47]，模拟退火[48,49]，粒子群[38,50]，禁忌搜索[51]，分布估计[52]，混合蛙跳[53]，差分进化[54]，蚁群优化[55]，分散搜索[56]，路径重连[57]等。

3.2 实验环境设置

实验采用 Matlab（R2015b）编程实现。从项目调度问题库 (Project Scheduling Problem Library, PSPLIB)[58]选取规模为 J10、J20 和 J30 不等的 1608 个（各 536 个）MRCPSP 算例作为实验数据集。采用 DELL 笔记本电脑，CPU Intel i7, 主频 2.6GHz，8G 内存。

设计了不同条件下的多个验证环节，并与当前公开文献提供的结果做出比较。

3.3 与最新文献中的计算结果的比较

把 RLHM 实验结果与最新文献[40]列出的多种用于求解 MRCPSP 问题的优化算法进行对比。这些算法大多都报告了 J10 和 J20 两组算例的结果。为了公平起见，本实验每组均选取全部 536 个算例，总计 1072 个算例。表 2 列出了对比结果（表中算法名称以文献作者姓

名缩写表示）。表中数据代表了执行 5000 次可行解得到的平均偏差值。

表 2 与最新文献列出的优化算法做出比较[35,40]

算法名称	J10	J20
MORE97	2.68	13.55
RANJ09	0.17	1.31
OZDA99	0.44	6.05
DAMA09	0.74	1.62
JOZE01	0.99	6.65
JARB08	0.22	2.44
TSEN09	0.32	1.47
RLHM	**0.13**	**1.29**

从实验结果可以发现，RLHM 是这些算法中表现最好的。由于公开文献缺乏关于 J30 算例的进一步报告，针对 J30 的 536 个算例，在此仅报告其计算结果（参见表 3）。RLHM 对 J30 算例的计算结果表明有多达 41 个算例获得了比当前公开文献报告的已知最优解更好的结果。

3.4 与元启发式算法计算结果的比较

RLHM 算法实现了多种元启发式算子的择优使用，针对不同的算例充分利用了不同算子的优势。为了验证这一点，从 PSPLIB 库[58]的 J10、J20 和 J30 随机选取每组 50 个算例，共计 150 个算例，每个算例执行 5000 次可行解，分别取各组算例的偏差均值做出比较。图 4 画出了 RLHM 与分别独立执行的四种元启发式算法（TB、PSO、ABC、ACS）计算结果的比较。从图 4 可以看出，RLHM 得到的目标偏差在三组算例中均小于其他四种元启发式算法，进一步验证了 RLHM 算法的优势。

表 3 获得改进的 J30 算例

算例	改进值	算例	改进值
J3013-3	-1	J3030-1	-6
J3013-9	-1	J3031-1	-1
J3014-9	-3	J3032-9	-3
J3016-7	-1	J3033-1	-4
J3017-4	-4	J3033-9	-1
J3017-8	-2	J3034-1	-5
J3017-10	-4	J3045-1	-1
J3018-3	-1	J3045-3	-4
J3018-6	-3	J3045-5	-1
J3019-9	-2	J3048-6	-2
J3020-3	-2	J3053-2	-11

J3020-6	-5	J3053-3	-4
J3020-7	-6	J3053-4	-1
J3020-10	-4	J3053-8	-2
J3022-1	-4	J3053-9	-1
J3025-2	-1	J3057-8	-1
J3025-5	-3	J3058-7	-2
J3027-3	-1	J3061-2	-1
J3027-6	-2	J3064-8	-7
J3027-10	-6	J3062-1	-1
J3028-7	-1		

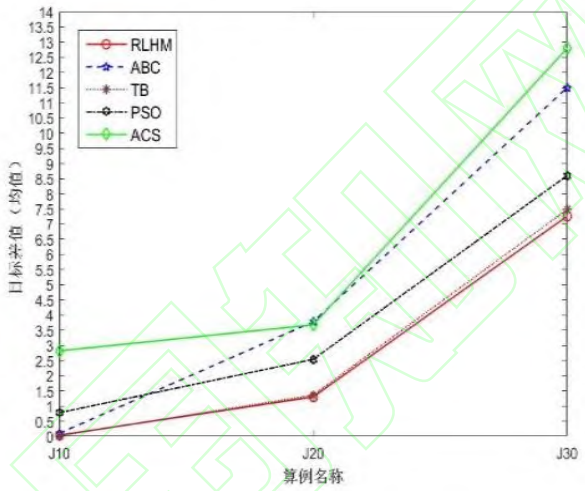

图 4　RLHM 与四种元启发式算法比较

3.5 与随机选择超启发式算法结果对比

RLHM 高层采用了改进接受和 Naive 接受两种预先指定的状态，使用 Q-学习指导 LLH 选择，与传统的随机机制选择 LLH（Random Heuristic Selection, RHS）相比，效果明显有所改善。现设定两种算法的有关参数（终止迭代次数，LLH 相关参数设置等）均一致，算例仍然采用 J10，J20 和 J30 不同规模 150 个算例进行计算，结果对比曲线如图 5 所示。

