华南理工大学

South China University of Technology

本科毕业设计（论文）

遗传编程辅助的强化学习技术研究

学　　院	计算机科学与工程学院
专　　业	计算机科学与技术
学生姓名	刘丁楠
学生学号	201636599108
指导教师	陈伟能
提交日期	2020 年 5 月 16 日

摘　要

本文将遗传编程算法与强化学习相结合，把强化学习中值函数的拟合问题转化为符号回归问题，通过遗传编程算法优化以二叉树模型为载体的符号表达式，实现了该表达式对状态值函数的拟合与预测，结合传统的决策控制方法，实现了智能体在运动控制问题中的自主学习能力。本文还将对算法在实验中遇到的学习速率缓慢，解的鲁棒性欠佳的问题提出改进方法，运用值函数的双重学习和智能体的集成化提高算法的性能和解的表现。实验部分使用了三个不同难度的控制问题来对算法进行测试，问题包括 Mountain Car，Cart Pole 和分阶段策略学习的 Cart Pole Swing Up，测试的结果初步验证了算法的学习能力和改进方法的有效性。本次设计探索了一个将强化学习与进化计算相结合的新的研究方向，相比于基于深度神经网络的强化学习，本文算法所使用的二叉树模型具有一定的可解释性，能够为算法的拓展提供更多可能；另一方面，二叉树模型的结构可以根据问题的难度自主优化，降低了算法调参的难度。本文还将利用群体算法中个体之间的独立性，实现算法的多线程版本设计，大大提高了算法的运算效率。

关键词：遗传编程算法；强化学习；符号回归；运动控制

Abstract

In this paper, we achieve the combination between genetic programming algorithm and reinforcement learning, and the problem of fitting the state value function of reinforcement learning is transformed into the problem of symbolic regression. The optimization of symbolic regression based on a binary tree model uses genetic programming algorithm to fit and predict the state value function. Combining this and the traditional action control method, the agent has the ability of self-learning in the motion control problem. We also propose improved methods to overcome the problems of slow learning rate and weak robustness we found in the experiments. These methods include dual learning of state value function and the integration of the agents. In the experiment part, we use three different motion control problems such as Mountain Car, Cart Pole and Cart Pole Swing Up to test the algorithm, and the results prove the learning ability of the algorithm and the effectiveness of the improved methods. This paper explores a new cross-research direction between reinforcement learning and evolutionary computation. Comparing with reinforcement learning based on deep neural networks, the binary tree model of the algorithm has the advantages of interpretability and can be extended in many aspects. On the other hand, the structure of the binary tree model can be automatically optimized according to the difficulty of the problem. This reduces the work of parameter adjustment. We also utilize the independence between individuals of the population to achieve the algorithm parallelization which greatly improves the efficiency of the algorithm.

Keywords: Genetic Programming Algorithm; Reinforcement Learning; Symbolic Regression; Motion Control

目　录

第一章 绪论

1.1 研究背景

近些年来，越来越多的科研人员探索智能体的自主学习方法，由于传统控制算法大多需要专家知识，且需要给出针对不同情况的不同策略，其过程十分复杂，需要耗费大量的人力物力，因此科研工作者试图设计算法让智能体能够自主的学习。强化学习是实现智能体自主学习的算法之一，其过程为让智能体在模拟环境中不断的试错，在与环境的交互中提高自身的策略控制，近些年在科技，工业领域得到了较为广泛的应用，例如 DeepMind 团队的围棋 AI AlphaGo[33]击败了世界冠军李世石，OpenAI 团队的 5 人团体竞技类游戏 AI[34]在规定的条件下击败世界冠军 OG，自动驾驶中根据感知信息自主学习路径规划和金融领域的自动化量化投资，强化学习在越来越多的生活场景和工业场景中得到应用，改善了人们的生活，提高了工业的生产效率；但强化学习技术目前也存在着一些问题，例如算法冷启动学习速率缓慢，算法在搜索过程中解存在偏差和较大方差，在稀疏奖励问题中难以学习有效的策略，训练模型需要耗费大量时间与计算资源，这些问题制约着强化学习算法的发展，是强化学习领域研究的重点与难点。

1.2 国内外研究现状

目前，大多数研究人员对于强化学习的探索在于将深度神经网络与传统强化学习理论相结合，在游戏 AI，自动驾驶[16]等具体应用中具有良好的表现，例如深度 Q 网络[2]，基于策略梯度的 DDPG[3]算法，A3C[4]算法；由文献[2]知，深度 Q 网络的思路是将神经网络与 Q 学习相结合，环境状态动作值函数是通过由卷积神经网络与全连接网络共同形成的网络来预测，卷积神经网络对于图像的信息进行了特征选取，实现了特征的自动构造与选择，然后在经过全连接神经网络预测状态动作值，网络需要的误差由 Q 学习得到，除此之外还使用了记忆回放的方法来取消不同状态动作奖励序列之间的关联性，减小训练偏差；文献[3]提到，DDPG 算法是基于策略梯度的算法，不同于基于值函数的算法，动作决策是直接由状态经过神经网络得到的，网络的整体结构使用了表演家-评判家结构，状态值首先经过一个策略网络得到一个动作，动作再同状态一

同输入评判网络中得到动作的一个评判估值，根据累计奖励函数优化评判网络，使评判网络的评判更为准确，再使用评判估值优化策略网络的参数得到更好的策略，整体网络有两个，一个作为训练的网络，一个作为目标网络来加快网络参数的学习；文献[4]中介绍的 A3C 算法采用了异步的思想来收集状态动作值序列更新网络的参数，结合计算机的多线程并行能力，同一时间开启多个训练得到基于一个策略多个训练序列，再结合不同训练序列得到的参数梯度，统一更新到主网络参数，此方法可替代记忆回放方法，实现在策略学习，另一方面也结合了硬件的发展，算法训练的过程因并行化得到加快，实验所需时间更少。

还有科研工作者尝试将强化学习与进化计算的若干算法相结合，利用进化算法的全局搜索能力，弥补神经网络梯度下降方法容易陷入局部最优解的不足，OpenAI 提出了将进化策略方法用于神经网络的参数优化[5]，利用进化策略的全局搜索能力和可并行训练的优势，智能体能够在一些游戏 AI 中取得不错的表现；也有学者将进化算法用于神经网络的结构优化[6][7][8]，通过进化算法自主优化神经网络的拓扑结构，无需人的专家知识构建网络，相比于全连接神经网络，该算法能够在达到相同效果的同时，减小网络大小，起到了网络压缩的作用，另一方面也可能寻找到不依赖于参数设置的网络，大大提高神经网络的鲁棒性。虽然进化算法在数值优化，组合优化上的优势为强化学习中模型的优化提供了其他的可能，但文献[1]中也提到，进化算法的优化往往是一种不需要先验知识的整体优化，在强化学习领域中，智能体与环境之间交互的信息能够为优化提供帮助，如何在进化算法中利用这些信息需要更深入的研究。

遗传编程算法是近些年来有望实现自动编程的算法之一，基于二叉树模型的遗传编程算法中的树形结构与程序编译过程中的抽象语法树十分相似，在已知输入与输出数据的情况下，可以通过优化算法中二叉树的组成与结构生成输入输出关系与已知数据相似的运算表达式，从而实现自动编程。遗传编程算法在解决符号回归问题上有着良好的表现，其生成的二叉树模型与神经网络相比具有一定的解释性和自适应性，具体表现为：二叉树模型由运算符组成且可以转化为符号表达式，可通过数学分析挖掘模型中的隐含信息；二叉树的结构无需专家知识设定，其大小可以根据问题的难度自主调整。得益于遗传编程算法的以上优点，有学者尝试将其与强化学习算法相结合，文献[9]中作者将强化学习中的神经网络模型替换为二叉树模型去拟合强化学习中的状

态值函数，模型的优化是通过遗传编程算法而不是梯度下降法，实验的结果表明，该算法在保证模型拟合准确度的同时大大减小了模型的复杂度，初步验证了将遗传编程算法与强化学习算法相结合的可能性。

1.3 研究目的

本文的研究目的为探索强化学习中状态值函数表示的一种新的方式，基于文献[9]中使用二叉树表达的符号运算式去拟合强化学习中状态值函数的思路，重新设计遗传编程算法辅助的强化学习算法，将神经网络模型替换为二叉树模型，一定程度上提高模型的可解释性；针对该算法在实验中出现的智能体鲁棒性较差和训练过程中累计奖励值易大幅震荡的问题提出了改进方法，增加了集成化智能体和状态值函数双重学习的方法，借助多个个体学到的不同策略来提高集成体策略的鲁棒性，状态值函数的学习不仅通过基因的进化还将利用优秀个体的信息指导学习的方向；完善算法的相关流程之后，实验部分设置了由文献[22-24]中描述的三个难度不同的控制问题来对算法进行测试，问题分为简单的低维状态值低维动作输出，难度中等的较高维状态值多动作输出和难度最大的分阶段策略学习问题，通过基础算法与改进算法实验结果的对比，验证集成化智能体和状态值函数双重学习的有效性；本实验还将结合种群算法中个体互相独立的特点，实现算法的多线程版本，大大降低算法的运行时间，并结合目前已有的可视化测试环境，将实验结果进行直观形象的展示。

