基于纳什深度Q网络方法的大规模交通信号控制

Yuli Zhang、Shangbo Wang、Ruiyuan Jiang

arXiv:2301.00637v1 [cs.GT] 2 Jan 2023

摘要—**强化学习（RL）目前是交通信号控制（TSC）中最常用的技术之一，能够根据实时交通数据自适应地调整交通信号相位和持续时间。然而，在多交叉口网络场景中，完全集中式RL方法面临诸多困难，因为随着交叉口数量的增加，状态-动作空间呈指数增长。多智能体强化学习（MARL）通过对每个局部强化学习智能体实施全局控制，可以克服高维问题，但同时也带来了新的挑战，例如马尔可夫决策过程（MDP）的非平稳性导致收敛失败。本文提出了一种异策略纳什深度Q网络（OPNDQN）算法，能够缓解完全集中式方法和MARL方法的不足。OPNDQN算法在每次迭代中采用虚拟博弈方法，在相邻交叉口之间寻找纳什均衡，从而解决传统算法无法应用于大状态-动作空间交通模型的问题；在该均衡下，没有任何交叉口有动机单方面偏离。OPNDQN的主要优势之一是，通过共享相邻交叉口的动作，考虑了相邻交叉口之间的相互影响，从而减轻了多智能体马尔可夫过程的非平稳性。另一方面，在大型交通网络的训练中，OPNDQN的收敛速度高于现有MARL方法，因为它不包含每个智能体的全部状态信息。我们使用城市交通仿真模拟器（SUMO）开展了大量实验，结果表明，在平均队列长度、回合训练奖励和平均等待时间方面，OPNDQN均明显优于现有的多种MARL方法。**

关键词—**多智能体、强化学习、深度Q网络、纳什均衡、交通信号灯控制。**

I. 引言

随着汽车保有量的增长，当前交通需求迅速上升。随着出行需求增加，交通拥堵和交通事故日益严重。主要原因之一是交通信号灯设置不合理。据报道， [1] 当前交通信号网络无法根据实时交通状态进行自适应响应。现有的交通信号灯控制要么采用定时模式而不考虑实时交通状况，要么仅在极有限程度上考虑交通状况 [2], , , 这会导致车辆在交叉路口积聚，并降低交通效率 [3]。因此，制定有效的交通信号控制策略对城市道路网络具有重要意义。

近年来，强化学习（RL）技术在交通信号控制（TSC）领域取得了许多显著成果，因为强化学习能够更好地解决序列动作的决策优化问题。在相关文献中，采用强化学习的交通信号控制方法可分为基于模型和无模型两类。基于模型的方法中，智能体尝试理解环境，并通过与环境交互来创建模型。无模型强化学习方法无需透彻理解交通系统的动态特性，即可学习最优策略基准 [4]。无模型方法简单直接且便于实现，因此在交通信号控制中得到了广泛应用。无模型强化学习有两种变体：基于数值的方法和基于策略的方法。Q学习是一种基于数值的强化学习方法，无需环境模型即可比较可用动作的预期效用。然而，当状态空间很大时，Q 表也会变得十分庞大，搜索和存储都需要耗费大量时间和内存，因此，Q学习方法的性能取决于交通特征设计的质量 [5], [6], [7]。深度Q网络（DQN）由 Mnih等人于2015年首次提出，它通过使用多层卷积神经网络近似Q函数，成功地将强化学习与深度学习相结合 [8]。从[9], [10]可以看出，在单智能体控制中，DQN能够在较小的状态-动作空间中取得良好性能。2019年，Liang和Du提出了双重对决深度Q网络方法，该方法融合了对决网络、双重网络和深度网络[11]，用于控制单个交叉路口的交通灯周期。

尽管强化学习技术在单交叉口场景的交通信号控制中取得了良好成效，但由于交叉口数量增加时状态-动作空间呈指数级增长，因此该技术不适用于包含多个交叉口的大规模城市道路网络。为克服联合动作空间维度过高的问题，一些研究人员提出了多智能体强化学习方法，将全局控制分配给各个局部智能体 [12], [13]。2020年，Chu等人 [14]在自适应交通信号控制环境中提出了一种优势演员-评论家（A2C）算法，该算法通过考虑其他智能体的控制策略，提高了每个智能体的可观测性。2021年，Chen等人通过采用差分奖励方法 [15]改进了A2C。然而，分布式A2C方法的缺点在于，当状态维度或周边交叉口数量增加时，交叉口之间会产生严重的相互影响，从而导致环境呈现非平稳性。