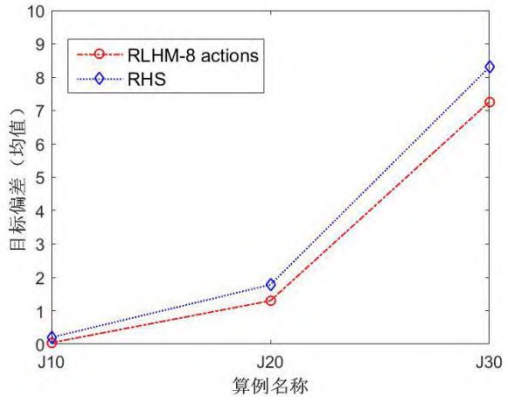

图 5 RLHM 与 RHS 的对比

从图 5 可知，RLHM 比 RHS 平均偏差更小。随着算例规模的增加，差距也在拉大。这说明 RLHM 中 Q-学习机制在选择 LLH 时随着问题规模越大，能力表现越突出。更主要的是 RLHM 没有刻意地去调整哪个参数以适应问题，而是利用 Q 学习机制自动地选择各个算子，实现了超启发式算法的初衷。

3.6 扩充 LLH 及其影响

扩充 LLH 意味着增加新算子的数量，有助于改进搜索空间的多样性，从而增大全局优化的可能性。RLHM 设计了两种增加 LLH 算子的可行方案，第一种方案是直接通过适当修改现有的元启发式算法并集成到低层 LLH 算子集中，正如图 3 低层左侧所示，把模拟退火算法、遗传算法、邻域搜索算法、混合蛙跳等元启发式算法都集成进来。这样的集成只需要前一个算子能够平滑地把本算子计算的结果传递给下一个算子。第二种方案是在现有的各算子基础上修改算法参数来获得新的算子。例如，Pandiri 等[59]曾根据参数值的不同组合改变算法的特性，有效地求解了 k-互连多仓库多旅行商问题。对于一个算子的某个参数值来说，有时其可选的范围很大，也很灵敏，因此，初始可以根据经验选定几种典型的参数，作为不同算子使用，当然也可以自适应地调整不同参数的组合。

为了验证增加新 LLH 带来的效果，本文在前 4 个 LLH 基础上分别设计了两种实验方案。第一方案增加了遗传算子 GA 和模拟退火算子 SA；第二方案改变了 TB 的禁忌表长度和 PSO 的学习因子 c_1, c_2 和惯性权重 ω 的值。

表 4 列出了不同算子（动作）数量下针对前述 150 个算例的计算结果。从表 4 可见，增加新动作带来的最重要的变化是缩短了计算时间（从 4 个算子的计算时间 210 分钟下降到 8 个算子的 107 分钟）。目标值平均偏差均有所下降，说明增加算子数量有助于及时跳出了

变化不大的局部搜索环节，提升算法效率和效果。

表 4 不同算子数量下目标值差均值

RLHM 动作数量		4	6	8
目标平均偏差	J10	0.06	0.06	0.04
	J20	2.02	2.00	0.88
	J30	8.98	8.24	6.54
时间(分钟)		210	160	107

3.7 LLH 算子调用频度分析

一个算子的调用频度定义为执行过程中该算子被调用的次数与全体算子被调用次数之比。该值越大说明该算子被调用的概率越大，因而可以说明算法对其依赖程度以及 Q 学习的效果。表 5 是执行 150 个算例时，LLH 算子平均调用频度统计结果。

表 5　LLH 算子平均调用频度统计

算例	算子平均调用次数			
	TB	PSO	ABC	ACS
J10	2.45	1.60	4.15	0.80
J20	12.90	11.80	10.20	6.85
J30	19.20	9.95	7.20	6.15
合计	11.52	7.78	7.18	4.60

