# 文献综述

多智能体强化学习算法是在单智能体强化学习算法的基础之上发展而来的，为了综述的完整性，本文的文献综述分为单智能体强化学习算法综述和多智能体强化学习算法综述，并总结现有方法的优缺点。

## 单智能体强化学习算法综述

在单智能体强化学习中，智能体需要学习到一个策略，使得它在与环境交互的过程中可以获得尽可能大的累计奖励；在单智能体强化学习方法中，Q-learning是最流行的强化学习方法的一种[1]。Q-learning算法基于时序差分让智能体学习每个状态下每个动作可能获得的回报的均值，从而做出最优决策。通常，Q-learning方法会让智能体采用ε-贪心策略进行动作选择[2]。

Q-learning被Watkins博士首次提出[3]。Watkins博士[3]将智能体与环境交互并进行学习的过程建模为马尔可夫决策过程（Markov Decision Process, MDP），并将时序差分、动态规划等思想结合到一起，提出了Q-learning的概念。Q-learning通过在程序内维护一张二维的Q表格，来计算智能体在状态s下采取动作a可以获得的回报的期望；每次与环境交互时，程序都会通过状态、动作以及从环境中获得的实际奖励来更新Q表格的一部分。在Q-learning被提出之后，Watkins等人[4]证明了Q-learning算法的收敛性：假如智能体重复探索过了所有状态下所有可能的动作，那么Q-learning中的Q表格会收敛到最优的状态-动作价值函数。

在Q-learning提出后，许多学者针对Q-learning算法做了相关改进。Q-learning算法需要在内存中维护一张状态-动作的二维Q表格，当状态、动作数量过多或者智能体处于连续观测空间时，Q-learning算法就会产生极高的内存开销；所以Martin[5]提出了神经拟合Q值（Neural Fitted Q, NFQ）算法，用神经网络计算在状态s下执行动作a的Q值，代替从Q表格中查询在状态s下执行动作a的Q值，很大程度上节省了内存并提高了数据利用的效率。Mnih等人[6]提出了深度Q网络（Deep Q-Network, DQN），DQN可以直接将每个状态的视觉图像作为输入，并且根据状态s计算所有动作的分数，通过经验回放让智能体根据过往的经验更新网络参数，并将智能体与环境的交互存放到经验池中，以便后续智能体进行学习；DQN在Atari游戏上的表现远优于当时的其它算法。值得说明的是，NFQ算法中神经网络的输入是状态s和动作a，输出是对应的Q值，也就是说对每个状态下的每个动作都要用神经网络计算其Q值；而DQN算法中神经网络的输入是状态s，输出是动作空间维度的向量，表示当前状态下每个动作对应的Q值；因此，DQN中对神经网络的利用显然是更高效的。

在DQN算法提出后，许多学者从不同角度对DQN进行了改进。Van等人[7]发现DQN中最终预测出的Q值过高的现象，于是他们提出了双网络DQN（Double DQN，DDQN），DDQN包含一个用于计算Q值的网络（策略网络）和一个计算目标值的网络（目标网络），每次训练时只更新策略网络的参数，一段时间后，目标网络的参数与策略网络同步；这样就保证了计算的Q值不会越来越高，减小了Q值的过估计。Wang等人[8]提出了对偶网络的DQN（Dueling DQN），将网络输出的Q值分解为动作Q值和状态Q值，这两个Q值加和起来就是智能体在某状态下选择某动作的Q值；这种网络架构除了让智能体学习到好的决策外，还能让智能体学习到环境中什么状态是好的状态。在一些场景中，Q值和当前状态高度相关，在某些状态下无论选择什么样的动作的Q值都相差不大，Dueling DQN的设计就可以让智能体在这些场景下学习到更好的策略。

上述强化学习算法的目标都是找到最优的状态-动作价值函数Q，即让智能体预测在状态s下选择动作a可以获得的回报的均值。然而，在环境中存在的随机性过大的情况下，仅仅计算回报的均值是不够的。因此，Guo等人[9]提出了级联时序差分学习的方法，利用两个级联的Q表格，让智能体接连预测在状态s下选择动作a可以获得的回报的均值和方差，并推导出了Q表格更新的公式；他们的算法在实际的交通路网上高效地解决了可靠最短路径的问题。