自纳什在 [16], 提出博弈论中的纳什均衡概念以来，已有一些关于纳什Q学习的研究。2004年，Hu和Wellman设计了一种在策略纳什Q学习算法，将Q学习扩展到非合作多智能体环境中 [17]。然而，这种在策略纳什Q学习算法有一个明显的缺陷，即需要知道或能够计算每个智能体的控制

策略。我们认为，在复杂交通网络中，在策略方法并不实用，因为获取准确的控制策略十分繁琐。为解决这一问题，本文提出一种异策略纳什深度Q网络（OPNDQN）算法，通过在相邻交叉口之间迭代搜索纳什均衡来找到最优策略，无需知道或计算控制策略，即根据当前状态选择动作的条件概率。主要贡献如下：

- 我们提出了 OPNDQN 算法，该算法无需了解或计算控制策略，即可使非合作智能体的期望累积奖励达到纳什均衡，并将其应用于大规模交通信号控制场景；

- 我们通过使用 SUMO（城市交通仿真） [18]开展大量实验，评估所提出方法的性能。我们将 OPNDQN 算法与完全分散式 DQN 方法、多智能体 Q 学习（ MAQL）、合作式多智能体深度强化学习方法（ Co-MARL） [19]以及多智能体优势演员-评论家（ MA2C）进行比较。结果表明，在多项交通指标——平均队列长度、回合训练奖励和平均等待时间——方面，所提出的 OPNDQN 方法均优于这些方法。

II. 方法论

为了让读者更好地理解 OPNDQN 方法，我们首先简要回顾马尔可夫决策过程。

A. 马尔可夫决策过程

马尔可夫决策过程（MDP）是描述强化学习问题的基本框架，它通过一个序贯决策过程来描述，该过程包含状态空间、动作空间、奖励、状态转移函数和折扣因子。强化学习是一个与环境交互，通过长期试错来最大化目标奖励函数的序贯决策过程。一个 MDP 可以用五元组 $\langle S，A，R，T\ \gamma\rangle$ 定义：

- 状态空间 S：所有可能状态的集合。状态是对环境的描述和概括。$s_t(s_t \in S)$ 是智能体在第 t 个时刻观察到的状态。

- 动作空间 A：所有可能动作的集合。$a_t\ (a_t \in A)$ 是智能体在第t 个时刻执行的动作。

- 奖励 R：奖励空间。智能体执行动作后，环境会向智能体返回一个数值（奖励）。$r_t\ (r_t \in R)$ 是智能体在第 t 个时刻获得的奖励。

- 状态转移函数 T：环境用于生成新状态的函数。它表示从一个状态转移到另一个状态的概率。状态转移函数 T 可表示为：$p\left(s_{t+1} \mid s_t, a_t\right) = \mathbb{P}\left(S' = s_{t+1} \mid S = s_t A = a_t\right)$，其中$p$为概率，$a_t$ 为智能体在第t个时刻采取的动作，s_t 为智能体的状态，而s_{t+1} 表示所有智能体在下一时刻转移到的状态。

- 折扣因子 γ：对未来奖励进行折扣。（$\gamma \in [0,1]$）

B. OPNDQN

为便于读者理解 OPNDQN方法，我们从一个单智能体系统入手。在 MDP 中，将s_t、a_t和r_t 分别定义为第t个时刻的状态矩阵、动作向量和奖励向量，并将从开始到结束的所有奖励定义为：$r_1, \cdots, r_t, \cdots, r_{t_{End}}$。结合折扣率，时刻$t$ 的未来折扣回报为U_t，可表示为：

$$U_t = r_t + \gamma \cdot r_{t+1} + \gamma^2 \cdot r_{t+2} + \cdots + \gamma^{t_{End}-t} \cdot r_{t_{End}} \tag{1}$$

U_t 的随机性来自时间t之后的所有动作和观测序列。为消除随机性的影响，我们对公式 (1) 取期望，并将动作价值函数$Q_\pi\ (s_t,a_t)$定义为

$$Q_\pi\left(s_t, a_t\right) = \mathbb{E}\left[U_t \mid S_t = s_t, A_t = a_t, \pi\right] \tag{2}$$

其中，π 是一个将序列映射为动作的策略。传统的深度Q网络方法使用最大化方法确定最优策略 π^*，并推导出最优动作价值函数：

$$Q^*(s_t, a_t) = \max_\pi \mathbb{E}\left[U_t \mid S_t = s, A_t = a, \pi^*\right] \tag{3}$$

