第 26 卷第 4 期
2020 年 4 月

计算机集成制造系统
Computer Integrated Manufacturing Systems

Vol. 26 No. 4
Apr. 2020

DOI:10.13196/j.cims.2020.04.025

基于强化学习的超启发算法求解有容量车辆路径问题

张景玲,冯勤炳,赵燕伟,刘金龙,冷龙龙

(浙江工业大学 特种装备制造与先进加工技术教育部重点实验室,浙江 杭州 310014)

摘 要:为了更高效地求解物流优化领域中的有容量车辆路径问题,减少陷入局部最优的情况,提出一种基于强化学习的超启发算法。设计了算法的高层启发式策略,包括选择策略和解的接受准则:基于学习机制,使用强化学习中的深度 Q 神经网络算法构造该算法的选择策略,对底层算子的性能进行奖惩评价;利用奖惩值以及模拟退火作为算法的接受准则,对优质解建立序列池,从而引导算法更有效地搜索解空间,并采用聚类思想提升初始解的质量。对有容量车辆问题的标准算例进行计算,并与其他算法对比,统计分析了最优值、误差率和平均值,实验结果表明了所提算法在该问题求解上的有效性和稳定性,总体求解效果优于对比算法。

关键词:车辆路径问题;强化学习;深度 Q 神经网络;超启发算法

中图分类号:TP301.6;U116.2 **文献标识码**:A

Hyper-heuristic for CVRP with reinforcement learning

ZHANG Jingling, *FENG Qinbing*, *ZHAO Yanwei*, *LIU Jinlong*, *LENG Longlong*

(Key Laboratory of Special Equipment Manufacturing and Advanced Processing Technology, Ministry of Education, Zhejiang University of Technology, Hangzhou 310014, China)

Abstract: To reduce the situation of falling into local optimum and solve the capacitated vehicle routing problem, a hyper-heuristic algorithm based on reinforcement learning was. A high-level heuristic strategy was designed, which included selection strategy and acceptance criteria. Based on the learning mechanism, the deep Q neural network algorithm in reinforcement learning was used to construct the selection strategy, and evaluate the performance of the underlying operator with rewards and punishments; Rewards and punishments as well as simulated annealing was used as the acceptance criteria, and a sequence pool was constructed for high-quality solutions, so as to guide the algorithm searching effectively. Also, the clustering method was used to improve the quality of the initial solution. The optimal value was analyzed, error rate and average value were compared with other algorithms. The experimental results show that the proposed algorithm was effect and stable in solving the problem, and the overall solution effect was better than the comparison algorithm.

Keywords: vehicle routing problem; reinforcement learning; deep Q neural network; hyper-heuristic algorithm

0 引言

为了有效分配物流资源,减少运输成本,车辆路径问题(Vehicle Routing Problem, VRP)一直是物流调度领域研究的热点问题。车辆路径问题最早由 Dantizg 等[1]于 1959 年提出,其基本问题为有容量车辆路径问题(Capacitated Vehicle Routing Problem, CVRP),该问题在满足车载量约束条件下,以服务所有客户的最小化车辆行驶距离为目标,达到降低物流配送成本的目的。因此,本文提出一种基于强化学习的超启发算法,致力于更有效地优化配送路径,解决车辆路径问题。

收稿日期:2019-09-17;修订日期:2019-10-24。Received 17 Sep. 2019; accepted 24 Oct. 2019.

基金项目:国家自然科学基金资助项目(61402409);浙江省自然科学基金资助项目(LY19F030017)。**Foundation items**: Project supported by the National Natural Science Foundation, China (No. 61402409), and the Natural Science Foundation of Zhejiang Province, China (No. LY19F030017).

近年来,各种算法被应用于求解 VRP 以及衍生类问题。精确算法方面,如:分支定界法[2],最小化 K-trees[3],动态规划算法[4]等,此类算法能求得全局最优解,但是当客户点规模较大时,求解时间较长;传统启发式算法方面,如:节约法[5],两阶段法[6]等,相较于精确算法,能够更加有效地找寻最优解,但同样在大规模问题上,效率不佳;智能算法,因其既能得到优解,又能保证效率的优点,得到许多研究者青睐。Osman[7]使用模拟退火以及禁忌搜索算法作为策略,在 VRP 上求得了不错的结果。Bullnheimer 等[8]和 Baker 等[9],分别提出了一种改进蚁群算法、遗传算法,用于 VRP,并将其与模拟退火算法和禁忌搜索算法效果相比较。Marinakis 等[10]在粒子群算法中融合了局部搜索和路径重构策略,用于随机需求车辆路径问题中,取得了较好的效果。Zhang 等[11]使用量子进化算法求解以路径及客户满意度两个方面为目标的多目标 VRP。Marshall 等[12]提出了基于语言进化的超启发算法,在解决问题域经验不足的情况下,同样可以求解 VRP 问题,并在实例中得到效果验证。

超启发算法(Hyper-heuristics Algorithm,HH)被称为"寻找启发式算法的启发式算法",其主要由高层启发式策略(High-Level Heuristic,HLH)和底层启发式算子(Low-Level Heuristics,LLH)两个部分组成。LLH 是对实际问题进行独立搜索,而 HLH 是对 LLH 进行科学合理的选择,并对产生的解判别接受。HLH 设计主要指选择策略和解的接受准则两个方面。

超启发算法由于具有通用性、高效性等优点,已被广泛运用于多个领域。Cowling 等[13]将超启发算法用于人员调度,减少了分配时间,提升了分配质量。Kheiri 等[14]研究了基于序列分析的超启发算法,将其用于基准水网调配优化。邱俊荧[15]提出了基于带 path-relinking 的 GRASP 超启发算法,由构造底层算子序列阶段、局部搜索阶段和 path-relinking 阶段组成,并将其与模拟退火算法进行了比较。Walker 等[16]在超启发算法局部搜索算子中添加自适应机制,将其用于带时间窗的车辆路径问题。Sabar 等[17]提出构建蒙特卡洛树搜索框架的超启发算法,即将底层算子看作一棵树,用蒙特卡洛法识别其不同序列。Soria-Alcaraz 等[18]对超启发算法的确定有效启发式子集进行了讨论,提出一种新颖的迭代局部搜索算子。Dokeroglu 等[19]利用标准库中有关最新的元启发式,模拟退火,鲁棒禁忌搜索,蚁群优化和突破局部搜索算子,来解决二次指派问题。Zamli 等[20]将禁忌搜索作为高层选择策略,将教学学习优化算法、全局领域算法、粒子群算法和布谷鸟算法作为底层搜索算子,改进超启发算法。Choong 等[21]设计了基于强化学习的超启发算法,将 Q 算法用于高层选择策略,验证了其算法与其他性能最好的算法效果相当。

鉴于超启发算法优秀的搜索性能,而强化学习又能基于学习评价动作,本文将强化学习策略——深度 Q 神经网络(Deep Q Network,DQN)算法引入超启发算法的高层策略设计,旨在利用学习机制,对底层启发式算子立即奖惩、将来奖惩进行权衡;设置较优序列池,同时使用模拟退火的接受准则,更有效地引导底层算子搜索,获得优质解。并利用所提算法对 CVRP 进行求解,验证其性能。本文首先介绍了 CVRP 的描述与模型,然后构造了一种基于 DQN 算法的超启发算法,最后使用该算法对标准算例进行计算。根据仿真实验的结果,并与其他算法进行比较分析,证明了基于 DQN 算法的超启发算法在解决 CVRP 上的有效性。