1.4 论文结构

本文分为四章，第一章简述了本次设计的研究背景，国内外研究现状和研究目的；第二章介绍了遗传编程算法和强化学习的基础知识；第三章对本文设计的算法和其改进算法进行说明；第四章通过不同的实验问题对算法进行测试，并结合结果对算法进行分析；最后一章总结本次毕业设计的研究内容，并对之后的研究作出展望。

第二章 遗传编程算法与强化学习基础知识

2.1 遗传编程算法

遗传编程算法[10][11][12] [21]是群体智能算法之一，其与传统的遗传算法相比算法流程大致相同，不同之处在于种群个体基因的组成和表现方式，遗传编程算法的基因由运算符节点和变量节点组成且可以表达为一个符号运算式，因为符号表达式可以表示某种映射关系，因此可以用于拟合函数等问题，遗传编程算法同样模拟了自然进化的若干过程，例如遗传，交叉，变异和选择，这些过程体现了达尔文进化论中优胜劣汰的思想，借助群体智慧的不断提高，产生出更为优秀的后代个体。群体智能算法本质上是一类随机算法，具有较强的全局搜索能力，相较于完全随机算法又具有一定启发性，能够加快搜索的过程，因此在组合问题优化，非线性不可微数值优化中有着不错的表现，但也存在参数敏感和算法鲁棒性欠佳的问题，如何提高算法的自适应性和鲁棒性是群体智能算法的研究重点。

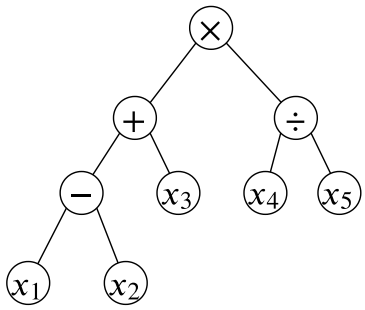

图 2-1 基因的组成和表达方式

2.1.1 基因组成与表达方式

遗传编程算法中，符号表达式需要某种载体来承载，而且还要便于优化其组成和结构，本文使用的是二叉树的方法，关于二叉树方法的具体讨论可以查阅文献[26]。种群中每个个体是一棵二叉树，树的叶子节点由变量节点组成，内部节点由运算符号节点组成，本文使用的运算符均为二元运算符，分别是$\{+,-,\times,\div\}$，变量的数量由问题决定，二叉树所代表的符号运算式可以通过遍历得到，例如图 2-1 这棵二叉树所表达的符号运算式为：$\big((x_1-x_2)+x_3\big)\times(x_4\div x_5)$。

2.1.2 基因交叉与变异操作

本文算法中基因的交叉与变异操作并不复杂，交叉的方式为从种群中随机选出若干个体组成竞争小组，挑出小组中适应度最好的两个个体，再分别从个体的二叉树中随机选择一棵子树，以概率CR_P来决定是否交换这两棵子树。图 2-2 给出交换时的示意图，蓝色和绿色节点为交换的节点：

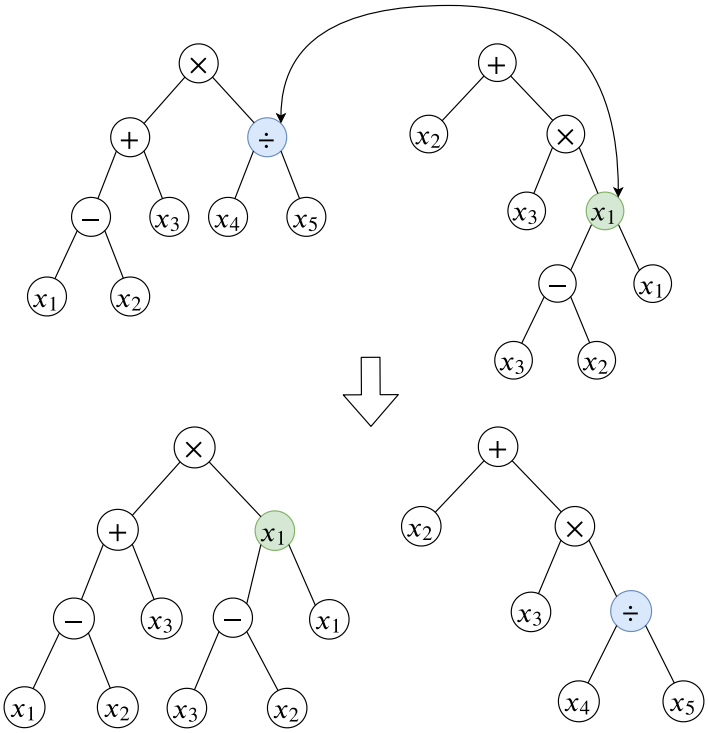

图 2-2 基因的交叉操作

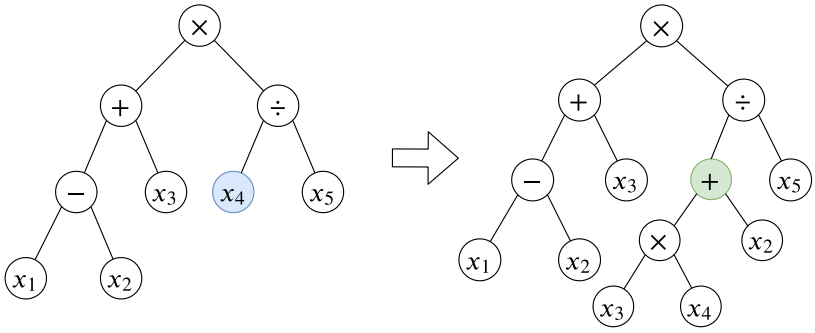

图 2-3 基因的变异操作

经过基因的交叉操作，再对其中的一个个体进行变异操作，同样随机选择出个体二叉树中的一棵子树，以概率M_P决定是否重新构建这颗子树，为了避免因变异操作导致二叉树的复杂度迅速提高，本文算法每次只重新构建一棵高度为 3 的子树，构建

的过程完全随机，图 2-3 展示了将蓝色节点为根的子树重新构造的过程。

2.2 强化学习算法

强化学习算法[1][31][32]是一类让智能体通过与环境的交互，自主学习动作策略的算法，其突出三大要素：状态，动作与回报。通常来说，在强化学习的过程中，智能体首先根据所处环境信息和策略产生一个动作，执行动作后，智能体会转移到一个新的状态并得到一个奖励信息，整个过程如图 2-4 所示。智能体的目标为最大化累计奖励，在与环境的不断交互中优化自己的策略，从而在动作选择中选择未来奖励更高的动作。

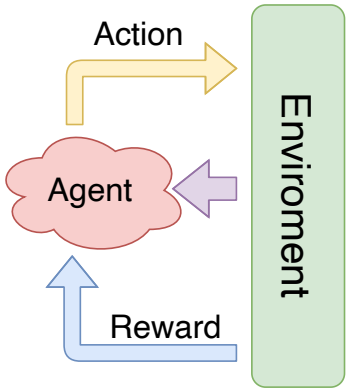

图 2-4 强化学习流程

2.2.1 强化学习基本概念

智能体的行为策略由环境与智能体之间的交互决定，首先我们定义对环境的观测值为智能体所处的状态$S(State)$，智能体根据状态S可以根据预设或自主学习到的策略π作出动作$A(Action)$，动作A结束之后，智能体会有概率$P(S'|S,A)$来到一个新的环境状态S'，并伴随奖励$R(Reward)$，智能体不断的重复上述过程，将得到一个状态动作奖励序列，即$S_0, A_0, R_0, S_1, A_1, R_1, S_2, \dots, S_t$，$S_t$为终止状态，我们将这样的过程成为马尔可夫决策过程(Markov Decision Processes)。初始状态的智能体的动作决策是随机的，没有偏好的，为了能够提高智能体在环境中的表现，智能体的决策要能够在与环境的交互中不断学习，智能体得到一个马尔可夫决策过程即为一个训练时期(Episode)，训练的奖励值G的表达式如公式(2-1)，G_t为一个训练时期中第t个状态S_t获得的累计奖励，S_T指最后结束的状态且$G_T = 0$，γ是折扣因子，用来调整之后状态的奖

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots + \gamma^{T-t-1} R_T = R_{t+1} + \gamma G_{t+1} \tag{2-1}$$

励在当前状态累计奖励值中的比重，经过多个训练时期，我们可以得到针对某个动作策略π下状态s的期望累计奖励$V_\pi(s)$，当训练次数为无穷多时，期望累计奖励近似等于真实累计奖励，即：

$$V_\pi(s) = E_\pi[G_t | S_t = s] \tag{2-2}$$

我们称$V_\pi(s)$为状态s的值函数，在我们得到每个状态的值函数后，智能体便可以在每次状态转移中选择能够到达值函数最大的状态的动作a，即$argmax_a V_\pi(s'|s,a)$，从而得到最优的动作策略π^*。

强化学习中，智能体所面临的环境模型分为已知与未知，若模型已知，我们可以通过动态规划算法来优化智能体的行为策略；而对于模型未知的情况，我们需要通过蒙特卡洛方法去拟合并预测环境信息，即智能体在环境中的状态值，因为我们并不知道环境的信息，状态转移之间的信息也未知，所以我们需要将状态值函数拓展状态动作值函数$Q(S,A)$，其表达式为公式(2-3)；未知环境信息情况下的智能体学习存在一个

$$Q(S,A) = E[G(S'|S,A) + R_A] \tag{2-3}$$

平衡问题，即对于环境信息的探索与利用，也就是智能体状态动作值函数的更新和基于状态动作值函数的策略更新，对于这个问题，经典且有效的方式为ε贪心方法，即公式(2-4)，以ε的可能性选择最优的动作，以$1-\varepsilon$的可能性随机选择动作。

$$Action = \begin{cases} argmax_a Q_\pi(s'|s,a), & random(0,1) < \varepsilon \\ a = random(A), & else \end{cases} \tag{2-4}$$