从表 5 可以看出，算子从高到低调用频度分别是 TB>PSO>ABC>ACS。这基本上符合单独应用这四种元启发式算法时的效果，也间接证明了 RLHM 算法在 LLH 选择上使用 Q 学习进行智能选择的可靠性。其次，随着问题规模的增加，优秀算子被调用频率更大但并没有放弃对其他算子的选择，说明了多样性的 Q 学习带来的灵活性。

4 结束语

在优化算法研究领域，超启发式算法和技术已经成为当前一大研究热点。其目的就是要解决传统的元启发式算法机制单一和面向问题定制等不足，能够大大提升解决问题的通用性。从这一视角看，超启发式算法的研究是比发明新算法更有意义的一项工作，能够实现领域内不同策略和技术的交叉融合。

本文提出了一种基于 Q-学习的超启发式算法 RLHM。首先，与传统的超启发式算法不同的是，低层算子不再采用简单的启发式序列，而是使用不同元启发式算法作为独立算子。元启发式算法不依赖于问题，而相同的问题可在不同元启发式算法上统一编码。其次，作为低层算子的元启发式算法可以随意扩充，而常见的组合优化问题的编码也可以根据不同的问

题随时扩充，大大增加了算法的灵活性和通用性。再其次，算法通过 Q-学习的评价机制智能地选择适当状态-动作组合，从而使 RLHM 在 LLH 选择上具备较高的灵活性和可靠性。实验结果证明了 RLHM 的良好特性。未来的研究中，将继续增加高层算子的选择策略，进一步提高低层算子的计算效率，进而提高算法的整体通用性。

参考文献：

[1] Ilhem B, Lepagnot J, Siarry P. A survey on optimization metaheuristics. Information Sciences, 2013, 237: 82–117.

[2] Li K, Fialho A, Kwong S. Adaptive operator selection with bandits for a multi objective evolutionary algorithm based on decomposition. IEEE Transactions on Evolutionary Computation 2014,18(1):114-30.

[3] Ochoa G, Hyde M, Curtois T. Hyflex: A benchmark framework for cross-domain heuristic search. in: Evolutionary Computation in Combinatorial Optimization, Springer, 2012:136–147.

[4] Blum C, Puchinger J, Raidl G R. Hybrid metaheuristics in combinatorial optimization: A survey. Applied Soft Computing Journal, 2011,11(6): 4135-51.

[5] Burke E K, Gendreau M, Hyde M. Hyper-heuristics: a survey of the state of the art. Journal of the Operational Research Society, 2013, 64(12): 1695-1724.

[6] Cui J S, Che M R. An intelligent recommendation system for optimization algorithms based on multi-classification support vector machine and its empirical analysis. Computer Engineering & Science, 2019, 41(1): 92-99.

[7]Wu G H，Mallipeddi R，Suganthan P N. Ensemble strategies for population-based optimization algorithms-a survey. Swarm and Evolutionary Computation, 2019, 44: 695-711.

[8] Burke E K, Kendall G, Newall J, Sonia S. Hyper-heuristics: An emerging direction in modern search technology. in: Handbook of Metaheuristics, Boston, MA: Springer, 2003: 457-474.

[9] Burke E K, Hyde M, Kendall G. A classification of hyper-heuristic approaches. in: Handbook of Metaheuristics, Boston, MA: Springer, 2010: 449-468.

[10] Drake J H, Kheiri A, Özcan E, Burke E K. Recent advances in selection hyper-heuristics. European Journal of Operational Research, 2020, 285(2): 405-428.

[11] Denzinger J, Fuchs M, Fuchs M. High performance ATP Systems by combining several AI methods. International Joint Conference on Artificial Intelligence. Morgan Kaufmann Publishers Inc. 1997.

[12] Cowling P, Kendall G, Soubeiga E. A hyper-heuristic approach to scheduling a sales summit. Practice and Theory of Automated Timetabling III, Third International Conference, Patat, Konstanz, Germany, 2001: 176-190.

[13] Storer R H, Wu S D, Vaccari R. Local search in problem and heuristic space for Job Shop scheduling genetic algorithms. Springer Berlin Heidelberg, 1992.

[14] Gratch J, Chien S, Dejong G. Learning search control knowledge for deep space network scheduling, 1993, 7(4): 135-142.

[15] Gratch J, Chien S. Adaptive problem-solving for large-scale scheduling problems: a case study. Journal of Artificial Intelligence Research, 1996, 4(1): 43-65.

[16] Grefenstette J J. Optimization of control parameters for genetic algorithms. IEEE Transactions on Systems, Man, and Cybernetics, 1986, 16(1): 122-128.