1　有容量车辆路径问题

1.1　问题描述

CVRP 描述为:假设有一个已知位置的配送中心,对多个已知位置和需求量的客户点进行货物配送。约束为:每个客户点有且仅有一辆车对其服务;每辆车有一定标准载重量;完成配送任务后,每辆车须返回配送中心。目标为:求在保证最小使用车辆数的前提下,所有车辆行驶的最短总距离(或最低总成本)以及路径分布。由于其是一个 NP-hard 问题,当客户规模较大时,传统的数学规划方式难以求解,而传统启发式算法、智能算法也容易陷入局部最优,无法获得全局最优解的情况。为此,设计能克服以上难点的算法,求得问题最优解,具有较大意义。

1.2　数学模型

将上述问题转化为数学模型[22]:配送中心设为 $i=0$,客户点设为 $L(i=1,2,3,\cdots,L)$,最多车辆数设为 $K(k=1,2,3,\cdots,K)$,每辆车具有相同载重量为 q,每个客户点需求量设为 $d_i(i=1,2,3,\cdots,L)$,客户 i 到客户 j 的距离设为 c_{ij}。定义以下变量:

$$y_{ik}=\begin{cases}1 & \text{客户 } i \text{ 由车辆 } k \text{ 配送}\\ 0 & \text{其他}\end{cases};$$

$$x_{ijk}=\begin{cases}1 & \text{车辆 } k \text{ 从客户点 } i \text{ 到 } j\\ 0 & \text{其他}\end{cases}。$$

数学模型如下：

$$\min \sum_{k=1}^{K}\sum_{i=0}^{L}\sum_{j=0}^{L}c_{ij}x_{ijk}; \tag{1}$$

$$\sum_{i=1}^{L}d_i y_{ik}\leqslant q_k,\forall k; \tag{2}$$

$$\sum_{k=1}^{K}y_{ik}=1,\forall i; \tag{3}$$

$$\sum_{i\in L}\sum_{k\in K}x_{ijk}=1,\forall j\in L; \tag{4}$$

$$\sum_{i,j\in S\times S}x_{ijk}\leqslant|S|-1,S\subset\{1,2,\cdots,L\},$$

$$\text{且}\neq\Phi\forall k。 \tag{5}$$

其中：式(1)表示所求目标函数，即最短总距离；式(2)为每辆车的标准载重量约束；式(3)保证每个客户点都被服务；式(4)保证每个客户有且仅有一辆车服务；式(5)表示消除子回路。

2 基于强化学习的超启发式算法设计

鉴于 CVRP 的 NP-hard 属性，设计了基于强化学习的超启发算法。利用聚类方式提升初始解的质量。在此基础上，将强化学习引入超启发算法的高层选择策略，为了更好地选择底层算子，设计常数 Ck 辅助判别 $State$ 值，利用学习机制，主动记录在所有状态下算子的相对优劣性；结合模拟退火算法，选择性接收优劣算子。并设计序列池，动态存储当前序列，从而一定程度上扩大全局搜索能力。

超启发算法的设计主要包括以下几方面：初始解的设计；高层启发式策略 HLH 设计；底层启发式算子 LLH 设计；算法流程框架设计。

2.1 初始解的设计

对一个可行解的要求是：能够包含所有客户，且每个客户点只出现一次；在满足车辆标准载重量的条件下，粗略划分路径数，即大致确定由 k 辆车运输；每条路径起始点、终点皆为配送中心。如果在算法初期，给予较优质初始解，则有助于减少算法搜索的时间，提升效率。因此，本文采用聚类[23]的思想。但由于通常的聚类方法都需要不断迭代，寻找聚类点，为了降低选取聚类点产生的耗时性，本文将按距离最短选择聚类点，虽不能获得准确点，但已经能达到优化初始解的目的。

具体步骤如下：

步骤 1 对于第 k 条路径，先设配送中心点为 $i=L+1$(也可设为“0”)，即该路径两端点都为 $i=L+1$；随机挑选客户点 $L(i=1,2,3,\cdots,L)$，加入首尾点中间，判断该车辆现载重量情况。

步骤 2 从剩下的客户点中继续随机挑选，依次加入路线，直到超出标准载重量，则产生第 $k+1$ 条路径；将超超出标准载重量的点，加入新路线中；重复循环，当所有客户点都被选取，则一个初始种群个体生成。

步骤 3 多次进行上述操作，生成一定数量个体的种群，数量为 $Npop$。对 $Npop$ 个个体进行路径判断，选出具有最短路径数的个体，记最短路径数为 k，将 k 作为划分块的数量。

步骤 4 计算所有客户点与仓库点的距离 $c_{i-L+1}(i=1,2,3,\cdots,L)$。为了节省聚类分类的时间，将 c_{i-L+1} 升序排列，只取前 k 个点作为聚类中心点，设为 $L^{KC}(KC=1,2,3,\cdots,k)$，$KC$ 代表聚类块，以除聚类中心点外的其他客户点，与各聚类中心的距离最短为原则，进行聚类。

步骤 5 随机排列 KC 块，按车辆载重量分配，依 KC 块排列顺序，随机挑选客户。若 KC 块中客户点未能满足 k 车辆载重，则向 $KC+1$ 块中随机抽取客户点，直至满足，反之则向后延用至 $k+1$ 辆车，共组成 k 条路径，由此产生一个初始解个体。

2.2 超启发算法的高层启发式策略 HLH 构建

超启发算法的高层启发式策略 HLH 主要包括底层算子的选择策略和解的接受准则两方面。这两方面的设计是超启发算法设计的关键。对于底层算子的选择策略，本文将强化学习中的 DQN 算法[24-25]引入超启发算法。对于解的接受准则，本文利用模拟退火的接受准则，对劣势解选择性接收，引导解向较好效果的方向发展。

2.2.1 基于 DQN 的底层算子的选择策略

DQN 算法主要基于 Q-Learning 算法的思想，利用当前状态 $State$，下一步行动 $Action$，下一步状态 $State'$，延用 Q 值的计算方法，建立损失函数，构建神经网络结构，起到预测下一步底层算子性能表现的作用。将 DQN 算法作为底层算子的选择策略，根据当前解的状态(或者解的发展趋势)，挑选下一步底层算子。

(1)$State$ 设计

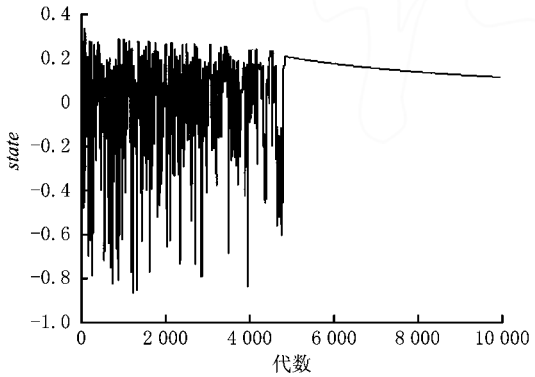
$State$ 表示事物当前的状态。设计目的：尽可能体现 $Action$ 对事物产生的影响以及能够为挑选之后的 $Action$ 起到预测选择作用。由于本文将算法

用于求解 CVRP,难以对 *State* 值提前划分确切范围,*State* 值完全取决于各 *Action* 对解的适应度值的影响,同时为了在挑选算子中,保证解的多样性,*State* 值在一定时期(未求到最优解的情况下),须在一定范围内不断变化。将 *State* 代表适应度值的变化程度,设计当前代的适应度值为 fit,上一代的适应度值为 fit',上一代的平均适应度为 $\mathrm{avg}(fit')$,设计 *State* 的表达式为以下两种:

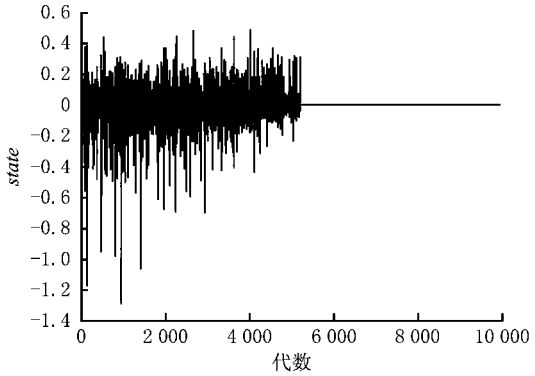
$$State = -(fit - avg(fit'))/avg(fit'), \quad (6)$$

$$State = -(fit - fit')/fit'。 \quad (7)$$

对以上两个表达式进行实验仿真,设初始状态为"0",结果如图 1 所示。



a 式(6)*State*值变化情况图



b 式(7)*State*值变化情况图

图1　式(6)与式(7)*State*值的变化情况图

由图 1 可得,两式在一定代数之内 *State* 因外部环境都呈现上下摇摆的情况。但是式(6)容易在求得局部最优解后,随着代数的增加,呈现"假 *State*"状态,此时的 $fit = fit'$ 为局部最优值,一直未发生变化,但 *State* 由于

$$\mathrm{avg}(fit') = \sum_{iter=1}^{G} fit'/iter, \quad (8)$$

式(8)中分子增长后的值小于分母($iter$ 为迭代次数),整体呈现逐渐下降趋势,是对当前状态的错误反应。而式(7)表示的 *State* 值在局部最优后稳定为 0,正确反映当前状态,有利于算法对产生局部最优状态的判别,更符合设计要求。

因为在此类求最优问题中,单纯的运行趋势无法确切表示当前状态。在算子选择中,本文采用了 3 种类型的算子,将其分为局部优化算子与变异算子两大类,鉴于两类算子优化性能不同,为了更好地体现算子对当前状态的影响,添加 *State* 值的辅助判别机制:利用常数 *Ck* 值,作为 *State* 基数,需满足的条件为,对 *State* 值的影响远大于由 fit 值变化引起的影响(起到区分算子种类作用,利于后续求解算子奖惩值),经测试观察,fit 值变化引起的式(7)大小变化范围见如 1 右图所示,值约为[−1.4,0.6],变化区间的长度约为 2。经分析,若局部优化算子 $Ck=10$,变化区间长度占 *Ck* 值的 20%,影响较大,基准性较差;取局部优化算子 $Ck=20$,变化区间长度占 *Ck* 值的 10%,满足既能扰动 *State* 值,且影响较小的条件;取局部优化算子 $Ck=60$,变化区间长度占 *Ck* 值的 3%,扰动性较低。因此,最终选择局部优化算子 $Ck=20$。因为基准值越大,*State* 值越大,越容易被选择。且由于变异算子较局部优化算子效果更具间接性,即变异算子根据适应度值较优的准则不容易被选择,为了平衡两种算子被选择的概率,定变异算子的 *Ck* 值为局部优化算子的 2 倍,变化区间长度占 *Ck* 值的 5%。由于变异算子相较于局部优化算子优化效果更间接性,定变异算子的 *Ck* 值为局部优化算子的 2 倍,变化区间长度占 *Ck* 值的 5%。因此定义如下:

$$Ck = \begin{cases} 20 & \text{局部优化算子} \\ 40 & \text{变异算子} \end{cases}。$$

则

$$State = -(fit - fit')/fit' + Ck。 \quad (9)$$

(2)*Action*,*Reward* 设计

由于本文在 HLH 中使用算法的目的是选择 LLH 的算子,利用算子在问题解的空间中,搜索优质解,*Action* 值为 LLH 算子的指代值(如算子"1",算子"2"……)。

在考虑 *State* 值时,已经对算子种类进行对应区分,因此扰动算子的 *Reward* 值判断方法,不再区别于其他种类算子。因此,利用 *Reward* 代表立即回报值,若当前解的质量比上代解的质量提升,则 $Reward=1$;未提升则 $Reward=0$;质量下降,则 $Reward=-1$。

(3)DQN 算法主要结构部分设计

DQN 算法结构中主要包括经验池、估值网络、

目标网络。

为了提供历史经验，对 Q 值计算提供参考，在 DQN 算法中设计经验池(Experience Pool，EP)。EP 存储的是 N^E 组数据，N 代表经验池的容量。每组数据由 $State$，$Action$，$Reward$，$State^t$ 四个部分组成，记录的是当前事物状态以及进行下一步动作后的立即回报值和下一步状态值。

传统的 Q 值计算与神经网络相结合的算法，只采用一个神经网络，若其中具有错误历史(即某个连续时刻，某个动作奖赏极大)，则会使该动作的 Q 值评价大于实际性能水平，因此设计估值网络与目标网络来均衡历史的影响。估值网络与目标网络是同结构的神经网络。包含同样的隐藏层数及结构，以及相同个数的神经元节点。输入层节点数根据 $State$ 输入值设置为一个，输出层节点数根据低层算子数量进行设置。利用随机，初始化估值网络的阈值 ω_e、权值 b_e 以及目标值网络的阈值 ω_t、权值 b_t。估值网络的输入值为 N^E 组中某一组的 $State$，目标值网络的输入值为对应组中的 $State^t$。在估值网络输出值中取对应第 N^E 组中 $Action$ 所对应的 Q_e，即 $Q_e(Action)$，而在目标值网络的输出值中取最大的输出值 $\max(Q_t)$，将两者根据下式进行计算，作为更新估值网络的损失函数值的判断：

$$Loss = ((Reward + \gamma \cdot \max(Q_t)) - Q_e(Action))^2。 \tag{10}$$

根据 $Loss$ 值，对 ω_e 和 b_e 作更新处理，其中 γ 为折扣率。一定代数以后，将 ω_e 和 b_e 分别替代 ω_t 和 b_t，以此更新目标网络。利用更新后的估值网络，对当前状态的下个动作产生的影响进行预测，根据预测情况选择 $Action$ 值。

2.2.2 解的接受准则

接受准则的目的就是判断是否需要用当前解代替先前解。当产生改进解时，必然接受；当产生非改进解时，一律接受，容易使算法整体运算缓慢；一律拒绝，则会导致种群缺乏多样性，最终陷入局部最优的情况。因此，接受准则对算法的收敛速度及优化精度有很大影响。张景玲等[26]采用模拟退火算法(Simulated Annealing，SA)作为接受准则，取得了不错的效果，因此本文也引用其作为算法的接受准则，以概率 p 选择性接受非改进解，

$$p = \exp(\Delta E / T_k), \tag{11}$$

$$T_{k+1} = T_k \cdot \beta。 \tag{12}$$

其中：ΔE 表示算子作用后，前后解的质量差，β 表示降温系数，k 为温度计数器。

2.2.3 搜索空间优化

本文的主体搜索方式为先分配后排序，以减小搜索空间的目的进行设计。由 2.1 节可知，采用先将客户点按载重分配，在满足载重的条件下，进行顺序优化的方法。在算法过程中，采用缩小解的搜索空间的思想[27]设计序列池(Sequence pool，SP)，保存算子优化后得到的最优解序列(一辆车的路径)，进行后续搜索。如图 2 所示。