2.2.2 Q 学习算法

结合马尔可夫决策过程和蒙特卡洛方法更新智能体的状态动作值函数需要获取一个完整的训练时期，若训练时期时间过长，则智能体的值函数更新十分缓慢，为了提高智能体的学习效率，我们可以改变值函数更新的方式，在每一次状态转移的同时更新值函数，如公式(2-4)，

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha[R_{t+1} + \gamma Q(S_{t+1}|A_{t+1}) - Q(S_t, A_t)] \tag{2-5}$$

A_t与A_{t+1}是由同一个策略π产生，此方法成为时序差分学习。为了能够更好地利用不同策略所产生的动作状态信息，我们可以将$R_{t+1} + \gamma Q(S_{t+1}|A_{t+1})$更改为$R_{t+1} + \gamma \max_a Q(S_{t+1}|a)$，我们称其为离线策略，得到公式(2-5)

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha \left[R_{t+1} + \gamma \max_a Q(S_{t+1}|a) - Q(S_t, A_t) \right] \quad (2\text{-}6)$$

这样，便得到了强化学习中的 Q 学习算法，Q 学习算法的优点在于将原本需要获取完整训练时期的信息之后才能更新状态动作值函数的过程改进为在每一次状态转移的同时更新，是一种增量式更新，并在更新的过程中，获取不同策略所产生的状态动作值函数，提高算法的学习效率。

2.3 本章小结

本章简单介绍了遗传编程算法与强化学习的基础知识，展示了遗传编程算法中基因组成和基因表达方式，基因的交叉过程和基因的编程过程；同时，介绍了强化学习的基础知识，如状态值函数，状态动作值函数，马尔可夫决策过程，蒙特卡洛方法，平衡智能体探索与利用的ε贪婪策略，时序差分学习和 Q 学习。

第三章 遗传编程算法辅助的强化学习算法设计

3.1 算法简介

本文尝试将遗传编程算法与强化学习相结合，用二叉树模型拟合强化学习中的状态值函数，再通过遗传编程算法优化二叉树的组成来提高模型的准确度。在得到状态值函数的拟合模型后，即可为动作选择提供依据，实现智能体的自主控制。相比于神经网络模型，二叉树模型具有一定的解释性，因为二叉树模型由运算符和变量节点组成，通过遍历二叉树可得到一个运算表达式，这时便可使用数学方法对表达式进行分析来获得模型的更多信息；另一方面，遗传编程算法所需参数较少，且不需要专家知识设计二叉树的结构，二叉树的组成与结构优化由算法自动完成，因此可以根据问题难度自动生成所需的二叉树模型，算法过程更加智能。

本章将基于上述思想介绍遗传编程算法辅助强化学习的基础算法，在实验过程中发现基础算法存在解的鲁棒性较差和策略学习过程易大幅震荡的问题，针对这些问题本章还提出了改进方法，分别为集成化智能体方法和值函数的双重学习方法，集成化的思想在文献[19][20]有所提及，值函数双重学习的思路可以查阅文献[18]；除此之外，借助群体算法个体之间的独立性，实现了基础算法与改进算法的多线程版本，大大缩短了实验所需的时间。

算法的整体流程大致为：首先对种群进行初始化操作，然后模拟种群的进化过程，进化中的每一代群体首先要经过问题的测试得到个体的累计奖励值，然后更新种群的若干信息和集成化智能体，再根据种群相较于上一代种群的变化和最优个体与集成体中最差的个体的优劣，得到个体适应度评判准则，依据该准则对种群中的个体进行选择，交叉，变异操作，从而得到下一代的群体，算法的伪代码如表(3-1)所示，代码的全部内容开源在 GitHub 仓库 https://github.com/LDNN97/GP-RL。

3.2 状态值函数的预测

本文提出的算法使用了二叉树模型去表示强化学习中的值函数模型，通过符号回归尝试找到与环境函数相同或成比例的函数模型，假设环境状态值是一个n维数组$S = (x_1, x_2, \dots, x_n)$，将数组的值代入一个运算式中即可得到对应的输出，对应二叉树表达

的运算式，将数组每一维的数值代入二叉树对应变量节点，对树进行遍历的同时计算二叉树对应运算式的输出，因为本文使用的运算符均为二元运算符，只要保证后序遍历，在得到一个节点左右子树的值就可以结合节点的运算得到结果，遍历回到二叉树的根节点时，便得到了状态值函数$V(S)$。

遗传编程算法优化二叉树组成的过程即是符号回归过程，群体智能算法优化的过程中，种群中的每个个体单独表示一棵二叉树，对每个个体进行评测得到适应度，再根据适应度挑选出优秀的个体进行交叉变异，在种群的进化中逐渐提高个体符号回归的准确度，从而产生出与状态值函数模型相同或相似的模型，以此为基础进行强化学习中动作决策。

3.3 智能体的动作决策

智能体动作决策本质上是对状态转移的一种选择，假设智能体t时间所处状态为S_t，可执行动作集合为$A = \{a_1, a_2, \dots, a_m\}$，执行动作后状态为$S_{t+1}$， 动作奖赏为$R_{(S_{t+1}|S_t, a)}$，则对应状态值函数关系式为公式(3-1)，由于遗传编程算法对于二叉树模型的优化并不需要具体的连续状态转移的值函数信息，所以此处并不需要折扣因子γ。根据式(3-1)，我们可以得到每种状态转移后状态值的大小关系，进而选择能够带来更

$$V(S_t) = R_{(S_{t+1}|S_t, a)} + V(S_{t+1}|S_t, a) \tag{3-1}$$

多奖励的动作，因此智能体的决策控制只需在动作集合中选择使当前状态值最大的动作，即公式(3-2)。

$$a^* = argmax_a \left\{ R_{(S_{t+1}|S_t, a)} + V(S_{t+1}|S_t, a) \right\} \tag{3-2}$$

有了如上所述的动作决策，智能体便可以完成状态转移时的选择，将多个状态转移连接在一起，就实现了智能体在环境交互中的自主控制。

3.4 种群个体适应值

在用遗传编程算法对种群个体二叉树进行优化时，需要对每个体进行测试得到适应度值，在传统的符号回归问题中，适应度值是符号运算式结果与真实结果之间的误差，算法整体属于监督学习，在强化学习中，智能体的学习并不是监督学习，无法得

到每个状态的真实值，因此无法用误差来衡量个体的优劣，但在智能体与环境交互的过程中，智能体会得到奖励值，可以用此奖励值间接的衡量个体的好坏，因此将智能体与环境交互得到的累计奖励值作为个体的适应度值 (Fitness)。这样的方式解决了群体算法中对个体进行选择时的基准问题，但也带来了一些问题，由于强化学习中，智能体得到的累计奖励值对于智能体的策略是十分敏感的，在算法学习的初期，因为没有找到合适的策略，智能体的累计奖励值大多较差，因此其无法对学习起到明显的指导作用，算法学习效果容易呈现较大震荡，影响智能体的学习速率。

3.5 集成化智能体

实验过程中发现，即使适应度表现良好的个体，在不同环境下的表现也可能差异较大，分析其原因发现，个体二叉树表示的符号运算式对于整个状态空间的拟合可能是局部良好的，当实验环境状态位于这部分之外，智能体的表现就无法预测，因此个体对于整体环境的预测能力不足，其表示的智能体鲁棒性较差。为了解决这个问题，很自然的想法是可否将多个个体集成在一起，得益于群体智能算法中个体之间的独立性，集成个体的实现比较容易，本文的方法为记录不同代际间表现良好的个体，使用大根堆维护一个大小固定的集合，根据个体的适应度值动态调整集合中所包含的个体，算法运行结束后，便得到了一个集成化智能体。集成化智能体的想法运用了种群的群体智慧，提高了智能体对于完整状态空间的拟合能力，提高了智能体在不同环境中的鲁棒性；集成化智能体的动作决策在本文中采用的是投票的方法，统计个体集合中不同个体对于同一状态的动作选择，选取票数最高的动作，完成智能体的动作决策。

3.6 值函数的双重学习

基础算法实验中发现虽然进化过程中可以得到适应度高的个体，但种群整体的表现并没有因为优秀个体的发现显著变好，而是呈现较大的震荡，经过分析，我们发现完全依赖累计奖励值作为个体的适应度来更新种群并不能掺选出潜在的优秀个体，因为智能体对于策略是十分敏感的，策略上细小的差异便会导致累计奖励值差异巨大，在评估次数有限的情况下，无法得到个体准确的奖励值估计；另一方面，由式(3-1)发

现，因为遗传编程算法优化的二叉树代表的是完整的状态值空间，因此$V(S_{t+1}|S_t,a)$在状态转移中是准确度未知的值，进而无法保证$V(S_t)$的准确性。以上的两个原因会造成种群整体适应度震荡，增长缓慢的问题。