最优动作价值函数满足一个重要恒等式，即贝尔曼方程 [8]。假设智能体已知后续状态的最优Q值，那么最优策略就是选择a_{t+1} ，使期望值$r_t + \gamma Q^*\ (s_{t+1},a_{t+1})$最大化。实际应用中，通常使用神经网络来近似Q值：

$$Q(s_t, a_t; \theta) \approx Q^*(s_t, a_t) \tag{4}$$

其中，θ 是神经网络的参数。我们将神经网络函数逼近器定义为Q网络。

然而，在MARL中，多智能体系统与同一环境进行交互，由于非平稳性，神经网络无法收敛。为弥补这一不足，我们尝试通过考虑邻近智能体所造成的影响，研究如何减轻MDP的非平稳性。一些文献倾向于让每个智能体收集所有其他智能体的信息，以提高模型训练过程中的可观测性，从而增强平稳性；然而，这会由于状态-动作空间的维度极大而对收敛速度产生负面影响 [19]。本文提出一种OPNDQN方法，该方法通过仅共享各自的动作，在每次迭代中寻找邻近智能体之间的纳什均衡策略。该方法的优势并不仅仅在于提高每个智能体的Q值，而是寻找智能体与其邻居之间的平衡，使每个智能体在不降低邻居奖励的情况下提高自身奖励。我们考虑一个包含N个智能体的非合作多智能体系统，其中每个智能体拥有n 个邻居。对于所有$s^1 \in S^1, \ldots, s^N \in S^N, a^1 \in A^1, \ldots, a^N \in A^N$，我们使用虚拟博弈方法，为多智能体的复杂无模型学习寻找纳什Q值。我们定义，智能体i （$i \in N$）的动作选择不仅取决于其自身的

状态、动作和奖励值，还取决于其^ ^邻居i （$i \in n$）的动作：

$$a_t^i = NashQ_t\left(s_t^i, a_t^i, a_t^{\hat{i}}; \theta_i\right) \tag{5}$$

虚拟博弈过程可描述如下：^在每个回合初始化动作 a_t^i 和 a_t^i ，然后更新群体动作，使每个智能体的Q值不断提高，直至达到纳什均衡；此时，任何智能体都无法通过单方面改变自己的动作来进一步提高其Q值（式6）。

$$NashQ^i\left(s^i, a^i, a^{\hat{i}}; \theta_i\right)_{\text{Step } j} \leq NashQ^i\left(s^i, a^i, a^{\hat{i}}; \theta_i\right)_{\text{Step } j-1} \tag{6}$$

其中，j 是虚拟博弈循环计数器。在此前选取的所有动作中，Q值最大的动作（要执行的实际动作）即为纳什均衡策略。值得注意的是，在上述虚拟博弈过程中，通过改变动作 a_t^i 来搜索纳什均衡的过程^是虚拟的，智能体i 及其邻居i 并不会执行实际动作。只有找到最终的纳什策略$a^i_{\text{Step } j}$ 后，智能体才会执行实际动作。虚拟博弈的流程见图1。OPNDQN中虚拟博弈的伪代码如下：

算法1 虚拟博弈

输入： 随机动作a^i
输出： 纳什联合动作a^i 或随机动作a^i
符号说明： 神经网络中的参数 θ，智能体数量n，纳什参数 ϵ
1：将a^i 初始化为随机动作。
2：**如果**概率 ϵ 选择随机动作**则**
3：输出a^i
4: **否则**
5: **当**步数 $j = 1$ **时**
6: $a^i_{\text{Step } j} = \text{argmax} Q^i\left(s^i, a^i_{\text{Step } j-1}, a^{\hat{i}}_{\text{Step } j-1}; \theta_i\right)$
7: **如果**等式 (6) **则**
8: 跳出循环
9: **结束如果**
10: **结束循环**
11: 输出 $a^i_{\text{Step } j-1}$
12: **结束如果**

我们定义目标 y_k^i ，并通过最小化一系列损失函数 $L_k^i(\theta_k)$ 来训练 Q网络；这些损失函数在每次迭代 k 时都会变化：

$$y_k^i = r_t^i + \gamma \cdot NashQ^i\left(s_{t+1}^i, a_{t+1}^{i,\hat{i}}; \theta_k\right) \tag{7}$$

$$L_k^i(\theta_k) = \mathbb{E}\left[\left(y_k^i - Q_t^i\left(s_t^i, a_t^{i,\hat{i}}; \theta_k\right)\right)^2\right] \tag{8}$$