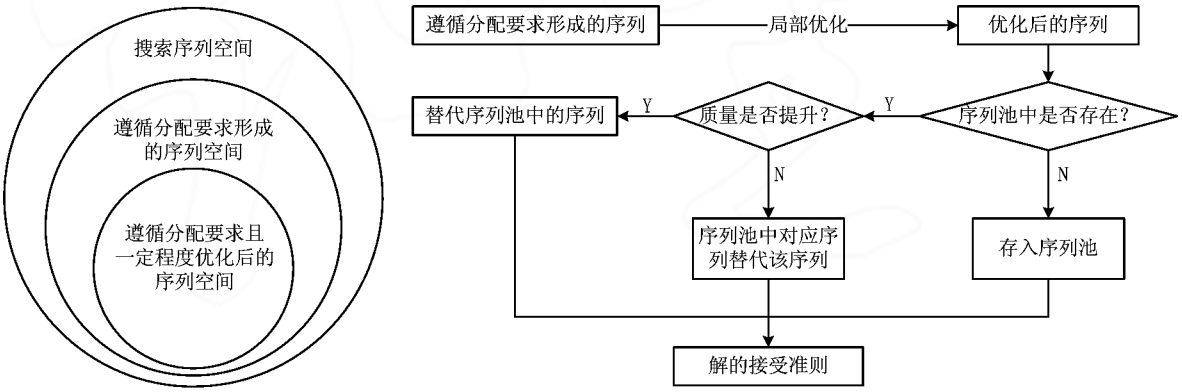


图2 搜索空间优化示意图

图 2 左图代表搜索空间所属，可知如果没有任何约束，只对序列进行排序，则搜索空间最大；若利用载重等约束条件进行限制，则相对缩小了搜索空间；在约束后的基础上，再次进行序列的优化，则能获得更小的搜索空间。图 2 右图代表实施过程，即因为采用局部搜索算子优化后，所得的序列(每辆车的路径分布)都是较优序列，搜索序列池，判断是否存在该序列点集。若不存在，则直接存入序列池；存在则对比，替代该序列或者更新序列池。

2.3　超启发算法的底层启发式算子 LLH 设计

根据 VRP 的特点,设计对应的几种算子,主要为局部优化算子(Local Research,LLH-L)、变异算子(Mutation,LLH-M)和破坏与重构算子(Location-based Radial Ruin,LLH-LR)3 大类。将变异算子与破坏重构算子合为一类,具体算子为如表 1 所示。

表 1　底层算子表

作用领域	算子类型
局部优化算子	1)LLH-L-2opt
	2)LLH-L-Swap
	3)LLH-L-Relocate
	4)LLH-L-D-2opt
	5)LLH-L-D-Swap
	6)LLH-L-D-Relocate
变异算子	7)LLH-M-2opt
	8)LLH-M-Interchange
	9)LLH-M-Oropt
	10)LLH-M-Removal Shaw
	11)LLH-M-Shift

将其根据作用分类是为了方便设计 $State$ 的基数。局部优化算子主要通过点或者部分交换以及重定位的方式,对路径进行优化,作用后能够绝对判别作用效果($Reward$ 值大,则效果好,反之效果差);变异算子主要起到扰动作用,作用后作用效果并非可以绝对判别。1)~3)是路径内局部优化算子;4)~6)是路径间局部优化算子;7)~11)为变异算子。

2.4　基于强化学习的超启发式算法框架设计

利用 DQN 算法的学习机制,对在不同状态下,使用不同 LLH 算子对解产生的影响作评价,保留较优序列和较优解,由此选择相对较优的 LLH 算子进行解的搜索,设计如下算法流程(如图 3):

步骤 1　初始化。先生成 $Npop$ 组个体的种群,得到使用的最小车辆数为 k,利用聚类思想将客户点区域划分为 k 块,称划分成的每块区域为"KC 块",由 KC 块随机挑选生成可行解组 $P(p_i=p_1,p_2,p_3,\cdots,p_{NP})$,计算种群适应度 $f(f_i=f_1,f_2,f_3,\cdots,f_{NP})$。随机挑选一组可行解 p_i 以及对应适应度值 f_i,设 P_B 为最优解个体,F_B 为最优适应度值,设 LLH 算子数量为 N^A,$Action$ 取值为 $(1,2,3,\cdots,N^A)$ 整数,初始化 $P_B=p_i$,$F_B=f_i$,$State=0$,$Action=\mathrm{random}(N^A)$(随机挑选一个范围 N^A 中的数)。

步骤 2　经验池、序列池存储。操作上步 $Action$ 后,产生的个体为 Ind,适应度值 fit,根据适应度值,判断立即回报值 $Reward$,此时状态即为"下一个状态",判断该 $State$ 和 $State^t$ 所属状态,利用式(12)计算 $State^t$ 值。设 EP 代表经验池,将上述值存入,则 $EP_{nE}=[State,Action,Reward,State^t]$,$nE$ 代表经验池中数据组数。当达到一定次数后,判断此时 $State$ 值所属状态,如果为 $15\leqslant State\leqslant 25$,则此时 $Action$ 为路径内算子,对此时的序列进行筛选,质量优则存入 SP,SP 代表序列池,反之,则更新序列。SP 设常量 Q_{sp} 为容量,且每次对比 SP 中序列,若此时序列在 SP 中有对应序列集,则 SP 中该序列计数一次。当 SP 容量已满,则刷新对比次数最少的序列。

步骤 3　解的接受保留。判断,如果 $fit<fit'$,则说明此时解的适应度值更好,则保存解及解的适应度值,令 $State=State^t$,$fit'=fit$;如果 $fit\geqslant fit'$,则采用模拟退火判别,若概率 $p>$随机值,则同样保留好解,同时更新状态,反之,则舍去该解,此时 $State^t=State$,$fit'=fit'$。

步骤 4　判断经验池容量。判断经验池内组数 nE,$n\geqslant N^E$,则进入步骤 7 学习环节,否则,进入步骤 5 选择 $Action$ 步骤。

步骤 5　选择 $Action$。设置 $epsilon$ 值,若随机值$>epsilon$,将 $State$ 值,输入估值网络,输出 Q_e 值,取 $\max(Q_e)$ 所对应的 $Action$,若随机值$<epsilon$,则根据此时 $State$ 值,令 $Action=random(N^A)$,此时 N^A 为对应 $State$ 值的算子序号。

步骤 6　保留全局最优解,判断算法是否结束。若 $fit\leqslant F_B$,更新全局最优解,$F_B=fit$,$P_B=Ind$,否则保留原有解。若 $nG\geqslant G$,返回步骤 2,否则算法结束。

步骤 7　选择学习样本,并初始化神经网络。从 EP 中随机挑选 N^S 组,作为学习样本,记为 ESP。初始化估值网络和目标网络的阈值和权值 ω_e、b_e、ω_t、b_t。

步骤 8　神经网络学习更新。估值网络中输入 $State_{nS}^{ESP}$(ESP 样本中第 nS 个样本中的 $State$ 值,余同),计算后取 $Q_e(Action_{nS}^{ESP})$,目标值网络中输入 $State_{nS}^{tESP}$,计算后取 $\max(Q_t)$,利用式(10)计算损失值 $Loss$,更新估值网络 ω_e、b_e。

步骤 9　更新目标值网络。判断学习代数 $Ln\geqslant LN$,则令 ω_t、b_t 替代 ω_e、b_e 的值。

步骤 10　判断学习结束情况。若学习代数 $Ln\leqslant(3/4)\times N^S$,则进入步骤 8 继续学习更新。反之,则进入步骤 5 选择 $Action$,返回主循环。