基于以上问题原因的分析，我们提出个体值函数的双重学习，在种群进化的过程中，将适应度优秀的个体挑出，组成集成体，并将其设为种群进化中的目标个体，在行为决策时，对同一环境，对比目标个体与当前个体之间的动作决策，并计算行为动作决策的相似度，作为累计奖励值之外的第二评判标准，如公式(3-3)—(3-5)所示。

$$a_t = argmax_a\left\{R_{(S_{t+1}|S_t,a)} + V(S_{t+1}|S_t,a)\right\} \tag{3-3}$$

$$a_t^* = argmax_a\left\{R_{(S_{t+1}|S_t,a)} + V^*(S_{t+1}|S_t,a)\right\} \tag{3-4}$$

$$similarity = \frac{\sum_{t=0}^{a_t \neq a_t^*} cmp(a_t, a_t^*)}{T}, cmp(a,b) = \begin{cases} 1, & a = b \\ 0, & a \neq b \end{cases} \tag{3-5}$$

在得到个体与目标个体之间的相似度后，评判个体优劣便存在两个标准：与环境交互得到的累计奖励值和与目标个体的相似度，如何平衡适应度中两个标准的权重十分重要，经过大量实验发现，如果过于依赖相似度，则会大大减小种群的多样性，使种群进化陷入停顿，所以本文采用的方法是根据种群适应度的平均值变化和种群最优个体与集成体中累计奖励最小的个体的大小关系动态调整累计奖励值与个体相似度权重值，具体方法如下(3-6)。

$$Rate_{reward} = \begin{cases} 1, & ACR \geq ACR_{lg} \\ 0.7, & ACR < ACR_{lg}, BCR > LCR_{agent} \\ 0.5, & else \end{cases} \tag{3-6}$$

其中ACR表示平均累计奖励，下标lg表示上一代，BCR表示种群得到的最高的累计奖励值，LCR_{agent}表示集成体中的最小累计奖励值，相似度的权重为$(1 - Rate_{reward})$。

3.7 本章小结

本章介绍了遗传编程算法辅助的强化学习算法的基础算法和改进算法，对算法流程与算法细节进行了详细的说明，介绍了基础算法中状态值函数预测的方法，智能体的动作决策和种群个体适应值的评估方法；之后根据基础算法实验所遇到的问题提出了两个改进方法：集成化智能体方法和值函数的双重学习方法，提高了智能体的鲁棒性和算法的学习效率。

表 3-1 遗传编程算法辅助的强化学习算法

算法 1:遗传编程算法辅助的强化学习

初始化：环境 ENV，种群 POP，集成化智能体 Agent，平均累计奖励值 R_A，
上代平均累计奖励值 L_A，最优奖励值个体 I_B，平均相似度 SIM_A
累计奖励值比例 R_R，相似度比例 S_R

过程：

```
1.FOR GEN = 1 TO MAX_GENERATION DO
2.    初始化环境 ENV，初始化种群累计奖励 Re，种群相似度 Sim
3.    智能体累计奖励值最小个体值 AF_Min
4.    PARALLEL FOR I = 1 TO POP_SIZE DO
5.        评估个体 I，得到对应的 Re[I]和 Sim[I]
6.    END
7.    更新 R_A，I_B
8.    更新集成化智能体(Agent, I_B)
9.    # 选择种群排序准则
10.   IF R_A > L_A AND SIM_A > 0.5 DO
11.       R_R = 1; S_R= 0;
12.   ELSE
13.       IF Re[I_B] >= AF_Min DO
14.           R_R = 0.7; S_R= 0.3;
15.       ELSE
16.           R_R = 0.5; S_R= 0.5;
17.       END
18.   END
19.   L_A = R_A
20.   Rank (POP, Re, Sim, R_R, S_R)
21.   FOR I = 1 TO POP_SIZE DO
22.       P1 = TS(POP) # Tournament Selection
23.       P2 = TS(POP) # Tournament Selection
24.       Crossover (P1, P2)
25.       Mutation (P1)
26.       NEW_POP[I] = P1
27.   END
28.   POP = NEW_POP
29. END
```

输出：集成化智能体 Agent

第四章 遗传编程算法辅助的强化学习实验与结果分析

4.1 算法实验设置和对比方法

基础算法与改进算法的参数设置如下表(4-1)，基础算法相比于改进算法缺少集成化智能体与值函数的双重学习，除此之外其他部分参数设置相同。

表 4-1 算法参数设置

POP-SIZE	INI-DEPTH	CR-P	MU-P	T-SIZE	ENS-SIZE	F-NODE
30	5	0.8	0.2	6	6	+,-,×,÷

POP-SIZE 指种群大小，INI-DEPTH 指初始个体子树大小，CR-P 指交叉概率，MU-P 指变异概率，T-SIZE 指竞争组大小，ENS-SIZE 指集成体中个体数目，F-NODE 是二叉树中间节点的运算符类型，树的变量节点数目为实验测试问题输入维度。

算法实验结果的对比分为以下三项：

1. 对比智能体累计奖励值的增长曲线，为了提高方法的可靠性，分别对比每代集成智能体中最大累计奖励值和最小累计奖励值的增长曲线，曲线增长的速度越快，代表算法的学习速率越快。
2. 对比基础算法在整个学习的过程中，种群中最优个体与集成化智能体获得的累计奖励值的分布，若集成化智能体在测试问题中所获得高累计奖励值的次数高于种群最优个体，证明集成化智能体方法有效。
3. 对比算法采用值函数双重学习前后，种群全部个体在测试问题中获得的累计奖励值的平均值的分布情况，若采用值函数双重学习方法获得的高平均累计奖励值次数越多，证明值函数双重学习方法有效。

为了降低随机算法测试结果的偶然性，算法在不同随机种子的情况下运行 5 次。第一个对比方法为 5 次测试中每一代集成体中最大累计奖励值的均值和最小累计奖励值的均值；第二个对比方法为算法在 5 次测试中，基础算法种群最优个体与集成化个体获得的累计奖励值的分布情况；第三个对比方法为 5 次测试中，采用值函数双重学习方法与不采用此方法所获得的平均累计奖励值分布情况。

4.2 Mountain Car

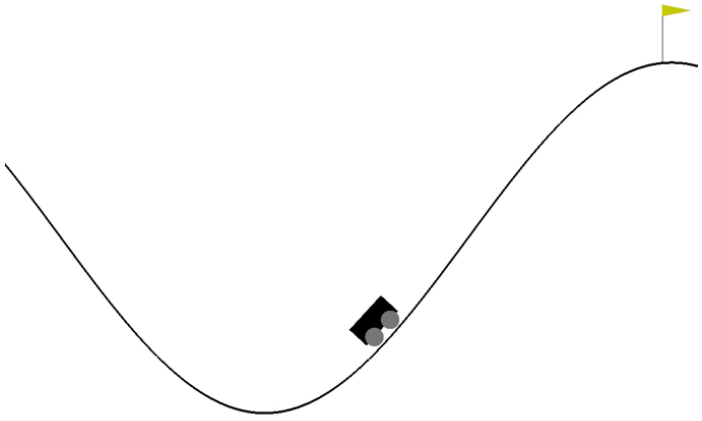

图 4-1 Mountain Car 实验

4.2.1 实验问题简介

实验部分第一个实验为 Mountain Car 实验，关于该实验的详细信息可以查阅文献[22]，实验截图如图 4-1，智能体在环境中的交互过程为智能体控制的小车在山坡的随机位置生成，小车的动力不足以冲上山坡，需要在两个山坡之间来回移动积累能量，智能体能够得到的关于小车的状态信息为小车的速度v和位置x，能够控制的动作为向左或向右提供一个固定的动力F，动力的大小不足以将小车开上山坡，整个环境状态函数为公式(4-1)-(4-3)。

$$a_t = [-1,0,1] \tag{4-1}$$

$$v_{t+1} = v_t + a_t \times 0.001 + \cos(3x_t) \times (-0.0025) \tag{4-2}$$

$$x_{t+1} = x_t + v_t \tag{4-3}$$

初始速度和位置为$v_0 \in (-0.07, 0.07), x_0 \in (-1.2,0.6)$，可选择动作为向左，不动或向右，每一次动作的奖励值在未到达终点之前均为-1，终点为$x_T > 0.6$。

4.2.2 实验结果及分析

实验结果对比第一项为集成化智能体中最小最大累计奖励值变化曲线，如下图 4-2 所示，首先可以看到，遗传编程算法辅助的强化学习在 Mountain Car 上的表现是优秀

的，最优个体在第 25 代左右即可学到将小车开上山丘的策略，且整个过程中，策略的提高是迅速的，集成体中表现最差的个体的学习速率与最优个体相似，同样不足 50 代便可学习到稳定可行的策略，集成化智能体中不同个体的二叉树组成不一样，表明算法在整个过程中学习到多个可行解，一定程度上展现了算法的学习能力；对比基础算法与改进算法，在此问题中并没有较大差异，可能是因为问题过于简单，值函数双重学习的过程与基础算法相似，结果差异较小。