计算损失L 相对于 θ 的梯度：

$$\begin{aligned}\nabla_{\theta_k} L_k(\theta_k) = \mathbb{E}[&r_t^i + \gamma \cdot NashQ^i(s_{t+1}^i, a_{t+1}^{i,\hat{i}}; \theta_k) \\ &- Q_t^i(s_t^i, a_t^{i,\hat{i}}; \theta_k))\nabla_{\theta_k} Q_t^i(s_t^i, a_t^{i,\hat{i}}; \theta_k)]\end{aligned} \tag{9}$$

使用随机梯度下降法优化损失函数 $L_k^i(\theta_k)$，随着训练步k数量的增加，神经网络的参数 θ_k 会收敛到一个固定值。

C. 目标网络

为了更新神经网络中的参数 θ ，定义一个目标值 θ^* ，以帮助指导更新过程。使用目标网络更新参数后，公式（10）可写为：

$$y_k^i = r_t^i + \gamma \cdot NashQ^i\left(s_{t+1}^i, a_{t+1}^{i,\hat{i}}; \theta_k^*\right) \tag{10}$$

D. 总体架构

算法 2 OPNDQN算法

输入： 回放记忆大小 M，小批量大小B，纳什参数 ϵ，预训练步数 tp，目标网络更新率 α，折扣因子 γ。

符号说明： 主神经网络的参数 θ、目标神经网络的参数 θ^*、回放记忆 m、智能体数量 n、训练步数 k、时间 t_c。

1：使用随机数值初始化参数 θ， θ^* 。
2：将m 初始化为空。
3：在交叉路口将s^i 初始化为起始场景。
4：将a^i 初始化为随机动作。
5：**对于**回合=1 **执行**
6: 初始化状态s^i
7: **当** 车辆数量 $N > 0$, k=1 **时**
8: **算法 1**$\rightarrow a_t^i$
9: 执行动作a^i $_t$
10: 观察奖励r^i $_t$ 以及新状态s^i $_t$，
11: 将回合添加到缓冲区
12: 将 s_{t+1}^i 分配给 s_t^i： $s_t^i \leftarrow s_{t+1}^i$。
13: $k \leftarrow k + 1$.
14: **如果** $|M| > B$ 且$k > tp$ **则**
15: 从回放记忆中选择B 个样本m
16: **算法 1** $\rightarrow a_{t+1}^i$
17: 纳什 $Q_{t+1}^i = Q_{t+1}\left(s_{t+1}^i, a_{t+1}^i, a_{t+1}^{\hat{i}}; \theta_i^*\right)$
18: 计算损失L:
19: $L = [(r + \gamma$纳什$Q_{t+1}^i) - Q(s_t^i, a_t^i, a_{它+1}^{\hat{i}}; \theta_i)]^2$.
20: 使用RMS优化器更新 θ_i 。
21: 将 θ_i 分配给 θ_i^*。
22: **如果**车辆数量 $N = 0$ **则**
23: 跳出循环
24: **结束如果**
25: **结束如果**
26: **结束循环**
27: **</style结束循环**

如图1所示，我们以四个交叉口为例，展示智能体 i（红框）的训练过程。可以看到，红框标示的交叉口有三个邻居（蓝框）。交叉口之间可以相互传递动作信息，智能体可以接收相邻交叉口希望改变的动作。当训练步数 k 大于或等于预训练步数值 tp后，智能体 i 的参数便在训练模型中进行训练。对于本文使用的深度Q网络，我们采用具有相同全连接网络结构的评估网络和目标网络。每个网络包含三个隐藏层，规模分别为