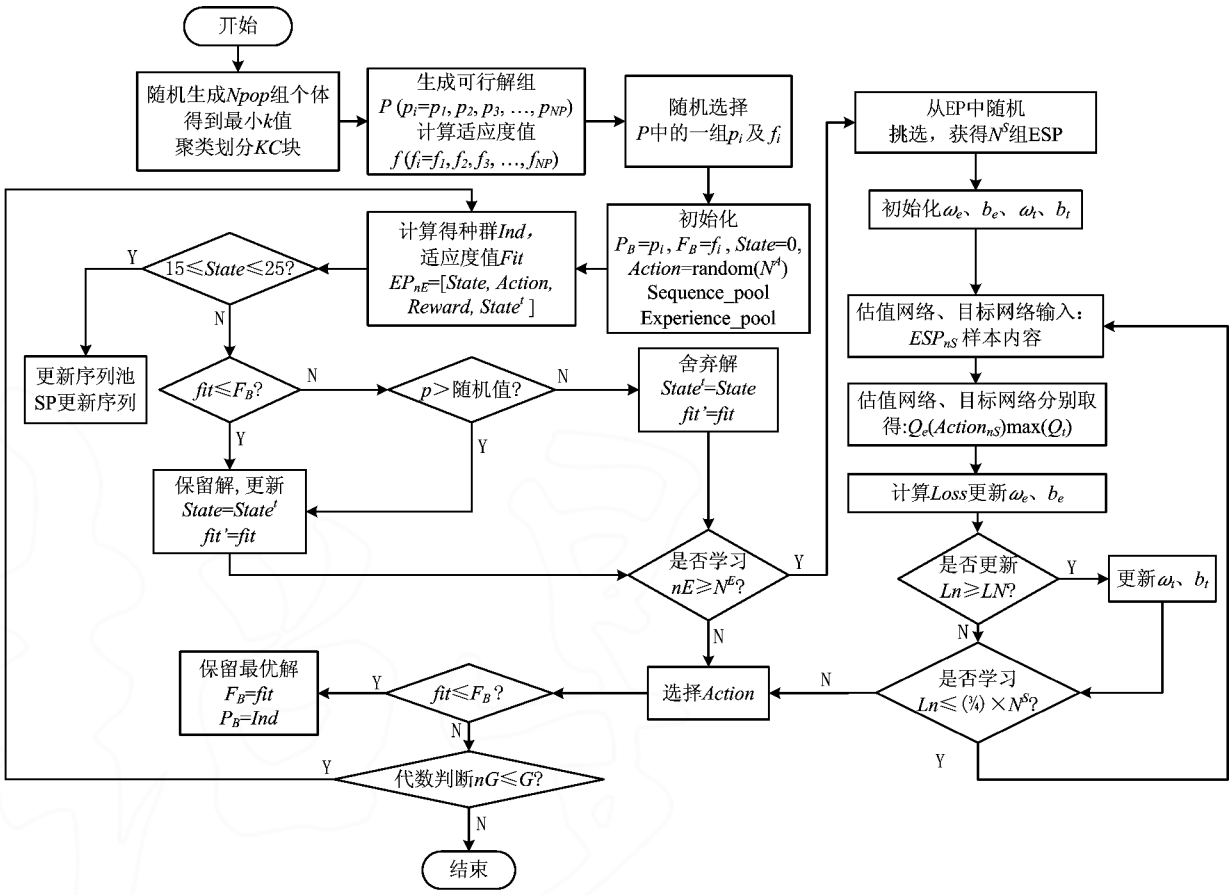


图3 算法框架图

2.5 基于强化学习的超启发式算法复杂度分析

对算法的时间复杂度进行分析是对算法运行时间性能的一种评估方法。本文算法为基于强化学习的超启发算法，简称为 HH-RLDQN。影响本文算法的复杂度计算的因素有：初始化种群的种群规模 NP，客户数量 N，车辆数量 K，迭代次数 $G_{\max}$，底层算子数量为 NL，序列池容量 SP，神经网络中学习样本数量 LN，神经网络的隐藏层节点数 HN。

对本文算法计算时间复杂度：随机化初始并聚类种群复杂度为 $O(2\times NP\times N^2+NP\times K)$，计算适应度值复杂度为 $O(NP)$；高层策略中，求解经验池(State，Action，Reward，State^t 三个值的求解以及写入经验池)复杂度为 $O(1+1+1+1)$，每次所得序列与序列池中序列的对比次数设为最大值(与全池序列对比)复杂度为 $O(SP)$，接受策略复杂度为 $O(1)$，神经网络学习复杂度约为 $O(LN\times(HN+3\times HN+1))$，选择 $Action$ 选耗时更长复杂度为 $O(HN)$；底层算子执行，由于每一步选择的算子未知，选择耗时最长的算子复杂度为 $O(N^2)$[28]。算法迭代一次的复杂度约为 $O(2\times NP\times N^2+NP\times K)+O(NP)+O(4)+O(SP)+O(1)+O(LN\times(HN+3\times HN+1))+O(HN)+O(N^2)$，算法总体计算复杂度约为：

$$\begin{aligned}O(\text{HH-RLDQN}) &= O(2\times NP\times N^2+NP\times K)+\\ &G_{\max}\times(O(NP)+O(4)+O(SP)+O(1)+\\ &O(LN\times(HN+3HN+1))+O(HN)+O(N^2))\\ &\approx O((G_{\max}+NP)\times N^2+G_{\max}\times LN\times HN)。\end{aligned}\tag{13}$$

由式(13)可得，算法的总体复杂度约为 $O((G_{\max}+NP)\times N^2+G_{\max}\times LN\times HN)$，即高层策略的复杂度除了神经网络部分，其他忽略不计。本文算法复杂度主要的影响因素还是迭代次数 $G_{\max}$，初始化种群的种群规模 NP，客户数量 N，神经网络中学习样本数量 LN，神经网络的隐藏层节点数 HN。文献[29-30]中计算总体复杂度用本文符号表示都为 $O(G_{\max}\times NP\times N^2)$，因为

$$O((G_{\max}+NP)\times N^2)<<O(G_{\max}\times NP\times N^2)\ \text{且}\ HN\approx NP。\tag{14}$$

本文的神经网络中学习样本数量 LN 取值范围始终小于 1 000,则 $LN \leqslant N^2$ 恒成立,即

$$O((G_{\max}+NP)\times N^2+G_{\max}\times LN\times HN) \leqslant O(G_{\max}\times NP\times N^2), \quad (15)$$

式(15)恒成立。由上述分析可得,本文算法的计算复杂度与其他群体智能进化算法计算复杂度在同一个量级,在计算机计算的可承受范围之内。

3　数值实验

实验环境 Intel(R) core-i5-3230M, 12 GB RAM,MATLAB 语言编写程序。经过反复测试,程序中使用的参数有 Q 值函数中折扣率 $\gamma=0.8$, $epsilon$ 初始值=0.5,迭代最大代数 $G_{\max}=10\hat{}6$,经验池 $N^E=800$,学习挑选样本 $N^S=600$。

实验利用 HH-RLDQN,对 CVRP 的标准算例进行求解。分别选取 Set A,Set E 和 Set P 的标准算例求解,对每个算例计算 20 次。所有算例可在网址: http://neo.lcc.uma.es/vrp/vrp-instances/capacitated-vrp-instances/下载。其中算例最优距离解(BK),实验所得最优解距离解(Min),所得最优解距离解平均值(Avg),实验所得最优解与算例最优解误差(DEV%)(DEV=(Min−BK)/BK)。