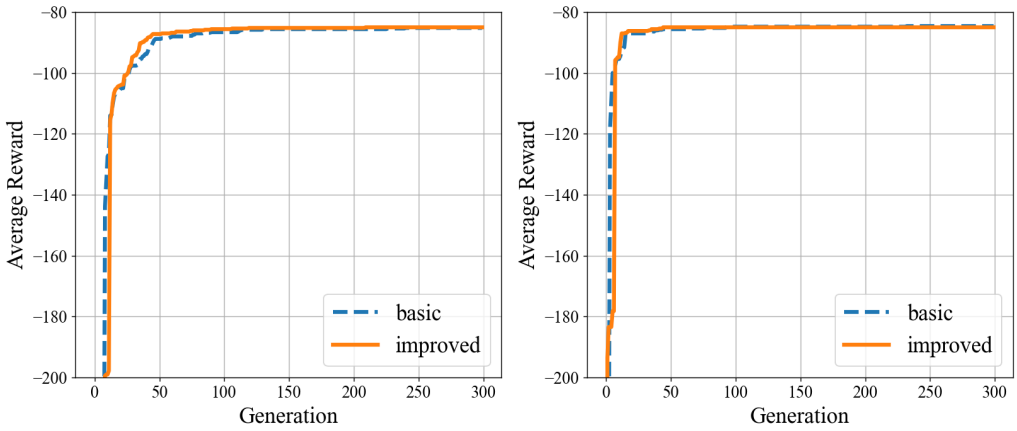

图 4-2 最小（左）和最大累计奖励值变化曲线

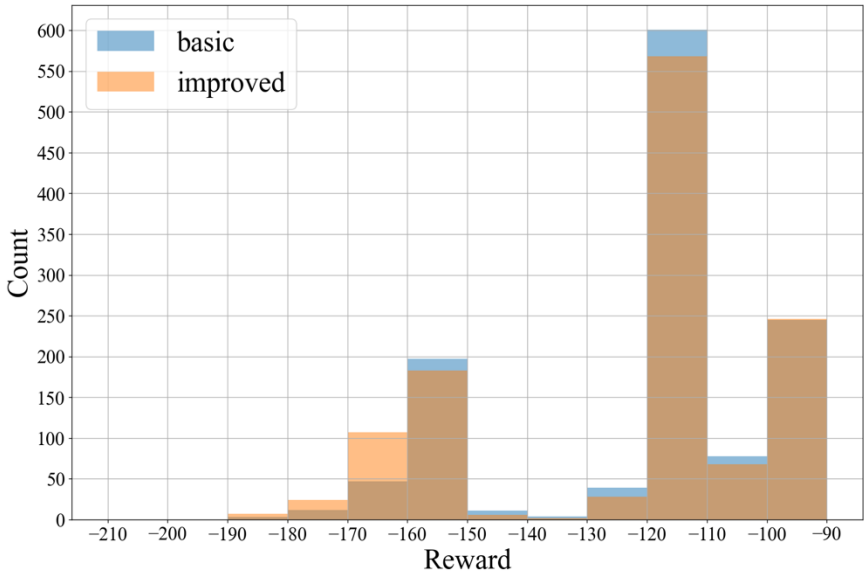

图 4-3 种群最优个体与集成体累计奖励值分布情况

实验结果对比的第二项为种群最优个体与集成体在全部测试中所获得累计奖励值

的分布情况，如图 4-3 所示，由表可以看出在高奖励值处，集成体与最优个体并无较大差异，在较低奖励值分布上，集成体获得的次数多于种群最优个体，以上结果表明，集成化智能体在保证与种群最优个体奖励值相似的情况下，提高了智能体学习初期所得到的累计奖励值，体现出集成化智能体在种群个体未学习到有效策略时，能够结合多个智能体学习到的策略，提高了智能体的表现。

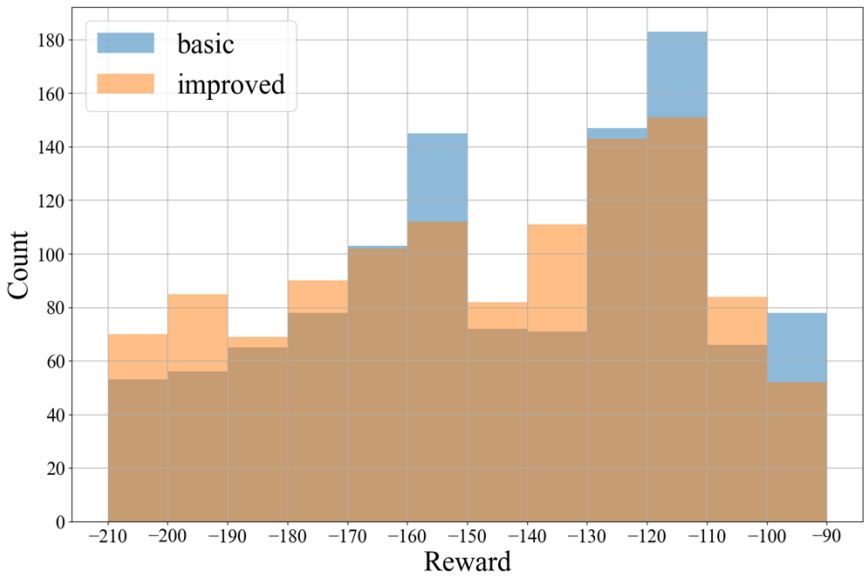

图 4-4 基础算法与改进算法平均累计奖励值的分布情况

实验结果对比第三项为基础算法与改进算法种群平均累计奖励值的分布情况，如图 4-4 所示，改进算法在大多数平均累计奖励值区间上所获的次数高于基础算法，一定程度上体现出值函数双重学习对种群整体值函数的学习的加速作用，如果在学习的过程中发现表现较好的个体，改进算法能够利用这些较好个体的策略信息引导种群整体的学习，从而提高整体的表现，虽然在部分奖励值区间上改进算法获得的次数低于基础算法，但从整体上看，还是可以验证改进方法的有效性的。

4.3 Cart Pole

4.3.1 实验问题简介

实验部分的第二个实验为 Cart Pole 实验，具体信息可以查阅文献[23] ，实验截图如图 4-5 所示，相比于第一个实验，该实验状态信息更多，状态值由 4 维变量

$(x, v_x, \theta, v_\theta)$ 组成，分别表示小车的位置，速度，摆杆与竖直向上之间的夹角和角速度，智能体的动作空间为$a \in (-1,1)$，表示向左或向右移动小车；实验中智能体与环境的交互过程为：摆杆初始化在一个与竖直方向夹角小于 60 度的位置，智能体通过尝试移动摆杆下方的小车让摆杆保持平衡，避免落下；物理模型的具体表达式为公式(4-4)-(4-7)所示：

$$a_\theta = \frac{gsin\theta + cos\theta\left[\frac{-F - mlv_\theta^2 sin\theta + \mu_c sgn(v_x)}{m_c + m}\right]}{l\left[\frac{4}{3} - \frac{mcos^2\theta}{m_c + m}\right]} \tag{4-4}$$

$$a_x = \frac{F + ml[v_\theta^2 sin\theta - a_\theta cos\theta]}{m_c + m} \tag{4-5}$$

$$v_{x_t} = v_{x_{t-1}} + \Delta t \times a_x, \qquad x_{t+1} = x_t + \Delta t \times v_{x_t} \tag{4-6}$$

$$v_{\theta_t} = v_{\theta_{t-1}} + \Delta t \times a_\theta, \qquad \theta_{t+1} = \theta_t + \Delta t \times v_{\theta_t} \tag{4-7}$$

上式中$g = -9.8\, m/s^2$, $m_c = 1.0kg, m = 0.1kg, l = 0.5m, F = \pm 10N$，状态转移时间长度$\Delta t = 0.02s$，状态终点为$x > 2.4 \ or \ x < -2.4, \theta > 60°$。由于此实验相比于第一个实验来说，状态维数大了一倍，因此拟合状态值模型的难度更大。

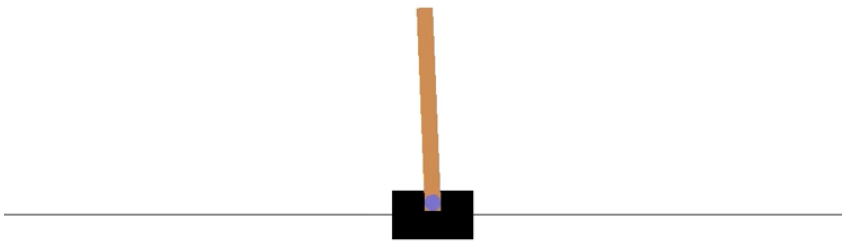

图 4-5 Cart Pole 实验

4.3.2 实验结果及分析

实验结果对比第一项为集成化智能体中最小最大累计奖励值变化曲线，如下图 4-6 所示，首先可以看到，遗传编程算法辅助的强化学习在 Cart Pole 实验的表现同样是十分优秀的，集成体中表现最优的个体在第 25 代左右即学习到维持杆子保持直立的策略，且累计奖励值在训练初期迅速提高，集成体中表现最差的个体的学习速率与最优

个体相似，所需代数增加到 50 代左右，同样十分优秀；对比基础算法与改进算法，改进算法稍稍晚于基础算法学习到累计奖励值为 500 的策略，但最小累计奖励值曲线在接近 50 代左右时改进算法是快于基础算法的，虽然改进算法发现最优策略的时间稍微晚了一些，但集成体中最差的个体也是好于基础算法集成体中最差的个体的，从一定程度上表明改进算法比基础算法容易学到多个较好的解；另一方面，小车立杆问题并不复杂，改进算法与基础算法学习性能整体上并未体现出明显差异。