## 二、多智能体强化学习算法综述

在多智能体强化学习中，每个智能体都需要学习到自己的最优策略，使得所有智能体一起行动时多智能体系统可以获得尽可能大的累积奖励[10]。随着多智能体强化学习方法广泛应用到人工智能相关领域，用于解决路径规划、博弈、任务分配等问题，越来越多的研究人员将目光聚集在了多智能体强化学习方法上[11]。多智能体强化学习方法可以根据智能体之间的关系大致分为三类：完全合作式、完全竞争式以及合作-竞争混合式[12]。在完全合作式多智能体强化学习中，所有智能体需要相互协作以完成共同的团队任务，它们的目标是最大化团队累积奖励；在完全竞争式多智能体强化学习中，每个智能体只关注自身行为，它们的目标是最大化自身累积奖励；在合作-竞争混合式多智能体强化学习中，每个智能体既要考虑自身利益也要考虑团队利益，它们的优化目标是在最大化自身累积奖励的同时通过与其他智能体的合作最大化团队累计奖励。

对于完全合作式、完全竞争式或是合作-竞争混合式，存在一种通用的多智能体强化学习方法，也就是独立Q学习（Independent Q-Learning, IQL）[13]。IQL算法中，每个智能体都会把其它智能体看作环境的一部分，也就是说，每个智能体都在解决一个单智能体强化学习任务；对于不同的任务，只需要修改每个智能体获得奖励的方式即可。然而，由于在IQL的建模中，环境中存在其它智能体，所以环境是非稳态的，算法的收敛性无法从理论上得到证明；且智能体之间无法进行任何形式的信息共享，然而，这种算法在工程实践中仍有一定价值。接下来，本文将分别对完全合作式、完全竞争式以及合作-竞争混合式这三类多智能体强化学习方法进行介绍。

### 1、完全合作式多智能体强化学习

在完全合作式多智能体强化学习中，智能体们会因为完成团队目标而统一获得奖励，不会有个体的奖励。这种共享团队奖励的方式会带来问题，即一个智能体由于只能获得自己的局部观测，从而无法确定是因为自己的行为或者其它智能体的行为而获得奖励；除此之外，由于部分智能体可以学习到比较好的策略以完成团队任务，其它智能体就会变得“懒惰”。为了解决上述问题，Sunehag等人[14]提出了价值分解网络（Value Decomposition Networks, VDN），将团队的Q值分解为各个智能体的Q值之和，每个智能体用深度神经网络来拟合自己的Q值；这样，每个智能体就可以分配到合理的奖励值，且一定程度上避免了懒惰智能体的问题。Rashid等人在VDN的基础上提出了QMIX[15]。Rashid等人通过混合网络将团队的Q值分解为各个智能体的Q值的复杂非线性组合，并在训练中加入全局信息，使得每个智能体学习到的策略更精确；且通过保证混合网络参数的非负性，保证团队Q值与每个智能体计算的Q值的关系是单调的，从而保证集中策略和分散策略的一致性。Kyunghwan等人[16]提出了QTRAN，为价值分解提供了更多了理论证明，并且将每个智能体的价值进行进一步映射，保证每个智能体学习到的状态-价值函数可以逼近最优。

## 2、完全竞争式多智能体强化学习

在完全竞争式多智能体强化学习中，不同的智能体的目标通常是不同的，甚至互相之间进行博弈，也就是每个智能体都需要学习到比其它智能体更好的策略，才可能获得更高的奖励。Micheal[17]针对二人零和随机博弈场景提出了Minimax-Q算法，使用Q-learning中的时序差分来迭代更新minimax算法的求解公式，即让每个智能体最大化在博弈中最差情况下的期望奖励值，最终学习到纳什均衡策略。Hu[18]等人针对多智能体博弈设计了Nash Q-learning，旨在在多智能体相互竞争或合作的环境中找到纳什均衡作为各个智能体的最优策略；其中，每个智能体在给定其它智能体策略的情况下，通过求解纳什均衡，选择一个能最大化自身收益的动作，并更新状态-动作价值函数。

### 3、合作-竞争混合式多智能体强化学习

在合作-竞争混合式多智能体强化学习中，智能体需要兼顾自身和团队的利益。Ryan等人[19]提出了多智能体深度确定性策略梯度（Multi-Agent Deep Deterministic Policy Gradient, MADDPG）算法；MADDPG算法是一种基于深度确定性策略梯度的扩展，用于解决多智能体环境中的协作和竞争问题。MADDPG通过独立的演员-评论家架构让每个智能体在训练中都可以了解其他智能体的策略信息，以改善学习效果和稳定性。Mnih等人[20]提出了异步优势演员-评论家（Asynchronous Advantage Actor-Critic, A3C）算法；A3C算法是一种策略梯度算法，它通过并行执行多个智能体来提高训练效率，即允许异步更新全局模型。A3C结合了价值和策略梯度方法，并使用n步回报来更新策略和价值函数，从而在复杂的围棋、星际争霸等任务中取得了显著的效果。

# 三、总结

总而言之，现有的多智能体强化学习方法只对智能体获得的回报的期望进行建模，从一定意义上讲，这些方法在随机性很强的多智能体环境中不会学习到鲁棒的最优策略。在单智能体强化学习方法中，有学者提出了CTD对智能体获得的回报的均值和方差同时进行建模，然而这种方法没有考虑一般强化学习中的折扣因子，只应用在了简单的交通路网环境中，并不能应用到其它领域。

# 参考文献

[1]. Thrun, S., & Littman, M. L. (2000). Reinforcement learning: an introduction. AI Magazine, 21(1), 103-103.