[17] Minton S. Automatically configuring constraint satisfaction programs: A case study. Constraints, 1996, 1(1-2): 7-43.
[18] Özcan E. A comprehensive analysis of hyper-heuristics. Intelligent Data Analysis, 2009, 12(1), 3-23.
[19] Adriaensen S, Brys T, Nowe A. Fair-share ILS: a simple state-of-the-art iterated local search hyperheuristic. Proceedings of the Conference on Genetic and Evolutionary Computation, ACM, 2014: 1303–1310.
[20] Jackson W G, Özcan E, Drake J H. Late acceptance-based selection hyper-heuristics for cross-domain heuristic search. in: Proceedings of the Thirteenth UK Workshop on Computational Intelligence, IEEE, 2013: 228–235 .
[21] Soria-Alcaraz J A, Ochoa G, Sotelo-Figeroa M A. A methodology for determining an effective subset of heuristics in selection hyper-heuristics. European Journal of Operational Research, 2017, 260(3): 972-983.
[22] Zamli K Z, Alkazemi B Y, Kendall G. A tabu search hyper-heuristic strategy for t-way test suite generation. Applied Soft Computing, 2016, 44: 57-74.
[23] Özcan E, Misir M, Ochoa G. A reinforcement learning: Great-deluge hyper-heuristic for examination timetabling. International Journal of Applied Metaheuristic Computing, 2010, 1: 39–59.
[24] Muklason A, Syahrani G B, Marom A. Great deluge based hyper-heuristics for solving real-world university examination timetabling problem: New data set and approach. Procedia Computer Science, 2019, 647-655.
[25] Ahmed L, Mumford C, Kheiri A. Solving urban transit route design problem using selection hyper-heuristics. European Journal of Operational Research, 2019, 274(216): 545-559.
[26] Nesi L C, Righi R R. H2-SLAN: A hyper-heuristic based on stochastic learning automata network for obtaining, storing, and retrieving heuristic knowledge. Expert Systems with Applications, 2020, 153,113426.
[27] Wang Y, Liu Y, Chen W. Target transfer Q-learning and its convergence analysis. Neuro computing, 2020, 392: 11-22.
[28] Choong S S, Wong L P, Lim C P. Automatic design of hyper-heuristic based on reinforcement learning. Information Science, 2018, 436: 89-107.
[29] Mohammad A K J, Mohammad A R, Lara Q. Reinforcement based mobile robot navigation in dynamic environment. Robot. Comput. Integrated Manuf., 2011, 27 (1): 135–149.
[30] Wei Q, Liu D, Shi G, A novel dual iterative-learning method for optimal battery management in smart residential environments. IEEE Trans. Ind. Electron, 2015, 62 (4): 2509–2518.
[31] Mnih V, Kavukcuoglu K, Silver D. Playing Atari with deep reinforcement learning. Technical Report, Deep Technologies, 2013.
[32] Kiumarsi B, Lewis F L, Modares H. Reinforcement Q-learning for optimal tracking control of linear discrete-time systems with unknown dynamics. Automatica, 2014, 50 (4): 1167–1175.
[33] Sim K. KSATS-H H: A simulated annealing hyper-heuristic with reinforcement learning and tabu-search. http://www.asap.cs.nott.ac.uk/external/chesc2011/ index.html.
[34]张景玲, 冯勤炳, 赵燕伟, 刘金龙, 冷龙龙. 基于强化学习的超启发算法求解有容量车辆路径问题. 计算机集成制造系统, 2020, 26 (4) : 1118-1129.