表 2 为本文使用的 HH-RLDQN 算法及文献[29-32]对 Set A 标准算例的计算结果。表中加粗且标有"*"的数字表示求得当前算例最优解。由表 2 可知,文献中的 LNS-ACO 和 SC-ESA 两种算法都求得 27 个算例中的 17 个算例的最优解,求最优解成功率为 62.96%;HVNSOS 算法求得 20 个最优解,求解成功概率为 74%;NHQEA 算法求得 23 个最优解,求解成功率为 85.18%。本文算法求得 22 个最优解,求解成功率为 81.48%,相较于前两种算法,提升 18.52%,提升较为明显,相较于第 3 种算法提升 7.48%,具有一定优势,相较于第 4 种算法,仅低了 3.7%。求得各算法对于该 27 个算例最优解的误差率平均值分别为:0.6%,0.16%,0.13%,0.07%,本文算法的误差平均值为 0.07%,相较于前 3 种算法降低了 0.53%,0.09%,0.06%。值得一提的是,本文算法在低于 NHQEA 算法最优值求解成功率的情况下,仍与其具有相同的误差平均值。在求解 n61,n64 和 n65 这 3 个算例上,本文算法在本次实验中相较于 NHQEA 算法,具有一定优势。对于 20 次实验的最优值平均值观察可得,有 8 个算例平均值均达到算例最优值,除 n69 和 n80 算例,平均值与算法所得最优值相差为 9 和 13,其他算例差值均在[1,6]之间,可见本文算法在此些算例中求解具有相对准确性。由此可得,本文算法相对于多种群搜索的其他 4 种算法而言,在 Set A 标准算例上,具有较好的搜索效果。

表 2　CVRP Set A-标准算例测试

Instance	BK	LNS-ACO[31]		SC-ESA[32]		HVNSOS[29]		NHQEA[30]		HH-RLDQN		
		Min	DEV%	Min	DEV%	Min	DEV%	Min	DEV%	Min	Avg	DEV%
A-n32-k5	784*	784*	0	748*	0	748*	0	748*	0	784*	784*	0
A-n33-k5	661*	661*	0	661*	0	661*	0	661*	0	661*	661*	0
A-n33-k6	742*	742*	0	742*	0	742*	0	742*	0	742*	742*	0
A-n34-k5	778*	778*	0	778*	0	778*	0	778*	0	778*	778*	0
A-n36-k5	799*	799*	0	799*	0	799*	0	799*	0	799*	799*	0
A-n37-k5	669*	669*	0	669*	0	669*	0	669*	0	669*	669*	0
A-n37-k6	949*	949*	0	949*	0	949*	0	949*	0	949*	950	0
A-n38-k5	730*	730*	0	730*	0	730*	0	730*	0	730*	730*	0
A-n39-k5	822*	822*	0	822*	0	822*	0	822*	0	822*	824	0
A-n39-k6	831*	831*	0	831*	0	831*	0	831*	0	831*	833	0
A-n44-k6	937*	937*	0	937*	0	937*	0	937*	0	937*	939	0
A-n45-k6	944*	958	1.48	944*	0	944*	0	944*	0	944*	960	0
A-n45-k7	1 146*	1 146*	0	1 146*	0	1 146*	0	1 146*	0	1 146*	1 149	0
A-n46-k7	914*	914*	0	914*	0	914*	0	914*	0	914*	914*	0
A-n48-k7	1 073*	1 084	1.03	1 084	1.03	1 073*	0	1 073*	0	1 073*	1 074	0
A-n53-k7	1 010*	1 010*	0	1 011	0.10	1 010*	0	1 010*	0	1 010*	1 016	0
A-n54-k7	1 167*	1 167*	0	1 168	0.09	1 167*	0	1 167*	0	1 167*	1 171	0
A-n55-k9	1 073*	1 073*	0	1 073*	0	1 073*	0	1 073*	0	1 073*	1 074	0

续表 2

A-n60-k9	1 354*	1 354*	0	1 355	0.07	1 354*	0	1 354*	0	1 354*	1 358	0
A-n61-k9	1 034*	1 067	3.19	1 034*	0	1 035	0.09	1 036	0.19	1 035	1 061	0.10
A-n62-k8	1 288*	1 308	1.55	1 298	0.78	1 291	0.23	1 288*	0	1 291	1 305	0.23
A-n63-k9	1 616*	1 649	2.04	1 624	0.50	1 628	0.74	1 616*	0	1 616*	1 629	0
A-n63-k10	1 314*	1 329	1.14	1 315	0.08	1 319	0.38	1 317	0.23	1 318	1 321	0.03
A-n64-k9	1 401*	1 415	1.00	1 409	0.57	1 414	0.93	1 414	0.57	1 412	1 418	0.79
A-n65-k9	1 174*	1 185	0.94	1 178	0.34	1 177	0.26	1 177	0.93	1 174*	1 179	0
A-n69-k9	1 159*	1 170	0.95	1 159*	0	1 159*	0	1 159*	0	1 159*	1 168	0
A-n80-k10	1 763*	1 815	2.95	1 776	0.74	1 779	0.91	1 763*	0	1 776	1 789	0.74
Average			0.6		0.16		0.13		0.07			0.07

表 3 是本文使用的 HH-RLDQN 算法，及文献[31-32]对 Set E 标准算例的计算结果，表中“～”表示文献中并未给出计算值。由表 3 可知，LNS-ACO 和 SC-ESA 两种算法分别求得 8 个算例中的 4 和 5 个算例的最优解，求最优解成功率为 50% 和 62.5%；本文算法求得 9 个算例中的 5 个算例最优解，求最优解成功率为 55.56%，求最优解成功率与两种算法效果相同。求得两种算法对于 8 个算例的最优值平均误差率为 0.68% 和 0.63%，本文算法为 0.32%，分别下降 0.36% 和 0.31%，几乎为一半，尤其在 n76-k7 和 n76-k8 算例上，误差率仅为 0.01% 和 0.03%，效果相较于两种算法，极其优秀。由以上数据可知，本文算法，在 Set E-标准算例上相对于 LNS-ACO 和 SC-ESA 同样具有较好搜索效果。

表 3 CVRPSet E-标准算例测试

Instance	BK	LNS-ACO[31]		SC - ESA[32]		HH - RLDQN		
		Min	DEV%	Min	DEV%	Min	Avg	DEV%
E-n22-k4	375*	375*	0	375*	0	375*	375*	0
E-n23-k3	569*	569*	0	569*	0	569*	569*	0
E-n30-k4	503*	503*	0	503*	0	503*	503*	0
E-n33-k4	835*	835*	0	839	0.48	835*	835*	0
E-n51-k5	521*	～	～	～	～	521*	521*	0
E-n76-k7	682*	695	1.91	696	2.05	683	688	0.01
E-n76-k8	735*	744	1.22	743	1.09	737	741	0.03
E-n76-k14	1 021*	1 030	0.88	1 021*	0	1 032	1 041	1.07
E-n101-k14	1 067*	1 082	1.41	1 082	1.41	1 086	1 094	1.78
Average			0.68		0.63			0.32

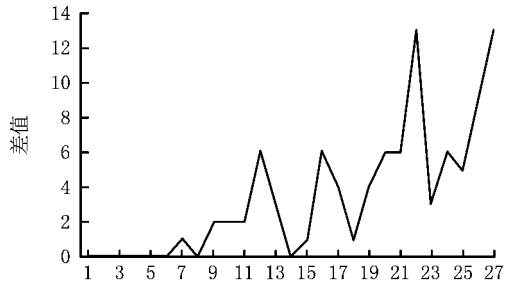
表 4 为本文使用的 HH-RLDQN 算法及文献[29-32]对 Set P 标准算例的计算结果。由表 4 可知，除了 SC-ESA 算法整体搜索求解效果不是很理想，17 个算例中，只求得 6 个，LNS-ACO，HVNSOS，NHQEA 以及本文算法都具有较好的搜索求解效果。LNS-ACO，HVNSOS，NHQEA 在 17 个算例中求得最优解算例的个数分别为 12，13 和 13 个，求最优解成功率分别为 75%，76.47% 和 76.47%，本文算法求得最优解算例个数为 16 个，求最优解成功率高达 94.12%，较 3 种算法提升近 20% 左右。本文未求得的算例 n50-k10，NHQEA 算法同样未求到，且求得为同值 697，差值仅为 1，误差率为 0.14%。在平均误差率上，由于本文算法就一例未求得，故仅为 0.02%，相较于其他算法分别降低 0.51%，0.65%，0.08%，0.02%。本文算法对表中前 7 个算例实验中平均值求得算例最优值，除 n50-k8 算例，其余算例与平均值差值范围均为[1，6]，具有较强准确性。