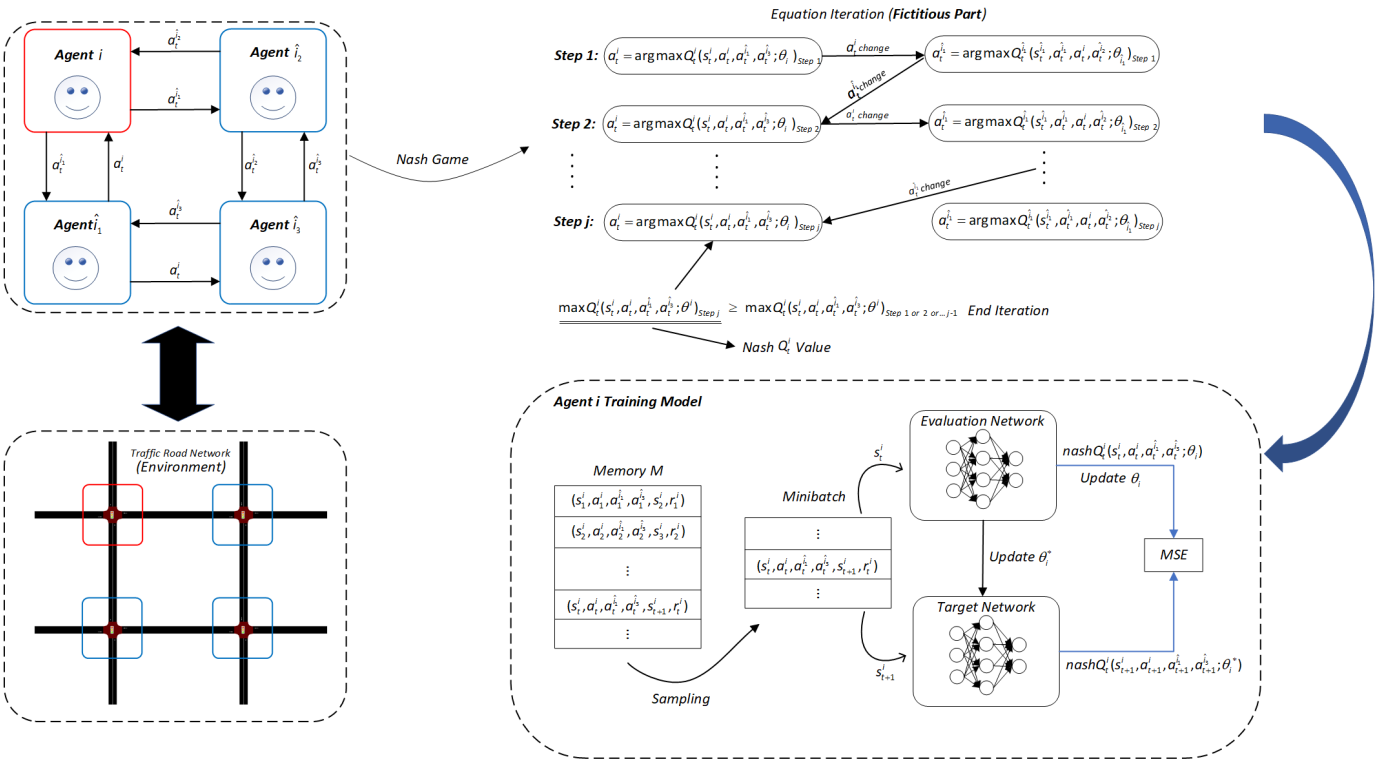


图1：OPNDQN方法的总体架构。

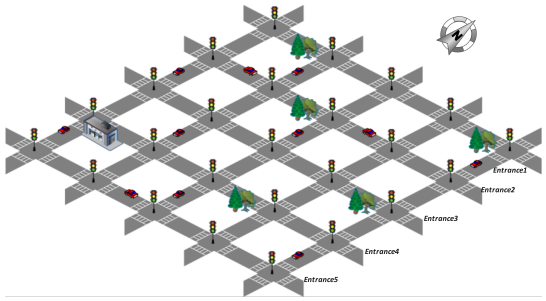


图2：包含25个交通信号控制智能体的道路网络。

32、64、64，每层均采用 *relu* 函数作为激活函数。输出层输出动作的Q值。我们选择学习率为0.0001的均方根（RMS）作为梯度优化器。训练过程中，我们采用批量大小为64的小批量方法。

III. 仿真环境

本节将介绍本研究所采用的仿真环境。首先阐述道路网络模型，随后定义状态、动作和奖励。

A. 道路网络

仿真道路网络是一个 5×5，如图2所示。网格的总尺寸为 600×600 米。道路网络的车辆信息，包括输入车辆数（IVN）（车辆数/回合）和交通流率（TFR）（车辆数/秒），汇总于表I中。

方向	入口	IVN	TFR
北	入口1	1000 车辆/小时	1/20 车辆/秒
	入口2	1000	1/10
	入口3	1100	1/15
	入口4	1050	1/20
	入口5	900	1/10
南	入口1	1100	1/15
	入口2	900	1/15
	入口3	900	1/20
	入口4	1000	1/10
	入口5	950	1/20
East	入口1	1050	1/20
	入口2	1000	1/15
	入口3	950	1/15
	入口4	1000	1/20
	入口5	850	1/10
West	入口1	950	1/15
	入口2	1000	1/20
	入口3	1050	1/10
	入口4	1000	1/10
	入口5	1000	1/20