[35] Ferreira A S, Gon R A, Pozo A T R. A multi-armed bandit hyper-heuristic. in: Proceedings of Brazilian Conference on Intelligent Systems (BRACIS), IEEE, 2015:13–18.
[36] Di Gaspero L, Urli T. Evaluation of a family of reinforcement learning cross-domain optimization heuristics. in: Learning and Intelligent Optimization, Springer, 2012: 384–389.
[37] Mosadegh H, Ghomi F, Süer G A. Stochastic mixed-model assembly line sequencing problem: Mathematical modeling and Q-learning based simulated annealing hyper-heuristics. European Journal of Operational Research, 2020, 282(216): 530-544.
[38] Jarboui B, Damak N, Siarry P. A combinatorial particle swarm optimization for solving multi-mode resource-constrained project scheduling problems. Applied Mathematics & Computation, 2008, 195(1): 299-308.
[39] Huang S R. Multi-mode resource constrained project scheduling based on ant colony system algorithm. Computer Applications and Software, 2012, 29(8): 153-159.[黄少荣. 蚁群系统算法求解多模式资源约束项目调度问题[J]. 计算机应用与软件, 2012, 29(8): 153-159.]
[40] Peteghem V, Vanhoucke M. An experimental investigation of metaheuristics for the multi-mode resource-constrained project scheduling problem on new dataset instances. European Journal of Operational Research, 2014, 235(1): 62-72.
[41] Cui J S. Project scheduling problem model and optimization method. Beijing: Science Press, 2018.[崔建双著.项目调度问题模型与算法.北京：科学出版社，2018.]
[42] Sprecher A, Drexl A. Multi-mode resource-constrained project scheduling by a simple, general and powerful sequencing algorithm. European Journal of Operational Research, 1998, 107(2): 431-450.
[43] Coelho J, Vanhoucke M. Multi-mode resource-constrained project scheduling using RCPSP and SAT solvers. European Journal of Operational Research, 2011, 213(1): 73-82.
[44] Peteghem V V, Vanhoucke, M. Using resource scarceness characteristics to solve the multi-mode resource-constrained project scheduling problem. Journal of Heuristics, 2011,17(6): 705-728.
[45] Geiger M J. A multi-threaded local search algorithm and computer implementation for the multi-mode resource-constrained multi-project scheduling. European Journal of Operational Research, 2016, 256(3): 729-741.
[46] Lova A, Tormos P, Cervantes M. An efficient hybrid genetic algorithm for scheduling projects with resource constraints and multiple execution modes. International Journal of Production Economics, 2009, 117(2): 302-316.
[47] Alcaraz J, Maroto C, Ruiz R. Solving the multi-mode resource-constrained project scheduling problem with genetic algorithms. Journal of the Operational Research Society, 2003, 54(6): 614-626.
[48] Bouleimen K, Lecocq H. A new efficient simulated annealing algorithm for the resource-constrained project scheduling problem and its multiple mode version. European Journal of Operational Research, 2003, 149(2): 268-281.
[49] Józefowska J, Mika M. Simulated annealing for multi-mode resource-constrained project scheduling. Annals of Operations Research, 2001,102(1-4): 137-155.
[50] Cui J S, Yan J H. Multiple resource constrained project scheduling problem with discrete particle swarm optimization. Computer Engineering and Applications. Computer Engineering and Applications, 2015, 51(14): 253-257. [崔建双, 杨建华. 多资源约束的项目调度问题离散粒子群算法. 计算机工程与应用, 2015, 51(14): 253-257.]

[51] Mika M, Waligóra G, Waglarz J. Tabu search for multi-mode resource-constrained project scheduling with schedule-dependent setup times. European Journal of Operational Research, 2008, 187(3): 1238-1250.
[52] Wang L, Fang C. An effective estimation of distribution algorithm for the multi-mode resource-constrained project scheduling problem. Computers & Operations Research, 2012, 39(2): 449-460.
[53] Wang L, Fang C. An effective shuffled frog-leaping algorithm for multi-mode resource-constrained project scheduling problem. 2011, 181(20): 4804-4822.
[54] Damak N, Jarboui B, Siarry P. Differential evolution for solving multi-mode resource-constrained project scheduling problems. Computers & Operations Research, 2009, 36(9): 2653-2659.
[55] Li H, Zhang H. Ant colony optimization-based multi-mode scheduling under renewable and nonrenewable resource constraints. Automation in Construction, 2013, 35: 431-438.
[56] Pourghaderi A R, Torabi S A, Talebi J. Scatter search for multi-mode resource-constrained project scheduling problems. IEEE International Conference on Industrial Engineering & Engineering Management, 2008.
[57] Fernandes M A. A path-relinking algorithm for the multi-mode resource-constrained project scheduling problem. Computers & Operations Research, 2018, 92: 145-154.
[58] Kolisch R, Sprecher A. PSPLIB-A project scheduling problem library. European Journal of Operational Research, 1997, 96(1): 205-216.
[59] Venkatesh P, Alok S. A hyper-heuristic based artificial bee colony algorithm for k-interconnected multi-depot multi-traveling salesman problem. Information Sciences, 2018, 463–464: 261–281.

作者简介：

崔建双(1961—), 男, 副教授, 博士, 研究方向：智能优化算法、项目优化调度、商务数据分析等, E-mail:cuijs@manage.ustb.edu.cn;

吕 玥(1996—), 女, 研究方向：智能优化算法、项目优化调度、机器学习算法的应用, E-mail: lvyue87@126.com;

徐子涵(1995—), 女, 研究方向：智能优化算法、项目优化调度、机器学习算法的应用, E-mail: julia1995@126.com.