表 4　CVRP Set P-标准算例测试

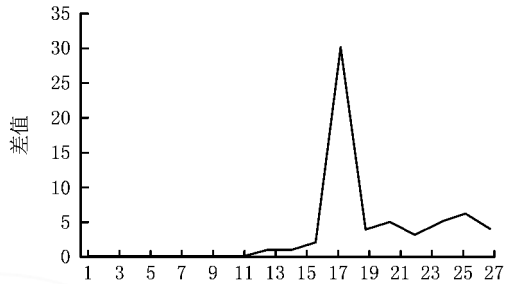
Instance	BK	LNS-ACO[31]		SC-ESA[32]		HVNSOS[29]		NHQEA[30]		HH-RLDQN		
		Min	DEV%	Min	DEV%	Min	DEV%	Min	DEV%	Min	Avg	DEV%
P-n16-k8	450*	450*	0	450*	0	450*	0	450*	0	450*	450*	0
P-n19-k2	212*	212*	0	219	3.30	212*	0	212*	0	212*	212*	0
P-n20-k2	216*	216*	0	218	0.93	216*	0	216*	0	216*	216*	0
P-n21-k2	211*	211*	0	212	0.47	211*	0	211*	0	211*	211*	0
P-n22-k2	216*	216*	0	216*	0	216*	0	216*	0	216*	216*	0
P-n22-k8	603*	～	～	～	～	603*	0	603*	0	603*	603*	0
P-n23-k8	529*	529*	0	529*	0	529*	0	529*	0	529*	529*	0
P-n40-k5	458*	458*	0	459	0.22	458*	0	458*	0	458*	459	0
P-n45-k5	510*	510*	0	511	0.20	510*	0	510*	0	510*	511	0
P-n50-k7	554*	554*	0	554*	0	554*	0	554*	0	554*	556	0
P-n50-k8	631*	643	1.9	637	0.95	632	0.16	631*	0	631*	661	0
P-n50-k10	696*	696*	0	697	0.14	696*	0	697	0.14	697	701	0.14
P-n51-k10	741*	747	0.81	741*	0	744	0.40	741*	0	741*	746	0
P-n55-k7	568*	568*	0	574	1.06	568*	0	568*	0	568*	571	0
P-n55-k10	694*	694*	0	695	0.14	698	0.58	695	0.14	694*	699	0
P-n60-k10	744*	755	1.48	745	0.13	748	0.54	745	0.13	744*	750	0
P-n60-k15	968*	977	0.93	968*	0	968*	0	971	0.31	968*	972	0
Average			0.53		0.67		0.10		0.04			0.02

图 4 所示为本文算法在 Set A/P 标准算例实验中所得平均值与最优值偏差,对两种类型的算例,平均值与所求得的最优值之间相差几乎不大,44 个例子中,只有 2 例偏差在[10,15]之间,1 例为 30(原因为各车容量相对紧凑,造成实验结果一直在最少车辆数为 8 和 9 之间交替出现,且当 9 辆车时最优值甚至求得 629,小于 8 辆车的最优值,而出现 8 辆车的最优值实验结果 742,远大于实验目标最优值),其余全在[0,6]之间,充分反映了本文算法的求解稳定性。

图 5 所示为计算 Set E 客户点数为 50 标准算例,所得的 DQN 神经网络收敛图。纵坐标为网络的均方误差值,横坐标为学习代数。在算法中设定,经验池刷新一定值后,才重新进入学习,因此横坐标代数少于实际算法运行代数,由此导致在算法中,学习代数不多,最后收敛时,均方误差为 0.016 7,未达到稳定状态。且其中变异算子并非绝对优化效果,即使在同样的 *State* 值情况下,依旧可能会出现负优化,造成样本数据偏差,因此图中曲线有起伏,但整体仍呈现下降趋势,体现了较好的学习情况。均方误差值逐渐减小,则 DQN 神经网络部分越接近真实,对底层算子性能的评价及其选择也越准确,提升超启发算法整体的搜索能力,算法的效果从之前的结果表、图中已经得到了体现。



a Set A-标准算例实验最优值与平均值偏差



b Set P-标准算例实验最优值与平均值偏差

图4　Set A/P-标准算例实验最优值与平均值偏差

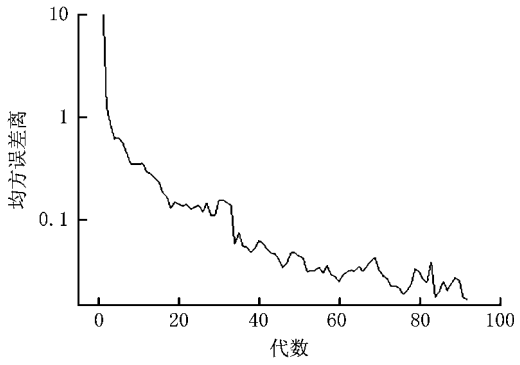


图5 DQN神经网络收敛图

通过上述 3 种标准算例的计算结果，可得本文所设计算法能在 CVRP 上得到了较好的效果，在保证能求得较多最优值的情况下，多次实验的平均值均接近最优值，从而表明算法具有较强的搜索能力和稳定性。通过与其他四种算法的对比表明，实验总体效果优于 LNS-ACO 算法和 SC-ESA 算法，较优于 HVNSOS 算法，与 NHQEA 算法效果相当，但在小客户点算例上，本文算法较 NHQEA 算法具有相对更好的搜索求解效果。

4 结束语

本文提出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。在初始种群的设计中，引入聚类思想，提升初始解的质量。在超启发算法高层策略设计中，引入强化学习中的 DQN 算法，结合 Q 学习算法的思想与神经网络算法的结构，对底层算子在解的每个状态下表现的性能进行评价，评分由立即奖惩和将来奖惩两部分组成，同时利用模拟退火的接受准则以及序列池，对得出的非改进解有条件接受，引导算法跳出局部最优情况，向优质解的空间持续搜索。

利用所提算法对 CVRP 三种标准算例进行实验求解，将求解所得结果与目前最优结果和文中所提 4 种其他算法所得结果相比较，通过最优值、误差率和平均值分析，验证了本文算法在该问题的求解上具有相对准确性和稳定性，总体求解效果优于对比算法。

下一步的工作将尝试采用该算法求解 CVRP 大规模客户点的问题及其衍生问题。同时，虽然本文算法在实验求解中整体获得不错的效果，但在客户规模较大的车辆路径问题上，仍具有较大的改进空间，之后将继续优化其高层策略，争取有所突破。

参考文献：

[1] DANTZIG G B, RAMSER J H. The truck dispatching problem[J]. Management Science, 1959, 6(1): 80-91.

[2] FISCHETTI M, TOTH P, VIGO D. A branch-and-bound algorithm for the capacitated vehicle routing problem on directed graphs[J]. Operations Research. 1994, 42(5): 846-859.

[3] FISHER M L. Optimal solution of vehicle routing problems using minimum K-trees[J]. Operations Research, 1994, 42(4): 626-642.

[4] CHRISTOFIDES N, MINGOZZI A, TOTH P. Space state relaxation procedures for the computation of bounds to routing problems[J]. Networks, 1981, 11(2): 145-164.

[5] CLARKE G, WRIGHT J W. Scheduling of vehicles from a central depot to a number of delivery points[J]. Operations Research, 1964, 12(4): 568-581.