表 I：道路网络的车辆输入信息。

B. 状态

图3和图4展示了如何设置状态值。图3展示了简单网络（例如包含四个交叉路口）中的交通状态快照，该网络被划分为方形网格。位置矩阵的大小与网格相同，如图4所示。车辆位置设为1，空白单元格表示相应网格中没有车辆，设为0。值得注意的是，每个网格的长度等于汽车的长度，并且一个网格只能取1或0。

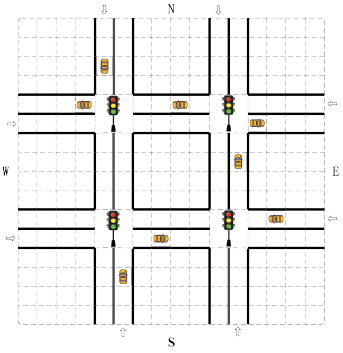


图3：某一时刻的交通状况快照。

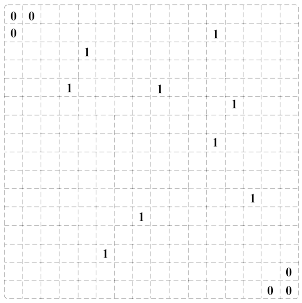


图4：图3对应的位置矩阵。

C. 动作

在我们的模型中，动作空间由每次阶段变化的绿灯相位持续时间定义。我们设定固定周期为40秒。每个交通信号灯的绿灯持续时间可以设置为10秒、15秒、20秒、25秒或30秒。剩余时间为红灯的持续时间（每40秒为一个历元，路线文件中的所有车辆运行完毕后，回合加1，并重置路线文件）。每个智能体具有相同的动作空间。

D. 奖励

奖励的作用是向强化学习模型反馈上一个动作的表现，因此奖励的定义尤为重要。我们将奖励定义为相邻两个历元之间累计等待时间的变化。设 T 历元中所有车辆的累计等待时间为 W_T，则 T 历元中的奖励定义为：

$$r_T = W_{T-1} - W_T \tag{11}$$

该奖励表示动作执行后所有车辆的累计等待时间增量。如果当前历元的奖励比之前更大，就意味着车辆的等待时间正在减少，从而达到优化的目的。

IV. 评估

本节将展示仿真结果，并对本研究中的OPNDQN进行性能分析。首先，我们介绍OPNDQN算法所使用的参数。

A. 评估参数

表II：离策略纳什深度Q网络中的参数

参数	数值
回放记忆大小 M	20000
小批量大小 B	64
开始时间 ϵ	1
结束时间 ϵ	0.01
预训练步数 tp	2000
目标网络更新迭代	100
折扣因子 γ	0.99
学习率 ϵ_r	0.0001

该模型以迭代的方式进行训练。一次迭代即为一个回合。整个开发环境均使用 TensorFlow [20]，编写，模型的参数如表 II 所示。

B. 基线方法

我们将 OPNDQN 的性能与以下基线方法进行比较：

(1) 多智能体Q学习（MAQL） [7]：我们在每个交叉路口分别使用 Q 学习方法，并应用 Q 表寻找最优的交通信号灯控制策略。每个交叉路口彼此独立，且不共享信息。

(2) 多智能体优势演员-评论家（MA2C） [14]：我们在每个交叉路口分别使用 A2C 方法。每个智能体使用评论家网络评估各演员的策略，并指导其优化策略。

(3) 完全去中心化 DQN 方法 [8]：我们在每个交叉路口分别使用传统的 DQN 方法。每个智能体主要受到其邻居最新动作的影响，同时通过 DQN 搜索控制交叉路口的最优策略。

(4) 协作式多智能体深度强化学习方法（Co-MARL）：Haddad 等人在2022 [19]中提出了一种 Co-MARL 方法。Co-MARL方法采用深度Q网络，并在学习过程中将从邻近智能体接收到的状态、动作和奖励传递到自身的损失函数中。与我们的 OPNDQN方法不同，Co-MARL 不让智能体参与博弈，而是将邻近智能体传递的信息用作学习状态，以实现智能体协作。