[2]. Jang, B., Kim, M., Harerimana, G., & Kim, J. W. (2019). Q-learning algorithms: A comprehensive classification and applications. IEEE access, 7, 133653-133667.

[3]. Watkins, C. J. C. H. (1989). Learning from delayed rewards.

[4]. Watkins, C. J., & Dayan, P. (1992). Q-learning. Machine learning, 8, 279-292.

[5]. Riedmiller, M. (2005). Neural fitted Q iteration–first experiences with a data efficient neural reinforcement learning method. In Machine learning: ECML 2005: 16th European conference on machine learning, Porto, Portugal, October 3-7, 2005. proceedings 16 (pp. 317-328). Springer Berlin Heidelberg.

[6]. Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., ... & Hassabis, D. (2015). Human-level control through deep reinforcement learning. nature, 518(7540), 529-533.

[7]. Van Hasselt, H., Guez, A., & Silver, D. (2016, March). Deep reinforcement learning with double q-learning. In *Proceedings of the AAAI conference on artificial intelligence* (Vol. 30, No. 1).

[8]. Wang, Z., Schaul, T., Hessel, M., Hasselt, H., Lanctot, M., & Freitas, N. (2016, June). Dueling network architectures for deep reinforcement learning. In *International conference on machine learning* (pp. 1995-2003). PMLR.

[9]. Guo, H., Hou, X., & Peng, Q. (2021). CTD: Cascaded temporal difference learning for the mean-standard deviation shortest path problem. IEEE Transactions on Intelligent Transportation Systems, 23(8), 10868-10886.

[10]. Oroojlooy, A., & Hajinezhad, D. (2023). A review of cooperative multi-agent deep reinforcement learning. Applied Intelligence, 53(11), 13677-13722.

[11]. 罗彪,胡天萌,周育豪,等.多智能体强化学习控制与决策研究综述[J/OL].自动化学报,1-30[2024-11-24].https://doi.org/10.16383/j.aas.c240392.

[12]. Zhang, K., Yang, Z., & Başar, T. (2021). Multi-agent reinforcement learning: A selective overview of theories and algorithms. Handbook of reinforcement learning and control, 321-384.

[13]. Tampuu, A., Matiisen, T., Kodelja, D., Kuzovkin, I., Korjus, K., Aru, J., ... & Vicente, R. (2017). Multiagent cooperation and competition with deep reinforcement learning. PloS one, 12(4), e0172395.

[14]. Sunehag, P., Lever, G., Gruslys, A., Czarnecki, W. M., Zambaldi, V., Jaderberg, M., ... & Graepel, T. (2017). Value-decomposition networks for cooperative multi-agent learning. arXiv preprint arXiv:1706.05296.

[15]. Rashid, T., Samvelyan, M., De Witt, C. S., Farquhar, G., Foerster, J., & Whiteson, S. (2020). Monotonic value function factorisation for deep multi-agent reinforcement learning. Journal of Machine Learning Research, 21(178), 1-51.

[16]. Son, K., Kim, D., Kang, W. J., Hostallero, D. E., & Yi, Y. (2019, May). Qtran: Learning to factorize with transformation for cooperative multi-agent reinforcement learning. In International conference on machine learning (pp. 5887-5896). PMLR.

[17]. Littman, M. L. (1994). Markov games as a framework for multi-agent reinforcement learning. In Machine learning proceedings 1994 (pp. 157-163). Morgan Kaufmann.

[18]. Hu, J., & Wellman, M. P. (2003). Nash Q-learning for general-sum stochastic games. Journal of machine learning research, 4(Nov), 1039-1069.

[19]. Lowe, R., Wu, Y. I., Tamar, A., Harb, J., Pieter Abbeel, O., & Mordatch, I. (2017). Multi-agent actor-critic for mixed cooperative-competitive environments. Advances in neural information processing systems, 30.

[20]. Mnih, V. (2016). Asynchronous Methods for Deep Reinforcement Learning. arXiv preprint arXiv:1602.01783.