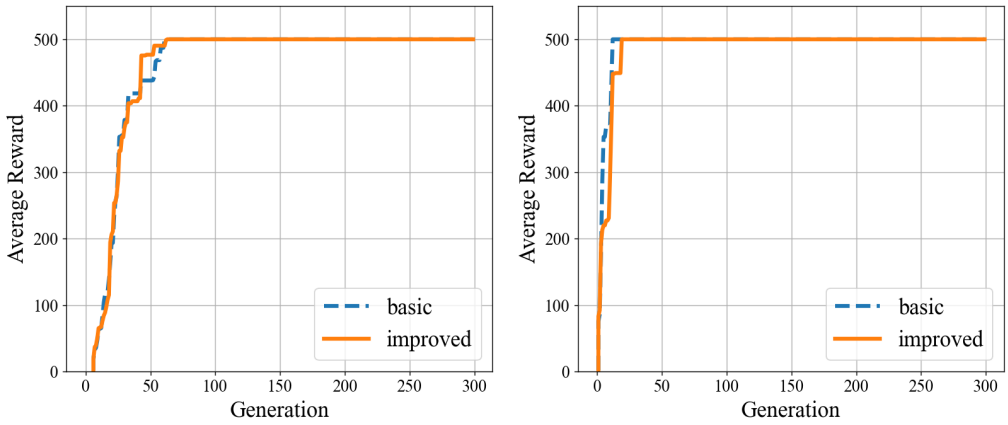

图 4-6 最小（左）和最大累计奖励值变化曲线

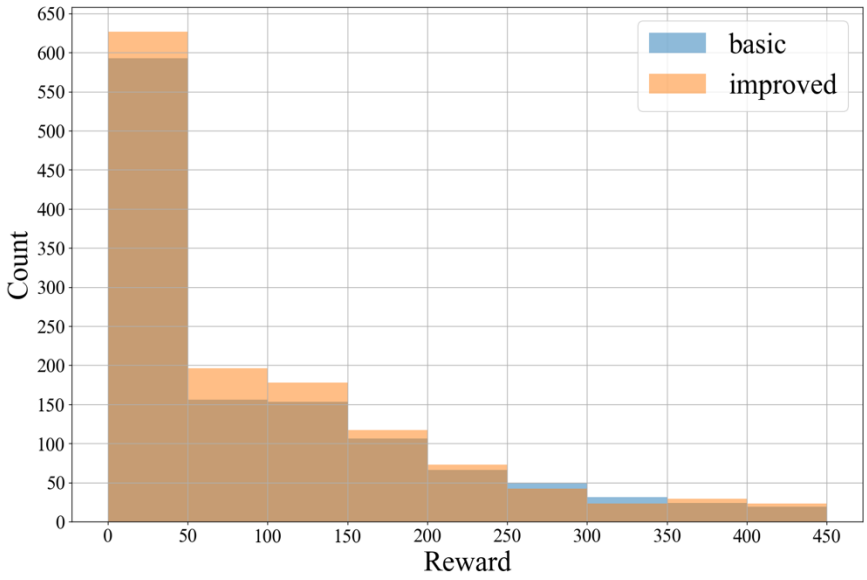

图 4-7 种群最优个体与集成体累计奖励值分布情况

实验结果对比的第二项为种群最优个体与集成体在全部测试中所获得累计奖励值的分布情况，如图 4-7 所示，由图可以看出，在大多数累计奖励值分布区间上，改进

算法获得的次数高于基础算法，在 250 至 350 区间中，改进算法获得的次数略微低于基础算法，结果表明集成化智能体方法能够提高其获得的累计奖励值，多个个体共同组成集成体时能够结合每个个体所获得的策略知识来提高集成体的策略表现，增强了智能体的鲁棒性。

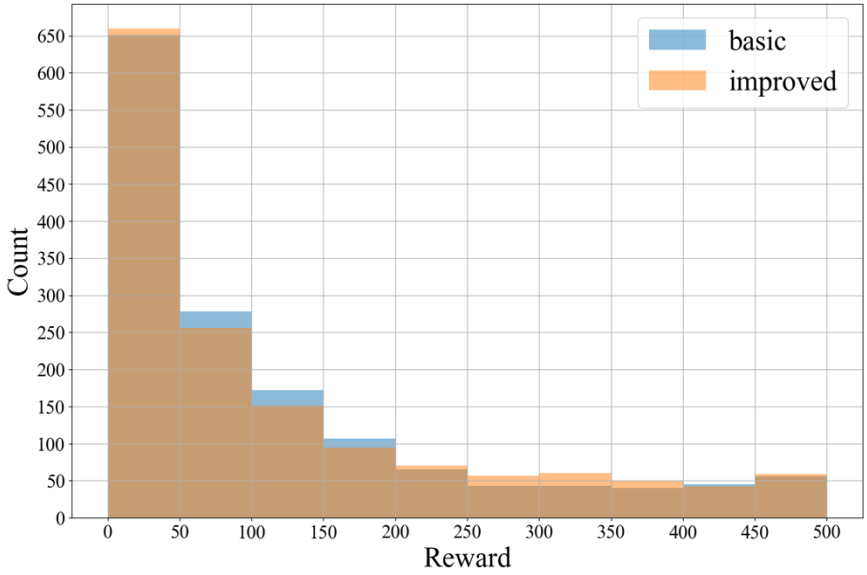

图 4-8 基础算法与改进算法平均累计奖励值的分布情况

实验结果对比第三项为基础算法与改进算法种群平均累计奖励值的分布情况，如图 4-8 所示，由图可以看出，在 200 至 400 和 450 至 500 的区间上改进算法获得的高平均累计奖励值次数高于基础算法，一定程度上体现出值函数双重学习方法对于种群整体表现的提高作用，虽然在 50 至 200 和 400 至 450 区间上，改进算法获得的累计奖励值次数略微低于改进算法，但在 200 之后，改进算法的表现在大多数累计奖励值区间上好于基础算法，表明改进算法在学习到更好策略之后，能够引导种群整体的学习，让种群整体学习到更好的策略，提高智能体在实验问题中的表现。

4.4 Cart Pole Swing Up

4.4.1 实验问题简介

实验部分的第三个实验为 Cart Pole Swing Up 实验，相较于第二个实验，该实验增加了状态信息和动作空间的维度，策略学习分为多个阶段，因此难度大大高于第二个

实验，关于实验的具体信息可以查阅文献[24]， 实验截图如图 4-9 所示。Cart Pole Swing Up 在 Cart Pole 实验的基础上，将摆杆的初始位置改为小车的下面，动作空间由原先的向左向右更改为 11 个不同的动作，分别表示向左或向右不同的推力或不给力，虽然看似环境并没有什么大的改变，但对于智能体的学习是完全不一样的难度。首先由于角度拓展为 360 度，状态空间约是之前的 3 倍，其次摆杆在小车上下需要学习不同的策略，首先智能体要找到将摆杆甩起的方法，其次将摆杆甩起后要学习与之前完全不同的维持摆杆不落下的方法，因此学习的策略分为两个阶段，难度大大提高；又由于动作空间由原先的 2 个动作增加为 11 个，智能体对于环境的感知与决策需要更加精细的控制。环境的物理模型如下公式(4-8)-(4-11)所示：

$$a_x = \frac{-2m_p l v_\theta^2 sin\theta + 3m_p g sin\theta cos\theta + 4F - 4bv_x}{4(m_c + m_p) - 3m_p \cos^2\theta} \tag{4-8}$$

$$a_\theta = \frac{-3m_p l v_\theta^2 \mathrm{sin\theta cos\theta} + 6(\mathrm{m_c} + m_p) g sin\theta + 6(F - bv_x) cos\theta}{4l(m_c + m_p) - 3m_p l c^2} \tag{4-9}$$

$$v_{x_t} = v_{x_{t-1}} + \Delta t \times a_x, \qquad x_{t+1} = x_t + \Delta t \times v_{x_t} \tag{4-10}$$

$$v_{\theta_t} = v_{\theta_{t-1}} + \Delta t \times a_\theta, \qquad \theta_{t+1} = \theta_t + \Delta t \times v_{\theta_t} \tag{4-11}$$

上式中$l = 0.6m, m_c = 0.5kg, m_p = 0.5kg, g = 9.82\frac{m}{s^2}, b = 0.1N/m$，每一次状态转移时间长度为$\Delta t = 0.01s$，状态初始值满足$S_0 \sim \mathcal{N}(\mu, \Sigma), \mu = \delta([0, \pi, 0, 0]^T), \Sigma^{\frac{1}{2}} = diag([0.2m, 0.2rad, 0.2m/s, 0.2rad/s])$，状态终点为$x < -2.4 \; or \; x > 2.4$。