C. 性能分析

仿真结果如图5、图6和图7所示。图5展示了在相同交通流量下，各回合累积奖励的性能比较。可以看出，

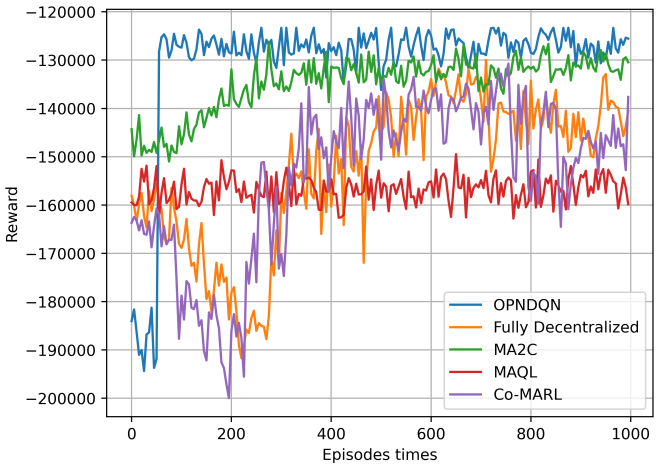


图5：奖励性能对比。

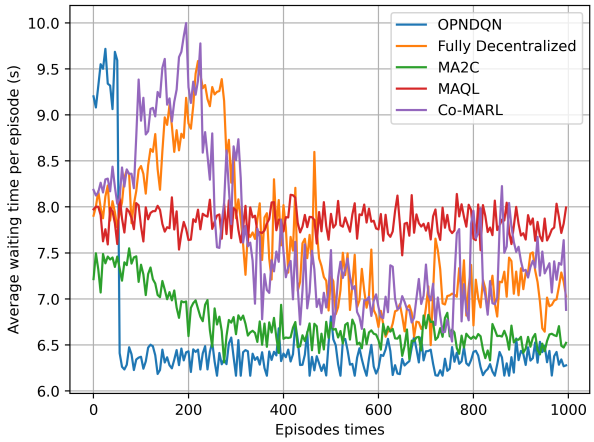


图6：平均等待时间性能对比。

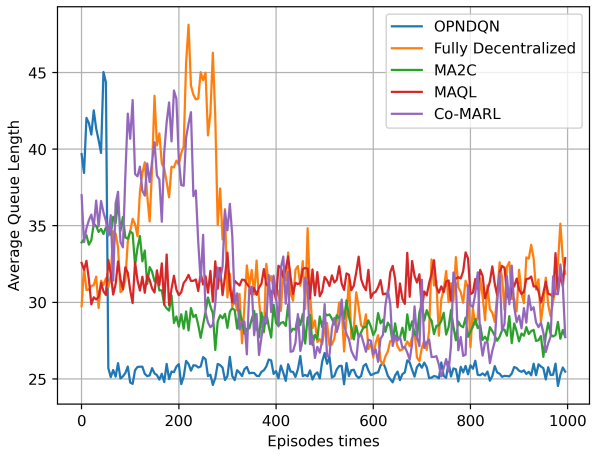


图7：平均队列长度性能对比。

当回合数达到约80时，我们的方法便已收敛。完全分散式DQN方法、MA2C方法和Co-MARL方法则需要250个回合才能收敛。MAQL方法效果最差，模型几乎无法收敛。同时，就训练效果而言，我们的方法收敛后的奖励数值高于其他四种方法。图6展示了五种方法每回合的车辆平均等待时间。可以看出，经过OPNDQN方法训练后，车辆平均等待时间缩短至约6.4秒，比MA2C方法少0.2秒，比完全分散式DQN方法和MA2C方法少0.4秒（由于MAQL无法收敛，因此未将其纳入比较）。虽然我们的方法在车辆平均等待时间方面并未明显优于其他方法，但在收敛速度方面远胜其他四种方法。为评估各种信号控制方法的性能，我们还在图7中比较了每回合各交叉路口的车辆平均队列长度。可以看出，OPNDQN的表现同样出色，在100个回合之前便已收敛，并将队列长度保持在约25。还可以发现，OPNDQN方法在收敛后具有更强的稳定性。完全分散式方法、MA2C方法和Co-MARL方法的三条曲线在收敛后出现较大振荡，这是由于交叉路口之间相互影响所致，这证明了我们的OPNDQN方法具有更好的鲁棒性和稳定性。

V. 结论

本文研究大型交通网络中的自适应交通信号控制问题，并提出一种 OPNDQN 方法。首先，我们采用虚拟博弈方法求得智能体之间的纳什均衡。其次，我们使用智能体之间的纳什Q值，而非最大Q值，使智能体能够通过学习达到均衡。同时，OPNDQN方法克服了完全集中式方法难以训练大规模神经网络的问题。实验结果表明，其收敛速度远高于此类方法。此外，我们的方法在算法收敛后具有较高的稳定性。实验还表明，当智能体数量较多时，我们的方法优势尤为突出。