[6] BRAMEL J, SIMCHI-LEVI D. A location based heuristic for general routing problems[J]. Operations Research, 1995, 43(4): 649-660.

[7] OSMAN I H. Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problem[J]. Annals of Operations Research, 1993, 41(4): 421-451.

[8] BULLNHEIMER B, HARTL R F, STRAUSS C. An improved ant system algorithm for the vehicle routing problem [J]. Annals of Operations Research, 1999, 89(1): 319-328.

[9] BARKER B M, AYECHEW M A. A genetic algorithm for the vehicle routing problem[J]. Computers & Operations Research, 2003, 30(5): 787-800.

[10] MARINAKIS Y, IORDANIDOU G, MARINAKI M. Particle swarm optimization for the vehicle routing problem with stochastic demands[J]. Applied Soft Computing, 2013, 13(4): 1693-1704.

[11] ZHANG J, WANG W, ZHAO Y, et al. Multiobjective quantum evolutionary algorithm for the vehicle routing problem with customer satisfaction[J]. Mathematical Problems in Engineering, 2012, 2012(10): 939-955.

[12] MARSHALL R J, JOHNSTON M, ZHANG M. Developing a hyper-heuristic using grammatical evolution and the capacitated vehicle routing problem[C]//Proceedings of Asia-Pacific Conference on Simulated Evolution & Learning, 2014.

[13] COWLING P, KENDALL G, SOUBEIGA E. Hyperheuristics: a tool for rapid prototyping in scheduling and optimisation[J]. Lecture Notes in Computer Science. 2002, 2279: 1-10.

[14] KHEIRI A, KEEDWELL E, GIBSON M J, et al. Sequenceanalysis-based hyper-heuristics for water distribution network optimisation[J]. Procedia Engineering, 2015, 119: 1269-1277.

[15] QIU Junying. A hyper-heuristic using GRASP with Path-Relinking[D]. Dalian: Dalian University of Technology, 2011(in Chinese). [邱俊荧. 基于带 Path-Relinking 的 GRASP 的超启发式方法[D]. 大连：大连理工大学，2011.]

[16] WALKER J D, OCHOA G, GENDREAU M, et al. Vehicle

routing and adaptive iterated local search within the hyflex hyper-heuristic framework[M]. Berlin, Germany: Springer-Verlag,2012:265-276.

[17] SABAR N R, KENDALL G. Population based Monte Carlo tree search hyper-heuristic for combinatorial optimization problems[J]. Information Sciences,2015,314:225-239.

[18] SORIA-ALCARAZ J A, OCHOA G, SOTELO-FIGEROA M A, et al. A methodology for determining an effective subset of heuristics in selection hyper-heuristics[J]. European Journal of Operational Research,2017,260(3):972-983.

[19] DOKEROGLU T, COSAR A. A novel multistart hyper-heuristic algorithm on the grid for the quadratic assignment problem[J]. Engineering Applications of Artificial Intelligence, 2016,52:10-25.

[20] ZAMLI K Z, ALKAZEMI B Y, KENDALL G. A Tabu Search hyper-heuristic strategy for t-way test suite generation[J]. Applied Soft Computing. 2016,44(C):57-74.

[21] CHOONG S S, WONG L P, LIM C P. Automatic design of hyper-heuristic based on reinforcement learning[J]. Information Sciences,2018,436-437:89-107.

[22] ZHAO Yanwei, PENG Dianjun, ZHANG Jingling, et al. Quantum evolutionary algorithm for capacitated vehicle routing problem[J]. Systems Engineering-Theory & Practice, 2009,29(2):159-166(in Chinese).[**赵燕伟,彭典军,张景玲,等.有能力约束车辆路径问题的量子进化算法**[J].**系统工程理论与实践**,2009,29(2):159-166.]

[23] BEASLEY J E. Route first-cluster second methods for vehicle routing[J]. Omega,1983,11(4):403-408.

[24] VOLODYMYR M, KORAY K, DAVID S, et al. Human-level control through deep reinforcement learning[J]. Nature,2015,518(7540):529.

[25] LECUNY, BENGIO Y, HINTON G. Deep learning.[J]. Nature,2015,521(7553):436.

[26] ZHANG Jingling, LIU Jinlong, ZHAO Yanwei, et al. Hyper-heuristic for time-dependent VRP with simultaneous deliveryand pickup[J/OL]. Computer Integrated Manufacturing Systems:1-19[2020-03-12]. http://kns.cnki.net/kcms/detail/11.5946.tp.20190118.1348.042.html.[**张景玲,刘金龙,赵燕伟,等.时间依赖型同时取送货** VRP **及超启发式算法**[J/OL].**计算机集成制造系统**:1-19[2020-03-12]. http://kns.cnki.net/kcms/detail/11.5946.tp.20190118.1348.042.html.]

[27] TOFFOLO T A M, VIDAL T, WAUTERS T. Heuristics for vehicle routing problems: Sequence or set optimization?[J]. Computers & Operations Research,2019,105:118-131.

[28] LENG Longlong, ZHAO Yanwei, ZHANG Chunmiao, et al. Quantum-inspired hyper-heuristics for low-carbon location-routing problem with simultaneous pickup and delivery[J]. Computer Integrated Manufacturing Systems,2018,22(1):1-22(in Chinese).[**冷龙龙,赵燕伟,张春苗,等.求解物流配送同时取送货低碳选址—路径问题的量子超启发式算法**[J].**计算机集成制造系统**,2018,22(1):1-22.]

[29] LI Yang, FAN Houming. Hybrid variable neighborhood symbiotic organisms search for capacitatedvehicle routing problem[J]. Control and Decision,2018,33(7):41-49(in Chinese).[**李　阳,范厚明.求解带容量约束车辆路径问题的混合变邻域生物共栖搜索算法**[J].**控制与决策**,2018,33(7):41-49.]

[30] CHAO Gaoli, HU Rong, QIAN bin, et al. Effective hybrid quantum evolutionary algorithm for capacitated vehicle problem[J]. Computer Integrated Manufacturing Systems,2015, 21(4):1101-1113(in Chinese).[**曹高立,胡　蓉,钱　斌,等.一种有效混合量子进化算法求解带容量约束的车辆路径优化问题**[J].**计算机集成制造系统**.2015,21(4):1101-1113.]

[31] AKPINAR S. Hybrid large neighbourhood search algorithm for capacitated vehicle routing problem[J]. Expert Systems with Applications,2016,61:28-38.

[32] STANOJEVIC M, STANOJEVIC B, VUJOSEVIC M. Enhanced savings calculation and its applications for solving capacitated vehicle routing problem[J]. Applied Mathematics and Computation,2013,219(20):10302-10312.

作者简介:

张景玲(1980—),**女**,**湖北黄冈人**,**副教授**,**博士**,**硕士生导师**,**研究方向**:**人工智能**、**物流优化调度**,E-mail:jlzhang@zjut.edu.cn;

冯勤炳(1994—),**男**,**浙江嘉兴人**,**硕士研究生**,**研究方向**:**物流配送与优化调度**,E-mail:fengqinbing@163.com;

赵燕伟(1959—),**女**,**河南郑州人**,**教授**,**博士生导师**,**研究方向**:**物流配送与优化调度等**,E-mail:ywz@zjut.edu.cn;

刘金龙(1993—),**男**,**安徽阜阳人**,**硕士研究生**,**研究方向**:**物流优化调度**,E-mail:1075815542@qq.com;

冷龙龙(1991—),**男**,**江西宜春人**,**博士研究生**,**研究方向**:**物流配送与优化调度**,E-mail:cyxlll@zjut.edu.cn。