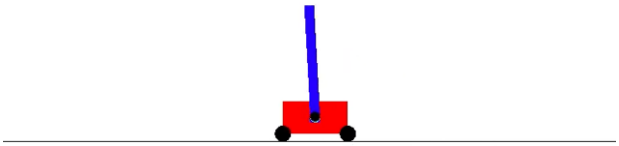

图 4-9 Cart Pole Swing Up 实验

4.4.2 实验结果及分析

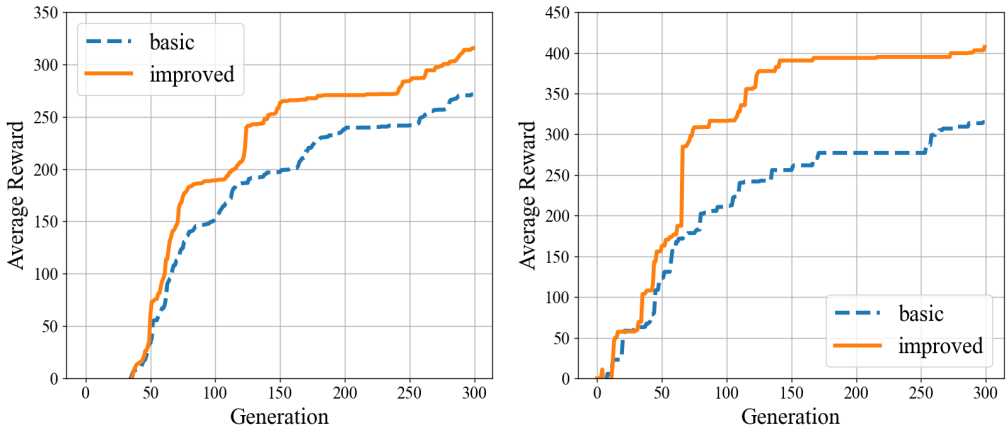

图 4-10 最小（左）和最大累计奖励值变化曲线

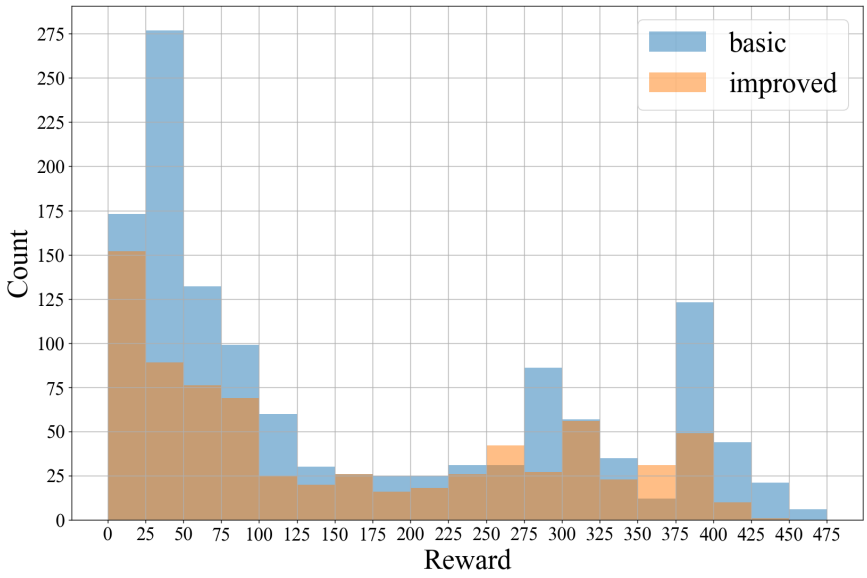

图 4-11 种群最优个体与集成体累计奖励值分布情况

实验结果对比第一项为集成化智能体中最小最大累计奖励值变化曲线，如图 4-10 所示，首先可以看到，相比于上面两个实验，此实验算法的学习速率是较为缓慢的，最大累计奖励值在 150 代左右趋近于一个渐进值，一直到 300 代左右都没有较大的变化，由此可以看出该问题学习过程是较为困难的，因为算法学习的策略空间增大为之前的数倍且策略需要分阶段学习，算法需要更多的时间以及更强的启发能力去搜索策

略；对比基础算法与改进算法，改进算法无论是在最大值还是最小值变化曲线上，表现都优于基础算法，仔细观察曲线的变化趋势，我们发现当算法学习到一个更优的策略之后，会迅速提高累计奖励值，如左图 120 代左右，右图 60 代左右，120 代左右，这些变化表现出值函数双重学习的方法能够充分利用学习到的新策略，对于累计奖励值的提升效果十分明显，在接近 300 代时，改进算法与基础算法都有略微的提升，受限于实验设备的性能，实验代数上限为 300 代，300 代之后的学习效果仍需更多实验探索。

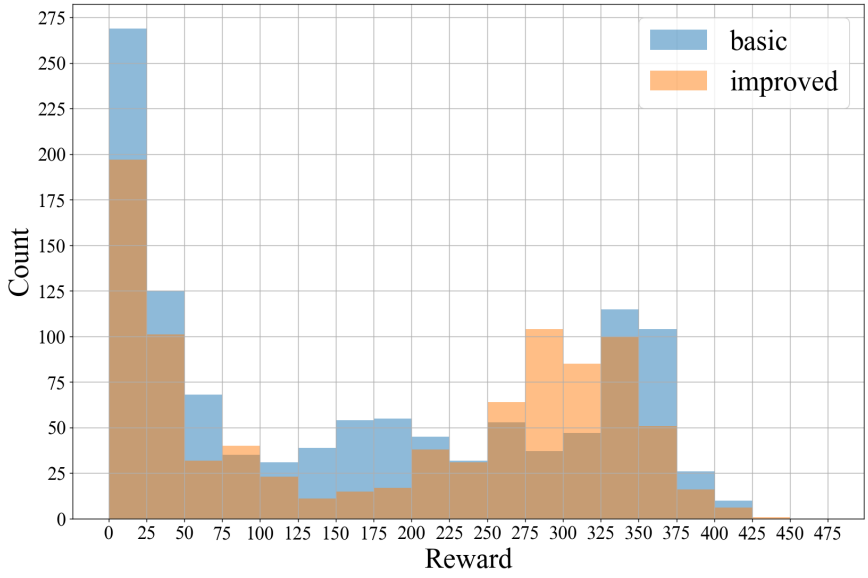

图 4-12 种群最优个体与集成体累计奖励值分布情况

实验结果对比的第二项为种群最优个体与集成体在全部测试中所获得累计奖励值的分布情况，如图 4-11 所示，由图可以看出，该实验结果与前两个实验截然不同，采用了集成化智能体的方法并没有比种群中的最优个体表现的更好，大部分累计奖励值区间上所获得的次数均小于基础算法，结果令人深思，分析其原因可能是由于该测试问题比较困难，算法在寻找策略的过程中策略波动较大，多个个体结合在一起并不能根据彼此之间所获得信息的交集来对动作进行决策，因此集成化智能体所表现出的策略接近于随机策略，测试结果性能较差，思考之余对比了采用了值函数双重学习方法的集成化智能体与种群最优个体的累计奖励值分布，结果如图 4-12 所示，较之前有明显提高，改进算法在 250 到 325 区间上表现好于基础算法，但可能由于还未学习到稳定的策略，在较高累计奖励值区间上，改进算法表现还是差于基础算法。

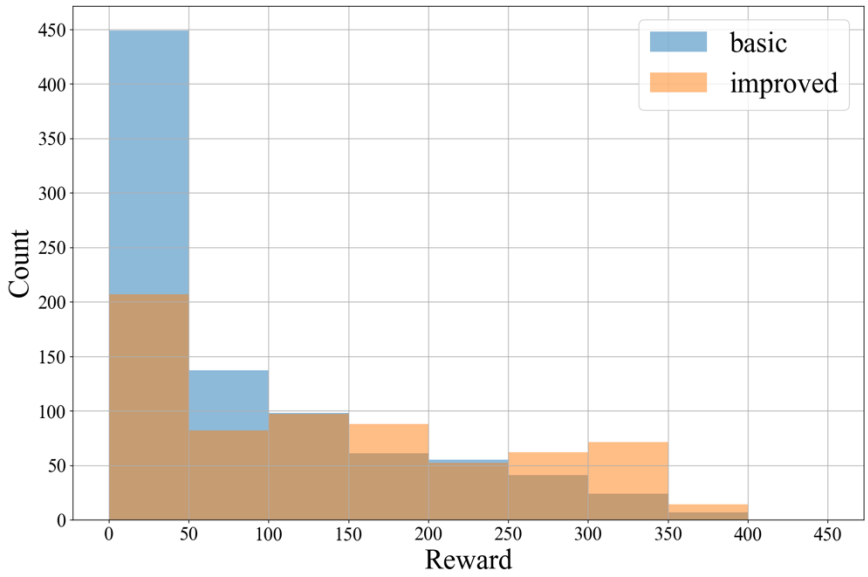

图 4-13 基础算法与改进算法平均累计奖励值的分布情况

实验结果对比第三项为基础算法与改进算法种群平均累计奖励值的分布情况，如图 4-13 所示，由图可以看出，改进算法在 250 至 400 高平均累计奖励值区间上所获的次数均高于基础算法，基础算法在 250 之后的区间数量呈现递降的趋势，而改进算法则先增后降，结果体现出值函数双重学习方法对于种群整体值函数学习的提高作用，基础算法在 0 至 50 区间获得次数远远高于改进算法，从一定程度上显现出基础算法在策略学习初期并不能根据学习到的较好策略去提高种群整体的表现，而改进算法则在累计奖励值 100 之后的区间接近基础算法或优于基础算法，此差异也初步验证了值函数双重学习改进算法的有效性，加快了智能体策略的学习。

4.5 实验总结

通过上述三个不同难度测试问题的结果对比，初步验证了集成化智能体方法能够提高智能体在测试问题上的鲁棒性，集成体结合多个个体所学习到的策略，利用了群体的智慧提高了在不同测试环境下的表现；对于算法采用值函数双重学习方法前后所得到的平均累计奖励值变化曲线与分布情况的对比，验证了该方法对于提高种群整体表现的有效性，在第三个测试问题中改进算法的平均累计奖励值变化曲线明显好于基础算法，表明提高种群整体表现的同时可能产生出策略更优的个体，体现了值函数双重学习对于策略提高的潜在作用。