参考文献

[1] X. Liang、T. Yan、J. Lee 和 G. Wang，《面向安全、效率与驾驶舒适性的分布式交叉口管理协议》，*IEEE*物联网期刊，2018年。[2] N. Casas，《用于城市交通信号灯控制的深度确定性策略梯度》，*arXiv*预印本 *arXiv:1703.09035*，2017年。[3] C. Li、W. Yue、G. Mao 和 Z. Xu，《城市道路网络中基于拥堵传播的瓶颈识别》，*IEEE*车辆技术汇刊，2020年。[4] J. Jin 和 X. Ma，《一种多目标基于智能体的控制方法及其在智能交通信号系统中的应用》，*IEEE*智能交通系统汇刊，第20卷，第10期，第3900–3912页，2019年。[5] T. Chu 和 J. Wang，《利用宏观基本图进行交通信号控制》，载于美国控制会议论文集，2015年。[6] T. Tan、T. S. Chu、B. Peng 和 J. Wang，《采用分散式模糊强化学习的大规模交通网格信号控制》，载于网络与系统讲义，2018年。[7] M. Abdoos、N. Mozayani 和 A. L. Bazzan，《基于多智能体Q学习的非平稳环境交通信号灯控制》，载于*2011*年第*14*届*IEEE*智能交通系统国际会议（*ITSC*）。IEEE，2011年，第1580–1585页。

[8] V. Mnih、K. Kavukcuoglu、D. Silver、A. A. Rusu、J. Veness、M. G. Bellemare、A. Graves、M. Riedmiller、A. K. Fidjeland、G. Ostrovski、S. Petersen、C. Beattie、A. Sadik、I. Antonoglou、H. King、D. Kumaran、D. Wierstra、S. Legg 和 D. Hassabis，《通过深度强化学习实现人类水平的控制》，《自然》，2015年。[9] T. Chu、S. Qu 和 J. Wang，《基于区域强化学习的大规模交通网格信号控制》，载于美国控制会议论文集，2016年。[10] R. Aragon-G´omez 和 J. B. Clempner，《面向连续时间马尔可夫博弈的交通信号控制强化学习方法》，《人工智能工程应用》，2020年。[11] X. Liang、X. Du、G. Wang 和 Z. Han，《用于交通信号灯周期控制的深度强化学习网络》，*IEEE*车辆技术汇刊，2019年。[12] K. Prabuchandran、H. K. AN 和 S. Bhatnagar，《用于交通信号控制的多智能体强化学习》，载于第*17*届*IEEE*智能交通系统国际会议（*ITSC*）。IEEE，2014年，第2529–2534页。[13] X. Wang、L. Ke、Z. Qiao 和 X. Chai，《采用新型多智能体强化学习的大规模交通信号控制》，《*IEEE*控制论汇刊》，第51卷，第1期，第174–187页，2020年。[14] T. Chu、J. Wang、L. Codeca 和 Z. Li，《面向大规模交通信号控制的多智能体深度强化学习》，*IEEE*智能交通系统汇刊，2020年。[15] Y. Chen、C. Li、W. Yue、H. Zhang 和 G. Mao，《大规模交通信号控制的工程实现：一种多智能体强化学习方法》，载于*IEEE INFOCOM 2021*——*IEEE*计算机通信会议研讨会，*INFOCOM WKSHPS 2021*，2021年。[16]《博弈论课程》，《计算机与数学及其应用》，1995年。[17] J. Hu 和 M. P. Wellman，《一般和随机博弈的纳什Q学习》，《机器学习研究杂志》，2004年。[18] D. Krajzewicz、J. Erdmann、M. Behrisch 和 L. Bieker，《SUMO——城市交通仿真的最新发展与应用》，《系统与测量进展国际期刊》，第5卷，第3&4期，2012年。[19] T. A. Haddad、D. Hedjazi 和 S. Aouag，《一种基于深度强化学习的多交叉口交通信号控制协作方法》，《人工智能工程应用》，第114卷，文章编号105019，2022年。 [在线]。可访问：https://www.sciencedirect.com/ science/article/pii/S0952197622001993

[20] F. Marcham，《TensorFlow：异构分布式系统上的大规模机器学习（初步白皮书，2015年11月9日 1929）》，图书馆，。