结论

1. 论文工作总结

本文基于强化学习算法的基本原理，将二叉树模型运用于状态值函数的拟合并通过遗传编程算法优化二叉树的组成，实现了遗传编程算法辅助的强化学习算法，又针对算法在实验中所遇到的问题提出改进算法，提高了智能体的学习效率和在不同环境中的鲁棒性。论文首先介绍了强化学习的相关背景和国内外研究现状，并列出了若干个将进化算法与强化学习相结合的算法实例，简要说明了将遗传编程算法用于强化学习的理论基础；论文的第二章介绍了遗传编程算法的基础知识，包括基因的组成与表达方式，基因的交叉与变异方法；还介绍了强化学习的基础知识，包括状态值函数预测与动作决策的基本方法，环境模型未知时的动作状态值函数和蒙特卡洛方法，更新间隔短学习效率更高的时序差分学习算法和 Q 学习算法；第三章对实现的遗传编程算法辅助的强化学习算法进行说明并介绍了其改进算法，具体包括基于二叉树模型所表示的状态值函数，动作决策控制和种群个体适应度评估方法，针对基础算法的不足，提出值函数的双重学习和集成化智能体的方法；第四章的实验部分设置了三个难度不同的控制问题对算法进行测试，问题分为简单的低维状态值低维动作输出，难度中等的较高维状态值和动作输出和难度最大的分阶段策略学习控制问题，并从三个不同的角度对基础算法和其改进算法的结果进行对比，初步验证了改进算法的有效性。

2. 工作展望

本文实现的遗传编程辅助的强化学习算法需要在已知环境转移信息的情况下进行学习，对于环境信息完全未知时的算法设计还需进一步的探索；本文实验部分测试问题的状态信息维度还较小，针对状态信息复杂情况下状态信息的选择与压缩还需更多的尝试[25]；算法执行过程中，二叉树的大小增长较快，树的规模越大遍历的时间成本越高，如何在保证二叉树模型表现能力的前提下优化二叉树的结构，文献[13][14]提出了不一样的方法；本文提出的算法还未与基于神经网络的强化学习算法进行对比，其学习效果的优劣还需更多的实验来说明。

参考文献

[1] Sutton, Richard S., and Andrew G. Barto. "Reinforcement learning: An introduction." (2011).

[2] Mnih, Volodymyr, et al. "Playing atari with deep reinforcement learning." *arXiv preprint arXiv:1312.5602* (2013).

[3] Lillicrap, Timothy P., et al. "Continuous control with deep reinforcement learning." *arXiv preprint arXiv:1509.02971*(2015).

[4] Mnih, Volodymyr, et al. "Asynchronous methods for deep reinforcement learning." *International conference on machine learning*. 2016.

[5] Salimans, Tim, et al. "Evolution strategies as a scalable alternative to reinforcement learning." *arXiv preprint arXiv:1703.03864* (2017).

[6] Stanley, Kenneth O., and Risto Miikkulainen. "Evolving neural networks through augmenting topologies." *Evolutionary computation* 10.2 (2002): 99-127.

[7] Stanley, Kenneth O., David B. D'Ambrosio, and Jason Gauci. "A hypercube-based encoding for evolving large-scale neural networks." *Artificial life* 15.2 (2009): 185-212.

[8] Gaier, Adam, and David Ha. "Weight agnostic neural networks." *Advances in Neural Information Processing Systems*. 2019.

[9] Kubalík, Jiří, et al. "Symbolic Regression Methods for Reinforcement Learning." *arXiv preprint arXiv:1903.09688*(2019).

[10] Langdon, William B., et al. "Genetic programming: An introduction and tutorial, with a survey of techniques and applications." *Computational intelligence: A compendium*. Springer, Berlin, Heidelberg, 2008. 927-1028.

[11] O'Reilly, Una-May, and Erik Hemberg. "Introduction to genetic programming." *Proceedings of the Genetic and Evolutionary Computation Conference Companion*. 2019.

[12] Ahvanooey, Milad Taleby, et al. "A Survey of Genetic Programming and Its Applications." *TIIS* 13.4 (2019): 1765-1794.

[13] Zhong, Jinghui, Liang Feng, and Yew-Soon Ong. "Gene expression programming: A survey." *IEEE Computational Intelligence Magazine* 12.3 (2017): 54-72.

[14] Zhong, Jinghui, Yew-Soon Ong, and Wentong Cai. "Self-learning gene expression programming." *IEEE Transactions on Evolutionary Computation* 20.1 (2015): 65-80.

[15] Jackson, David. "A new, node-focused model for genetic programming." *European Conference on Genetic Programming*. Springer, Berlin, Heidelberg, 2012.

[16] Qin, Zhiwei, Jian Tang, and Jieping Ye. "Deep Reinforcement Learning with Applications in Transportation." *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2019.

[17] Akimoto, Youhei, Anne Auger, and Nikolaus Hansen. "CMA-ES and advanced adaptation mechanisms." *Proceedings of the 2016 on Genetic and Evolutionary Computation Conference Companion*. 2016.

[18] Hasselt, Hado V. "Double Q-learning." *Advances in neural information processing systems*. 2010.

[19] Bhowan, Urvesh, et al. "Evolving diverse ensembles using genetic programming for classification with unbalanced data." *IEEE Transactions on Evolutionary Computation* 17.3 (2012): 368-386.

[20] Gagné, Christian, et al. "Ensemble learning for free with evolutionary algorithms?." *Proceedings of the 9th annual conference on Genetic and evolutionary computation*. 2007.

[21] Koza, John R., and John R. Koza. *Genetic programming: on the programming of computers by means of natural selection*. Vol. 1. MIT press, 1992.

[22] Singh, Satinder P., and Richard S. Sutton. "Reinforcement learning with replacing eligibility traces." *Machine learning*22.1-3 (1996): 123-158.

[23] Barto, Andrew G., Richard S. Sutton, and Charles W. Anderson. "Neuronlike adaptive elements that can solve difficult learning control problems." *IEEE transactions on systems, man, and cybernetics* 5 (1983): 834-846.

[24] Gal, Yarin, Rowan McAllister, and Carl Edward Rasmussen. "Improving PILCO with

Bayesian neural network dynamics models." *Data-Efficient Machine Learning workshop, ICML*. Vol. 4. 2016.

[25] Tran, Binh, Bing Xue, and Mengjie Zhang. "Genetic programming for feature construction and selection in classification on high-dimensional data." *Memetic Computing*8.1 (2016): 3-15.

[26] Cramer, Nichael Lynn. "A representation for the adaptive generation of simple sequential programs." *Proceedings of the first international conference on genetic algorithms*. 1985.

[27] 蒋毅恒,白焰,朱耀春,樊丽.基于遗传编程的智能建模方法及应用.微计算机信息,2008(12):150-152.

[28] 曹波,蒋宗礼.基于自助法的遗传编程泛化能力.计算机工程与设计,2017,38(3):768-772.

[29] 毕莹,薛冰,张孟杰.GP 算法在图像分析上的应用综述.郑州大学学报:工学版,2018,39(6):3-13.

[30] 胡建军,唐常杰,段磊,左劼,彭京,元昌安.基因表达式编程初始种群的多样化策略.计算机学报,2007,30(2):305-310.

[31] 赵冬斌,邵坤,朱圆恒,李栋,陈亚冉,王海涛,刘德荣,周彤,王成红.深度强化学习综述：兼论计算机围棋的发展.控制理论与应用,2016,33(6):701-717.

[32] 朴松昊,洪炳熔.一种动态环境下移动机器人的路径规划方法.机器人,2003,25(1):18-21.

[33] Silver, David, et al. "Mastering the game of Go with deep neural networks and tree search." *nature* 529.7587 (2016): 484.

[34] Berner, Christopher, et al. "Dota 2 with Large Scale Deep Reinforcement Learning." *arXiv preprint arXiv:1912.06680*(2019).

致谢

大学四年时光很快就过去了，回顾大学的生活还是感到充实与快乐的，毕业设计在我内心中并不是一次衡量我能否毕业的考核，而是对我大学所学习知识与技能的完整回顾，还记得在去年 10 月份对于自己毕业设计的规划，我希望自己的“最后一舞”既能够总结本科期间在进化计算领域所学到的知识，也能够完成一次进化计算与机器学习相结合的尝试，所以从那时起便开始为毕业设计做准备，经过半年的努力，终于完成这篇算法设计论文，在此我想对一些人表示感谢，感谢他们在生活学习中对我的帮助。

首先我要感谢我的父母，感谢他们在我感受到学习压力时的安慰与开导，在我遭遇挫折时的陪伴与鼓励；感谢他们为我提供了一个能够心无旁骛专注于算法研究的环境，是他们的支持让我能够坚定追寻自己的梦想。

其次我要感谢陈伟能老师和钟竞辉老师对我的指导和栽培，是他们带领我进入进化计算的研究领域，并在本科的科研工作中给予我极大的关照，在他们的指导下，我取得了一系列的学术成果，给我未来的科研道路打下坚实的基础，也感谢老师在我留学申请时提供的巨大帮助。

我还要感谢我的室友，大队长，树清和元泓，感谢室友这四年里对我的宽容与帮助，融洽的寝室氛围让我在异乡漂泊时感受到如同家般的温暖，让我的大学生活感到轻松愉快。

最后我还要感谢我自己，感谢自己在这四年中始终如一的努力，无论外界有多么嘈杂，也能够保持自己的初心，追寻内心的想法；也感谢这四年来的不断学习让我对人工智能这门学科有了更为清晰的认识，对未来的规划有了更加明确的方向。